Visual HyperScript API Specification

Ucamco Software Product Group
March 2018

Ucamco
Former Barco ETS

March 2018

Correspondence regarding this publication can be sent to:

Ucamco NV
Bijenstraat 19,
B-9051 Gent,
Belgium

For more information:

Our web site: http://www.ucamco.com

E-mail: info@ucamco.com

About Ucamco

Ucamco (formerly Barco ETS) is a market leader in PCB CAM software, photoplotting and direct imaging systems, with a global network of sales and support centers. Headquartered in Ghent, Belgium, Ucamco has over 25 years of ongoing experience in developing and supporting leading-edge photoplotters and front-end tooling solutions for the global PCB industry. Key to this success is the company's uncompromising pursuit of engineering excellence in all its products. Ucamco also owns the IP rights on the Gerber File Format through its acquisition of Gerber Systems Corp. (1998).

Helpdesk

Europe, Middle East, Africa, Latin Amerika	Asia Pacific	North America
Customer Support for Plotters, Software en ManiaBarco AOI Monday - Friday: 08.00 AM - 6.00 PM MET ✆ + 32 9 216 99 00 General support: support@ucamco.com License: license@ucamco.com Java™ HyperTool: hypertool@ucamco.com HyperScript: hyperscript@ucamco.com General information: info@ucamco.com Sales: sales@ucamco.com	Please contact our business partner for your country during support working hours see contacts page	Customer Support for Plotters, Software en ManiaBarco AOI Monday - Friday: 08.00 AM - 6.00 PM Pacific Time Saturday : 10 am to 4 pm Pacific Time +1 949 632 6895 General support: support@ucamco.us License: license@ucamco.us Java™ HyperTool: hypertool@ucamco.com HyperScript: hyperscript@ucamco.com General information: info@ucamco.us

Hyper Script API JAPAN

構成

- class **Arc**
- class **Line**
- class **Point**
- class **Rectangle**

関数

- void **abort** (String sInfo)
- void **activate** (String layClass, String laySubclass, int layNum, boolean layAct)
- void **activate** (ObjectList layerID, boolean layAct)
- void **activate** (String layClass, String laySubclass, boolean layAct)
- void **activateAllLayers** ()
- void **activateBottomLayers** (boolean layAct)
- void **activateToggle** ()
- void **activateTopLayers** (boolean layAct)
- void **activityClean** (String sMode)
- void **activityClean** ()
- void **activityRestore** (String sMode)
- void **activityRestore** ()
- void **activityStore** (String sMode)
- void **activityStore** ()
- void **addBreak** (**Point** line_fp, **Point** line_tp)
- void **addBreak** (double line_fx, double line_fy, double line_tx, double line_ty)
- void **addBreak** (**Line** line)
- int **addCFMEEAlignmentPoint** (double point_x, double point_y)
- int **addCFMEEAlignmentPoint** (**Point** point)
- void **addDPF** (String dpf, String layName, String subClass, int layPos, String readable)
- void **addDPFDrill** (String dpf, int layPos, int drillTop, int drillBot)
- void **addDPFExtra** (String dpf, String attach, boolean bNoChecks)
- void **addDPFExtra** (String dpf, String attach)
- void **addDPFLayer** (String dpf, int layPos)
- int **addETMComponentHiPot** (int etmId, int primId, int NetPrim, int NetSecond, String TestVolt, String Duration, String LeakCurrent, String VoltType, String StartVolt, String VoltRise)
- int **addETMComponentHiPot** (int etmId, int primId, double XStart, double YStart, String AccStart, double XEnd, double YEnd, String AccEnd, int NetPrim, int NetSecond, String TestVolt, String Duration, String LeakCurrent, String VoltType, String StartVolt, String VoltRise)
- void **addFault** (String sType, double oRectangle_xmin, double oRectangle_ymin, double oRectangle_xmax, double oRectangle_ymax, String sInfo)
- void **addFault** (String sType, **Rectangle** oRectangle, String sInfo)
- void **addFault** (String sType, **Line** oLine, String sInfo)
- void **addFault** (String sType, double oPt_x, double oPt_y, String sInfo)
- void **addFault** (String sType, **Point** oPt, String sInfo)
- void **addHyperScriptMenuItem** (String sScriptPath, String sMenuItemLabel)
- void **addMaskLayer** (double dThicken)
- void **addObjectAttribute** (String sAttrName)
- void **addObjectAttribute** (String sAttrName, String sAttrValue)
- void **addOptimizedMaskLayer** (double dMinRing, double dMaxRing, double dMaskToCopper, double dMaskToMask, double dBigRing)
- void **addRefPoint** (int iIndex, double pPnt_x, double pPnt_y, boolean bOnAllActiveLay)
- void **addRefPoint** (int iIndex, **Point** pPnt, boolean bOnAllActiveLay)
- void **addShavedMaskLayer** (double dThicken, double dPadToTrack, double dPadToPad)
- void **addTeardrops** (int iMode, double dRelDiam, double dRelDist, double dAbsDiam, double dAbsDist, double dMinClr, int iOnRect, int iOnHoles)
- void **addTeardrops** (int iMode, double dRelDiam, double dRelDist, double dAbsDiam, double dAbsDist, double dMinClr, int iOnRect)
- int **addYsphotechAlignmentPoint** (int region, double point_x, double point_y)

int **addYsphotechAlignmentPoint** (int region, **Point** point)
- void **align_blocks** (double dTolerance, boolean bOnAllLayers)
- void **AmliAddUser** (String sUser, String sPassword, String sAuthorityLevel)
- void **AmliChangeUserPwd** (String sUser, String sPassword, String sNewPassword)
- String **AmliCheckUser** (String sUser, String sPassword)
- void **AmliRemoveUser** (String sUser)
- String **analyzeExternal** (String sExtName)
- void **angle** (double angle)
- double **angle** ()
- void **apeAnamorphicScale** (double dDistanceX, double dDistanceY, boolean bProportional)
- void **apeAnamorphicScale** (double dScaleX, double dScaleY)
- void **apeAttribute** (String name, String value)
- String **apeAttribute** (String name)
- String **apeAttribute** ()
- void **apeCorners** (String sCorners)
- String **apeCorners** ()
- void **apeCreateBox** (int apeNum, double xsize, double ysize, String corners, double xcutoff, double ycutoff)
- void **apeCreateCircle** (int apeNum, double dia)
- void **apeCreateContour** (int apeNum, double stroke)
- void **apeCreateDonut** (int apeNum, double outer, double inner, String kind)
- void **apeCreateOblong** (int apeNum, double xsize, double ysize)
- void **apeCreateOctagon** (int apeNum, double size)
- void **apeCreateRectangle** (int apeNum, double xsize, double ysize)
- void **apeCreateText** (int iApeNum, double dHeight, String sText, double dRotation)
- void **apeCreateText** (int iApeNum, double dHeight, String sText)
- void **apeCreateThermal** (int apeNum, double outer, double inner, double gap, int numGap, double angle, String kind)
- **Rectangle apeEnclosingBox** ()
- int **apeExtlinkCheck** ()
- String **apeExtlinkPath** ()
- String **apeExtlinkPathString** ()
- boolean **apeExtlinkRelative** ()
- int **apeExtlinkStatus** ()
- void **apeGap** (double dGap)
- double **apeGap** ()
- boolean **apeHasPattern** (boolean bUsed)
- void **apeHeight** (double dHeight)
- double **apeHeight** ()
- int **apeIndex** ()
- String **apeInfo** ()
- void **apeInner** (double dInner)
- double **apeInner** ()
- void **apeKind** (String sKind)
- String **apeKind** ()
- int **apeMaxNetNumber** ()
- void **apeMirror** (String sMirror)
- String **apeMirror** ()
- void **apeName** (String sName)
- String **apeName** ()
- void **apeNumber** (int iNumber)
- int **apeNumber** ()
- void **apeNumberGap** (int iNumGap)
- int **apeNumberGap** ()
- int **apeNumberObject** (String sObjectClass)
- int **apeNumberObject** ()
- int **apeNumberRegions** ()
- int **apeNumContours** ()
- void **apeOuter** (double dOuter)
- double **apeOuter** ()
- void **apePattern** (String sPattern)
- String **apePattern** ()

void **apePatternAngle** (double dPatternAngle)
- double **apePatternAngle** ()
- void **apePatternStep** (double dPatternStep)
- double **apePatternStep** ()
- void **apePatternWidth** (double dPatternWidth)
- double **apePatternWidth** ()
- void **apePatternX** (double dX)
- double **apePatternX** ()
- void **apePatternY** (double dY)
- double **apePatternY** ()
- **Rectangle apeRectangle** ()
- void **apeReverse** (boolean bReverse)
- boolean **apeReverse** ()
- void **apeRotation** (double dRotation)
- double **apeRotation** ()
- void **apeScale** (double dScale)
- double **apeScale** ()
- boolean **apeSelection** ()
- **Rectangle apeSelectionEnclosingBox** ()
- String **apeShape** ()
- void **apeSize** (double dSize)
- double **apeSize** ()
- void **apeStartAngle** (double dStartAngle)
- double **apeStartAngle** ()
- void **apeString** (String sString)
- String **apeString** ()
- void **apeStroke** (double dStroke)
- double **apeStroke** ()
- double **apeSurface** ()
- void **apeThickenThin** (double value, boolean keepArcs)
- void **apeWidth** (double dWidth)
- double **apeWidth** ()
- void **apeXCutOff** (double dXCutOff)
- double **apeXCutOff** ()
- void **apeXSize** (double dXSize)
- double **apeXSize** ()
- void **apeYCutOff** (double dYCutOff)
- double **apeYCutOff** ()
- void **apeYSize** (double dYSize)
- double **apeYSize** ()
- int **applyHorns** (String hornType, double minimumClearance, ObjectList params)
- **Arc Arc** (double ptFromX, double ptFromY, double ptToX, double ptToY, double ptCenterX, double ptCenterY, String sSense, String sUnits)
- **Arc Arc** (double ptFromX, double ptFromY, double ptToX, double ptToY, double ptCenterX, double ptCenterY, String sSense)
- **Arc Arc** (**Arc** oArc)
- **Arc Arc** (**Point** ptFrom, **Point** ptTo, **Point** ptCenter, String sSense)
- void **autofixtureBuildFixture** (boolean bFixtureBuild, String sFixture)
- void **autofixtureDo** ()
- void **autofixtureMicroAdjustment** (boolean bMicroAdjustment, int iNbrOfTestPoints, double dTestPointDiameter, double dTestPointShiftEdge, double dTestPointShiftValue, double dTestPointPitch, double dClearanceFactor, double dCenterDiameter)
- void **autofixtureNetlist** (boolean bNetlist, boolean bNetlistBuild, boolean bNetlistExpand)
- void **autofixtureOutput** (boolean bOutput)
- void **autofixtureTestpoints** (boolean bTestPoints, int iLoop, boolean bUseMasks, boolean bProbeSwaping, boolean bHandlePaintedPads, boolean bCircuitryCheck, boolean bFilterCopperAreas, boolean bViaOfSMDs, boolean bDrillsWithoutPad)
- void **autofixtureTestpointsBot** (boolean bPointsBot1, boolean bPointsBot2, boolean bPointsBot3, boolean bPointsBot4, boolean bPointsBot5, boolean bPointsBot6, boolean bPointsBot7)
- void **autofixtureTestpointsTop** (boolean bPointsTop1, boolean bPointsTop2, boolean bPointsTop3, boolean bPointsTop4, boolean bPointsTop5, boolean bPointsTop6, boolean bPointsTop7)

void **blockEdit** ()
- void **blockMultiEdit** ()
- int **BlockReconstruct** ()
- void **boardSnapshot** (boolean graph, String templPath, boolean pio, String pioPath)
- void **buildSubJobs** ()
- void **calculateImpedance** (String sImpConfig, ObjectList parameters)
- boolean **canRead** (ObjectList fileInfo)
- boolean **canWrite** (ObjectList fileInfo)
- void **center** (double center_x, double center_y)
- void **center** (**Point** center)
- **Point center** ()
- void **centerX** (double centerX)
- void **centerY** (double centerY)
- void **chain** ()
- void **chamferJoin** (double pt_x, double pt_y, double disX, double disY)
- void **chamferJoin** (**Point** pt, double disX, double disY)
- void **changeDirection** (double p_x, double p_y)
- void **changeDirection** (**Point** p)
- int **changePrioPlotQueue** (String sRipHost, int iJobId, int iPriority)
- boolean **checkDrillInfo** (boolean bSelNonPlated, boolean bAssignAttributes, boolean bBlocksOnly)
- String **chooseDirPath** (String sTitle, String sStartDir)
- String **chooseDirPath** (String sTitle)
- String **chooseDirPath** ()
- String **chooseFilePath** (String sTitle, String sStartDir, String sFileMask)
- String **chooseFilePath** (String sTitle)
- String **chooseFilePath** (String sStartDir, String sFileMask)
- String **chooseFilePath** ()
- void **cleanApertures** ()
- void **cleanApeTables** ()
- void **cleanETMComponentLayers** (int type)
- void **cleanSubJobs** ()
- void **cleanUfd** (String sUfdName)
- void **cleanUnderBlo** ()
- void **cleanup** (double dReconstructArcs, double dValidateArcs, double dRemoveObsoleteObjects, double dRemoveSmallObjects, double dReconnectObjects, boolean bReconstructArcs, boolean bValidateArcs, boolean bRemoveObsoleteObjects, boolean bRemoveSmallObjects, boolean bReconnectObjects)
- void **clearance** (double clearance)
- double **clearance** ()
- void **clearanceCheckMAT** (double dPadSpread, double dSmdSpread, double dTrackSpread, double dAreaSpread, double dPadPadClearance, double dPadSmdClearance, double dPadTrackClearance, double dPadAreaClearance, double dSmdSmdClearance, double dSmdTrackClearance, double dSmdAreaClearance, double dTrackTrackClearance, double dTrackAreaClearance, double dAreaAreaClearance, boolean bCheckSameNetSpacing, boolean bFastMode, int iShiftMode, double dMinCopper)
- void **clearanceCheckMAT** (double dPadSpread, double dSmdSpread, double dTrackSpread, double dAreaSpread, double dPadPadClearance, double dPadSmdClearance, double dPadTrackClearance, double dPadAreaClearance, double dSmdSmdClearance, double dSmdTrackClearance, double dSmdAreaClearance, double dTrackTrackClearance, double dTrackAreaClearance, double dAreaAreaClearance, boolean bCheckSameNetSpacing, boolean bFastMode, int iShiftMode)
- void **clearanceCheckMAT** (double dPadSpread, double dSmdSpread, double dTrackSpread, double dAreaSpread, double dPadPadClearance, double dPadSmdClearance, double dPadTrackClearance, double dPadAreaClearance, double dSmdSmdClearance, double dSmdTrackClearance, double dSmdAreaClearance, double dTrackTrackClearance, double dTrackAreaClearance, double dAreaAreaClearance, boolean bCheckSameNetSpacing, boolean bFastMode)
- void **clearMessages** ()
- boolean **clipping** (int iClipReference, String sClipSide, double dClipClr, double dMinLineLength, boolean bRounded)
- boolean **clipSilk** (double dClr, double dMinLen)
- void **closeAMLIJobManager** ()
- void **closeAnamorphicScale** ()
- void **closeApeCreator** ()

void **closeApeEditor** ()
- void **closeApertureAttributes** ()
- void **closeApertureManager** ()
- void **closeAttributeEditor** ()
- void **closeAttributeManager** ()
- void **closeAutoDrill** ()
- void **closeAutoDrillEditor** ()
- void **closeAutoFixture** ()
- void **closeBarcode** ()
- void **closeBarcode128** ()
- void **closeBoardAnalyzer** ()
- void **closeBoardSnapshot** ()
- void **closeCalculatorSetup** ()
- void **closeCamtek** ()
- void **closeCheckList** ()
- void **closeCheckListDefineChecklist** ()
- void **closeCheckListDefineSteps** ()
- void **closeClipping** ()
- void **closeColor** ()
- void **closeConnect** ()
- void **closeContourHandling** ()
- void **closeConvertAttributes** ()
- void **closeCopperBalance** ()
- void **closeCopperRepair** ()
- void **closeCoverlayOptimizer** ()
- void **closeCU9000Dialog** ()
- void **closeDatums** ()
- void **closeDistort** ()
- void **closeDrawSlots** ()
- void **closeDRC** ()
- void **closeDrillInfo** ()
- void **closeDrillMap** ()
- void **closeDrillOptimizer** ()
- void **closeDrillRoutSetups** ()
- void **closeDrillTolerance** ()
- void **closeDrillToolManager** ()
- void **closeDsAoi** ()
- void **closeDSAOIialog** ()
- void **closeDsAoiPreview** ()
- void **closeEditingToolbox** ()
- void **closeEditVectorText** ()
- void **closeErrors** ()
- void **closeEtchCompensation** ()
- void **closeExpand** ()
- void **closeExternalLinkManager** ()
- void **closeFiducials** ()
- void **closeFillAngledPattern** ()
- void **closeFillPattern** ()
- void **closeFillVector** ()
- void **closeFlashMaker** ()
- void **closeFlexManager** ()
- void **closeFlipJob** ()
- void **closeFrame** (String sFrameName)
- void **closeGridParameters** ()
- void **closeHiPot** ()
- void **closeImageCompare** ()
- void **closeImpedanceControl** ()
- void **closeImportODBxx** ()
- void **closeInsertContourText** ()
- void **closeInsertVectorText** ()
- void **closeJobDefinition** ()

void **closeJobEdit** ()
- void **closeJobEditor** ()
- void **closeJobEditorOptions** ()
- void **closeJobMerge** ()
- void **closeJobPlaneSetup** ()
- void **closeJobPrint** ()
- void **closeLayerEdit** ()
- void **closeLegendOptimizer** ()
- void **closeLoadCheckList** ()
- void **closeMagnifier** ()
- void **closeMarkupAssistant** ()
- void **closeMessages** ()
- void **closeMLIOutput** ()
- void **closeModels** ()
- void **closeNetCompare** ()
- void **closeNonFunctionalPad** ()
- void **closeNumbers** ()
- void **closeObjectAttributes** ()
- void **closeObjectCompare** ()
- void **closeOutputAccumatch** ()
- void **closeOutputAOI** ()
- void **closeOutputCAD** ()
- void **closeOutputCamtek** ()
- void **closeOutputDrillRout** ()
- void **closeOutputDsDi** ()
- void **closeOutputDsDiPreview** ()
- void **closeOutputNetlist** ()
- void **closeOutputOrbot** ()
- void **closeOutputSapphire** ()
- void **closeOutputScoring** ()
- void **closeOutputSmartArgos** ()
- void **closeOutputTrackscan** ()
- void **closeOutputUxpAutomanager** ()
- void **closeOutputUxpEtec** ()
- void **closePanelFramesCoupons** ()
- void **closePanelLinks** ()
- void **closePanelPlus** ()
- void **closePanelReproduce** ()
- void **closePanelSetup** ()
- void **closePanelStepRepeat** ()
- void **closePlaneAdjuster** ()
- void **closePlotParameters** ()
- void **closePPMonitor** ()
- void **closeQueryNet** ()
- void **closeQueryObject** ()
- void **closeReferencePoints** ()
- void **closeRegister** ()
- void **closeRemoveAttributes** ()
- void **closeRepair** ()
- void **closeRoutManager** ()
- void **closeRoutManagerCleanUp** ()
- void **closeRoutManagerDimensioning** ()
- void **closeRoutManagerEditor** ()
- void **closeRoutManagerTools** ()
- void **closeSaveLayout** ()
- void **closeSecureEtchCompensation** ()
- void **closeSelections** ()
- void **closeSetupOptions** ()
- void **closeSetupSave** ()
- void **closeShavePads** ()
- void **closeSignalLayerAdjuster** ()

void **closeSignalLayerAdjusterAssistant** ()
- void **closeSilkOptimizer** ()
- void **closeSmartCamtek** ()
- void **closeSmartDRC** ()
- void **closeSmartFix** ()
- void **closeSmartplot** ()
- void **closeSmartSR** ()
- void **closeSmartStart** ()
- void **closeSoldermask** ()
- void **closeSoldermaskOptimizer** ()
- void **closeTearDrop** ()
- void **closeTechnicalAnalyzer** ()
- void **closeTestpointEdit** ()
- void **closeToolbarManager** ()
- void **closeToolbars** ()
- void **closeTransformObjects** ()
- void **closeTransformObjectsBGAPads** ()
- void **closeTransformObjectsBGATracks** ()
- void **closeTransformObjectsEdit** ()
- void **closeTransformObjectsRescale** ()
- void **closeUcamDbEditor** ()
- void **closeUndoRedoDetails** ()
- void **closeUTest** ()
- void **closeUtestUtilities** ()
- void **closeValidateLayer** ()
- void **closeVectorTextFont** ()
- void **closeVerifyArcsDraws** ()
- void **closeViewGuide** ()
- void **colorAll** (String exclSubClass, boolean bKeepLayActivity)
- void **colorAll** (String exclSubClass)
- void **compareImage** (String reference, boolean bAutoAlign, double missingTol, double exceedingTol, int iErrorAccuracy, ObjectList revPolArr, int compSelMode)
- void **compareImage** (String reference, boolean bAutoAlign, double missingTol, double exceedingTol, int iErrorAccuracy, ObjectList revPolArr)
- void **compareImage** (double missingTol, double exceedingTol, int iErrorAccuracy, boolean bRevPolarityCur, boolean bRevPolarityRef, int compSelMode)
- void **compareImage** (double missingTol, double exceedingTol, int iErrorAccuracy, boolean bRevPolarityCur, boolean bRevPolarityRef)
- void **compareNet** (int iMode, boolean bCheckFlash)
- void **compareNet** (int iMode, boolean bCheckFlash, String sReferenceFile, boolean bPanelize)
- void **compareNet** (int iMode, boolean bCheckFlash, boolean bIgnoreOutline, double dOutlineMargin, boolean bIgnoreNPTHPads, double dNPTHExpandMargin, String sReferenceFile, boolean bPanelize)
- void **compareNet** (boolean bShorts, boolean bOpens, boolean bLostElements, boolean bDoubleNet, boolean bNotCovered, boolean bCheckFlash, boolean bIgnoreOutline, double dOutlineMargin, boolean bIgnoreNPTHPads, double dNPTHExpandMargin, String sReferenceFile, boolean bPanelize, boolean bBuildNetlist)
- void **compareObjects** (String referenceJob, double xTol, double yTol, boolean bWindow, boolean bObjMoved, boolean bObjAdded, boolean bObjNet, boolean bApeShape, boolean bApeSize, boolean bApeOrder)
- void **compensate** (String sSense, double dis)
- void **complexEdit** ()
- int **connectPadTrack** (double dActiveRadius, double dSnapRadius, boolean bUseNetlist)
- void **connectTracks** ()
- boolean **contourizeBitmap** (int iPpi, double dMargin, double dDxdy, double dDx, double dDy)
- void **contourizeExact** ()
- void **contourizeExactAperture** ()
- boolean **contourizePatterns** (int iPpi)
- boolean **contourizePatternsinJob** (int iPpi)
- void **contourThickenThin** (double value, boolean bKeepArcs)
- boolean **convertGar** (String sInputFile, String sGarFile, String sOutputFile)
- void **copperBalancePad** (double dMinClrToCopper, double dMinClrToBoard, double dMinConSurface,

String sFillPattern, double dPatternClr, double dApeSize, String sApeShape)
- void **copperBalanceSolid** (double dMinClrToCopper, double dMinClrToBoard, double dMinConSurface, double dApeSize)
- void **copperBalanceTrack** (double dMinClrToCopper, double dMinClrToBoard, double dMinConSurface, String sLineStyle, double dPatternClr, double dApeSize, double dRotation)
- void **copperCount** (String sOpt)
- void **copperRepair** (String sOpt, double smallerThan, double minSize, double expand)
- void **copy** (double pt_x, double pt_y)
- void **copy** (**Point** pt)
- void **copyOutline** (double refPoint_x, double refPoint_y, double offset_x, double offset_y, double rotation)
- void **copyOutline** (**Point** refPoint, **Point** offset, double rotation)
- void **copyToClipboard** ()
- void **coreInfo** ()
- int **countAmbiguousContours** ()
- int **countAmbiguousContoursOnLayer** ()
- int **countInvalidArcs** ()
- int **countInvalidArcsOnLayer** ()
- int **countInvalidDraws** ()
- int **countInvalidDrawsOnLayer** ()
- int **countOpenContours** ()
- int **countOpenContoursOnLayer** ()
- int **countOverlapContours** ()
- int **countOverlapContoursOnLayer** ()
- int **countUndefinedApertures** ()
- int **countUndefinedAperturesOnLayer** ()
- int **countZeroDrawsArcs** (double dMaxLength, boolean bFunctional, boolean bNonFunctional)
- int **countZeroDrawsArcsOnLayer** (double dMaxLength, boolean bFunctional, boolean bNonFunctional)
- void **countZeroLengthDrawsArcs** (double dMaxLength, boolean bFunctional, boolean bNonFunctional)
- void **createAperture** (int iApeNum, String sApeName, String sApeDef, ObjectList attrArray)
- boolean **createBarcode128** (double dHeight, double dNarrowX, String sValue)
- boolean **createBarcode39** (double dHeight, double dNarrowX, double dRatio, String sValue)
- boolean **createBarcodeInterleaved25** (double dHeight, double dNarrowX, String sValue)
- void **createBlockAperture** (int iApeNum, String sApeName, String sApeDef, ObjectList attrArray, int iMode, boolean bWithCenter, double pt_x, double pt_y, String sExtFile)
- void **createBlockAperture** (int iApeNum, String sApeName, String sApeDef, ObjectList attrArray, int iMode, boolean bWithCenter, **Point** pt, String sExtFile)
- void **createComplexAperture** (int iApeNum, String sApeName, String sApeDef, ObjectList attrArray, boolean bUseRegion, boolean bWithCenter, double pt_x, double pt_y)
- void **createComplexAperture** (int iApeNum, String sApeName, String sApeDef, ObjectList attrArray, boolean bUseRegion, boolean bWithCenter, **Point** pt)
- void **createDataMatrix** (String sTextToEncode, String sMode, String sFormat, double dDotSize, double dRotation, String sMirror, double dClearance, boolean bReverse)
- void **createDrill** (String layName, String subClass, int from, int to)
- void **createDrill** (int laynum, int drillFrom, int drillTo)
- void **createExtra** (String layName, String subClass, String attach, int index)
- void **createExtra** (String layName, String subClass, String attach)
- void **createExtra** (String attach)
- void **createLayer** (String layName, String subClass, int layPos, String readable)
- void **createLayer** (int laynum)
- void **createQRCode** (String sCode, double dDotSize, double dAngle, String sMirror, double dClr, double dLabelClr, String sLabelPos, boolean bMicroQR, boolean bReverse)
- void **createQRCode** (String sCode, double dDotSize, double dAngle, String sMirror, double dClr, String sLabel, double dLabelClr, String sLabelPos, boolean bMicroQR, boolean bReverse)
- void **createQRCode** (String sCode, double dDotSize, double dAngle, String sMirror, double dClr, boolean bMicroQR, boolean bReverse)
- void **createSubJob** (int from, int to, int selectedSubJob, int selectedLevel)
- void **createVoronoiDiagram** (int iEdgeTypes)
- void **createVoronoiDiagram** (int iEdgeTypes, boolean bExpandArcs)
- void **createVoronoiEdgesExtFile** (boolean bExpandArcs, String sFilePath, String sOptions)
- void **CU9000ApplyPlotstamps** (ObjectList plotstamps)
- boolean **CU9000CheckPlotstamps** ()

boolean **CU9000DetectAutoAreas** (String resultLayerName, String referenceLayerName, double margin)
- boolean **CU9000DetectExactAreas** (String resultLayerName, int blockMode, String pcbName, String referenceLayerName, double margin, double outline)
- int **CU9000DetectGlobalAlignment** (String sAPRefLayerName)
- int **CU9000DetectGlobalAlignment** ()
- int **CU9000DetectLocalAlignmentPoints** (String sAPRefLayerName)
- int **CU9000DetectLocalAlignmentPoints** ()
- boolean **CU9000DetectRectangularAreas** (String resultLayerName, int blockMode, String pcbName, String referenceLayerName, double margin)
- ObjectList **CU9000GetPlotstamps** ()
- void **CU9000GUIApply** ()
- void **CU9000GUILoadAlignment** (String sAlignmentPath)
- void **CU9000GUILoadBrd** (String sBrdPath)
- void **CU9000GUILoadRgi** (String sRgiPath)
- void **CU9000GUISaveAlignment** (String sAlignmentPath)
- boolean **CU9000LoadBoardSetup** (String path)
- boolean **CU9000LoadResistSetup** (String path)
- boolean **CU9000LoadResources** (String sPropertiesPath, String sPropertiesName, String sConversionFileName)
- ObjectList **CU9000OrderPlotstamps** (Object[] plotstamps, String sLevel, String sAtLevel, String sStart, String sOrder)
- boolean **CU9000Output** (String machine)
- void **CU9000SaveBPIs** ()
- void **CU9000SaveLocalAlignmentPoints** (String sOutputFilePath)
- void **CU9000SaveLocalAlignmentPoints** (String sOutputFilePath, boolean bShareAlignmentMarks)
- boolean **CU9000SetParameters** (String xmlFile)
- void **cutToClipboard** ()
- void **dbBoolean** (String dbKey, Boolean bValue)
- boolean **dbBoolean** (String dbKey)
- boolean **dbBooleanDef** (String dbKey, boolean bDefault)
- void **dbDouble** (String dbKey, Boolean dValue)
- double **dbDouble** (String dbKey)
- double **dbDoubleDef** (String dbKey, double dDefault)
- void **dbInteger** (String dbKey, Integer iValue)
- int **dbInteger** (String dbKey)
- int **dbIntegerDef** (String dbKey, int iDefault)
- void **dbPath** (String dbKey, String sPath)
- String **dbPath** (String dbKey)
- String **dbPathDef** (String dbKey, String sDefault)
- void **dbString** (String dbKey, String sValue)
- String **dbString** (String dbKey)
- String **dbStringDef** (String dbKey, String sDefault)
- void **dbUnitValue** (String dbKey, String sValue)
- void **dbUnitValue** (String dbKey, Double dValue)
- double **dbUnitValue** (String dbKey)
- double **dbUnitValueDef** (String dbKey, double dDefault)
- double **dbUnitValueDef** (String dbKey, String sDefault)
- void **defaultOrder** ()
- void **defineFirst** (double p_x, double p_y)
- void **defineFirst** (**Point** p)
- void **defineGroup** (double p_x, double p_y, int iGroupNumber)
- void **defineGroup** (**Point** p, int iGroupNumber)
- void **defineSelectedGroup** ()
- void **delete** ()
- void **deleteAllCFMEEAlignmentPoints** ()
- void **deleteAllRefPoints** (boolean bOnAllActiveLay)
- void **deleteAllYsphotechAlignmentPoints** (int region)
- void **deleteAperture** ()
- void **deleteApertureAttribute** (String sAttributeName)
- void **deleteCFMEEAlignmentPoint** (int point)
- void **deleteDouble** ()

void **deleteLayerByClass** (String className, String subClass, int num, String side)
- void **deleteLayersByActivation** (boolean active)
- void **deleteLayersByName** (String layName)
- void **deleteLayersByNames** (String layNames)
- void **deleteLayersByPlane** (int plane)
- void **deleteRefPoint** (int iIndex, boolean bOnAllActiveLay)
- void **deleteRefPoints** (ObjectList Indexes, boolean bOnAllActiveLay)
- void **deleteSubJob** (int index, int level)
- void **deleteTrueObjects** ()
- void **deleteWithApe** ()
- void **deleteWorkspace** (String sWorkspaceName)
- void **deleteYsphotechAlignmentPoint** (int region, int point)
- void **deleteZeroLengthDraws** (double dMaxLength, boolean bFunctional, boolean bNonFunctional)
- void **deselectAll** ()
- void **deselectAllApertures** ()
- void **deselectAperture** (ObjectList apeIndexArray)
- void **deselectAperture** ()
- void **deselectObjectAttribute** (String sAttrName, String sAttrValue)
- void **deselectObjectAttribute** (String sAttrName)
- void **deselectObjectByAttribute** (String sAttrName, String sAttrValue)
- void **deselectObjectByAttribute** (String sAttrName)
- boolean **detectPCBOutlines** (String sLayName, String sParams)
- boolean **DetectPlaceHolders** (String sHandlerName, String sParams)
- boolean **DetectPlaceHoldersAtLayer** (String sHandlerName, String sParams)
- void **dimensioning** (String sType, double dApertureSize, ObjectList oPoints, boolean bShowErrors, double dArrowHeadWidth, double dArrowHeadHeight, double dRuleToElement, double dRuleToDimLine, double dTextToDimLine, double dTextHeight, double dTextWidth, double dTolerancePos, double dToleranceNeg, double dToleranceScale, int iFormat, boolean bProjectionHorizontal, boolean bProjectionVertical, String sFontName, int iFontStyle, int iFontSize, String sLabel)
- String **direction** ()
- void **direction** (String sDirection)
- void **distance** (double distance)
- double **distance** ()
- boolean **distort** (double x, double y, double pCenter_x, double pCenter_y)
- boolean **distort** (double x, double y, **Point** pCenter)
- boolean **distort** (double x, double y)
- void **doActiveFunction** ()
- void **doCancelActiveFunction** ()
- void **doCopy** (double offset_x, double offset_y)
- void **doCopy** (**Point** offset)
- void **doMove** (double offset_x, double offset_y)
- void **doMove** (**Point** offset)
- void **doOption** (String sOption)
- String **doOption** ()
- void **doRemoveAttribute** (boolean jobAttr, boolean layAttr, boolean apeAttr, boolean objAttr)
- void **drag** (double clickp_x, double clickp_y, double dRadius, double offset_x, double offset_y, double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax)
- void **drag** (**Point** clickp, double dRadius, **Point** offset, **Rectangle** rect)
- void **drag** (double clickp_x, double clickp_y, double dRadius, double offset_x, double offset_y)
- void **drag** (**Point** clickp, double dRadius, **Point** offset)
- void **dragAngle** (double pt_x, double pt_y, double dRadius, double dist, double mlen, boolean bUseLimit)
- void **dragAngle** (**Point** pt, double dRadius, double dist, double mlen, boolean bUseLimit)
- void **dragAngle** (double pt_x, double pt_y, double dRadius, double dist, double mlen)
- void **dragAngle** (**Point** pt, double dRadius, double dist, double mlen)
- void **dragLayer** (String prevClass, String newClass, int prevPosition, int newPosition, boolean duplicate)
- **Line dragLine** (String sLabel)
- **Rectangle dragRectangle** (String sLabel)
- void **drawLastPlanesInFront** (boolean bDo)
- void **drawSlots** (String sDPFApeRef, double dTolerance, String sDPFSlotApe)
- void **drawSlotsSelect** (String sDPFApe, double dTolerance)
- void **drillMapReplace** (String sSymbolFilePath, ObjectList oMappingTable)

void **DSAOIAlignmentApply** ()
- void **DSAOIAlignmentDetect** (String sMachineType, String sObjectRestrictions, boolean bPositive, boolean bNegative, double dMinimumSize, double dMaximumSize)
- boolean **DSAOIApply** ()
- void **DSAOIAreaDetection** (boolean detectImportantLine, boolean detectImportantSpace, boolean detectImportantClearance, boolean detectImportantDrill, boolean detectImportantFutureDrill, boolean detectImportantMaskOpenings, boolean detectProhibitCopper, boolean detectProhibitSpace, boolean detectProhibitNonfunctionalCopper, ObjectList importantLineWidth, Object[] importantSpaceWidth, double importantClearanceWidth, double importantDrillSpreadValue, double prohibitLineWidth, double prohibitSpaceWidth, double minSliverSize, String pathStrategy, boolean mergeOutput, boolean clipProhibitWithImportant, boolean outputNormalAreas, boolean maskPolarity, boolean paintedArea, double paintedAreaValue, String PCBName, boolean outputDrillBinary)
- void **DSAOIAreaDetection** (boolean detectImportantLine, boolean detectImportantSpace, boolean detectImportantClearance, boolean detectImportantDrill, boolean detectImportantFutureDrill, boolean detectImportantMaskOpenings, boolean detectProhibitCopper, boolean detectProhibitSpace, boolean detectProhibitNonfunctionalCopper, ObjectList importantLineWidth, Object[] importantSpaceWidth, double importantClearanceWidth, double importantDrillSpreadValue, double prohibitLineWidth, double prohibitSpaceWidth, double minSliverSize, String pathStrategy, boolean mergeOutput, boolean clipProhibitWithImportant, boolean outputNormalAreas, boolean maskPolarity, String PCBName, boolean outputDrillBinary)
- void **DSAOIAreaDetection** (boolean detectImportantLine, boolean detectImportantSpace, boolean detectImportantClearance, boolean detectImportantDrill, boolean detectImportantFutureDrill, boolean detectImportantMaskOpenings, boolean detectProhibitCopper, boolean detectProhibitSpace, boolean detectProhibitNonfunctionalCopper, ObjectList importantLineWidth, Object[] importantSpaceWidth, double importantClearanceWidth, double importantDrillSpreadValue, double prohibitLineWidth, double prohibitSpaceWidth, double minSliverSize, String pathStrategy, boolean mergeOutput, boolean clipProhibitWithImportant, boolean outputNormalAreas, boolean maskPolarity, String PCBName)
- void **DSAOIAreaDetection** (boolean detectImportantLine, boolean detectImportantSpace, boolean detectImportantClearance, boolean detectImportantDrill, boolean detectImportantFutureDrill, boolean detectImportantMaskOpenings, boolean detectProhibitCopper, boolean detectProhibitSpace, boolean detectProhibitNonfunctionalCopper, ObjectList importantLineWidth, double importantSpaceWidth, double importantClearanceWidth, double importantDrillSpreadValue, double prohibitLineWidth, double prohibitSpaceWidth, double minSliverSize, String pathStrategy, boolean mergeOutput, boolean clipProhibitWithImportant, boolean outputNormalAreas, boolean maskPolarity, String PCBName)
- void **DSAOIAreasApply** ()
- void **DSAOIDetectRectangularArea** (double dMargin, String sBlockMode, String sPCBName, String sReferenceLayerName, boolean bIsSingleLevel, boolean bIsInspection, boolean bIsPmiP1, boolean bIsPmiP2, boolean bIsDrcP1, boolean bIsDrcP2)
- void **DSAOILayerList** ()
- void **DSAOILayerListAreaApply** ()
- void **DSAOILayerListAreaOutput** ()
- void **DSAOILayerListAreaSelect** ()
- void **DSAOILayerListGroupValue** (String sNewValue)
- void **DSAOILayerListRowDeselect** (int iIndexFrom, int iIndexTo)
- void **DSAOILayerListRowDeselect** (int iRow)
- void **DSAOILayerListRowSelect** (int iIndexFrom, int iIndexTo)
- void **DSAOILayerListRowSelect** (int iRow)
- boolean **DSAOILoadLayerListProfile** (String pro)
- boolean **DSAOIOutput** ()
- void **DSAOIpinpointDetection** (double dStep, double dInfinity, String sOutputFilePath)
- void **DSAOIpinpointDetection** (double dStep, double dInfinity, String sPCBName, String sOutputFilePath, String sLocation)
- void **DSAOIPositionApply** ()
- void **DSAOISetApplyToBackLayers** (boolean bValue)
- void **DSAOISetApplyToFrontLayers** (boolean bValue)
- void **DTMCalculate** (String sPlatingType, String sToleranceScript)
- boolean **DTMCreateSymbolDrawing** ()
- boolean **DTMLoadData** ()
- void **DTMLoadToleranceFile** (String sToleranceFileName)
- void **DTMRemoveAttributes** ()
- void **DTMSaveDataToAttributes** (String sJobName)

void **DTMUpdateDPF** (String sJobName)
- void **duplicateAperture** ()
- void **duplicateLayer** (String layName, String newName)
- void **dwAnnotate** (String annotationLayerName, String sChipID)
- void **dwAnnotate** (String annotationLayerName)
- void **dwApplyTransform** (String sResultFilePath)
- void **editAperture** (int iApeNum, String sApeName, String sApeDef)
- void **emptyClipboard** ()
- void **enlargePads** (String absRel, double aVal, boolean bExclcon, boolean bUseBGA)
- String **envString** (String name)
- void **equalizeTrackSpace** ()
- void **etchCompensation** (boolean bUseExcludeAreas, boolean bCreateLayerBackup, boolean bShowLayerBackup, int iOutStyle, boolean bAsymmetricPadTrackComp, ObjectList arrPrefOffset)
- void **etchCompensation** (boolean bUseExcludeAreas, boolean bUseCompensateAreas1, boolean bUseCompensateAreas2, boolean bCreateLayerBackup, boolean bShowLayerBackup, int iOutStyle, boolean bAsymmetricPadTrackComp, ObjectList arrPrefOffset, Object[] arrCompLay1, Object[] arrCompLay2)
- void **expandArcs** ()
- void **expandBlock** ()
- void **expandNibble** (double overlapValue, double pitchValue, boolean useOverlap)
- void **expandText** ()
- void **expandTrueObjects** ()
- void **expandVtx** ()
- void **externalLinkManagerCheck** ()
- void **filletJoin** (double pt_x, double pt_y, double dis)
- void **filletJoin** (**Point** pt, double dis)
- void **fillPolygon** (boolean bDirection)
- void **fillPolygonCCW** ()
- void **fillPolygonCW** ()
- void **fillWithAngledPattern** (String shape, double size, double pitch, double angle)
- void **fillWithPatternPads** (String sKind, boolean bKeepEdge, double pGridOrigin_x, double pGridOrigin_y, double pGridStep_x, double pGridStep_y)
- void **fillWithPatternPads** (String sKind, boolean bKeepEdge, **Point** pGridOrigin, **Point** pGridStep)
- void **fillWithPatternStarburst** (int iSegments, String sKind, int dBlack, boolean bWithCenter, double pCenter_x, double pCenter_y, boolean bKeepEdge, double dEdgeWith)
- void **fillWithPatternStarburst** (int iSegments, String sKind, int dBlack, boolean bWithCenter, **Point** pCenter, boolean bKeepEdge, double dEdgeWith)
- void **fillWithPatternTracks** (String sPattern, double dStep, double dWidth, double dRotation, boolean bKeepEdge)
- int **fillWithVectors** (double dOverlap, double dDiameter, int iApeCount, int iApeNum, String sFillOpt)
- void **findSections** (String szOptions)
- void **findSlots** ()
- int **findStandardShape** (double dTolerance, String szOpt, String szAction)
- void **flashMakerDeleteComplex** ()
- void **flashMakerDeselectComplex** ()
- void **flashMakerFind** ()
- void **flashMakerFindStandardShapes** (Uxjob oJob)
- void **flashMakerReplace** ()
- void **flashMakerReplaceStandardShapes** (Uxjob oJob)
- void **flashMakerSetup** (double minCutoff, double minSize, double maxSize, boolean useTol, double tol, boolean useMask, boolean deselNonModel)
- void **flashMakerSetup** (double minCutoff, double minSize, double maxSize, boolean useTol, double tol, boolean useMask, boolean completelyFree, boolean deselNonModel)
- void **flipJob** (String mirror, boolean bFlipBuildup, boolean bFlipAttachNone, String suffix)
- void **forEachApe** (String sType)
- void **forEachApe** ()
- void **forEachArc** ()
- void **forEachDraw** ()
- void **forEachDrill** (String sSubClass)
- void **forEachDrill** ()
- void **forEachExtra** (String sSubClass, String sAttach)

void **forEachExtra** (String sSubClass)
- void **forEachExtra** ()
- void **forEachFlash** ()
- void **forEachI8Job** (String serverName, String dbname, String username, String password, String context, String i8path)
- void **forEachInRectangle** (**Rectangle** rect, boolean opt)
- void **forEachInRectangle** (**Rectangle** rect)
- Object **forEachItem** (ObjectList items)
- void **forEachJobNet** ()
- void **forEachLayer** (String sClass, String sSubClass, String sAttach)
- void **forEachLayer** (String sClass, String sSubClass)
- void **forEachLayer** (String sClass)
- void **forEachLayer** ()
- void **forEachLayerNet** ()
- void **forEachNet** (int iNet)
- void **forEachObject** (String sClass)
- void **forEachObject** ()
- void **forEachPEInputJob** ()
- void **forEachPEPanelJob** ()
- void **forEachPESolution** ()
- void **forEachRegion** ()
- void **forEachSignal** (String sSubClass)
- void **forEachSignal** ()
- void **forEachVtxt** ()
- int **GDSII_outLayer** (String filename)
- int **GDSII_outLayer** (String filename, String options)
- boolean **generateContours** (double dGap, double dOverlap)
- boolean **generateContoursOnLayer** (double dGap, double dOverlap)
- ObjectList **getAttrCategories** ()
- ObjectList **getAttrNames** (String sCategory)
- ObjectList **getAttrValues** (String sAttributeName)
- int **getCount** (String sType)
- ObjectList **getFaultTypes** ()
- String **getFileLastModified** (ObjectList fileInfo)
- String **getFileName** (ObjectList fileInfo)
- String **getFileParent** (ObjectList fileInfo)
- long **getFileSize** (ObjectList fileInfo)
- Ulayer **getLayer** (ObjectList layerID)
- ObjectList **getLayerNames** ()
- ObjectList **getLayers** ()
- **Rectangle getLocationOnScreen** (String sFrameName)
- ObjectList **getMode** ()
- ObjectList **getNetAttrNames** ()
- ObjectList **getNetNames** ()
- int **getNetNumberByClick** (double pt_x, double pt_y)
- int **getNetNumberByClick** (**Point** pt)
- Ulayer **getNextLayer** ()
- ObjectList **getODBxxSteps** (String sPath)
- int **getPlotParam** (String sKey, int iDefValue)
- double **getPlotParam** (String sKey, double dDefValue)
- String **getPlotParam** (String sKey, String sDefValue)
- void **grabWidget** ()
- void **gridAlign** (double dStep)
- void **gridCross** (boolean bCross)
- boolean **gridCross** ()
- void **gridOrigin** (double ptOrigin_x, double ptOrigin_y)
- void **gridOrigin** (**Point** ptOrigin)
- **Point gridOrigin** ()
- void **gridOutline** (double refPoint_x, double refPoint_y, double offset_x, double offset_y, int repeatX, int repeatY)
- void **gridOutline** (**Point** refPoint, **Point** offset, int repeatX, int repeatY)

void **gridStep** (double dStepX, double dStepY)
- **Point gridStep** ()
- double **gridStepX** ()
- double **gridStepY** ()
- void **gridVisible** (boolean bVisible)
- boolean **gridVisible** ()
- void **groupApeBy** (String spec)
- void **groupApertureDefinitions** ()
- void **groupApertureNumbers** ()
- void **groupAperturesByPolarity** ()
- double **groupUFD** (double dDistance)
- void **groupUFD** ()
- void **helpOnContext** ()
- void **hideAll** ()
- void **hideBlockStructure** ()
- void **HitachiSpotDiameterCompensation** (double dOffset, double dArcExpandMarginOverrideMicrons, int iMode, int iFastMode, boolean bAmSkipFlag, boolean bChangePolarity)
- void **HitachiSpotDiameterCompensation** (double dOffset, double dArcExpandMarginOverrideMicrons, int iMode, boolean bFastMode, boolean bAmSkipFlag, boolean bChangePolarity)
- String **i8Jobarticleid** ()
- String **i8JobBoardid** ()
- String **i8JobCustomer** ()
- boolean **i8Jobdelete** ()
- int **i8JobDuration** ()
- Date **i8JobFinishtime** ()
- int **i8JobFullduration** ()
- int **i8JobId** ()
- String **i8JobLocation** ()
- int **i8JobPriority** ()
- String **i8JobProgress** ()
- int **i8JobQueueposition** ()
- Date **i8JobStarttime** ()
- Date **i8JobSubmittime** ()
- void **importEpc** (String sPath)
- int **importExternal** (String sExtName, String sWheName, String sLanguage, boolean bKeepExtension, String sLayClass, String sLayAtt, String sStatus, String sWheLang, boolean bAnalyzed, String sWheFile, int iLocale)
- int **importExternal** (String sExtName, String sWheName, String sLanguage, boolean bKeepExtension)
- int **importExternal** (String sExtName)
- void **importFile** (String sScriptPath)
- void **importGwk** (String sPath)
- String **importHeptaCSV** (String sCSVFile, String sDXFFile, String sOptions)
- void **importHouei** (String sPath)
- void **importIpc** (String sPath, String sVersion)
- void **importIPC2581** (String sPath, String sStep, String sLayer)
- void **importODBxx** (String sPath, String sStep, String sLayer, ObjectList oReplaceCodeMap)
- void **importODBxx** (String sPath, String sStep, String sLayer)
- void **importODBxx** (String sPath, String sStep, ObjectList oReplaceCodeMap)
- void **importODBxx** (String sPath, String sStep)
- void **importPolarBuildup** (String sPolarFilePath)
- void **importScript** (String sScript)
- void **importWf** (String sPath)
- void **innerCopperCount** (boolean bUseMask, boolean bConfirmMaskUsage)
- void **innerCopperCount** ()
- void **insertAperture** (boolean bBefore, String sSrcLayer, ObjectList srcApeIndex)
- int **insertArc** (double arc_fx, double arc_fy, double arc_tx, double arc_ty, double arc_cx, double arc_cy, String arc_sense, int iNet, String sSelection)
- int **insertArc** (**Arc** arc, int iNet, String sSelection)
- void **insertArc3Point** (double pnt1_x, double pnt1_y, double pnt2_x, double pnt2_y, double pnt3_x, double pnt3_y)
- void **insertArc3Point** (**Point** pnt1, **Point** pnt2, **Point** pnt3)

void **insertArc3Point** (double pnt1_x, double pnt1_y, double pnt2_x, double pnt2_y, double pnt3_x, double pnt3_y, int iNet, String sSelection)
- void **insertArc3Point** (**Point** pnt1, **Point** pnt2, **Point** pnt3, int iNet, String sSelection)
- void **insertArcCenterStart** (double pntCenter_x, double pntCenter_y, double pntFrom_x, double pntFrom_y, double pntTo_x, double pntTo_y, String sDirection)
- void **insertArcCenterStart** (**Point** pntCenter, **Point** pntFrom, **Point** pntTo, String sDirection)
- void **insertArcCenterStart** (double pntCenter_x, double pntCenter_y, double pntFrom_x, double pntFrom_y, double pntTo_x, double pntTo_y, String sDirection, int iNet, String sSelection)
- void **insertArcCenterStart** (**Point** pntCenter, **Point** pntFrom, **Point** pntTo, String sDirection, int iNet, String sSelection)
- boolean **insertArcConcentric** (boolean bSelection, ObjectList oArcs)
- void **insertBreak** (**Point** line_fp, **Point** line_tp)
- void **insertBreak** (double line_fx, double line_fy, double line_tx, double line_ty)
- void **insertBreak** (**Line** line)
- void **insertContourText** (double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax, String sText, String sFontName, int iFontStyle, String sMirror, boolean bReverse, boolean bAllowDistortion, String sSelection)
- void **insertContourText** (**Rectangle** rect, String sText, String sFontName, int iFontStyle, String sMirror, boolean bReverse, boolean bAllowDistortion, String sSelection)
- void **insertCopper** (int number, String attach, String material, double thickness, String reference, double tolerance, String supplier)
- void **insertCore** (int layNum, boolean matTop, boolean matBot, String material, String topMaterial, String botMaterial, double thickness, double topThickness, double botThickness, String reference, double tolerance, double erConstant, String supplier, ObjectList attrNames, Object[] attrValues, boolean revInsert)
- void **insertCore** (int layNum, boolean matTop, boolean matBot, String material, String topMaterial, String botMaterial, double thickness, double topThickness, double botThickness, String reference, double tolerance, double erConstant, String supplier, ObjectList attrNames, Object[] attrValues)
- void **insertCore** (int layTop, int layBot, boolean matTop, boolean matBot, String material, String topMaterial, String botMaterial, double thickness, double topThickness, double botThickness, String reference, double tolerance, double erConstant, String supplier, ObjectList attrNames, Object[] attrValues, boolean revInsert)
- void **insertCore** (int layTop, int layBot, boolean matTop, boolean matBot, String material, String topMaterial, String botMaterial, double thickness, double topThickness, double botThickness, String reference, double tolerance, double erConstant, String supplier, ObjectList attrNames, Object[] attrValues)
- int **insertDraw** (double line_fx, double line_fy, double line_tx, double line_ty, int iNet, String sSelection)
- int **insertDraw** (**Line** line, int iNet, String sSelection)
- int **insertDraw** (double line_fx, double line_fy, double line_tx, double line_ty)
- int **insertDraw** (**Line** line)
- int **insertFlash** (double pt_x, double pt_y, int iNet, String sSelection)
- int **insertFlash** (**Point** pt, int iNet, String sSelection)
- int **insertFlash** (double pt_x, double pt_y)
- int **insertFlash** (**Point** pt)
- void **insertFlashRepeat** (double pt_x, double pt_y, int iNx, double dXstep, int iNy, double dYstep, String sSelection)
- void **insertFlashRepeat** (**Point** pt, int iNx, double dXstep, int iNy, double dYstep, String sSelection)
- void **insertFlashRepeat** (double pt_x, double pt_y, int iNx, double dXstep, int iNy, double dYstep)
- void **insertFlashRepeat** (**Point** pt, int iNx, double dXstep, int iNy, double dYstep)
- void **insertFullArc3Point** (double pnt1_x, double pnt1_y, double pnt2_x, double pnt2_y, double pnt3_x, double pnt3_y)
- void **insertFullArc3Point** (**Point** pnt1, **Point** pnt2, **Point** pnt3)
- void **insertFullArc3Point** (double pnt1_x, double pnt1_y, double pnt2_x, double pnt2_y, double pnt3_x, double pnt3_y, int iNet, String sSelection)
- void **insertFullArc3Point** (**Point** pnt1, **Point** pnt2, **Point** pnt3, int iNet, String sSelection)
- void **insertFullArcCenterRadius** (double pntCenter_x, double pntCenter_y, double dRadius, String sDirection)
- void **insertFullArcCenterRadius** (**Point** pntCenter, double dRadius, String sDirection)
- void **insertFullArcCenterRadius** (double pntCenter_x, double pntCenter_y, double dRadius, String sDirection, int iNet, String sSelection)
- void **insertFullArcCenterRadius** (**Point** pntCenter, double dRadius, String sDirection, int iNet, String sSelection)
- boolean **insertParallel** (boolean bSelection, ObjectList oLines)

void **insertPolydrawRect** (double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax, boolean bSel)
- void **insertPolydrawRect** (**Rectangle** rect, boolean bSel)
- void **insertPolydrawRect** (double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax)
- void **insertPolydrawRect** (**Rectangle** rect)
- void **insertPolydrawRect** (double drawRectangle_xmin, double drawRectangle_ymin, double drawRectangle_xmax, double drawRectangle_ymax, boolean bRectCW, boolean bSel)
- void **insertPolydrawRect** (**Rectangle** drawRectangle, boolean bRectCW, boolean bSel)
- void **insertPolydrawRect** (**Point** p1, **Point** p2, boolean bRectCW, boolean bSel)
- void **insertPolygon** (boolean bSelection, ObjectList polygon)
- void **insertPrePreg** (int topLayer, int bottomLayer, String sPosition, String material, double thickness, String reference, double tolerance, double erConstant, String supplier, ObjectList attrNames, Object[] attrValues)
- void **insertTab** (double p_x, double p_y, double dis, String pat)
- void **insertTab** (**Point** p, double dis, String pat)
- void **insertVectorText** (double pt_x, double pt_y, String sText, String sFont, double dWidth, double dSpacing, String sMirror, double dRotation, double dScale)
- void **insertVectorText** (**Point** pt, String sText, String sFont, double dWidth, double dSpacing, String sMirror, double dRotation, double dScale)
- void **insertVectorText** (double pt_x, double pt_y, String sText, String sFont, double dWidth, double dSpacing, String sMirror, double dRotation, double dScaleX, double dScaleY)
- void **insertVectorText** (**Point** pt, String sText, String sFont, double dWidth, double dSpacing, String sMirror, double dRotation, double dScaleX, double dScaleY)
- void **intersectDraws** (double pt_x, double pt_y)
- void **intersectDraws** (**Point** pt)
- boolean **isDirectory** (ObjectList fileInfo)
- boolean **isEqual** (Object oParam1, Object oParam2)
- boolean **isFile** (ObjectList fileInfo)
- boolean **isHidden** (ObjectList fileInfo)
- boolean **isLayerInPlane** (int iPlaneNumber)
- void **job_save_shm_and_release** (String sShmName)
- int **jobApeMaxNumber** ()
- String **jobATEMachine** ()
- void **jobAttribute** (String name, String value)
- String **jobAttribute** (String name)
- String **jobAttribute** ()
- void **jobCopperCount** (boolean bUseMask, boolean bConfirmMaskUsage)
- void **jobCopperCount** ()
- void **jobCustomer** (String sCustomer)
- String **jobCustomer** ()
- void **jobDRCParameters** (String sDrc)
- String **jobDRCParameters** ()
- **Rectangle jobEnclosingBox** ()
- void **jobExtension** (String sExtension)
- void **jobFixture** (String sFixture)
- String **jobFixture** ()
- boolean **jobHasPattern** (boolean bUsed)
- void **jobInfo** (String[] sInfo)
- void **jobInfo** (String sInfo)
- String **jobInfo** ()
- void **jobLayMask** (String sLayMask)
- String **jobLayMask** ()
- int **jobMaxNetnumer** ()
- void **jobName** (String sName)
- String **jobName** ()
- boolean **jobNetlist** ()
- int **jobNumApes** ()
- int **jobNumBothExtras** ()
- int **jobNumBothExtras** (String subClass)
- int **jobNumBottomExtras** ()
- int **jobNumBottomExtras** (String subClass)

int **jobNumCores** ()
- int **jobNumDrills** ()
- int **jobNumDrills** (String subClass)
- int **jobNumExtras** ()
- int **jobNumExtras** (String subClass)
- int **jobNumLayers** ()
- int **jobNumNoneExtras** ()
- int **jobNumNoneExtras** (String subClass)
- int **jobNumPrepregs** (int start)
- int **jobNumPrepregs** ()
- int **jobNumSignals** ()
- int **jobNumSignals** (String subClass)
- int **jobNumTopExtras** ()
- int **jobNumTopExtras** (String subClass)
- void **jobPath** (String sPath)
- String **jobPath** ()
- void **jobRevision** (String sRevision)
- String **jobRevision** ()
- int **jobSelectCount** (String sOption)
- int **jobSelectCount** ()
- boolean **jobSelection** ()
- **Rectangle jobSelectionEnclosingBox** ()
- void **jobSize** (double pntSize_x, double pntSize_y)
- void **jobSize** (**Point** pntSize)
- void **jobSize** (String sUnit, double pntSize_x, double pntSize_y)
- void **jobSize** (String sUnit, **Point** pntSize)
- void **jobSize** (String sSize)
- **Point jobSize** ()
- void **jobSpec** (String sSpec)
- String **jobSpec** ()
- void **jobUserData** (String sUserData)
- String **jobUserData** ()
- void **lajCleanLegendLayer** (boolean DoMask, double MaskClearance, boolean DoCu, double CuClearance, boolean DoCuPads, double CuPadClearance, boolean bDoCuFOM, double iCuFOMClearance, boolean bDoCuPadsFOM, double iCuPadsFOMClearance, boolean doPlatedDrills, double platedDrillClearance, boolean doUnplatedDrills, double UnplatedDrillClearance, boolean DoSmallDraws, double MinDrawSize)
- void **lajDefineWord** ()
- void **lajDeselectAllWords** ()
- void **lajDragWord** (double pt_x, double pt_y, double radius, double offset_x, double offset_y, double limit, boolean enforcelimit)
- void **lajDragWord** (**Point** pt, double radius, **Point** offset, double limit, boolean enforcelimit)
- void **lajLegendDRC** (boolean bDoLineWidth, double dMinLineWidth, boolean bDoMask, double dMaskClearance, boolean bDoCu, double dCuClearance, boolean bDoCuPads, double dCuPadClearance, boolean bDoCuFOM, double dCuFOMClearance, boolean bDoCuPadsFOM, double dCuPadsFOMClearance, boolean bDoPlatedDrills, double dPlatedDrillClearance, boolean bDoUnplatedDrills, double dUnplatedDrillClearance, boolean bDoSmallDraws, double dMinDrawSize)
- void **lajLegendTextToWords** (double dMaxSize, int iMaxSpacing)
- void **lajMoveWord** (String value, double dx, double dy, double limit)
- void **lajMoveWord** (String value, double dx, double dy)
- void **lajScaleWord** (String value, double factor, double limit)
- void **lajScaleWord** (String value, double factor)
- void **lajScaleWordOnPt** (double pt_x, double pt_y, double radius, double scale, double limit, boolean enforcelimit)
- void **lajScaleWordOnPt** (**Point** pt, double radius, double scale, double limit, boolean enforcelimit)
- void **lajSelectAllWords** ()
- void **lajUndefineWord** ()
- void **layActive** (ObjectList layerID, boolean bActive)
- boolean **layActive** (ObjectList layerID)
- void **layActive** (boolean bActive)
- boolean **layActive** ()

void **layAlias** (String sAlias)
- String **layAlias** ()
- int **layApeCount** ()
- void **layAttach** (String sAttach)
- String **layAttach** ()
- void **layAttribute** (String name, String value)
- String **layAttribute** (String name)
- String **layAttribute** ()
- void **layClass** (String sNewClass)
- String **layClass** ()
- void **layCopperCount** (boolean bUseMask, boolean bConfirmMaskUsage)
- void **layCopperCount** ()
- **Rectangle layEnclosingBox** ()
- void **layerViewSplit** (boolean on)
- int **layExtractPlotStamps** (String dstLayName, String sOptions, ObjectList sFilters)
- void **layFrom** (int layFrom)
- int **layFrom** ()
- boolean **layHasPattern** (boolean bUsed)
- ObjectList **layID** ()
- void **layIndex** (int iIndex)
- int **layIndex** ()
- void **layInfo** (String sText)
- String **layInfo** ()
- void **layMaterial** (String sMaterial)
- String **layMaterial** ()
- void **layName** (String sName)
- String **layName** ()
- void **layNumber** (int iNumber)
- int **layNumber** ()
- void **layReadable** (String sSide)
- String **layReadable** ()
- void **layReverse** (boolean bReverse)
- boolean **layReverse** ()
- boolean **laySelection** ()
- **Rectangle laySelectionEnclosingBox** ()
- void **laySubClass** (String sSubClass)
- String **laySubClass** ()
- void **layThickness** (double dThickness)
- double **layThickness** ()
- void **layTo** (int layTo)
- int **layTo** ()
- double **layZPos** ()
- void **liftUpUpcbBlocks** ()
- **Line Line** (double ptFromX, double ptFromY, double ptToX, double ptToY, String units)
- **Line Line** (double ptFromX, double ptFromY, double ptToX, double ptToY)
- **Line Line** (**Line** line)
- **Line Line** (**Point** ptFrom, **Point** ptTo)
- ObjectList **listFrames** ()
- void **loadApertures** (String sDpfFile)
- void **loadBuildup** (String buildupSpec)
- void **loadFrames** (boolean bVerbose, boolean bLoadOnce)
- void **loadFrames** ()
- void **loadSplitConfig** (String sConfigName)
- void **loadUFD** (String sUFDName)
- void **loadWorkspace** (String sWorkspaceName)
- void **loadWorkspace** ()
- double **maxInvalidArcsDeviation** ()
- int **measureFingers** (String szOption)
- void **measureLayers** ()
- void **measureObjects** (double p1_x, double p1_y, double p2_x, double p2_y)
- void **measureObjects** (**Point** p1, **Point** p2)

void **measurePoints** (double pt_x, double pt_y)
- void **measurePoints** (**Point** pt)
- void **measurePoints** (double p1_x, double p1_y, double p2_x, double p2_y)
- void **measurePoints** (**Point** p1, **Point** p2)
- void **mergeContours** ()
- void **mergeContoursSingle** ()
- void **mergeContoursSingleAdd** ()
- void **mergeLayers** (String posNegAlt, boolean delLay)
- void **mirror** (String axis, boolean bUseCenter, boolean bOnRefPoints)
- void **models** (String sModelShape, double dTolerance)
- void **models** (String sModelShape)
- boolean **modelsCreateComplex** ()
- boolean **modelsCreateStandard** (double dTolerance)
- **Rectangle modelsDefineSelections** ()
- int **modelsReplace** (double pntTolerance_x, double pntTolerance_y)
- int **modelsReplace** (**Point** pntTolerance)
- int **modelsSelect** (double pntTolerance_x, double pntTolerance_y)
- int **modelsSelect** (**Point** pntTolerance)
- void **modifyCore** (int iTopLay, String sAtt, int iNewTopLay, int iNewBotLay, String sNewAtt, double dThickness, String sMaterial, String sInfo)
- void **modifyDrill** (String sName, String sAlias, String sClass, String sSubClass, int iFrom, int iTo, double dThickness)
- void **modifyExtra** (String sName, String sAlias, String sClass, String sSubClass, String sAttach, int iNumber, boolean bReverse, String sMaterial)
- void **modifyFeedback** (String sName, String sAlias, String sClass, String sSubClass, String sAttach)
- void **modifyLayer** (String sName, String sAlias, String sClass, String sSubClass, int iNumber, boolean bReverse, double dZPosition, String sReadable, String sMaterial, double dThickness)
- void **modifyPrePreg** (int iTopLay, int iIndex, int iNewTopLay, int iNewBotLay, int iNewIndex, double dThickness, String sMaterial, String sInfo)
- void **move** (double pt_x, double pt_y, boolean bOnRefPoints)
- void **move** (**Point** pt, boolean bOnRefPoints)
- void **netlistBuild** (String target)
- void **netlistClear** ()
- void **netlistReference** (String target)
- void **newJob** (String jobPath, String jobName)
- void **notImplemented** (String sFuncName)
- void **objAttribute** (String name, String value)
- String **objAttribute** (String name)
- String **objAttribute** ()
- **Point objCenterPoint** ()
- double **objClearance** ()
- **Rectangle objEnclosingBox** ()
- **Point objFlash** ()
- **Point objFromPoint** ()
- String **objInfo** ()
- int **objNet** ()
- **Point objPoint** ()
- double **objRing** ()
- void **objSelect** (String sel)
- boolean **objSelect** ()
- String **objSense** ()
- String **objShape** ()
- void **objString** (String vtxString)
- String **objString** ()
- **Point objToPoint** ()
- String **objType** ()
- void **offset** (double offset_x, double offset_y)
- void **offset** (**Point** offset)
- **Point offset** ()
- void **offsetX** (double offsetX)
- void **offsetY** (double offsetY)

void **openAboutUcamco** ()
- void **openAboutUcamX** ()
- void **openAdvantools** ()
- void **openAMLIJobManager** ()
- void **openAnamorphicScale** ()
- void **openApeCreator** ()
- void **openApeEditor** ()
- void **openApertureAttributes** ()
- void **openApertureManager** ()
- void **openAttributeEditor** ()
- void **openAttributeManager** ()
- void **openAutoDrill** ()
- void **openAutoDrillEditor** ()
- void **openAutoFixture** ()
- void **openBarcode** ()
- void **openBarcode128** ()
- void **openBoardAnalyzer** ()
- void **openBoardSnapshot** ()
- void **openCalculatorSetup** ()
- void **openCamtek** (String sMachineCfg)
- void **openCFMEEOutput** ()
- void **openCheckList** ()
- void **openCheckListDefineChecklist** ()
- void **openCheckListDefineSteps** ()
- void **openClipping** ()
- void **openColor** ()
- void **openConnect** ()
- void **openContourHandling** ()
- void **openConvertAttributes** ()
- void **openCopperBalance** ()
- void **openCopperRepair** ()
- void **openCoverlayOptimizer** ()
- void **openCU9000Dialog** ()
- void **openDatums** ()
- void **openDistort** ()
- void **openDraw** (double pt_x, double pt_y, double dis)
- void **openDraw** (**Point** pt, double dis)
- void **openDrawSlots** ()
- void **openDRC** ()
- void **openDrillInfo** ()
- void **openDrillMap** ()
- void **openDrillOptimizer** ()
- void **openDrillRoutSetups** ()
- void **openDrillTolerance** ()
- void **openDrillToolManager** ()
- void **openDsAoi** ()
- void **openDsAoiAdvanced** ()
- void **openDSAOIialog** ()
- void **openDsAoiPreview** ()
- void **openDsAoiQueue** ()
- void **openEditingToolbox** ()
- void **openEditVectorText** (double pickPoint_x, double pickPoint_y)
- void **openEditVectorText** (**Point** pickPoint)
- void **openErrors** ()
- void **openEtchCompensation** ()
- void **openExpand** ()
- void **openExternalLinkManager** ()
- void **openFiducials** ()
- void **openFillAngledPattern** ()
- void **openFillPattern** ()
- void **openFillVector** ()

void **openFlashMaker** ()
- void **openFlexManager** ()
- void **openFlipJob** ()
- void **openFrame** (String sFrameName)
- void **openGridParameters** ()
- void **openHelp** ()
- void **openHelpOnHelp** ()
- void **openHelpOnHypertool** ()
- void **openHelpOnResources** ()
- void **openHelpOnVersion** ()
- void **openHiPot** ()
- void **openImageCompare** ()
- void **openImpedanceControl** ()
- void **openImportIPC356** ()
- void **openImportMET** ()
- void **openImportODBxx** ()
- void **openImportWF** ()
- void **openInsertContourText** ()
- void **openInsertVectorText** ()
- void **openJob** (String jobName)
- void **openJob_shm** (String sShmName)
- void **openJobCreate** ()
- void **openJobDefinition** ()
- void **openJobEdit** ()
- void **openJobEditor** ()
- void **openJobEditorOptions** ()
- void **openJobLoad** ()
- void **openJobMerge** ()
- void **openJobPlaneSetup** ()
- void **openJobPrint** ()
- void **openJobView** ()
- void **openLayerEdit** ()
- void **openLegendOptimizer** ()
- void **openLicenseHelp** ()
- void **openLoadCheckList** ()
- void **openMagnifier** ()
- void **openMarkupAssistant** ()
- void **openMessages** ()
- void **openMLIOutput** ()
- void **openModels** ()
- void **openNetCompare** ()
- void **openNetfixSetup** ()
- void **openNonFunctionalPad** ()
- void **openNumbers** ()
- void **openObjectAttributes** ()
- void **openObjectCompare** ()
- void **openOutputAccumatch** ()
- void **openOutputAOI** ()
- void **openOutputCAD** ()
- void **openOutputCamtek** ()
- void **openOutputDrillRout** (String sDrillMachine)
- void **openOutputDsDi** ()
- void **openOutputDsDiPreview** ()
- void **openOutputDsDiQueue** ()
- void **openOutputNetlist** ()
- void **openOutputOrbot** ()
- void **openOutputSapphire** ()
- void **openOutputScoring** ()
- void **openOutputSmartArgos** ()
- void **openOutputTrackscan** ()
- void **openOutputUxpAutomanager** ()

void **openOutputUxpEtec** ()
- void **openOutputUxpLocal** ()
- void **openPanelFramesCoupons** ()
- void **openPanelLinks** ()
- void **openPanelPlus** ()
- void **openPanelReproduce** ()
- void **openPanelSetup** ()
- void **openPanelStepRepeat** ()
- void **openPlaneAdjuster** ()
- void **openPlotParameters** ()
- void **openPPMonitor** ()
- void **openProductionStagesEditor** ()
- void **openQueryNet** ()
- void **openQueryObject** ()
- void **openReferencePoints** ()
- void **openRegister** ()
- void **openRemoveAttributes** ()
- void **openRepair** (String szLabname)
- void **openRoutManager** ()
- void **openRoutManagerCleanUp** ()
- void **openRoutManagerDimensioning** ()
- void **openRoutManagerEditor** ()
- void **openRoutManagerTools** ()
- void **openSaveJobAs** ()
- void **openSaveLayout** ()
- void **openSecureEtchCompensation** ()
- void **openSelections** ()
- void **openSetupOptions** ()
- void **openSetupSave** ()
- void **openShavePads** ()
- void **openSignalLayerAdjuster** ()
- void **openSignalLayerAdjusterAssistant** ()
- void **openSilkOptimizer** ()
- void **openSmartCamtek** (String sMachineCfg)
- void **openSmartDRC** ()
- void **openSmartFix** ()
- void **openSmartplot** ()
- void **openSmartSR** ()
- void **openSmartStart** ()
- void **openSoldermask** ()
- void **openSoldermaskOptimizer** ()
- void **openTearDrop** ()
- void **openTechnicalAnalyzer** ()
- void **openTestpointEdit** ()
- void **openToolbarManager** ()
- void **openToolbars** ()
- void **openTransformObjects** ()
- void **openTransformObjectsBGAPads** ()
- void **openTransformObjectsBGATracks** ()
- void **openTransformObjectsEdit** ()
- void **openTransformObjectsRescale** ()
- void **openUcamDbEditor** ()
- void **openUndoRedoDetails** ()
- void **openUTest** ()
- void **openUtestUtilities** ()
- void **openValidateLayer** ()
- void **openVectorTextFont** ()
- void **openVerifyArcsDraws** ()
- void **openViewGuide** ()
- void **openYsphotechOutput** ()
- void **optimizeDrill** (int nPasses, int optMode, double yxTime)

void **optimizeMaskLayer** (double dMinRing, double dMaxRing, double dMaskToCopper, double dMaskToMask, double dBigRing)
- String **osChDir** ()
- String **osChDir** (String sDir)
- int **osCopy** (String sSrcName, String sDstName)
- String **osCreateTmpDir** (String sBasePath)
- String **osCreateTmpDir** ()
- int **osDelete** (String sFileName)
- ObjectList **osFileInfo** (String sPath)
- ObjectList **osGetFileList** (String sDir, String sFileMask, boolean bRecurse, boolean bFullPath, boolean bWithDirs)
- ObjectList **osGetFileList** (String sDir, String sFileMask, boolean bRecurse, boolean bFullPath)
- ObjectList **osGetFileList** (String sDir, String sFileMask, boolean bRecurse)
- ObjectList **osGetFileList** (String sDir, String sFileMask)
- ObjectList **osGetFileList** (String sDir, boolean bRecurse)
- ObjectList **osGetFileList** (String sDir, boolean bRecurse, boolean bFullPath, boolean bWithDirs)
- ObjectList **osGetFileList** (String sDir)
- int **osMarkAsTmp** (String sName)
- int **osMkDir** (String sDirName)
- int **osMove** (String sSrcName, String sDstName)
- void **osRmDir** (String sDirName)
- void **osRmTree** (String sDirName)
- int **osUnTgz** (String sTgzArchive, String sDstDir)
- int **osUnZip** (String sZipArchive, String sDstDir)
- void **outAtgFixture** (String key, String sTool, String iRes, int iSession)
- void **output274x** (String sRes)
- void **outputAft** (String res)
- void **outputAoi** (boolean bCadData, boolean bReference)
- void **outputAtf** ()
- ObjectList **outputAutoDrill** (String sDrjFile)
- void **outputCFMEE** (String outputPath, boolean reverse, double marginx, double marginy, boolean distort, double distortx, double distorty, double resizex, double resizey, boolean deleteOutside)
- void **outputCli** ()
- void **outputColorPDF** (String sPdfFullPath)
- void **outputDp40** (double pt_x, double pt_y, boolean bPositive, boolean bMirrorx, boolean bMirrory, double dLaserPower, int iPolygonSpeed, int iPcbFormat, String unit)
- void **outputDp40** (**Point** pt, boolean bPositive, boolean bMirrorx, boolean bMirrory, double dLaserPower, int iPolygonSpeed, int iPcbFormat, String unit)
- void **outputDxf** (String unit, int iConturize, int iKeepTXT, double dExpandArcs, int iCenterLine, int iAllInOne)
- void **outputDxfV6** (String unit)
- void **outputEie** (String sRes, ObjectList par)
- void **outputEtec** (String sResistOuter, String sResistInner, double mediaX, double mediaY, int iAlignType, int iLevelType, int iCycles, String sDate, String sTime, boolean bAuto, double dScaleX, double dScaleY, double dScaleOriX, double dScaleOriY, String sMDFfile, String sResource)
- void **outputExt** (String lan, String too, String res, ObjectList resdb, String inc1, String inc2, int session, Object[] pre, Object[] pos, Object notUsed)
- void **outputExt** (String lan, String too, String res, ObjectList resdb, String inc1, String inc2, int session, Object[] pre, Object[] pos)
- void **outputHimt** (double datum_x, double datum_y, double offset_x, double offset_y, String mirror, String rotation)
- void **outputHimt** (**Point** datum, **Point** offset, String mirror, String rotation)
- void **outputIpc2581** ()
- void **outputIpcUfd** (String key, String res, String version)
- void **outputLpg** (int iPpi, int iChoke, double offset_x, double offset_y)
- void **outputLpg** (int iPpi, int iChoke, **Point** offset)
- int **outputManiaSapphire** (String sOutputPath, String sDescription, String sGeometryfile, boolean bStatistics, boolean bDrill)
- void **outputMda** (String sPath, ObjectList par, Object[] subpar, int iApr, int iSubfig, int iRenum)
- void **outputNec** (String too)
- void **outputOdbxx** (String res)

void **outputOdbxxv7** (String res)
- void **outputOif** (String oifVersion, int byJob, int pan, int fillin)
- boolean **outputOrbot** ()
- void **outputPdf** ()
- void **outputProbe** (String sLang, int iSession, int iAccuracy)
- int **outputRaid** ()
- void **outputRpd** (int iPpi, double datum_x, double datum_y, double offset_x, double offset_y, String sMirror, String sRotation)
- void **outputRpd** (int iPpi, **Point** datum, **Point** offset, String sMirror, String sRotation)
- boolean **outputSchmid** (int resolution, double maskRectangle_xmin, double maskRectangle_ymin, double maskRectangle_xmax, double maskRectangle_ymax, int maskRotation, String maskMirror, String maskPolarity, int equipmentRotation, String equipmentMirror, double offsetX, double offsetY, ObjectList fiducials, String imagePath, String configPath, String batchFile)
- boolean **outputSchmid** (int resolution, **Rectangle** maskRectangle, int maskRotation, String maskMirror, String maskPolarity, int equipmentRotation, String equipmentMirror, double offsetX, double offsetY, ObjectList fiducials, String imagePath, String configPath, String batchFile)
- void **outputSl13** (String sResources, String sKey)
- void **outputSprint** (String sOutputFolder, boolean bStandardMark, String sCopperName, String sRefName, String sAttributeZero, int iZeroPointNumber, String sAttributeCamera, int iCameraNumber, int iRotation0, int iRotation90, int iRotation180, int iRotation270, String sText1, String sText2, String sText3, String sCleanOption)
- void **outputSys** ()
- void **outputTiff** (String sPath, String sExt, String sOptions, int iResolution)
- int **outputTs3** (boolean bDrilledBoards, double dSpace)
- void **outputWf2** (ObjectList par)
- void **outputXdpf** (String sPath)
- void **outputYsphotech** (boolean imagecomp, String outputPath, String reverse, double marginx, double marginy, boolean mirrorx, boolean mirrory, double rotate, double distortx, double distorty, double resize, boolean keepArrays, boolean deleteOutside, boolean autoDetected, String layername)
- void **outputYsphotech** (String outputPath, String reverse, double marginx, double marginy, boolean mirrorx, boolean mirrory, double rotate, double distortx, double distorty, double resize, boolean keepArrays, boolean deleteOutside, boolean autoDetected, String layername)
- void **pajPlaneAdjust** (double dPlatedClearance, double dUnplatedClearance, double dRingSize, double dRingClearance, double dLineWidth, double dCuClearance, boolean bDoCut, double dOutlineCLearance)
- void **pajPlaneAdjust** (double dPlatedClearance, double dUnplatedClearance, double dRingSize, double dRingClearance, double dLineWidth, double dCuClearance, boolean bDoCut, double dOutlineClearance, boolean bOutputAsContour, boolean bSaveBackup, boolean bErrorPopups)
- void **pajPlaneAdjust** (double dPlatedClearance, double dUnplatedClearance, double dRingSize, double dRingClearance, double dLineWidth, double dCuClearance, boolean bDoCut, double dOutlineClearance, boolean bOutputAsContour, boolean bSaveBackup)
- void **panelStepRepeat** (double pStart_x, double pStart_y, int iRepeatX, int iRepeatY, double dStepX, double dStepY, String sFlashPoint)
- void **panelStepRepeat** (**Point** pStart, int iRepeatX, int iRepeatY, double dStepX, double dStepY, String sFlashPoint)
- void **panelStepRepeatCenter** (double pStart_x, double pStart_y, int iRepeatX, int iRepeatY, double dStepX, double dStepY)
- void **panelStepRepeatCenter** (**Point** pStart, int iRepeatX, int iRepeatY, double dStepX, double dStepY)
- void **panelStepRepeatJobZero** (double pStart_x, double pStart_y, int iRepeatX, int iRepeatY, double dStepX, double dStepY)
- void **panelStepRepeatJobZero** (**Point** pStart, int iRepeatX, int iRepeatY, double dStepX, double dStepY)
- void **panelStepRepeatMiddle** (double pStart_x, double pStart_y, int iRepeatX, int iRepeatY, double dStepX, double dStepY)
- void **panelStepRepeatMiddle** (**Point** pStart, int iRepeatX, int iRepeatY, double dStepX, double dStepY)
- int **panelStepRepeatValidate** ()
- void **pasteFromClipboard** ()
- String **peGetInputFile** ()
- boolean **peGetInputJobBooleanProperty** (String name)
- double **peGetInputJobDoubleProperty** (String name)
- int **peGetInputJobIntegerProperty** (String name)
- String **peGetInputJobProperty** (String name)
- String **peGetJobList** ()

int **peGetOptionalQuantity** ()
- boolean **peGetPanelJobBooleanProperty** (String name)
- double **peGetPanelJobDoubleProperty** (String name)
- int **peGetPanelJobIntegerProperty** (String name)
- String **peGetPanelJobProperty** (String name)
- int **peGetPCBQuantity** ()
- boolean **peGetSingleOptimization** ()
- boolean **peGetSolutionBooleanProperty** (String name)
- double **peGetSolutionDoubleProperty** (String name)
- int **peGetSolutionIntegerProperty** (String name)
- String **peGetSolutionProperty** (String name)
- boolean **peGetUseFrameSet** ()
- void **peSetInputFile** (String fileName)
- void **peSetInputJobProperty** (String name, boolean value)
- void **peSetInputJobProperty** (String name, double value)
- void **peSetInputJobProperty** (String name, int value)
- void **peSetInputJobProperty** (String name, String value)
- void **peSetOptionalQuantity** (int quantity)
- void **peSetPanelJobProperty** (String name, boolean value)
- void **peSetPanelJobProperty** (String name, double value)
- void **peSetPanelJobProperty** (String name, int value)
- void **peSetPanelJobProperty** (String name, String value)
- void **peSetSingleOptimization** (boolean set)
- void **peSetSolutionProperty** (String name, boolean value)
- void **peSetSolutionProperty** (String name, double value)
- void **peSetSolutionProperty** (String name, int value)
- void **peSetSolutionProperty** (String name, String value)
- void **peSetUseFrameSet** (boolean set)
- void **pickAperture** (double pt_x, double pt_y, double radius)
- void **pickAperture** (**Point** pt, double radius)
- **Point pickPoint** (String sLabel)
- void **plotAddLayerToMerge** (String sJobName, String sPath)
- void **plotAddLayerToMerge** ()
- boolean **plotLayer** (String sPath, int iFillPercentage, boolean bSeparator, boolean bClean)
- boolean **plotLayer** (int iFillPercentage, boolean bSeparator, boolean bClean)
- boolean **plotMergedLayers** (int iFillPercentage, boolean bSeparator, boolean bClean)
- void **plotResetParams** ()
- void **plotSetAttribute** (ObjectList oLayID, String sName, String sValue)
- void **plotSetAttribute** (String sName, String sValue)
- void **plotSetAttribute** (String sName)
- void **plotSetParam** (ObjectList oLayID, String sKey, String sName, double dValue)
- void **plotSetParam** (String sKey, String sName, double dValue)
- void **plotSetParam** (ObjectList oLayID, String sKey, boolean bValue)
- void **plotSetParam** (String sKey, boolean bValue)
- void **plotSetParam** (ObjectList oLayID, String sKey, int iValue)
- void **plotSetParam** (String sKey, int iValue)
- void **plotSetParam** (ObjectList oLayID, String sKey, double dValue)
- void **plotSetParam** (String sKey, double dValue)
- void **plotSetParam** (ObjectList oLayID, String sKey, String sValue)
- void **plotSetParam** (String sKey, String sValue)
- void **plotSetRipHost** (String sRIP)
- void **plotStartNew** ()
- **Point Point** (**Point** point)
- **Point Point** (double x, double y, String units)
- **Point Point** (double x, double y)
- void **point1** (double point1_x, double point1_y)
- void **point1** (**Point** point1)
- **Point point1** ()
- void **point1Active** (boolean bActivate)
- boolean **point1Active** ()
- void **point1X** (double pt1X)

void **point1Y** (double pt1Y)
- void **point2** (double point2_x, double point2_y)
- void **point2** (**Point** point2)
- **Point point2** ()
- void **point2Active** (boolean bActivate)
- boolean **point2Active** ()
- void **point2X** (double pt2X)
- void **point2Y** (double pt2Y)
- void **printListRefPoints** (boolean bOnAllActiveLay)
- void **printListRefPoints** ()
- boolean **promptBoolean** (String optName, boolean def)
- double **promptDouble** (String doubleName, double def)
- void **promptEnd** ()
- String **promptFileName** (String strLabel, String def)
- int **promptInteger** (String intName, int def)
- void **promptLabel** (String labelText)
- **Line promptLine** (String lineName, double fromX, double fromY, double toX, double toY)
- **Line promptLine** (String lineName, **Line** defLine)
- String **promptOption** (String optName, ObjectList options, String def)
- **Point promptPoint** (String pointName, double ptX, double ptY)
- **Point promptPoint** (String pointName, **Point** point)
- **Rectangle promptRectangle** (String rectangleName, double xmin, double xmax, double ymin, double ymax)
- **Rectangle promptRectangle** (String rectangleName, **Rectangle** rectangle)
- void **promptStart** (String sSetName, String sTitle)
- void **promptStart** (String sSetName)
- void **promptStart** ()
- String **promptString** (String strName, String def)
- double **promptUnit** (String unitName, double def, String units)
- void **qmerge** (String sOptions)
- void **quitBlockEdit** (boolean bSave, boolean bKeepLink)
- void **quitBlockEdit** (boolean bSave)
- void **quitBlockMultiEdit** (boolean bSave, boolean bKeepLink)
- void **quitBlockMultiEdit** (boolean bSave)
- void **quitComplexEdit** (boolean bSave)
- void **quitConfirm** ()
- void **readAmli** (String sPath)
- void **recognizeContours** (String sConName)
- **Rectangle Rectangle** (**Rectangle** rect)
- **Rectangle Rectangle** (double xmin, double ymin, double xmax, double ymax, String units)
- **Rectangle Rectangle** (double xmin, double ymin, double xmax, double ymax)
- **Rectangle Rectangle** (**Point** ptPoint1, **Point** ptPoint2)
- void **redo** ()
- void **registerJobOnPoints** ()
- void **registerLayers** ()
- void **registerOnGrid** (double GridStep_x, double GridStep_y, double GridOri_x, double GridOri_y, double dXRadius, double dYRadius)
- void **registerOnGrid** (**Point** GridStep, **Point** GridOri, double dXRadius, double dYRadius)
- void **registerOnPads** (double dRadius, boolean bOnFlashPoint)
- void **removeApeAttr** ()
- void **removeJobAttr** ()
- void **removeLayAttr** ()
- void **removeNetAttr** (int iNetNumber, String sAttrName, String sAttrValue)
- void **removeNetAttr** (int iNetNumber, String sAttrName)
- void **removeNetAttr** (String sAttrName, String sNetName)
- void **removeNetAttr** (String sAttrName)
- void **removeObjAttr** ()
- void **removeObjectAttribute** (String sAttrName, String sAttrValue)
- void **removeObjectAttribute** (String sAttrName)
- void **removeYsphotechPlotstamp** (int plotstampID)
- void **replaceApeByCurrent** ()

void **replaceApertures** ()
- void **replaceBitmapByContours** ()
- void **replaceDrawsWithArcs** (double tolerance)
- void **replaceInnersByOuters** ()
- void **replaceZeroLengthDraws** (double dMaxLength, boolean bFunctional, boolean bNonFunctional)
- void **reproducePanel** (String report)
- void **reset** ()
- void **resetCFMEE** ()
- void **resetCores** (int iTop, String sAttach)
- void **resetCores** ()
- void **resetWorkspace** ()
- void **resetYsphotech** ()
- void **restoreArcs** (boolean bPreferFullArc)
- void **restoreContours** (boolean bPreferFullArc)
- void **returnVariables** (ObjectList returnVariables)
- void **reverse** ()
- void **reverseLayer** ()
- void **reverseLayers** ()
- void **rotate** (double angle, boolean bUseCenter, boolean bOnRefPoints)
- void **roundDraw** (double pt_x, double pt_y, double dis)
- void **roundDraw** (**Point** pt, double dis)
- void **routStatistics** (String doOption)
- void **routStatistics** ()
- void **runDRC** (String sCfgFile, boolean bBuildNetlist, String sUseNetlist, boolean bSelErrors)
- String **runFile** (String sScriptPath, ObjectList argv)
- String **runFile** (String sScriptPath)
- String **runScript** (String sScript, ObjectList argv)
- String **runScript** (String sScript)
- ObjectList **runScriptWithReturn** (String sScript, Object[] argv)
- ObjectList **runScriptWithReturn** (String sScript)
- void **saveAmli** (String sPath)
- void **saveBuildup** (String sSpec, String sCustomer, String sDrcPar, String sCoreRef, String sPrePregMat, String sCopperMat, String sJobFlow, String sTechCheck, String sAttrSet, String sDatumList, ObjectList layList)
- int **saveJob** ()
- int **saveJobAs** (String fullPath, String sVersion)
- int **saveJobAs** (String fullPath)
- void **saveJobAsV3** ()
- void **saveJobAsV6** ()
- void **saveJobAsV9** ()
- void **saveLayer** (String sClass, String sSubClass, int iLayIndex, String sFullPath)
- void **saveLayer** (String layName, String fullPath)
- void **saveMessagesAs** (String sFilePath)
- void **saveOrder** ()
- void **saveSplitConfig** (String sConfigName)
- void **saveUFD** (String sUFDName)
- void **saveWorkspace** ()
- void **saveWorkspace** (String sWorkspaceName)
- void **saveWorkspaceAs** (String sWorkspaceName)
- void **scale** (double dScaleValue, boolean bUseCenter)
- void **scaleObjectOnAttribute** (String sName, double dScaleFactor, double dMinClearance)
- void **screendump** ()
- void **secureEtchCompensation** (double dPadSpread, double dSmdSpread, double dTrackSpread, double dAreaSpread, double dPadPadClearance, double dPadSmdClearance, double dPadTrackClearance, double dPadAreaClearance, double dSmdSmdClearance, double dSmdTrackClearance, double dSmdAreaClearance, double dTrackTrackClearance, double dTrackAreaClearance, double dAreaAreaClearance, String sContourMethod, boolean bProcessSameNetSpacing, boolean bBackupOriginalLayer, boolean bCheckMissingCopper, boolean bFastMode, int iShiftMode, double dMinCopper)
- void **secureEtchCompensation** (double dPadSpread, double dSmdSpread, double dTrackSpread, double dAreaSpread, double dPadPadClearance, double dPadSmdClearance, double dPadTrackClearance,

double dPadAreaClearance, double dSmdSmdClearance, double dSmdTrackClearance, double dSmdAreaClearance, double dTrackTrackClearance, double dTrackAreaClearance, double dAreaAreaClearance, String sContourMethod, boolean bProcessSameNetSpacing, boolean bBackupOriginalLayer, boolean bCheckMissingCopper, boolean bFastMode, int iShiftMode)
- void **secureEtchCompensation** (double dPadSpread, double dSmdSpread, double dTrackSpread, double dAreaSpread, double dPadPadClearance, double dPadSmdClearance, double dPadTrackClearance, double dPadAreaClearance, double dSmdSmdClearance, double dSmdTrackClearance, double dSmdAreaClearance, double dTrackTrackClearance, double dTrackAreaClearance, double dAreaAreaClearance, String sContourMethod, boolean bProcessSameNetSpacing, boolean bBackupOriginalLayer, boolean bCheckMissingCopper, boolean bFastMode)
- void **secureEtchCompensationUndo** ()
- void **selectAll** ()
- void **selectAll** (String selectMode)
- void **selectAllApertures** ()
- void **selectAllContours** (String selectMode, String conMode)
- void **selectAllContours** (String selectMode, double xSize, double ySize, String conMode)
- void **selectAmbiguousContours** (String selectMode)
- void **selectAperture** (ObjectList apeIndexArray)
- void **selectAperture** ()
- void **selectAperturesBiggerThan** (String selectMode, double dx, double dy)
- void **selectAperturesSmallerThan** (String selectMode, double dx, double dy)
- void **selectByApeAttributeNames** (String selectMode, String[] sName)
- void **selectByApertureShape** (String selectMode, String apertureShapes)
- void **selectByAttributeName** (String selectMode, String sName)
- void **selectByAttributeValue** (String selectMode, String sName, String sValue)
- void **selectByObjectType** (String selectMode, String objectTypes)
- void **selectChained** (String selectMode, double pt_x, double pt_y)
- void **selectChained** (String selectMode, **Point** pt)
- void **selectChained** (String selectMode, double pt_x, double pt_y, double dTolerance)
- void **selectChained** (String selectMode, **Point** pt, double dTolerance)
- void **selectChainedObjects** (String selectMode, double pnt_x, double pnt_y, double pixelRadius, double dOffCenter, boolean bSameApe, boolean bSameOrientation)
- void **selectChainedObjects** (String selectMode, **Point** pnt, double pixelRadius, double dOffCenter, boolean bSameApe, boolean bSameOrientation)
- void **selectContourByClick** (String selectMode, double pt_x, double pt_y, String conMode)
- void **selectContourByClick** (String selectMode, **Point** pt, String conMode)
- void **selectContoursInWindow** (String selectMode, double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax, String conMode)
- void **selectContoursInWindow** (String selectMode, **Rectangle** rect, String conMode)
- void **selectContoursWithThinRegion** (double dThin, String selectMode)
- void **selectCurrentAperture** (String selectMode)
- void **selectCurrentApertureDefinition** (String selectMode)
- void **selectCurrentObject** (String selectMode)
- void **selectDoubles** (String selectMode, double tolerance)
- void **selectEmbedded** (String selectMode, double tolerance)
- void **selectFlashesLongerThan** (String selectMode, double rRefRatio)
- int **selectHornablePads** (String selectMode, String apertureShapes)
- boolean **selectInvalidArcs** ()
- int **selectInvalidArcs** (String sSelectMode, double dDeviation, String sLimit)
- void **selectIsolatedFlashes** (String selectMode)
- void **selectMesh** (String selectMode)
- int **selectNetByClick** (String selectMode, double pt_x, double pt_y)
- int **selectNetByClick** (String selectMode, **Point** pt)
- int **selectNetByName** (String selectMode, String sNetName)
- void **selectNetByNumber** (String selectMode, int net, boolean bSelectShaved, boolean bSelectBroken)
- void **selectNetByNumber** (String selectMode, int net)
- void **selectNetByTestpoints** (String selectMode, int nbt)
- void **selectNetsWithoutPads** (String selectMode)
- String **selectNonFunctionalPads** ()
- void **selectObjectAttribute** (String sAttrName, String sAttrValue)
- void **selectObjectAttribute** (String sAttrName)

void **selectObjectByAttribute** (String sAttrName, String sAttrValue)
- void **selectObjectByAttribute** (String sAttrName)
- void **selectObjectByAttributeName** (String selectMode, String sName)
- void **selectObjectByAttributeValue** (String selectMode, String sName, String sValue)
- void **selectObjectByShape** (String selectMode, String apertureShapes)
- void **selectObjectByType** (String selectMode, String objectTypes)
- void **selectOpenContours** (String selectMode)
- void **selectOverlappingContours** (String selectMode)
- void **selectOverlaps** (String selectMode)
- void **selectPainted** (String selectMode)
- int **selectPaintedAreas** (boolean bUseLoops, boolean bExcludeChains)
- int **selectPaintedAreas** ()
- void **selectPlotStamps** (String selectMode)
- void **selectPolygon** (boolean bInside, boolean bOutside, boolean bCrossing, String sShapes, String sObjects, String selectMode, ObjectList polygonPoints)
- void **selectReferenceLayer** (String selectMode)
- void **selectReverse** (String selectMode)
- void **selectSmallContours** (String selectMode, double xSize, double ySize, String conMode)
- void **selectSmallSurface** (String selectMode, double surface, String conMode)
- void **selectSmallTracks** (String selectMode, String lenMode, String dstMode, double maxLength)
- void **selectTouchingObjects** (String selectMode, double pnt_x, double pnt_y, double pixelRadius)
- void **selectTouchingObjects** (String selectMode, **Point** pnt, double pixelRadius)
- void **selectWindow** (String selectMode, double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax, String winopt, String sShapes, String sObjects)
- void **selectWindow** (String selectMode, **Rectangle** rect, String winopt, String sShapes, String sObjects)
- void **selectZeroLengthDraws** (double dMaxLength, boolean bFunctional, boolean bNonFunctional)
- boolean **setApe** (int index)
- void **setApertureAttribute** (String sAttributeName, String sAttributeValue)
- void **setAttributeOnObject** (String attrName, String attrValue)
- boolean **setCurrentAperture** (int iIndex)
- void **setInPlane** (int newPlane, int iIndex)
- void **setInPlane** (int newPlane, String layName)
- void **setInPlane** (int newPlane, String layClass, String laySubclass, String attach, int index)
- void **setInPlane** (int newPlane, String layClass, String laySubclass, String attach, int index, boolean activate)
- void **setInPlane** (int newPlane, ObjectList layerID, boolean activate)
- void **setInPlaneByName** (int newPlane, String layName, boolean activate)
- void **setLayerViewBottom** (ObjectList nameArray)
- void **setLayerViewDrill** (ObjectList nameArray)
- void **setLayerViewMain** (ObjectList nameArray)
- void **setLayerViewTop** (ObjectList nameArray)
- void **setMode** (ObjectList oMode)
- void **setMode** (String sParams)
- void **setMode** (String sOption, String sUnit, String sSnapMode)
- Ulayer **setNextLayerToPlane1** ()
- void **setOrigin** (double p_x, double p_y)
- void **setOrigin** (**Point** p)
- void **setOrigin** (double pt_x, double pt_y, boolean bOnRefPoints)
- void **setOrigin** (**Point** pt, boolean bOnRefPoints)
- void **setOriginCenter** (double p_x, double p_y, boolean useOutline)
- void **setOriginCenter** (**Point** p, boolean useOutline)
- void **setOriginToCenter** (boolean bUseOutline, boolean bOnRefPoints)
- void **setPlotParam** (String sKey, int iValue)
- void **setPlotParam** (String sKey, double dValue)
- void **setPlotParam** (String sKey, String sValue)
- void **setResolution** (int resolution)
- void **setSnap** (String sMode)
- void **setSnapOnContour** (boolean on)
- void **setUnit** (String unit)
- void **setYsphotechAlignmentPointType** (int region, int point, String type)
- void **setYsphotechPlotstamp** (int plotstampID, double rec_xmin, double rec_ymin, double rec_xmax,

double rec_ymax, String type)
- void **setYsphotechPlotstamp** (int plotstampID, **Rectangle** rec, String type)
- void **shavePads** (double dPadTraClr, double dPadPadClr, int iClip, boolean bShaveInsideCom)
- void **shavePads** (double dPadTraClr, double dPadPadClr, int iClip)
- void **shavePadsOnMaskLayer** (double dPadToTrack, double dPadToPad)
- void **showBlockStructure** ()
- void **showMeasureValues** (double p1_x, double p1_y, double dx, double dy, double clr, double rng)
- void **showMeasureValues** (**Point** p1, double dx, double dy, double clr, double rng)
- void **showNetlistProfile** ()
- void **silkOptimize** (int iReference, double dClearanceToReference, boolean bCompensateBumps, double dBumpClearance, int iMethod, double dMinimumDrawLength)
- int **smoothen** (String mode, double dMaxDeviation)
- int **smoothen** (String mode, double dMaxDeviation, int iMinReplacePoints)
- void **spawn_func** (String sCommand)
- int **splitContour** (double dOverlapX, double dOverlapY, double dMinX, double dMinY)
- void **splitContours** ()
- boolean **stackupByGerAttr** ()
- void **standardizeBoxes** ()
- void **testVDPathfinder** (int iStart, int iEnd)
- void **testVDPathfinder2** (double dStartX, double dStartY, double dEndX, double dEndY)
- void **toggleApertureSelections** ()
- void **toggleSelections** ()
- void **toggleViewInBlocks** ()
- void **toggleViewMode** ()
- void **toggleViewObjects** ()
- void **toggleViewRefPoints** ()
- void **toggleViewZero** ()
- void **trimDraws** (double p1_x, double p1_y, double p2_x, double p2_y)
- void **trimDraws** (**Point** p1, **Point** p2)
- void **undo** ()
- void **undoClear** ()
- void **unload** (String sClass, String sSubClass, int iLayIndex, boolean bSave)
- void **updateProbes** ()
- void **updateTestPoints** ()
- void **updateTestPointsAndProbes** ()
- void **updateZPosition** ()
- void **utestCheck3DProbeClearance** (boolean bCheck3DProbeClearance, double dClearance)
- void **utestCreateEtmComponentLayers** (boolean bCreateCompLay)
- void **utestDedicatedFixtures** (boolean bDedicatedFixtures, boolean bFirstProbeNumber0, double dFontScale, int iShowProbeNumberEvery, boolean bShowConnectorNumberAlways, boolean bContinousNumbering, boolean bContinousNumberingOnBottom)
- void **utestDo** ()
- void **utestFiducals** (boolean bFiducals)
- void **utestFixtureSizeSplit** (boolean bFixtureSizeSplit, int iSplitSet)
- void **utestGuidePlates** (boolean bGuidePlates)
- void **utestKelvin4WireTest** (boolean bKelvin4WireTest, boolean bUsedMidPoints, boolean bTestOnBlindHoles, boolean bTestOnlyThroughHoles, boolean bTestAllPads, double dMinDrillDia, double dMaxDrillDia, int iSearchDepthLimit)
- void **utestKelvin4WireTest** (boolean bKelvin4WireTest, boolean bUsedMidPoints, boolean bTestOnBlindHoles, boolean bTestOnlyThroughHoles, double dMinDrillDia, double dMaxDrillDia, int iSearchDepthLimit)
- void **utestMachine** (int iSession, String sMachName, String sAccesstype)
- void **utestMicroAdjustment** (boolean bMicroAdjustment, int iNbrOfTestPoints, double dTestPointDiameter, double dTestPointShiftEdge, double dTestPointShiftValue, double dTestPointPitch, double dClearanceFactor, double dCenterDiameter)
- void **utestNetlist** (boolean bNetlist, boolean bNetlistBuild, boolean bNetlistExpand)
- void **utestOutput** (boolean bOutput, boolean bDrillFixture, String sDrillFixture, boolean bNetlist, String sNetlist, boolean bElectTest, boolean bPinInserter, boolean bRepairAid, String sRepairAid)
- void **utestProbeAssignment** (boolean bProbeAssignment, boolean bStagger, String sStagpnt, boolean bStagopttrian, boolean bStagoptlined, double dPitch, double dTolerance, double dSetBack, boolean bReverse, boolean bAxis)

void **utestProbeMapping** (boolean bProbeMapping)
- void **utestTestpoints** (boolean bTestPoints, int iLoop, boolean bUseMasks, boolean bProbeSwaping, boolean bHandlePaintedPads, boolean bCircuitryCheck, boolean bFilterCopperAreas, boolean bViaOfSMDs, boolean bDrillsWithoutPad)
- void **utestTestpointsBOT** (boolean bPointsBot1, boolean bPointsBot2, boolean bPointsBot3, boolean bPointsBot4, boolean bPointsBot5, boolean bPointsBot6, boolean bPointsBot7)
- void **utestTestpointsTOP** (boolean bPointsTop1, boolean bPointsTop2, boolean bPointsTop3, boolean bPointsTop4, boolean bPointsTop5, boolean bPointsTop6, boolean bPointsTop7)
- void **validateInvalidArcs** ()
- void **viewAmbiguous** ()
- void **viewGrid** (boolean bVisible, double ptOrigin_x, double ptOrigin_y, double dXStep, double dYStep, boolean bCross)
- void **viewGrid** (boolean bVisible, **Point** ptOrigin, double dXStep, double dYStep, boolean bCross)
- void **viewGrid** (boolean bVisible, **Point** ptOrigin, **Point** ptStep, boolean bCross)
- void **viewGrid** (boolean bVisible)
- void **viewGrid** ()
- void **viewGuide** ()
- void **viewHistory** ()
- void **viewInBlocks** (boolean trueFalse)
- void **viewMessages** ()
- void **viewMode** (String sMode)
- void **viewModeFilled** ()
- void **viewModeOutline** ()
- void **viewModeSkeleton** ()
- void **viewNumbers** ()
- void **viewObjects** (boolean trueFalse)
- void **viewPan** (double p1_x, double p1_y, double p2_x, double p2_y)
- void **viewPan** (**Point** p1, **Point** p2)
- void **viewRefPoints** (boolean trueFalse)
- void **viewRepaint** ()
- void **viewWarning** (String message)
- void **viewZero** (boolean trueFalse)
- void **viewZoom** (String sZoom)
- void **viewZoomIn** ()
- void **viewZoomOut** ()
- void **viewZoomSelections** ()
- void **viewZoomTotal** ()
- void **viewZoomWindow** ()
- void **viewZoomWindow** (double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax, boolean bScreenCenter)
- void **viewZoomWindow** (**Rectangle** rect, boolean bScreenCenter)
- void **viewZoomWindow** (double rect_xmin, double rect_ymin, double rect_xmax, double rect_ymax)
- void **viewZoomWindow** (**Rectangle** rect)
- void **xmlAdd** (String sDataName, String sElementName, String sContent, ObjectList attrArray)
- void **xmlAdd** (String sDataName, String sElementName, String sContent)
- void **xmlAdd** (String sElementName, String sContent)
- void **xmlAddData** (String sParentDataName, String sDataName)
- void **xmlAddData** (String sDataName)
- void **xmlCreateData** (String sDataName)
- void **xmlDocument** (String rootName)
- void **xmlSave** (String sDestFilePath)
- void **YachiyoAOI_clearOutput** (String name)
- void **YachiyoAOI_defineArea** (int grpIndex, int areaIndex, double pos_x, double pos_y)
- void **YachiyoAOI_defineArea** (int grpIndex, int areaIndex, **Point** pos)
- void **YachiyoAOI_defineGroup** (int index, double area_xmin, double area_ymin, double area_xmax, double area_ymax, String types)
- void **YachiyoAOI_defineGroup** (int index, **Rectangle** area, String types)
- void **YachiyoAOI_defineMask** (int grpIndex, int maskIndex, double area_xmin, double area_ymin, double area_xmax, double area_ymax, String type)
- void **YachiyoAOI_defineMask** (int grpIndex, int maskIndex, **Rectangle** area, String type)
- void **YachiyoAOI_finish** ()

boolean **YachiyoAOI_generateCalibration** (String name, double pos_x, double pos_y, double startRes, double endRes, double step, String options)
- boolean **YachiyoAOI_generateCalibration** (String name, **Point** pos, double startRes, double endRes, double step, String options)
- boolean **YachiyoAOI_generateOutput** (String name, String lens, String fixture, String options)
- String **YachiyoAOI_getStrings** (String kind)
- boolean **YachiyoAOI_init** (String iniFile)
- void **YachiyoAOI_reset** ()
- void **YachiyoAOI_setRefPoint** (int index, double pos_x, double pos_y)
- void **YachiyoAOI_setRefPoint** (int index, **Point** pos)

変数

- int **FILE_ATTRIBUTES** = 2
- int **FILE_MODIFICATION_DATE** = 4
- int **FILE_NAME** = 5
- int **FILE_PARENT** = 1
- int **FILE_SIZE** = 3
- int **FILE_TYPE** = 0
- int **LAYER_ACTIVITY** = 5
- int **LAYER_APERTURES** = 6
- int **LAYER_ATTACH** = 3
- int **LAYER_CLASS** = 1
- int **LAYER_INDEX** = 4
- int **LAYER_NAME** = 0
- int **LAYER_SUBCLASS** = 2

関数

void abort (String *sInfo*)

スクリプトの実行を中断する

引数:

sInfo 中断理由についての情報テキスト

例外:

AbortException

void activate (String *layClass*,
String *laySubclass*,
int *layNum*,
boolean *layAct*
)

Activate/Deactivate layer by class, subclass and index

引数:

layClass Class of layer(s) to be activated (layer, drill, extra, feedback or all)

laySubclass Subclass of layer(s) to be activated/deactivated

layNum Index within the specified copper layer class or subclass e.g. to activate the bottom copper layer of a 6 layer job:

```
activate("layer",null , 6, true);
```


```
activate("layer",null , 6, true);
```


The value 6 refers to the 6th layer or

```
activate("layer", "outer", 2, true);
```


```
activate("layer", "outer", 2, true);
```


The value 2 refers to the second layer of subclass "outer". Index within the extra layer class All "extra" layers attached to "top" are numbered from top to bottom, for "extra" layers attached to "bottom" the numbering continues from bottom to top. To activate the extra layer with index 4:

```
activate("extra", null, 4, true);
```


```
activate("extra", null, 4, true);
```


Index within the specified drill class or subclass Drill layers are numbered from left to right To activate the first plated drill layer of a job with 1 unplated and 2 plated layers

```
activate("drill",null , 2, true);
```


```
activate("drill",null , 2, true);
```


The value 2 refers to the second drill layer or

```
activate("drill", "plated", 1, true);
```


```
activate("drill", "plated", 1, true);
```


The value 1 refers to the first plated drill layer **Note:** The `extra.order` ucam.db key can change the order of the extra layers. Scripts relying on the index of extra layers could fail to execute correctly on other systems.

layAct Set true to activate and false to deactivate layers

void activate (ObjectList *layerID,*
boolean *layAct*
)

レイヤIDによりレイヤをアクティブ/非アクティブにする

引数:

layerID レイヤを表すレイヤID
layAct レイヤをアクティブにするには true を 非アクティブにするには false を設定

参照:

getLayerID(Ulayer)

getLayerID(Ulayer, String)

void activate (String *layClass,*
String *laySubclass,*
boolean *layAct*
)

サブクラスによりレイヤをアクティブ/非アクティブにする

引数:

layClass アクティブ化するレイヤクラス(layer, drill, extra, feedback, allのいずれか)
laySubclass アクティブ・非アクティブにするレイヤのサブクラス
layAct レイヤのアクティブ化にはtrueを 非アクティブ化にはfalseを設定

void activateAllLayers ( )

activateAllLayers

void activateBottomLayers (boolean *layAct*)

Activate/deactivate the bottom layer in all level-1 subjobs.

引数:

layAct Set true to activate and false to deactivate layers

void activateToggle ( )

このファンクションでは 全レイヤのトグル有効化は記録されない

void activateTopLayers (boolean *layAct*)

Activate/deactivate the top layer in all level-1 subjobs.

引数:

layAct Set true to activate and false to deactivate layers

void activityClean (String *sMode*)

Cleans attributes uActivityZoom or uActivityPlane or uActivityActive or all according to given sMode.

引数:

sMode string "plane", "zoom", "activity" or "*" for all previous values.

void activityClean ( )

Cleans all attributes uActivityZoom, uActivityPlane and uActivityActive

void activityRestore (String *sMode*)

Restores viewport stored in job attribute uActivityZoom or layer planes of all layers stored in layer attribute named uActivityPlane or layer activity stored in layer attribute named uActivityActive or all according to given sMode.

引数:

sMode string "plane", "zoom", "activity" or "*" for all previous values.

void activityRestore ( )

Restores viewport stored in job attribute uActivityZoom, planes of all layers stored in layer attribute named uActivityPlane and layer activity stored in layer attribute named uActivityActive

void activityStore (String *sMode*)

Stores viewport in job attribute uActivityZoom or layer planes of all layers in layer attribute named uActivityPlane or layer activity in layer attribute named uActivityActive or all according to given sMode.

引数:

sMode string "plane", "zoom", "activity" or "*" for all previous values.

void activityStore ( )

Stores current viewport in job attribute uActivityZoom, planes of all layers in layer attribute named uActivityPlane and layer activity in layer attribute named uActivityActive

void addBreak (Point *line_fp*,
Point *line_tp*
)

与えられたライン上のドローにブレイクを追加する

引数:

line_fp (始点) 与えられたライン
line_tp (終点) 与えられたライン

void addBreak (double *line_fx*,
double *line_fy*,
double *line_tx*,
double *line_ty*
)

与えられたライン上のドローにブレイクを追加する

引数:

line_fx (始点X座標) 与えられたライン
line_fy (始点Y座標) 与えられたライン
line_tx (終点X座標) 与えられたライン
line_ty (終点Y座標) 与えられたライン

void addBreak (Line *line*)

与えられたライン上のドローにブレイクを追加する

引数:

line 与えられたライン

int addCFMEEAlignmentPoint (double *point_x*,

```
                                    double  point_y
                                )
```

Add alignment point to CFMEE

引数:

point_x (X coordinate) the point of the alignment point
point_y (Y coordinate) the point of the alignment point

```
int addCFMEEAlignmentPoint ( Point  point  )
```

Add alignment point to CFMEE

引数:

point the point of the alignment point

```
void addDPF ( String  dpf,
              String  layName,
              String  subClass,
              int     layPos,
              String  readable
            )
```

DPFレイヤを追加する

引数:

dpf dpfファイルのパス
layName レイヤ名
subClass レイヤサブクラス
layPos レイヤ位置
readable レイヤ面視

```
void addDPFDrill ( String  dpf,
                   int     layPos,
                   int     drillTop,
                   int     drillBot
                 )
```

DPFドリルを追加する

引数:

dpf dpfファイルのパス
layPos 追加レイヤ位置 1からドリルレイヤ総数までのいずれかの値
drillTop トップドリルレイヤのインデックス
drillBot ボトムドリルレイヤのインデックス

```
void addDPFExtra ( String  dpf,
                   String  attach,
```

```
                    boolean  bNoChecks
                )
```

Add DPF Extra

引数:

dpf	Path to a dpf file
attach	"top" or "bottom"
bNoChecks	if true, disable overlap check

```
void addDPFExtra ( String  dpf,
                   String  attach
                )
```

DPF Extra を追加する

引数:

dpf	dpf ファイルのパス
attach	“top”もしくは”bottom”

```
void addDPFLayer ( String  dpf,
                   int     layPos
                )
```

DPFレイヤを追加する

引数:

dpf	dpfファイルのパス
layPos	追加レイヤの位置 1からシグナルレイヤ総数までのいずれかの値

```
int addETMComponentHiPot ( int     etmId,
                           int     primId,
                           int     NetPrim,
                           int     NetSecond,
                           String  TestVolt,
                           String  Duration,
                           String  LeakCurrent,
                           String  VoltType,
                           String  StartVolt,
                           String  VoltRise
                         )
```

Adds a HiPot component to the HiPot etm component and etm connect layers - searches coordinates of largest test point

引数:

etmId	The unique etm id of the component
primId	The primary ID for a primary group
NetPrim	Primary net number
NetSecond	Secondary net number

TestVolt	Test parameter Test Voltage [V]
Duration	Test parameter Duration [s]
LeakCurrent	Test parameter Test Leakage Current [mA]
VoltType	Test parameter Voltage Type - AC or DC
StartVolt	Test parameter Start Voltage [V]
VoltRise	Test parameter Voltage Rise [s]

戻り値:
0: ok, 1: an error occurred

int addETMComponentHiPot (**int** ***etmId,***
int ***primId,***
double ***XStart,***
double ***YStart,***
String ***AccStart,***
double ***XEnd,***
double ***YEnd,***
String ***AccEnd,***
int ***NetPrim,***
int ***NetSecond,***
String ***TestVolt,***
String ***Duration,***
String ***LeakCurrent,***
String ***VoltType,***
String ***StartVolt,***
String ***VoltRise***
)

Adds a HiPot component to the HiPot etm component and etm connect layers

引数:

etmId	The unique etm id of the component
primId	The primary ID for a primary group
XStart	Start coordinate (primary net)
YStart	
AccStart	Connect access for start coordinate (primary net): top or bottom
XEnd	End coordinate (secondary net)
YEnd	
AccEnd	Connect access for end coordinate (secondary net): top or bottom
NetPrim	Primary net number
NetSecond	Secondary net number
TestVolt	Test parameter Test Voltage [V]
Duration	Test parameter Duration [s]
LeakCurrent	Test parameter Test Leakage Current [mA]
VoltType	Test parameter Voltage Type - AC or DC
StartVolt	Test parameter Start Voltage [V]
VoltRise	Test parameter Voltage Rise [s]

戻り値:
0: ok, 1: an error occurred

```
void addFault ( String   sType,
                double   oRectangle_xmin,
                double   oRectangle_ymin,
                double   oRectangle_xmax,
                double   oRectangle_ymax,
                String   sInfo
              )
```

Add **Rectangle** as an user fault to current UFD

引数:

sType	fault type name
oRectangle_xmin	(left boundary of rectangle) fault **Rectangle**
oRectangle_ymin	(bottom boundary of rectangle) fault **Rectangle**
oRectangle_xmax	(right boundary of rectangle) fault **Rectangle**
oRectangle_ymax	(top boundary of rectangle) fault **Rectangle**
sInfo	fault info

```
void addFault ( String      sType,
                Rectangle   oRectangle,
                String      sInfo
              )
```

Add **Rectangle** as an user fault to current UFD

引数:

sType	fault type name
oRectangle	fault **Rectangle**
sInfo	fault info

```
void addFault ( String   sType,
                Line     oLine,
                String   sInfo
              )
```

Add **Line** as an user fault to current UFD

引数:

sType	fault type name
oLine	fault **Line**
sInfo	fault info

```
void addFault ( String   sType,
                double   oPt_x,
                double   oPt_y,
                String   sInfo
              )
```

Add point as an user fault to current UFD

引数:

sType fault type name
oPt_x (X coordinate) **Point** of the fault
oPt_y (Y coordinate) **Point** of the fault
sInfo fault info

void addFault (String *sType,*
Point *oPt,*
String *sInfo*
)

Add point as an user fault to current UFD

引数:

sType fault type name
oPt **Point** of the fault
sInfo fault info

void addHyperScriptMenuItem (String *sScriptPath,*
String *sMenuItemLabel*
)

UcamのHyperScriptに メニューアイテムを追加する

引数:

sScriptPath スクリプトファイルの完全なパス
sMenuItemLabel メニューアイテムのラベル

void addMaskLayer (double *dThicken*)

ソルダマスク層の新規作成

引数:

dThicken - パッドの太らせ

void addObjectAttribute (String *sAttrName*)

addObjectAttribute Sets the object attribute with the given name. Takes objects from active layers from current job by Ucam options. (e.g. all selected objects)

引数:

sAttrName The object attribute name

void addObjectAttribute (String *sAttrName,*
String *sAttrValue*
)

addObjectAttribute Sets the object attribute with the given name and value. Takes objects from active layers from current job by Ucam options. (e.g. all selected objects)

引数:

sAttrName The object attribute name
sAttrValue The object attribute value

void addOptimizedMaskLayer (double *dMinRing,*
double *dMaxRing,*
double *dMaskToCopper,*
double *dMaskToMask,*
double *dBigRing*
)

新たなソルダマスク最適化レイヤを作成する

引数:

dMinRing - 最小リング
dMaxRing - 最大リング
dMaskToCopper - マスクから銅へ
dMaskToMask - マスクからマスクへ
dBigRing - 大型パッドリング

void addRefPoint (int *iIndex,*
double *pPnt_x,*
double *pPnt_y,*
boolean *bOnAllActiveLay*
)

Adds a reference point to layer.

引数:

iIndex The reference point number.
pPnt_x (X coordinate) The point coordinates.
pPnt_y (Y coordinate) The point coordinates.
bOnAllActiveLay if true it work on all active layers, otherwise only on active loaded layer in plane 1

void addRefPoint (int *iIndex,*
Point *pPnt,*
boolean *bOnAllActiveLay*
)

Adds a reference point to layer.

引数:

iIndex The reference point number.
pPnt The point coordinates.
bOnAllActiveLay if true it work on all active layers, otherwise only on active loaded layer in plane 1

```
void addShavedMaskLayer ( double  dThicken,
                          double  dPadToTrack,
                          double  dPadToPad
                        )
```

新たなソルダマスクレイヤの作成とソルダーマスクのシェービング

引数:

dThicken - パッドの太らせ
dPadToTrack - マスクからトラックへ
dPadToPad - マスクからパッドへ

```
void addTeardrops ( int     iMode,
                    double  dRelDiam,
                    double  dRelDist,
                    double  dAbsDiam,
                    double  dAbsDist,
                    double  dMinClr,
                    int     iOnRect,
                    int     iOnHoles
                  )
```

Makes teardrop shaped pads where a track enters a cirle/rectangle pad.

引数:

iMode	Defines how the teardrop is generated. 0 : using a circular flash. 1 : using draws. 2 : using arcs.
dRelDiam	The relative size of the circular aperture used to create the teardrop. This size is relative to the size of the connected pad in case of sub-lands, or relative to the diameter of the connecting draws/arcs.
dRelDist	The distance of the flash point of the subland relative to the size of the connected pad in sublandmode. This is the length of the bissection relative to the size of the connected pad in track or arc mode.
dAbsDiam	The size of the circular aperture used to create the teardrop.
dAbsDist	The distance of the flash point of the subland in sublandmode, the length on the bissection in track or arc mode.
dMinClr	The minimum clearance.
iOnRect	Generates teardrops on rectangles when 1.
iOnHoles	Generates teardrops only on pads with a drill hole present.

```
void addTeardrops ( int     iMode,
                    double  dRelDiam,
                    double  dRelDist,
                    double  dAbsDiam,
                    double  dAbsDist,
                    double  dMinClr,
                    int     iOnRect
                  )
```

Makes teardrop shaped pads where a track enters a cirle/rectangle pad.

引数:

iMode	Defines how the teardrop is generated. 0 : using a circular flash. 1 : using draws. 2 : using arcs.
dRelDiam	The relative size of the circular aperture used to create the teardrop. This size is relative to the size of the connected pad in case of sub-lands, or relative to the diameter of the connecting draws/arcs.
dRelDist	The distance of the flash point of the subland relative to the size of the connected pad in sublandmode. This is the lengthof the bi-section relative to the size of the connected pad in track or arc mode.
dAbsDiam	The size of the circular aperture used to create the teardrop.
dAbsDist	The distance of the flash point of the subland in sublandmode, the length on the bissection in track or arc mode.
dMinClr	The minimum clearance.
iOnRect	Generates teardrops on rectangles when 1.

int addYsphotechAlignmentPoint (int *region,*
double *point_x,*
double *point_y*
)

Add alignment point to Ysphotech

引数:

region	the region number (0 for global)
point_x	(X coordinate) the point of the alignment point
point_y	(Y coordinate) the point of the alignment point

int addYsphotechAlignmentPoint (int *region,*
Point *point*
)

Add alignment point to Ysphotech

引数:

region	the region number (0 for global)
point	the point of the alignment point

void align_blocks (double *dTolerance,*
boolean *bOnAllLayers*
)

Performs automatic block align on this job.

引数:

dTolerance	required percentage of the overlap
bOnAllLayers	true means ignore layer activity This method is licensed.

void AmliAddUser (String *sUser,*

String sPassword,
String sAuthorityLevel
)

Add user to the system

引数:

sUser	user name
sPassword	user's password
sAuthorityLevel	"admin", "operator" or "engineer"

void AmliChangeUserPwd (String *sUser*,
String *sPassword*,
String *sNewPassword*
)

Changes user's password.

引数:

sUser	user name
sPassword	user password
sNewPassword	user new password

String AmliCheckUser (String *sUser*,
String *sPassword*
)

Get user's authority level

引数:

sUser	user name
sPassword	user password

void AmliRemoveUser (String *sUser*)

Delete user from the system

引数:

sUser	user name to be removed

String analyzeExternal (String *sExtName*)

任意のファイルの外部フォーマットを解析する

引数:

sExtName	外部ファイルの完全なパス

戻り値:

言語

void angle (double *angle*)

角度値を設定する

引数:
angle 角度値

double angle ( )

角度値を取得する

戻り値:
角度値

void apeAnamorphicScale (double *dDistanceX,*
double *dDistanceY,*
boolean *bProportional*
)

Anamorphic Scale - Scale X and/or Y size of the aperture of a layer. Only the pad sizes are affected. Anamorphic Scale works on active layer in the plane 1 and on selected objects in the layer.

引数:
dDistanceX - a multiplication value, X distance value, absolute
dDistanceY - a multiplication value,Y distance value, absolute
bProportional - if true, the scale will be proportional

void apeAnamorphicScale (double *dScaleX,*
double *dScaleY*
)

Anamorphic Scale - Scale X and/or Y size of the aperture of a layer. Only the pad sizes are affected. Anamorphic Scale works on active layer in the plane 1 and on selected objects in the layer.

引数:
dScaleX - a multiplication value, X scale value [%]
dScaleY - a multiplication value,Y scale value [%]

void apeAttribute (String *name,*
String *value*
)

任意のアパーチャ属性に任意の値を設定 すでに属性が存在する場合は新しい値に変更され それ以外は新規作成される 値が`null`の場合 設定した属性名の属性は削除される

引数:
name アパーチャ属性名

value アパーチャ属性値

String apeAttribute (String *name*)

Returns the value of the aperture attribute with given name.

引数:

name the aperture attribute name

戻り値:

the value of the aperture attribute with given name or null if the attribute is not defined in the aperture.

String apeAttribute ( )

Returns comma separated list of the all aperture attributes.

Example: "uPCB=cc_d01,num=1"

戻り値:

comma separated list of the all aperture attributes.

void apeCorners (String *sCorners*)

カレントボックスアパーチャのコーナータイプを設定する

引数:

sCorners 可能なString値：“rounded”（丸いコーナー） “straight”（内部にコーナーをもつ） “cut”（コーナーとしての2つの内部で直交するライン） ”antique”（内部にコーナーをもつ）

String apeCorners ( )

カレントボックスアパーチャのコーナータイプを取得する

戻り値:

“rounded”（ラウンド） “straight”（ストレート） “cut”（カット） ”antique”（逆ラウンド）

void apeCreateBox (int *apeNum,*
double *xsize,*
double *ysize,*
String *corners,*
double *xcutoff,*
double *ycutoff*
)

Create Box aperture

引数:

apeNum Aperture Number

xsize Box size in X
ysize Box size in Y
corners Box corners types : "r" for rounded, "a" for antique, "c" for cut and "s" for straight
xcutoff Box corner cutoff value in X
ycutoff Box corner cutoff value in Y

void apeCreateCircle (int *apeNum,*
double *dia*
)

円形アパーチャの作成

引数:
apeNum アパーチャ番号
dia 直径

void apeCreateContour (int *apeNum,*
double *stroke*
)

プレーン1（Plane1）のレイヤに輪郭アパーチャを作成する

引数:
apeNum アパーチャ番号
stroke ストローク値

void apeCreateDonut (int *apeNum,*
double *outer,*
double *inner,*
String *kind*
)

ドーナッツ型アパーチャを作成する

引数:
apeNum アパーチャ番号
outer ドーナッツ外径
inner ドーナッツ内径
kind ドーナッツ型種類: rr=円/円, ss=四角/四角, sr=四角/円

void apeCreateOblong (int *apeNum,*
double *xsize,*
double *ysize*
)

楕円形アパーチャを作成する

引数:

apeNum アパーチャ番号
xsize 楕円形サイズX
ysize 楕円形サイズY

void apeCreateOctagon (int *apeNum,*
double *size*
)

8角形アパーチャを作成する

引数:
apeNum アパーチャ番号
size 8角形サイズ

void apeCreateRectangle (int *apeNum,*
double *xsize,*
double *ysize*
)

長方形アパーチャを作成する

引数:
apeNum アパーチャ番号
xsize 長方形サイズX
ysize 長方形サイズY

void apeCreateText (int *iApeNum,*
double *dHeight,*
String *sText,*
double *dRotation*
)

テキストアパーチャを作成

引数:
iApeNum アパーチャ番号
dHeight テキスト高さ
sText アパーチャに設定するテキスト
dRotation アパーチャ回転角度 正の値は反時計回り（左回り）です

void apeCreateText (int *iApeNum,*
double *dHeight,*
String *sText*
)

テキストアパーチャを作成

引数:

iApeNum アパーチャ番号
dHeight テキスト高さ
sText アパーチャに設定するテキスト

void apeCreateThermal (int *apeNum,*
double *outer,*
double *inner,*
double *gap,*
int *numGap,*
double *angle,*
String *kind*
)

サーマルアパーチャを作成する

引数:
apeNum アパーチャ番号
outer サーマル外径
inner サーマル 内径
gap サーマルギャップ値
numGap サーマルのギャップ数
angle サーマル角度
kind サーマル種類: "rr", "rs", "ss" or "sr"

Rectangle apeEnclosingBox ( )

このアパーチャに定義されたオブジェクトは 全ての長方形囲み領域を取得する

戻り値:
このアパーチャに定義されたオブジェクトは 全ての長方形囲み領域

int apeExtlinkCheck ( )

カレントアパーチャのextlink（外部リンク）をチェックし ステータスを返す

戻り値:
ステータス:

0 - OK
1 - 発生すべきでない
2 - dpfファイルが見つからない
3 - extlinkで指定されたdpfファイルが読み込めない
4 - アパーチャサイズがdpfファイルのブロックサイズに合わない
5 - アパーチャに外部リンクがない
6 - リンクには有効なタイムスタンプがない
7 - メモリー割り当てエラー
8 - パス名が展開／圧縮できない
9 - パスネームを展開／圧縮する際にバッファーあふれが発生
10 - Ucam内の定義が変わっていないがファイルは変更された
11 - ライセンスチェック失敗
12 - バッファーが小さすぎてパスネームを返せない
13 - 外部リンクが自分を指し示す
14 - リンクのタイムスタンプが異なっている
15 - リンクで使用可能なdpfが異なる（dat_equal）

String apeExtlinkPath ( )

アパーチャが外部リンク（extlink）に関連付けられているならば パス名が返される

戻り値:
外部リンクパス アパーチャに該当リンクがなければ“”

String apeExtlinkPathString ( )

アパーチャが外部リンク（extlink）に関連付けられているならば パス名が返される

戻り値:
外部リンクパス アパーチャに該当リンクがなければ“”

boolean apeExtlinkRelative ( )

外部リンクへのパスが相対パスである場合に情報を返す

戻り値:
外部リンクへのパスが相対パスであればtrue

int apeExtlinkStatus ( )

アパーチャのextlinkステータスを取得

戻り値:
ステータス:

0 - OK
1 - 発生すべきでない.
2 - dpfファイルが見つからない
3 - extlinkで指定されたdpfファイルが読み込めない
4 - アパーチャサイズがdpfファイルのブロックサイズに合わない
5 - アパーチャに外部リンクがない
6 - リンクには有効なタイムスタンプがない
7 - メモリー割り当てエラー
8 - パス名が展開／圧縮できない
9 - パスネームを展開／圧縮する際にバッファーあふれが発生
10 - Ucam内の定義が変わっていないがファイルは変更された
11 - ライセンスチェック失敗
12 - バッファーが小さすぎてパスネームを返せない
13 - 外部リンクが自分を指し示す
14 - リンクのタイムスタンプが異なっている
15 - リンクで使用可能なdpfが異なる（dat_equal）

void apeGap (double *dGap*)

カレントサーマルアパーチャのギャップを設定する

引数:
dGap double型 カレントサーマルアパーチャのギャップ

double apeGap ( )

カレントサーマルアパーチャのギャップを取得する

戻り値:
カレントサーマルアパーチャのギャップ

boolean apeHasPattern (boolean *bUsed*)

Returns `true` if the aperture has a pattern.

引数:

bUsed pattern is used when it affects the image.

戻り値:

`true` if the aperture has a pattern.

void apeHeight (double *dHeight*)

カレントテキストアパーチャの高さを設定する

引数:

dHeight double型 カレントテキストアパーチャの高さ

double apeHeight ( )

カレントテキストアパーチャの高さを取得する

戻り値:

カレントテキストアパーチャの高さ

int apeIndex ( )

カレントアパーチャのインデックスを取得する

戻り値:

レイヤパチャーリストにおけるカレントアパーチャのインデックス

String apeInfo ( )

Gets the information associated with the current aperture.

戻り値:

the information associated with the current aperture. `null` if the current aperture is not set.

void apeInner (double *dInner*)

ドーナッツ型もしくはサーマル型カレントアパーチャの内径を設定する

引数:

dInner ドーナッツ型もしくはサーマル型カレントアパーチャの新しい内径

double apeInner ( )

ドーナッツ型もしくはサーマル型カレントアパーチャの内径を取得する

戻り値:
ドーナッツ型もしくはサーマル型カレントアパーチャの内径

void apeKind (String *sKind*)

カレントサーマルアパーチャの種類を設定する

引数:
sKind String型 "rr": 円-円, "rs": 円-四角, "ss": 四角-四角, "sr": 四角-円

String apeKind ( )

Gets the kind of the current thermal or donut aperture.

戻り値:
for Thermal:

- "rr" - round-round
- "rs" - round-square
- "ss" - square-square
- "sr" - square-round. for Donut:
- "RR" : Circular outer and circular inner shape (round/round), the default.
- "SS" : Square outer and square inner shape (square/square).
- "SR" : Square outer and circular inner shape (square/round).

int apeMaxNetNumber ( )

カレントアパーチャの最大ネット番号を取得する

戻り値:
カレントアパーチャの最大ネット番号

void apeMirror (String *sMirror*)

カレントアパーチャのミラーオプションを設定する

引数:
sMirror String型 カレントアパーチャのミラーオプション 値は"" "X" "Y" "XY"のいずれか

String apeMirror ( )

カレントアパーチャのミラーオプションを取得する

戻り値:
カレントアパーチャのミラーオプション 値は"" "X" "Y" "XY"のいずれか

void apeName (String *sName*)

カレントアパーチャの名前を設定する

引数:

sName String型 カレントアパーチャの名前

String apeName ( )

カレントアパーチャ名を取得する

戻り値:

カレントアパーチャ名

void apeNumber (int *iNumber*)

カレントアパーチャ番号を設定する

引数:

iNumber int型 カレントアパーチャの新番号

int apeNumber ( )

カレントアパーチャ番号を取得する

戻り値:

カレントアパーチャ番号

void apeNumberGap (int *iNumGap*)

カレントサーマルアパーチャのギャップ数を設定する

引数:

iNumGap Integer型 カレントサーマルアパーチャのギャップ数

int apeNumberGap ( )

カレントサーマルアパーチャのギャップ数を取得する

戻り値:

カレントサーマルアパーチャのギャップ数

int apeNumberObject (String *sObjectClass*)

Gets the number of objects in the current aperture object list. Objects are flashes, draws, arcs and vectortext.

引数:

sObjectClass Specifies the class of the objects to count. "f" for flashes, "d" for draws, "a" for arcs and "v" for vectortext.

戻り値:

the number of objects in the current aperture object list.

int apeNumberObject ( )

Gets the number of objects in the current aperture object list. Objects are flashes, draws, arcs and vectortext.

戻り値:

the number of objects in the current aperture object list.

int apeNumberRegions ( )

カレントアパーチャにおけるコンプレックス・サーマル・輪郭内の領域数を取得する

戻り値:

カレントアパーチャにおけるコンプレックス・サーマル・輪郭内の領域数

int apeNumContours ( )

カレントアパーチャにおけるコンプレックス・サーマル・輪郭内の輪郭エリア数を取得する

戻り値:

カレントアパーチャにおけるコンプレックス・サーマル・輪郭内の輪郭エリア数

void apeOuter (double *dOuter*)

カレントアパーチャにおけるサークル・ドーナッツ・サーマルの外径を設定する

引数:

dOuter double型 カレントアパーチャの新しい外径

double apeOuter ( )

カレントアパーチャにおけるサークル・ドーナッツ・サーマルの外径を取得する

戻り値:

カレントアパーチャの外径

void apePattern (String *sPattern*)

DPFフォーマットでアパーチャパターンを設定する

引数:

sPattern String型 DPFフォーマットにおけるアパーチャパターン

String apePattern ( )

DPFフォーマットでアパーチャパターンを取得する

戻り値:
DPFフォーマットにおけるアパーチャパターン

void apePatternAngle (double *dPatternAngle*)

Sets the angle of the pattern of the aperture.

引数:
dPatternAngle double the angle of the pattern of the aperture in degrees.

double apePatternAngle ( )

Gets the angle of the pattern of the aperture.

戻り値:
the angle of the pattern of the aperture.

void apePatternStep (double *dPatternStep*)

アパーチャのパターンステップを設定する

引数:
dPatternStep double - アパーチャのパターンのステップ

double apePatternStep ( )

アパーチャのパターンステップを取得する

戻り値:
アパーチャのパターンステップ

void apePatternWidth (double *dPatternWidth*)

アパーチャのパターン幅を設定する

引数:
dPatternWidth double型 アパーチャのパターン幅

double apePatternWidth ( )

アパーチャのパターン幅を取得する

戻り値:
アパーチャのパターン幅

void apePatternX (double *dX*)

Sets the x size of the pattern of the aperture.

引数:
dX double the x size of the pattern of the aperture in degrees.

double apePatternX ( )

Gets the x size of the pattern of the aperture.

戻り値:
the x size of the pattern of the aperture.

void apePatternY (double *dY*)

Sets the y size of the pattern of the aperture.

引数:
dY double the y size of the pattern of the aperture in degrees.

double apePatternY ( )

Gets the y size of the pattern of the aperture.

戻り値:
the y size of the pattern of the aperture.

Rectangle apeRectangle ( )

長方形のアパーチャ定義サイズを取得する

戻り値:
長方形のアパーチャ定義サイズ

void apeReverse (boolean *bReverse*)

Sets the reverse status for the current aperture.

引数:
bReverse boolean `true` for reverse, `false` otherwise.

boolean apeReverse ( )

Gets the reverse status for the current aperture.

戻り値:
`true` for reverse, `false` otherwise.

void apeRotation (double *dRotation*)

カレントアパーチャの回転を設定する

引数:
dRotation double型 カレントアパーチャの回転

double apeRotation ( )

カレントアパーチャの回転を取得する

戻り値:
カレントアパーチャの回転

void apeScale (double *dScale*)

カレントアパーチャのスケールファクターを設定する

引数:
dScale double型 カレントアパーチャのスケールファクター

double apeScale ( )

カレントアパーチャのスケールファクターを取得する

戻り値:
カレントアパーチャのスケールファクター

boolean apeSelection ( )

カレントアパーチャが選択アイテムを含むか否かをチェックする

戻り値:
選択があれば1 なければ0

Rectangle apeSelectionEnclosingBox ( )

Gets the enclosed rectangle of all selected objects in current aperture.

戻り値:

the enclosed rectangle of all selected objects in current aperture.

String apeShape ( )

Gets the shape of the current aperture.

戻り値:
the shape of the current aperture ("cir","blo","box","com","con","don","rec","txt","the","oct","und").

void apeSize (double *dSize*)

8角形（オクタゴン）のカレントアパーチャにサイズを設定する

引数:
dSize double型 8角形（オクタゴン）のカレントアパーチャのサイズ

double apeSize ( )

カレント8角形アパーチャのサイズを取得する

戻り値:
カレント8角形アパーチャのサイズ

void apeStartAngle (double *dStartAngle*)

カレントサーマルアパーチャの角度を設定する

引数:
dStartAngle double型 カレントサーマルアパーチャの角度

double apeStartAngle ( )

カレントサーマルアパーチャの角度を取得する

戻り値:
カレントサーマルアパーチャの角度

void apeString (String *sString*)

カレントテキストアパーチャのテキストを設定する

引数:
sString String型 カレントテキストアパーチャのテキスト

String apeString ( )

カレントテキストアパーチャのテキストを取得する

戻り値:
カレントテキストアパーチャのテキスト

void apeStroke (double *dStroke*)

カレント輪郭アパーチャのストローク幅を設定する

引数:
dStroke double型 カレント輪郭アパーチャのストローク幅

double apeStroke ( )

カレント輪郭アパーチャのストローク幅を取得する

戻り値:
カレント輪郭アパーチャのストローク幅

double apeSurface ( )

カレント輪郭アパーチャの表面積をカレント平方単位で取得する

戻り値:
カレント輪郭アパーチャのカレント平方単位の表面積

**void apeThickenThin (double *value,*
boolean *keepArcs*
)**

アパーチャに対し 太らせ/細らせ（広げる/絞る）機能を実行する これによりアパーチャサイズが変わる

引数:
value 太らせ（プラス）／細らせ（マイナス）量
keepArcs trueならば アークはコンプレックスアパーチャ内に維持される

void apeWidth (double *dWidth*)

カレントテキストアパーチャ幅を設定する

引数:
dWidth double型 カレントテキストアパーチャ幅

double apeWidth ( )

カレントテキストアパーチャ幅を取得する

戻り値:
カレントテキストアパーチャ幅

void apeXCutOff (double *dXCutOff*)

カレントボックスアパーチャのコーナーに対し x カットオフ値を設定する

引数:
dXCutOff double型 コーナーのxカットオフ値

double apeXCutOff ( )

カレントボックスアパーチャのコーナーのxカットオフ値を取得する

戻り値:
コーナーのxカットオフ値

void apeXSize (double *dXSize*)

長方形もしくはボックス型のカレントアパーチャにxサイズを設定する

引数:
dXSize double型 カレントアパーチャのxサイズ

double apeXSize ( )

長方形もしくはボックス型のカレントアパーチャのxサイズを取得する

戻り値:
カレントアパーチャのxサイズ

void apeYCutOff (double *dYCutOff*)

カレントボックスアパーチャのコーナーの y カットオフ値を設定する

引数:
dYCutOff double型 コーナーの y カットオフ値

double apeYCutOff ( )

カレントボックスアパーチャのコーナーの y カットオフ値を取得する

戻り値:
コーナーの y カットオフ値

void apeYSize (double *dYSize*)

長方形もしくはボックス型のカレントアパーチャに y サイズを設定する

引数:

dYSize double型 カレントアパーチャの y サイズ

double apeYSize ( )

長方形もしくはボックス型のカレントアパーチャから y サイズを取得する

戻り値:

カレントアパーチャの y サイズ

int applyHorns (String *hornType,*
double *minimumClearance,*
ObjectList *params*
)

Applies horns to all hornable pads in current selection or in current layer if there is no selection.

引数:

hornType	the type of horn to apply. Can be "up", "up_tilted", "side", "corner_rec", "corner_square" or "corner_flat". (See drawings at [we need a location to add drawings, this is impossible to explain otherays]).
minimumClearance	minimum free distance to other objects after applying horns
params	an array containing all the parameters needed for the specific horn type. (See drawings at [we need a location to add drawings, this is impossible to explain otherays]).

戻り値:

amount of pads to which horns where applied if successful -1 if params array contains non-floats -2 if an invalid horn type has been given -3 if params array was not of the right length

Arc Arc (double *ptFromX,*
double *ptFromY,*
double *ptToX,*
double *ptToY,*
double *ptCenterX,*
double *ptCenterY,*
String *sSense,*
String *sUnits*
)

6座標・描画方向・単位からアークを作成する

引数:

ptFromX	始点X座標
ptFromY	始点Y座標
ptToX	終点X座標
ptToY	終点Y座標

ptCenterX 中心X座標
ptCenterY 中心Y座標
sSense アーク描画方向
sUnits Ucam単位

戻り値:
アーク

Arc Arc (double *ptFromX*,
double *ptFromY*,
double *ptToX*,
double *ptToY*,
double *ptCenterX*,
double *ptCenterY*,
String *sSense*
)

6座標と描画方向からアークを作成する

引数:
ptFromX 始点X座標
ptFromY 始点Y座標
ptToX 終点X座標
ptToY 終点Y座標
ptCenterX 中心X座標
ptCenterY 中心Y座標
sSense アーク描画方向

戻り値:
アーク

Arc Arc (Arc *oArc*)

アークのコピーを作成する

引数:
oArc 元のアーク

戻り値:
the arc

Arc Arc (Point *ptFrom*,
Point *ptTo*,
Point *ptCenter*,
String *sSense*
)

3つのポイントと描画方向からアークを作成する

引数:
ptFrom アーク開始ポイント

ptTo アーク終了ポイント
ptCenter アーク中心ポイント
sSense アーク描画方向

戻り値:
アーク

void autofixtureBuildFixture (boolean *bFixtureBuild,*
String *sFixture*
)

自動治具作成ダイアログにパラメータを設定する

引数:
bFixtureBuild 治具設定を有効にする
sFixture 治具設定

例外:
AbortException

void autofixtureDo ( )

自動治具作成ダイアログの全データを生成する

void autofixtureMicroAdjustment (boolean *bMicroAdjustment,*
int *iNbrOfTestPoints,*
double *dTestPointDiameter,*
double *dTestPointShiftEdge,*
double *dTestPointShiftValue,*
double *dTestPointPitch,*
double *dClearanceFactor,*
double *dCenterDiameter*
)

Sets parameters to dialog Autofixture and to subdialog Micro Adjustment

引数:

bMicroAdjustment	enable Micro Adjustment Setup
iNbrOfTestPoints	The following alignment point parameters are used for all selected test points.
dTestPointDiameter	The aperture diameter of the pads used as alignment points (in the current unit).
dTestPointShiftEdge	The distance between the test point and the first alignment point's center (in the current unit).
dTestPointShiftValue	The distance between each of the alignment points' center on the axis of the shortest side (in the current unit).
dTestPointPitch	The distance between each of the alignment points center on the axis of the longest side (in the current unit).
dClearanceFactor	The minimum clearance required between other copper areas and the edge of the test point (at the longest side).
dCenterDiameter	The aperture diameter of the center point (in the current unit).

void autofixtureNetlist (boolean *bNetlist*,
boolean *bNetlistBuild*,
boolean *bNetlistExpand*
)

Sets parameters to dialog Autofixture to section Netlist

引数:

bNetlist	if sets true, generate the netlist for the current job.
bNetlistBuild	if sets true, generates or regenerates a netlist of the job.
bNetlistExpand	if sets true, generates the netlist for a panelized job

void autofixtureOutput (boolean *bOutput*)

自動治具作成ダイアログにパラメータを設定する

引数:

bOutput アウトプット設定を有効にする

void autofixtureTestpoints (boolean *bTestPoints*,
int *iLoop*,
boolean *bUseMasks*,
boolean *bProbeSwaping*,
boolean *bHandlePaintedPads*,
boolean *bCircuitryCheck*,
boolean *bFilterCopperAreas*,
boolean *bViaOfSMDs*,
boolean *bDrillsWithoutPad*
)

Sets parameters to dialog Autofixture to section Testpoints

引数:

bTestPoints	if true, calculate the test points of a job and to create one or two test point layers for the job.
iLoop	sets how to test pads in a loop
bUseMasks	if true, takes all solder mask layers into account for test point calculation.
bProbeSwaping	if true, marks all test points that can be technically tested on the other side of the pcb.
bHandlePaintedPads	if thue, handle painted pads
bCircuitryCheck	if true,enables the generation of test points according to electrical test Optimization Rules.
bFilterCopperAreas	if true, reduces the number of test points generated in large coppers by removing all unnecessary test points that are satisfactorily surrounded by copper.
bViaOfSMDs	if true, generates the test points only on the via holes of SMD's, according to the attribute settings for uVia.
bDrillsWithoutPad	if true, generates test points on drill holes without pad.

```
void autofixtureTestpointsBot ( boolean bPointsBot1,
                                boolean bPointsBot2,
                                boolean bPointsBot3,
                                boolean bPointsBot4,
                                boolean bPointsBot5,
                                boolean bPointsBot6,
                                boolean bPointsBot7
                              )
```

自動治具作成ダイアログのテストポイントセクションのボトム側の回路解析設定ルールにパラメータを設定する

引数:

bPointsBot1 ドリル済みパッドのうちトラック（トップもしくはボトムレイヤ）に接続していないパッドを対象に テストポイントを設定する

bPointsBot2 ドリル済みパッドのうち テストポイント側で1つのトラックのみに接続し レイヤの反対側ではいずれのトラックにも接続していないパッドを対象に テストポイントを設定する

bPointsBot3 ドリル済みパッドのうち テストポイント側ではトラックに接続していないが 反対側で1つのトラックのみに接続しているパッドを対象に テストポイントを設定する

bPointsBot4 ドリル済みパッドのうち テストポイント側や反対側ではいずれのトラックにも接続していないが 内層の1つレイヤに接続しているパッドを対象にテストポイントを設定する

bPointsBot5 ドリル済みパッドのうち テストポイント側ではいずれのトラックにも接続していないが レイヤ反対側で2つ以上のトラックに接続しているパッドを対象にテストポイントを設定する

bPointsBot6 ドリル済みパッドのうち レイヤのテストポイント側や反対側ではいずれのトラックにも接続していないが 内層の2つ以上のレイヤに接続しているパッドを対象にテストポイントを設定する

bPointsBot7 ドリル済みパッドではなく かつ2つ以上のトラックに接続しているパッドを対象に テストポイントを設定する

```
void autofixtureTestpointsTop ( boolean bPointsTop1,
                                boolean bPointsTop2,
                                boolean bPointsTop3,
                                boolean bPointsTop4,
                                boolean bPointsTop5,
                                boolean bPointsTop6,
                                boolean bPointsTop7
                              )
```

Sets parameters to dialog Autofixture to section Testpoints - Optimization Rule top side

引数:

bPointsTop1 Sets a test point on a pad that is a drilled pad and that is not connected with any track (on the top or bottom layer).

bPointsTop2 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side and connected with only one track on the opposite layer.

bPointsTop3 Sets a test point on a pad that is a drilled pad and that is connected with only one track on the test point side and not connected with any track on the opposite layer.

bPointsTop4 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side or the opposite layer, but is connected to one inner layer.

bPointsTop5 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side and connected with more than one track on the opposite layer.

bPointsTop6 Sets a test point on a pad that is a drilled pad and that is not connected with any track on

the test point side or the opposite layer, but is connected to two or more inner layers.

bPointsTop7 Sets a test point on a pad that is not a drilled pad and that is connected with more than one track.

void blockEdit ( )

アパーチャマネジャー: カレントブロックアパーチャのブロック定義編集モードに入る

void blockMultiEdit ( )

アパーチャマネジャー: カレントブロックアパーチャのブロック定義マルチ編集モードに入る

int BlockReconstruct ( )

Makes a block out of the current selection of the layer in plane1 and replaces any occurrences of the same data with a block flash.

戻り値:

a negative number if a problem was detected or the number of blocks that were created otherwise.

void boardSnapshot (boolean *graph,*
String *templPath,*
boolean *pio,*
String *pioPath*
)

ボード・スナップショットを生成する

引数:

graph trueならばグラフィックアウトプットを生成
templPath テンプレートファイルへのパス
pio trueならば製品情報のテキストアウトプットを生成
pioPath 情報テキストへのパス

void buildSubJobs ( )

デフォルトサブジョブを作成する

void calculateImpedance (String *sImpConfig,*
ObjectList *parameters*
)

インピーダンスの計算を行う

引数:

sImpConfig　インピーダンス設定
parameters　パラメーター配列　例：{[width, height, thickness, er]}

boolean canRead (ObjectList *fileInfo*)

Tests whether the application can read the file denoted by this fileInfo.

引数:
fileInfo objectlist with the file information

戻り値:
`true` if and only if the file denoted by this file info and it is allowed to read from the file; `false` otherwise

参照:
HSH_base::osFileInfo(String)

boolean canWrite (ObjectList *fileInfo*)

Tests whether the application can modify the file denoted by this fileInfo.

引数:
fileInfo objectlist with the file information

戻り値:
`true` if and only if the file denoted by this file info and it is allowed to write to the file; `false` otherwise

参照:
HSH_base::osFileInfo(String)

**void center (double *center_x,*
double *center_y*
)**

ナンバーズダイアログの中心を設定する

引数:
center_x　（X座標）中心
center_y　（Y座標）中心

void center (Point *center*)

ナンバーズダイアログの中心を設定する

引数:
center　センター

Point center ( )

ナンバーズダイアログの中心を取得する

戻り値:
センター

void centerX (double *centerX*)

ナンバーズダイアログの中心X座標を設定する

引数:
centerX 中心のX座標

void centerY (double *centerY*)

ナンバーズダイアログの中心y座標を設定する

引数:
centerY 中心のy座標

void chain ( )

If the order of the draws in a chain is not consecutive or if some of the draws have their vector-direction reversed you can use Chain to make the (selected) chains continuous.

void chamferJoin (double *pt_x*,
double *pt_y*,
double *disX*,
double *disY*
)

既存のドローに面取りを施す

引数:
pt_x (X 座標) 2つのドローの交差点
pt_y (Y 座標) 2つのドローの交差点
disX 水平方向の長さ（面取り 交差点）
disY 垂直方向の長さ（交差点 面取り）

void chamferJoin (Point *pt*,
double *disX*,
double *disY*
)

既存のドローに面取りを施す

引数:
pt 2描画の交差点
disX 水平方向の長さ（面取り面 交差点）
disY 垂直方向の長さ（交差点 面取り面）

**void changeDirection (double *p_x,*
double *p_y*
)**

最近接のルートチェーンの方向を変える ポイントは X・Y座標により定義される

引数:

p_x (X 座標) ルートチェーンを探すポイント
p_y (Y 座標) ルートチェーンを探すポイント

void changeDirection (Point *p*)

最近接のルートチェーンの方向を変える ポイントは X・Y座標により定義される

引数:

p ルートチェーンを探す場所

**int changePrioPlotQueue (String *sRipHost,*
int *iJobId,*
int *iPriority*
)**

Change plot queue job priority

引数:

sRipHost	name of the Rip host
iJobId	ID of the rip job
iPriority	new priority

戻り値:

0 if OK, otherwise is an error

**boolean checkDrillInfo (boolean *bSelNonPlated,*
boolean *bAssignAttributes,*
boolean *bBlocksOnly*
)**

Drill Info Generates drill info and selects non plated holes in the active layers.

引数:

bSelNonPlated	When true, non plated drill holes are selected.
bAssignAttributes	When true, the drill info results are stored as attributes UdrillStat on the drill holes
bBlocksOnly	When true, drill info is only calculated for objects in Panel StepRepeat blocks

戻り値:

a status, false = ok

**String chooseDirPath (String *sTitle,*
String *sStartDir*
)**

Opens Open File dialog and let the user select directory path

引数:

sTitle Dialog title
sStartDir Starting directory

戻り値:

Directory path

例外:

AbortException After Cancel button the script is aborted.

String chooseDirPath (String *sTitle*)

Opens Open File dialog and let the user select directory path

引数:

sTitle Dialog title

戻り値:

Directory path

例外:

AbortException After Cancel button the script is aborted.

String chooseDirPath ( )

Opens Open File dialog and let the user select directory path

戻り値:

Directory path

例外:

AbortException After Cancel button the script is aborted.

**String chooseFilePath (String *sTitle,*
String *sStartDir,*
String *sFileMask*
)**

Opens Open File dialog and let the user select file path **Example:**

```
openJob(chooseFilePath("Select Job", "D:/MyJobs/Test", "*.job"));
openJob(chooseFilePath("Select Job", "D:/MyJobs/Test", "*.job"));
```

引数:

sTitle Dialog title
sStartDir initial directory when Open dialog is opened
sFileMask file name mask

戻り値:
Selected file path

例外:
AbortException After Cancel button the script is aborted.

参照:
chooseFilePath()

chooseFilePath()

chooseFilePath(String)

chooseFilePath(String)

chooseFilePath(String, String)

chooseFilePath(String, String)

String chooseFilePath (String *sTitle*)

Opens Open File dialog and let the user select file path

引数:
sTitle Dialog title

戻り値:
Selected file path

例外:
AbortException After Cancel button the script is aborted.

参照:
chooseFilePath(String, String, String)

chooseFilePath(String, String, String)

**String chooseFilePath (String *sStartDir,*
String *sFileMask*
)**

Opens Open File dialog and let the user select file path

引数:
sStartDir initial directory when Open dialog is opened
sFileMask file name mask

戻り値:
Selected file path

例外:
AbortException After Cancel button the script is aborted.

参照:
chooseFilePath(String, String, String)

chooseFilePath(String, String, String)

String chooseFilePath ( )

Opens Open File dialog and let the user select file path

戻り値:
Selected file path

例外:
AbortException After Cancel button the script is aborted.

参照:
chooseFilePath(String, String, String)

chooseFilePath(String, String, String)

void cleanApertures ( )

アパーチャマネジャー: プレーン1のアクティブレイヤにおけるネガポジ極性によるアパーチャのグループ化

void cleanApeTables ( )

全てのアクティブレイヤのアパーチャテーブルをクリーンする

void cleanETMComponentLayers (int *type*)

Creates or cleans existing component layers

引数:
type The type of the component layers to clean: 0: CAPACITOR; 1: INDUCTOR; 2: KELVIN; 3: RESISTOR; 4: DIODE; 5: HIPOT

void cleanSubJobs ( )

Clean all sub-jobs

void cleanUfd (String *sUfdName*)

Create new empty fault database with the given name and add it to Error Manager

引数:
sUfdName fault database name

void cleanUnderBlo ( )

Clean data under BLO with uPcb attribute using CLIPPING and MERGE.

void cleanup (double *dReconstructArcs*,
double *dValidateArcs*,
double *dRemoveObsoleteObjects*,
double *dRemoveSmallObjects*,
double *dReconnectObjects*,
boolean *bReconstructArcs*,
boolean *bValidateArcs*,
boolean *bRemoveObsoleteObjects*,
boolean *bRemoveSmallObjects*,
boolean *bReconnectObjects*
)

引数:

dReconstructArcs	アーク再構成許容量
dValidateArcs	アーク有効化許容量
dRemoveObsoleteObjects	余計なオブジェクト削除の許容量
dRemoveSmallObjects	微小オブジェクト削除の許容量
dReconnectObjects	オブジェクト再接続の許容量
bReconstructArcs	trueの場合 アーク再構成
bValidateArcs	trueの場合 アーク有効化
bRemoveObsoleteObjects	trueの場合 余計なオブジェクト削除
bRemoveSmallObjects	trueの場合 微小オブジェクト削除
bReconnectObjects	trueの場合 オブジェクト再接続

void clearance (double *clearance*)

クリアランス値を設定する

引数:

clearance クリアランス値

double clearance ( )

クリアランス数値を取得する

戻り値:

クリアランス値

void clearanceCheckMAT (double *dPadSpread*,
double *dSmdSpread*,
double *dTrackSpread*,
double *dAreaSpread*,
double *dPadPadClearance*,
double *dPadSmdClearance*,
double *dPadTrackClearance*,
double *dPadAreaClearance*,

```
                    double  dSmdSmdClearance,
                    double  dSmdTrackClearance,
                    double  dSmdAreaClearance,
                    double  dTrackTrackClearance,
                    double  dTrackAreaClearance,
                    double  dAreaAreaClearance,
                    boolean bCheckSameNetSpacing,
                    boolean bFastMode,
                    int     iShiftMode,
                    double  dMinCopper
                )
```

Secure Etch Compensation

引数:

dPadSpread	The spread value to apply to circular pads
dSmdSpread	The spread value to apply to non circular pads
dTrackSpread	The spread value to apply to draws
dAreaSpread	The spread value to apply to areas
dPadPadClearance	The clearance to keep between circular pads
dPadSmdClearance	The clearance to keep between circular pads and non circular pads
dPadTrackClearance	The clearance to keep between circular pads and draws
dPadAreaClearance	The clearance to keep between circular pads and areas
dSmdSmdClearance	The clearance to keep between non circular pads
dSmdTrackClearance	The clearance to keep between non circular pads and draws
dSmdAreaClearance	The clearance to keep between non circular pads and areas
dTrackTrackClearance	The clearance to keep between draws
dTrackAreaClearance	The clearance to keep between draws and areas
dAreaAreaClearance	The clearance to keep between areas
bCheckSameNetSpacing	Whether to check spacing between objects of the same net as well
bFastMode	If true, skip slow "select embedded" step to make check faster but sometimes less correct
iShiftMode	Mode to shift clearances, 0: no shift, 1: shift to cut only areas, 2: like 1, but keep original copper of areas
dMinCopper	The minimum copper width to keep

```
void clearanceCheckMAT ( double  dPadSpread,
                         double  dSmdSpread,
                         double  dTrackSpread,
                         double  dAreaSpread,
                         double  dPadPadClearance,
                         double  dPadSmdClearance,
                         double  dPadTrackClearance,
                         double  dPadAreaClearance,
                         double  dSmdSmdClearance,
                         double  dSmdTrackClearance,
                         double  dSmdAreaClearance,
                         double  dTrackTrackClearance,
                         double  dTrackAreaClearance,
                         double  dAreaAreaClearance,
                         boolean bCheckSameNetSpacing,
```

```
                    boolean  bFastMode,
                    int      iShiftMode
                )
```

Secure Etch Compensation

引数:

dPadSpread	The spread value to apply to circular pads
dSmdSpread	The spread value to apply to non circular pads
dTrackSpread	The spread value to apply to draws
dAreaSpread	The spread value to apply to areas
dPadPadClearance	The clearance to keep between circular pads
dPadSmdClearance	The clearance to keep between circular pads and non circular pads
dPadTrackClearance	The clearance to keep between circular pads and draws
dPadAreaClearance	The clearance to keep between circular pads and areas
dSmdSmdClearance	The clearance to keep between non circular pads
dSmdTrackClearance	The clearance to keep between non circular pads and draws
dSmdAreaClearance	The clearance to keep between non circular pads and areas
dTrackTrackClearance	The clearance to keep between draws
dTrackAreaClearance	The clearance to keep between draws and areas
dAreaAreaClearance	The clearance to keep between areas
bCheckSameNetSpacing	Whether to check spacing between objects of the same net as well
bFastMode	If true, skip slow "select embedded" step to make check faster but sometimes less correct
iShiftMode	Mode to shift clearances, 0: no shift, 1: shift to cut only areas, 2: like 1, but keep original copper of areas

```
void clearanceCheckMAT ( double  dPadSpread,
                         double  dSmdSpread,
                         double  dTrackSpread,
                         double  dAreaSpread,
                         double  dPadPadClearance,
                         double  dPadSmdClearance,
                         double  dPadTrackClearance,
                         double  dPadAreaClearance,
                         double  dSmdSmdClearance,
                         double  dSmdTrackClearance,
                         double  dSmdAreaClearance,
                         double  dTrackTrackClearance,
                         double  dTrackAreaClearance,
                         double  dAreaAreaClearance,
                         boolean bCheckSameNetSpacing,
                         boolean bFastMode
                       )
```

Secure Etch Compensation

引数:

dPadSpread	The spread value to apply to circular pads
dSmdSpread	The spread value to apply to non circular pads
dTrackSpread	The spread value to apply to draws
dAreaSpread	The spread value to apply to areas

dPadPadClearance	The clearance to keep between circular pads
dPadSmdClearance	The clearance to keep between circular pads and non circular pads
dPadTrackClearance	The clearance to keep between circular pads and draws
dPadAreaClearance	The clearance to keep between circular pads and areas
dSmdSmdClearance	The clearance to keep between non circular pads
dSmdTrackClearance	The clearance to keep between non circular pads and draws
dSmdAreaClearance	The clearance to keep between non circular pads and areas
dTrackTrackClearance	The clearance to keep between draws
dTrackAreaClearance	The clearance to keep between draws and areas
dAreaAreaClearance	The clearance to keep between areas
bCheckSameNetSpacing	Whether to check spacing between objects of the same net as well
bFastMode	If true, skip slow "select embedded" step to make check faster but sometimes less correct

void clearMessages ( )

Clears messages window

boolean clipping (int *iClipReference,*
String *sClipSide,*
double *dClipClr,*
double *dMinLineLength,*
boolean *bRounded*
)

Clipping

引数:

iClipReference	Ref layer for clipping (0 for outline, 1, 2, 3 or 4 for plane colors)
iClipReference	Ref layer for clipping (0 for outline, 1, 2, 3 or 4 for plane colors)
sClipSide	Side where clipping should be applied ("outside" or "inside")
dClipClr	Clearance after clipping
dMinLineLength	Minimum length of clipped objects
bRounded	True if endpoints of clipped objects should be round

戻り値:

status value

boolean clipSilk (double *dClr,*
double *dMinLen*
)

シルク最適化は ジョブの全シルクレイヤをクリップする 対応するマスクレイヤは クリッピングのリファレンスとして用いられる もしくは マスクレイヤが見つからない場合は 対応する外層レイヤが用いられる シルクレイヤでは パッドとトラックのみがクリップされ またパッドだけがクリッピングリファレンスとして用いられる

引数:

dClr	は クリッピングに不可欠なクリアランス値 シルククリアランスがシルクレイヤオブジェクトと参照レイヤオブジェクトの間の値になるように シルククリッピングによってシルクレイヤからデータが削除される

dMinLen は どのサイズより小さいトラックを削除するか ここで値を設定する

戻り値:
クリッピングされていなければtrue 問題がなく実行されればfalse

void closeAMLIJobManager ( )

close AMLI Job Manager

void closeAnamorphicScale ( )

close AnamorphicScale dialog

void closeApeCreator ( )

close Aperture Creator

void closeApeEditor ( )

close Aperture Editor

void closeApertureAttributes ( )

close Aperture Attributes dialog

void closeApertureManager ( )

アパーチャマネジャーダイアログを閉じる

void closeAttributeEditor ( )

close Attribute Editor dialog

void closeAttributeManager ( )

close Attribute Manager dialog

void closeAutoDrill ( )

自動ドリルダイアログを閉じる

void closeAutoDrillEditor ( )

Close AutoDrill Editor

void closeAutoFixture ( )

close AutoFixture dialog

void closeBarcode ( )

close Barcode dialog

void closeBarcode128 ( )

close Barcode 128 dialog

void closeBoardAnalyzer ( )

基板解析ダイアログを閉じる

void closeBoardSnapshot ( )

ボードスナップショットダイアログを閉じる

void closeCalculatorSetup ( )

演算設定ダイアログを閉じる

void closeCamtek ( )

Camtekダイアログを閉じる

void closeCheckList ( )

close CheckList Dialog

void closeCheckListDefineChecklist ( )

Close "CheckList: Define Checklist" Dialog

void closeCheckListDefineSteps ( )

Close "CheckList: Define Steps" Dialog

void closeClipping ( )

クリッピングダイアログを閉じる

void closeColor ( )

Close Color dialog

void closeConnect ( )

接続ダイアログを閉じる

void closeContourHandling ( )

輪郭ハンドリングダイアログを閉じる

void closeConvertAttributes ( )

close Attribute Converter dialog

void closeCopperBalance ( )

close Copper Balance Dialog

void closeCopperRepair ( )

ピンホール修正ダイアログを閉じる

void closeCoverlayOptimizer ( )

close Coverlay Optimizer dialog

void closeCU9000Dialog ( )

Close DS DI output dialog

void closeDatums ( )

データムダイアログを閉じる

void closeDistort ( )

スケール補正ダイアログを閉じる

void closeDrawSlots ( )

close Draw Slots Dialog

void closeDRC ( )

DRCダイアログを閉じる

void closeDrillInfo ( )

ドリル情報ダイアログを閉じる

void closeDrillMap ( )

close Drill Map Dialog

void closeDrillOptimizer ( )

close Smart Drill Optimizer

void closeDrillRoutSetups ( )

Close Drill/Rout Setups dialog

void closeDrillTolerance ( )

Close Drill Tolerance dialog

void closeDrillToolManager ( )

ドリルツールマネージャを閉じる

void closeDsAoi ( )

Close DS AOI dialog

void closeDSAOIialog ( )

Close DS DI output dialog

void closeDsAoiPreview ( )

Close DS AOI dialog

void closeEditingToolbox ( )

ツールボックス編集ダイアログを閉じる

void closeEditVectorText ( )

ベクトル文字編集ダイアログを閉じる

void closeErrors ( )

エラーダイアログを閉じる

void closeEtchCompensation ( )

close Etch Compensation dialog

void closeExpand ( )

展開ダイアログを閉じる

void closeExternalLinkManager ( )

close External Link Manager

void closeFiducials ( )

close Fiducials dialog

void closeFillAngledPattern ( )

close Fill Angled Pattern Dialog

void closeFillPattern ( )

close Fill Pattern Dialog

void closeFillVector ( )

close Fill Vector Dialog

void closeFlashMaker ( )

フラッシュメーカーダイアログを閉じる

void closeFlexManager ( )

close uFlex Manager

void closeFlipJob ( )

close Flip Job dialog

void closeFrame (String *sFrameName*)

Closes custom dockable frame with given identification

引数:

sFrameName identification frame given by getFrameID() method of CustomFrame class

void closeGridParameters ( )

グリッドパラメータダイアログを閉じる

void closeHiPot ( )

close HiPot dialog

void closeImageCompare ( )

close Image Compare dialog

void closeImpedanceControl ( )

close Impedance Control dialog

void closeImportODBxx ( )

ODBxxステップインポートダイアログを閉じる

void closeInsertContourText ( )

close Insert Contour Text dialog

void closeInsertVectorText ( )

ベクトル文字の挿入ダイアログを閉じる

void closeJobDefinition ( )

ジョブ定義ダイアログを閉じる

void closeJobEdit ( )

ジョブ編集ダイアログを閉じる

void closeJobEditor ( )

Close Job Editor dialog

void closeJobEditorOptions ( )

Close Job Editor Options dialog

void closeJobMerge ( )

ジョブ合成ダイアログを閉じる

void closeJobPlaneSetup ( )

Close Job Plane Setup dialog

void closeJobPrint ( )

Close Job Print dialog

void closeLayerEdit ( )

close Layer Modify dialog

void closeLegendOptimizer ( )

close Legend Optimizer dialog

void closeLoadCheckList ( )

close Load CheckList Dialog

void closeMagnifier ( )

拡大レンズウィンドウを閉じる

void closeMarkupAssistant ( )

close Markup Assistant

void closeMessages ( )

メッセージログウィンドウを閉じる

void closeMLIOutput ( )

close MLI Output dialog

void closeModels ( )

モデルダイアログを閉じる

void closeNetCompare ( )

close Net Compare dialog

void closeNonFunctionalPad ( )

close Non-Functional pads dialog

void closeNumbers ( )

ナンバーズダイアログを閉じる

void closeObjectAttributes ( )

オブジェクトの属性ダイアログを閉じる

void closeObjectCompare ( )

close Object Compare dialog

void closeOutputAccumatch ( )

Close Accumatch Output dialog

void closeOutputAOI ( )

Close AOI Output dialog

void closeOutputCAD ( )

CAD出力ダイアログを閉じる

void closeOutputCamtek ( )

Close Camtek Output dialog

void closeOutputDrillRout ( )

ドリル／ルータダイアログを閉じる

void closeOutputDsDi ( )

Close DS DI output dialog

void closeOutputDsDiPreview ( )

Close DS DI Preview dialog

void closeOutputNetlist ( )

ネットリスト出力ダイアログを閉じる

void closeOutputOrbot ( )

Close Orbot Output dialog

void closeOutputSapphire ( )

Close Sapphire Output dialog

void closeOutputScoring ( )

スクアリング出力ダイアログを閉じる

void closeOutputSmartArgos ( )

Close SmartArgos Output dialog

void closeOutputTrackscan ( )

Close Trackscan Output dialog

void closeOutputUxpAutomanager ( )

Close Output UXP Automanager dialog

void closeOutputUxpEtec ( )

Close Output UXP Etec dialog

void closePanelFramesCoupons ( )

Close Panel Frames Coupons dialog

void closePanelLinks ( )

Close Panel Links dialog

void closePanelPlus ( )

パネルプラスダイアログを閉じる

void closePanelReproduce ( )

PanelReproduceダイアログを閉じる

void closePanelSetup ( )

Close Panel Setup dialog

void closePanelStepRepeat ( )

パネルステップリピートダイアログを閉じる

void closePlaneAdjuster ( )

close Plane Adjuster dialog

void closePlotParameters ( )

close Plot Parameters dialog

void closePPMonitor ( )

close PPMonitor dialog

void closeQueryNet ( )

close Query Net dialog

void closeQueryObject ( )

オブジェクト解析ダイアログを閉じる

void closeReferencePoints ( )

リファレンスポイントダイアログを閉じる

void closeRegister ( )

重ね合わせダイアログを閉じる

void closeRemoveAttributes ( )

close Remove Attributes dialog

void closeRepair ( )

Close Repair dialog

void closeRoutManager ( )

Rout Editor（ルータ編集）ダイアログを閉じる

void closeRoutManagerCleanUp ( )

Rout Editor（ルータ編集）ダイアログを閉じる

void closeRoutManagerDimensioning ( )

Rout Editor（ルータ編集）ダイアログを閉じる

void closeRoutManagerEditor ( )

Rout Editor（ルータ編集）ダイアログを閉じる

void closeRoutManagerTools ( )

Rout Editor（ルータ編集）ダイアログを閉じる

void closeSaveLayout ( )

レイアウトの保存ダイアログを閉じる

void closeSecureEtchCompensation ( )

close Secure Etch Compensation dialog

void closeSelections ( )

選択ダイアログを閉じる

void closeSetupOptions ( )

Close Setup Options dialog

void closeSetupSave ( )

Close Save dialog

void closeShavePads ( )

パッドのシェービングダイアログを閉じる

void closeSignalLayerAdjuster ( )

close Signal Layer Adjuster dialog

void closeSignalLayerAdjusterAssistant ( )

close Signal Layer Adjuster Assistant dialog

void closeSilkOptimizer ( )

close Silk Optimizer

void closeSmartCamtek ( )

SmartCamtekダイアログを閉じる

void closeSmartDRC ( )

Smart DRCダイアログを閉じる

void closeSmartFix ( )

close SmartFix dialog

void closeSmartplot ( )

Close Smartplot dialog

void closeSmartSR ( )

Close Smart S&R dialog

void closeSmartStart ( )

ファイル入力ダイアログを閉じる

void closeSoldermask ( )

close Soldermask Dialog

void closeSoldermaskOptimizer ( )

close Soldermask Optimizer Dialog

void closeTearDrop ( )

close Tear Drop Dialog

void closeTechnicalAnalyzer ( )

close Technical Analyzer dialog

void closeTestpointEdit ( )

close Testpoint edit dialog

void closeToolbarManager ( )

close Toolbar Manager Dialog

void closeToolbars ( )

close Toolbars Dialog

void closeTransformObjects ( )

移動ダイアログを閉じる

void closeTransformObjectsBGAPads ( )

移動BGAパッドダイアログを閉じる

void closeTransformObjectsBGATracks ( )

移動BGAトラックダイアログを閉じる

void closeTransformObjectsEdit ( )

移動編集ダイアログを閉じる

void closeTransformObjectsRescale ( )

移動リスケールダイアログを閉じる

void closeUcamDbEditor ( )

Close Ucamdb Editor dialog

void closeUndoRedoDetails ( )

close Undo/Redo Details

void closeUTest ( )

検査ダイアログを閉じる

void closeUtestUtilities ( )

close Utest Utilities dialog

void closeValidateLayer ( )

close Layer Validation dialog (or not, if everything is fine)

void closeVectorTextFont ( )

ベクトル文字ダイアログを閉じる

void closeVerifyArcsDraws ( )

アーク&ドローの検証ダイアログを閉じる

void closeViewGuide ( )

ビューガイドダイアログを閉じる

**void colorAll (String *exclSubClass,*
boolean *bKeepLayActivity*

)

Assigns a plane to all layers that have no plane assigned yet.

引数:

exclSubClass null or comma separated string, list subclasses that won't be colored.
bKeepLayActivity `true` keeps layer activity, `false` deactivates all layers without color if active.

void colorAll (String *exclSubClass*)

プレーン未割り当ての全レイヤに プレーンを割り当てる

引数:

exclSubClass プレーンカラー未割り当て場合 null 又はコンマ区切りのサブクラスリスト

void compareImage (String *reference,*
boolean *bAutoAlign,*
double *missingTol,*
double *exceedingTol,*
int *iErrorAccuracy,*
ObjectList *revPolArr,*
int *compSelMode*
)

Image Compare (external reference)

引数:

reference Reference job or reference layer to compare with
bAutoAlign Automatically align current and reference layers
missingTol Allowed tolerance of missing copper
exceedingTol Allowed tolerance of exceeding copper
iErrorAccuracy The display accuracy for differences
revPolArr Array of layers with reverse polarity, e.g. [{["lay1", false, true}], [{"lay2", true, true}]}] means: lay1 of current job not reversed but reference reversed; lay2 current and reference both reversed
compSelMode 0: compare all, 1: blocks - compare on primary, 2: compare in areas, 3: compare outside areas

void compareImage (String *reference,*
boolean *bAutoAlign,*
double *missingTol,*
double *exceedingTol,*
int *iErrorAccuracy,*
ObjectList *revPolArr*
)

画像比較（外部リファレンス）

引数:

reference 比較するリファレンスジョブもしくはリファレンスレイヤ

bAutoAlign カレントレイヤとリファレンスレイヤを自動で位置合わせする
missingTol 銅不足に対する許容量
exceedingTol 銅超過に対する許容量
iErrorAccuracy エラー表示精度
revPolArr 逆極性レイヤの配列　例:　[{[{"lay1", false, true}], [{"lay2", true, true}]}] カレントジョブのレイヤ#1はリバース（反転）されていないが リファレンスはリバースされている カレントジョブのレイヤ#2とリファレンスは両者がリバースされている

```
void compareImage ( double  missingTol,
                    double  exceedingTol,
                    int     iErrorAccuracy,
                    boolean bRevPolarityCur,
                    boolean bRevPolarityRef,
                    int     compSelMode
                  )
```

Image Compare (Layer 1 - Layer 2)

引数:

missingTol Allowed tolerance of missing copper
exceedingTol Allowed tolerance of exceeding copper
iErrorAccuracy The display accuracy for differences
bRevPolarityCur If true, layer 1 (current) has reversed polarity,
bRevPolarityRef If true, layer 2 (reference) has reversed polarity,
compSelMode 0: compare all, 1: blocks - compare on primary, 2: compare in areas, 3: compare outside areas

```
void compareImage ( double  missingTol,
                    double  exceedingTol,
                    int     iErrorAccuracy,
                    boolean bRevPolarityCur,
                    boolean bRevPolarityRef
                  )
```

画像（レイヤ# 1 −レイヤ#2）

引数:

missingTol 銅欠落の許容量
exceedingTol 銅過剰の許容量
iErrorAccuracy エラー表示精度
bRevPolarityCur trueであれば レイヤ1（カレント）は逆極性である
bRevPolarityRef trueであれば レイヤ2（リファレンス）は逆極性である

```
void compareNet ( int     iMode,
                  boolean bCheckFlash
                )
```

ネット比較

引数:

iMode 比較モード 1:基本テスト 2:1＋複数ネット 3:2＋完全に包括されていない
bCheckFlash true: フラッシュ欠落に対するテスト

void compareNet (int *iMode,*
boolean *bCheckFlash,*
String *sReferenceFile,*
boolean *bPanelize*
)

ネット比較

引数:

iMode 比較モード 1:基本テスト 2:1＋複数ネット 3:2＋完全に包括されていない
bCheckFlash true:フラッシュ欠落に対するテスト
sReferenceFile 複数ジョブのネット比較用のリファレンスファイル
bPanelize 面付けリファレンス

void compareNet (int *iMode,*
boolean *bCheckFlash,*
boolean *bIgnoreOutline,*
double *dOutlineMargin,*
boolean *bIgnoreNPTHPads,*
double *dNPTHExpandMargin,*
String *sReferenceFile,*
boolean *bPanelize*
)

Net Compare

引数:

iMode Compare mode - 1: basic tests, 2: 1 + multiple nets, 3: 2 + not fully covered
bCheckFlash true: test for missing flash
bIgnoreOutline true: ignore nets outside the outline
dOutlineMargin Net references inside the outline and near the outline edge within the chosen margin are also ignored.
bIgnoreNPTHPads true: ignore for the 'Lost Elements' test those pads which correspond to a NPTH hole
dNPTHExpandMargin Net references which correspond to a NTPH hole after expanding the hole with this margin are ignored
sReferenceFile Reference file for multi job net compare
bPanelize Panelize reference

非推奨:

use **compareNet(boolean, boolean, boolean, boolean, boolean, boolean, boolean, double, boolean, double, String, boolean, boolean)**

use **compareNet(boolean, boolean, boolean, boolean, boolean, boolean, boolean, double, boolean, double, String, boolean, boolean)**

void compareNet (boolean *bShorts,*

```
                boolean bOpens,
                boolean bLostElements,
                boolean bDoubleNet,
                boolean bNotCovered,
                boolean bCheckFlash,
                boolean bIgnoreOutline,
                double  dOutlineMargin,
                boolean bIgnoreNPTHPads,
                double  dNPTHExpandMargin,
                String  sReferenceFile,
                boolean bPanelize,
                boolean bBuildNetlist
              )
```

Net Compare

引数:

bShorts	true: test for shorts
bOpens	true: test for opens
bLostElements	true: test for missing copper elements
bDoubleNet	true: test for reference pads touching more than one net
bNotCovered	true: test for reference pads not fully covered by copper
bCheckFlash	true: test for missing flash
bIgnoreOutline	true: ignore nets outside the outline
dOutlineMargin	Net references inside the outline and near the outline edge within the chosen margin are also ignored.
bIgnoreNPTHPads	true: ignore for the 'Lost Elements' test those pads which correspond to a NPTH hole
dNPTHExpandMargin	Net references which correspond to a NTPH hole after expanding the hole with this margin are ignored
sReferenceFile	Reference file for reference job comparison
bPanelize	Panelize reference
bBuildNetlist	Build Job Netlist

```
void compareObjects ( String  referenceJob,
                      double  xTol,
                      double  yTol,
                      boolean bWindow,
                      boolean bObjMoved,
                      boolean bObjAdded,
                      boolean bObjNet,
                      boolean bApeShape,
                      boolean bApeSize,
                      boolean bApeOrder
                    )
```

オブジェクト比較

引数:

referenceJob	比較するリファレンスジョブ
xTol	X方向の比較における許容量
yTol	Y方向の比較における許容量

bWindow エリア定義に用いるドリルレイヤのリファレンスポイント
bObjMoved 移動されたオブジェクトにマークを付ける
bObjAdded カレントジョブのみに存在するオブジェクトにマークを付ける
bObjNet ネット番号をチェックする
bApeShape アパーチャ形状の変更をチェックする
bApeSize アパーチャサイズの変更をチェックする
bApeOrder アパーチャリストの順序をチェックする

void compensate (String *sSense,*
double *dis*
)

ルータツールの厚さに対し ルート補正を行う

引数:
sSense ルータツールのサイズを左方向もしくは右方向へ補正
dis 補正距離

void complexEdit ( )

アパーチャマネジャー: カレントコンプレックスアパーチャのコンプレックス定義編集モードに入る

int connectPadTrack (double *dActiveRadius,*
double *dSnapRadius,*
boolean *bUseNetlist*
)

connectPadTrack

引数:
dActiveRadius Maximum length of connect vector
dSnapRadius Maximum distance over which a vector can be snapped
bUseNetlist If true only vectors of identical, valid net can be connected

戻り値:
error number (100 = no netlist) or number of errors

void connectTracks ( )

このトラック接続は 次の操作に相当する: 移動 > BGAトラック > 接続トラック

boolean contourizeBitmap (int *iPpi,*
double *dMargin,*
double *dDxdy,*
double *dDx,*
double *dDy*

)

ビットマップ形式の輪郭化

引数:

iPpi
dMargin
dDxdy
dDx
dDy

戻り値:

ステータス

void contourizeExact ( )

高精度の輪郭化

void contourizeExactAperture ( )

Exact Contourize the (current) aperture

boolean contourizePatterns (int *iPpi*)

Contourize apertures with patterns in the layer in plane 1

引数:

iPpi given DPI

戻り値:

status false = 0 = ok;

boolean contourizePatternsinJob (int *iPpi*)

Contourize apertures with patterns in all active layers

引数:

iPpi given DPI

戻り値:

status false = 0 = ok;

void contourThickenThin (double *value,*
boolean *bKeepArcs*
)

輪郭に対し 太らせ／細らせ（広げる／絞る）機能を実行する アパーチャサイズは変更される

引数:

value 太らせ量（+）もしくは細らせ量（-）
bKeepArcs trueならば アークが保持される

boolean convertGar (String *sInputFile,*
String *sGarFile,*
String *sOutputFile*
)

GARファイルからアウトプットファイルを作成するルールに基づき インプットファイルを変換する

引数:
sInputFile インプットファイルの完全なパス
sGarFile GARファイル（ルール）の完全なパス
sOutputFile アウトプットファイルの完全なパス

戻り値:
ステータス

void copperBalancePad (double *dMinClrToCopper,*
double *dMinClrToBoard,*
double *dMinConSurface,*
String *sFillPattern,*
double *dPatternClr,*
double *dApeSize,*
String *sApeShape*
)

Creates the venting pattern. The area will be filled with pads

引数:
dMinClrToCopper - Minimum Clearance to Copper, specify the clearance between the copper from the affected layer and the newly created pattern. If the layer is drilled, this value will also be used as clearance between the copper pattern and the holes.
dMinClrToBoard - Minimum Clearance to Board. Specify the clearance from the board edge to the outline of the pattern contour. The original outline will be shrinked with this value to meet the desired clearance.
dMinConSurface - Minimum Contour Surface. All the copper surfaces, smaller than the specified value, are removed.
sFillPattern - Choose one of the following options: - full - pads are placed on each grid points - even - pads are placed only on even grid points - odd - pads are placed only on odd grid points
dPatternClr - Specify the grid step used to place the pads or tracks.
dApeSize - Specify the size of the aperture used to fill the area.
sApeShape - Choose one of the following options - circle, hexagon, diamond

void copperBalanceSolid (double *dMinClrToCopper,*
double *dMinClrToBoard,*
double *dMinConSurface,*
double *dApeSize*
)

銅バランスパターンの作成 このエリアは ベタ塗り輪郭が用いられる

引数:

dMinClrToCopper - 銅との最小クリアランス 関係するレイヤにある銅と新規作成されたパターン間のクリアランスを指定する レイヤがドリルされている場合 この値は 銅パターンとホール間のクリアランスとしても用いられる

dMinClrToBoard - 基板との最小クリアランス 基板の端からパターン輪郭のアウトラインへのクリアランスを指定する 望ましいクリアランスになるように 元のアウトラインはこの値に従って縮小される

dMinConSurface - 最小輪郭サイズ 銅エリアにおいて 指定値よりも小さいものは全て削除される

dApeSize - エリアを埋めるためのアパーチャサイズを指定する

void copperBalanceTrack (double *dMinClrToCopper,*
double *dMinClrToBoard,*
double *dMinConSurface,*
String *sLineStyle,*
double *dPatternClr,*
double *dApeSize,*
double *dRotation*
)

Creates the venting pattern. The area will be filled with tracks using a circle aperture

引数:

dMinClrToCopper - Minimum Clearance to Copper, specify the clearance between the copper from the affected layer and the newly created pattern. If the layer is drilled, this value will also be used as clearance between the copper pattern and the holes.

dMinClrToBoard - Minimum Clearance to Board. Specify the clearance from the board edge to the outline of the pattern contour. The original outline will be shrinked with this value to meet the desired clearance.

dMinConSurface - Minimum Contour Surface. All the copper surfaces, smaller than the specified value, are removed.

sLineStyle - Select the track style to be applied: "parallel lines" or "crosshatched by lines"

dPatternClr - Specify the grid step used to place the pads or tracks.

dApeSize - Specify the size of the aperture used to fill the area.

dRotation - set a rotation angle for tracks

void copperCount (String *sOpt*)

非推奨:

Copper count without mask layer usage

引数:

sOpt Set to "job", "layer", or "inner"

void copperRepair (String *sOpt,*
double *smallerThan,*
double *minSize,*
double *expand*
)

銅修正

引数:

sOpt	修正オプション: "pinholes"（ピンホール）, "peelables"（同一ネットの近接箇所）"slivers"(輪郭エリアの細り箇所)
smallerThan	この値より小さいピンホール／ピーラブル（同一ネットの近接箇所）／スライバー(輪郭エリアの細り箇所)を修正する
minSize	この値より大きいピーラブルもしくはスライバーを修正する
expand	ピンホール／ピーラブル／スライバーの膨張値を設定する

void copy (double *pt_x*,
double *pt_y*
)

Duplicate (selected) object(s) using board coordinates

Example:

```
setInPlane(1,1);
direction("");
copy(100,200);
copy(100,200);
doCopy(100,200);

direction("h");
copy(100,200);
copy(100,200);
doCopy(100,200);

direction("v");
copy(100,200);
copy(100,200);
doCopy(100,200);
```

引数:

pt_x (X coordinate) Offset (vector) where to create the copy of the source objects

参照:

com.barco.ets.ucam.hypershell.HyperShell::doCopy(Upoint)

com.barco.ets.ucam.hypershell.HyperShell::direction(String)

引数:

pt_y (Y coordinate) Offset (vector) where to create the copy of the source objects

参照:

com.barco.ets.ucam.hypershell.HyperShell::doCopy(Upoint)

com.barco.ets.ucam.hypershell.HyperShell::direction(String)

void copy (Point *pt*)

Duplicate (selected) object(s) using board coordinates

Example:

```
setInPlane(1,1);
direction("");
copy(100,200);
copy(100,200);
doCopy(100,200);
```

```
direction("h");
copy(100,200);
copy(100,200);
doCopy(100,200);

direction("v");
copy(100,200);
copy(100,200);
doCopy(100,200);
```

引数:

pt Offset (vector) where to create the copy of the source objects

参照:

com.barco.ets.ucam.hypershell.HyperShell::doCopy(Upoint)

com.barco.ets.ucam.hypershell.HyperShell::direction(String)

void copyOutline (double *refPoint_x*,
double *refPoint_y*,
double *offset_x*,
double *offset_y*,
double *rotation*
)

PCB images in a flat data need to be outlined. We can manually construct outline contour in this (usually the layer in plane 1 and outline extra layer) layer according to a PCB image in reference layer. We mark reference point in PCB image and give an offset of the same reference data in second PCB image we want to outline. The rotation says the PCB outline on target PCB should be rotated 0,90,180,270 degree.

引数:

refPoint_x	(X coordinate) A point where the PCB data are taken as an reference (for alignment)
refPoint_y	(Y coordinate) A point where the PCB data are taken as an reference (for alignment)
offset_x	(X coordinate) a target PCB should have the same objects (from reference point) at the offset position
offset_y	(Y coordinate) a target PCB should have the same objects (from reference point) at the offset position
rotation	the target Outline has to be rotated with given angle (0,90,180,270)

戻り値:

Uapeobj can be `null` in case the new outline couldn't be created. The return aperture can be the same in case the target is without rotation. Only the object(flash) is the new. In case there is a rotation the return object is completely new aperture with the (one) new flash.

void copyOutline (Point *refPoint*,
Point *offset*,
double *rotation*
)

PCB images in a flat data need to be outlined. We can manually construct outline contour in this (usually the layer in plane 1 and outline extra layer) layer according to a PCB image in reference layer. We mark reference point in PCB image and give an offset of the same reference data in second PCB image we want to outline. The rotation says the PCB outline on target PCB should be rotated 0,90,180,270 degree.

引数:

refPoint A point where the PCB data are taken as an reference (for alignment)

offset a target PCB should have the same objects (from reference point) at the offset position
rotation the target Outline has to be rotated with given angle (0,90,180,270)

戻り値:

Uapeobj can be `null` in case the new outline couldn't be created. The return aperture can be the same in case the target is without rotation. Only the object(flash) is the new. In case there is a rotation the return object is completely new aperture with the (one) new flash.

void copyToClipboard ( )

プレーン1 のレイヤにある全ての（選択された）オブジェクトをコピーする オブジェクトはクリップボードに保管される

void coreInfo ( )

Core/Prepreg Info Gives information to log window (console) Adds information like attribute to dpf and job files

int countAmbiguousContours ( )

Count ambiguous contours.

戻り値:

count of ambiguous contours

int countAmbiguousContoursOnLayer ( )

Count ambiguous contours.

戻り値:

count of ambiguous contours

int countInvalidArcs ( )

Count invalid arcs.

戻り値:

count of ambiguous contours

int countInvalidArcsOnLayer ( )

Count invalid arcs.

戻り値:

count of ambiguous contours

int countInvalidDraws ( )

Count invalid draws.

戻り値:
count of invalid draws

int countInvalidDrawsOnLayer ( )

Count Invalid Draws on layer.

戻り値:
count of invalid draws

int countOpenContours ( )

Count Open Contours.

戻り値:
count of open contours

int countOpenContoursOnLayer ( )

Count Open Contours on layer.

戻り値:
count of open contours

int countOverlapContours ( )

Count Overlap Contours.

戻り値:
count of overlap contours

int countOverlapContoursOnLayer ( )

Count Overlap Contours on layer.

戻り値:
count of overlap contours

int countUndefinedApertures ( )

Count Undefined apertures.

戻り値:
count of undefined apertures

int countUndefinedAperturesOnLayer ( )

Count Undefined apertures on lauer.

戻り値:
count of undefined apertures

**int countZeroDrawsArcs (double *dMaxLength,*
boolean *bFunctional,*
boolean *bNonFunctional*
)**

Count ZeroDrawsArcs.

引数:

dMaxLength	Maximum length of objects to be selected
bFunctional	Select objects within functional copper if true
bNonFunctional	Select objects within non-functional copper if true

戻り値:
count of ZeroDrawsArcs

**int countZeroDrawsArcsOnLayer (double *dMaxLength,*
boolean *bFunctional,*
boolean *bNonFunctional*
)**

Count ZeroDrawsArcs on layer.

引数:

dMaxLength	Maximum length of objects to be selected
bFunctional	Select objects within functional copper if true
bNonFunctional	Select objects within non-functional copper if true

戻り値:
count of ZeroDrawsArcs

**void countZeroLengthDrawsArcs (double *dMaxLength,*
boolean *bFunctional,*
boolean *bNonFunctional*
)**

Count Zero Length Draws and Arcs

引数:

dMaxLength	Maximum length of objects to be selected
bFunctional	Select objects within functional copper if true
bNonFunctional	Select objects within non-functional copper if true

```
void createAperture ( int          iApeNum,
                      String       sApeName,
                      String       sApeDef,
                      ObjectList   attrArray
                    )
```

アパーチャマネジャー：Create an Aperture（アパーチャ作成）

引数:

iApeNum	新規作成アパーチャの番号：0未満であれば 次のフリーな番号が使われる
sApeName	新規作成アパーチャの名前
sApeDef	アパーチャのDPFスタイル定義 String型　例）"REC,1.905,0.3048"
attrArray	アパーチャ属性の配列 例）[{"attr1=v1", "attr2=", "attr3=v3"}]

```
boolean createBarcode128 ( double  dHeight,
                           double  dNarrowX,
                           String  sValue
                         )
```

Create barcode 128 like block aperture and add the aperture to the layer. If you want add a flash, you must call insertFlash too.

引数:

dHeight	The barcode height.
dNarrowX	The width of narrow bars.
sValue	a string, what will be transform tu barcode

戻り値:

status, if return true, the function was ok

```
boolean createBarcode39 ( double  dHeight,
                          double  dNarrowX,
                          double  dRatio,
                          String  sValue
                        )
```

Create barcode 39 like block aperture and add the aperture like current to the layer. If you want add a flash, you must call insertFlash too.

引数:

dHeight	The barcode height.
dNarrowX	The width of narrow bars.
dRatio	The barcode ratio.
sValue	a string, what will be transform tu barcode

戻り値:

status, if return true, the function was ok

```
boolean createBarcodeInterleaved25 ( double  dHeight,
                                     double  dNarrowX,
```

```
                                    String   sValue
                                )
```

Create barcode interleaved 25 like block aperture and add the aperture like current to the layer. If you want add a flash, you must call insertFlash too.

引数:

dHeight	The barcode height.
dNarrowX	The width of narrow bars.
sValue	a string, what will be transform tu barcode

戻り値:

status, if return true, the function was ok

```
void createBlockAperture ( int        iApeNum,
                           String     sApeName,
                           String     sApeDef,
                           ObjectList attrArray,
                           int        iMode,
                           boolean    bWithCenter,
                           double     pt_x,
                           double     pt_y,
                           String     sExtFile
                         )
```

アパーチャマネジャー: Create a Block Aperture（ブロックアパーチャの作成）

引数:

iApeNum	新規作成アパーチャの番号: 0未満であれば 次のフリーな番号が使われる
sApeName	新規作成アパーチャの名前
sApeDef	アパーチャのDPFスタイル定義 String型 'BLO' と 'size'は不要
attrArray	アパーチャ属性の配列 例）[{"attr1=v1", "attr2=", "attr3=v3"}]
iMode	0:空のブロック 1:選択 2:外部dpfファイルにリンク
bWithCenter	センターオフセットを減らす （iMode 1のみで使用）
pt_x	(X 座標) センター
pt_y	(Y 座標) センター
sExtFile	外部リンクファイル（iMode 2のみで使用）

```
void createBlockAperture ( int        iApeNum,
                           String     sApeName,
                           String     sApeDef,
                           ObjectList attrArray,
                           int        iMode,
                           boolean    bWithCenter,
                           Point      pt,
                           String     sExtFile
                         )
```

アパーチャマネジャー: Create a Block Aperture（ブロックアパーチャ作成）

引数:

iApeNum 新規作成アパーチャの番号: 0未満であれば 次のフリーな番号が使われる
sApeName 新規作成アパーチャの名前
sApeDef アパーチャのDPFスタイル定義 String型 'BLO' と 'size'は不要
attrArray アパーチャ属性の配列 例）[{"attr1=v1", "attr2=", "attr3=v3"}]
iMode 0:空のブロック 1:選択 2:外部dpfファイルにリンク
bWithCenter 中心座標利用（iMode 1のみで使用）
pt センター
sExtFile 外部リンクファイル（iMode 2のみで使用）

```
void createComplexAperture ( int        iApeNum,
                             String     sApeName,
                             String     sApeDef,
                             ObjectList attrArray,
                             boolean    bUseRegion,
                             boolean    bWithCenter,
                             double     pt_x,
                             double     pt_y
                           )
```

アパーチャマネジャー: Create a Complex Aperture（コンプレックスアパーチャ作成）

引数:

iApeNum 新規作成アパーチャの番号: 0未満であれば 次のフリーな番号が使われる
sApeName 作成するアパーチャの名前
sApeDef アパーチャのDPFスタイル定義 String型 'COM' と 'size'は不要
attrArray アパーチャ属性の配列 例）[{"attr1=v1", "attr2=", "attr3=v3"}]
bUseRegion true:リージョンのみを使用 false:全てのセレクション（選択対象）を使用
bWithCenter 中心座標利用
pt_x (X座標) センターポイント
pt_y (Y座標) センターポイント

```
void createComplexAperture ( int        iApeNum,
                             String     sApeName,
                             String     sApeDef,
                             ObjectList attrArray,
                             boolean    bUseRegion,
                             boolean    bWithCenter,
                             Point      pt
                           )
```

アパーチャマネジャー: Create a Complex Aperture（コンプレックスアパーチャ作成）

引数:

iApeNum 新規作成アパーチャの番号: 0未満であれば 次のフリーな番号が使われる
sApeName 作成するアパーチャの名前
sApeDef アパーチャのDPFスタイル定義 String型 'COM' と 'size'は不要
attrArray アパーチャ属性の配列 例）[{"attr1=v1", "attr2=", "attr3=v3"}]
bUseRegion true:リージョンのみを使用 false:全てのセレクション（選択対象）を使用
bWithCenter 中心座標利用
pt センターポイント

```
void createDataMatrix ( String  sTextToEncode,
                        String  sMode,
                        String  sFormat,
                        double  dDotSize,
                        double  dRotation,
                        String  sMirror,
                        double  dClearance,
                        boolean bReverse
                      )
```

Create barcode - data matrix like block aperture and add the aperture like current to the layer. If you want add a flash, you must call insertFlash too.

引数:

sTextToEncode	Text of the DataMatrix
sMode	Mode of the encoding (MODE_ASCII, MODE_C40, MODE_TEXT, MODE_BASE256, MODE_NONE, MODE_AUTO)
sFormat	Format of the DataMatrix
dDotSize	Size of the dot in the DataMatrix
dRotation	rotation
sMirror	The mirror setting, either "", "X", "Y" or "XY".
dClearance	clearance
bReverse	Indication whether the block aperture needs to be reversed or not.

```
void createDrill ( String layName,
                   String subClass,
                   int    from,
                   int    to
                 )
```

ドリルを作成する

引数:

layName	レイヤ名
subClass	レイヤのサブクラス
from	nレイヤから
to	nレイヤまで

```
void createDrill ( int laynum,
                   int drillFrom,
                   int drillTo
                 )
```

ドリルレイヤを作成する

引数:

laynum	ドリルレイヤの番号
drillFrom	最上部のドリルレイヤのインデックス
drillTo	最下部のドリルレイヤのインデックス

void createExtra (String *layName,*
String *subClass,*
String *attach,*
int *index*
)

特殊レイヤを作成する

引数:

layName レイヤ名
subClass レイヤのサブクラス
attach トップ／ボトム／なしのいずれかに添付
index レイヤインデックス

void createExtra (String *layName,*
String *subClass,*
String *attach*
)

特殊レイヤを作成する

引数:

layName レイヤ名
subClass レイヤサブクラス
attach トップ／ボトム／なしのいずれかに添付

void createExtra (String *attach*)

特殊レイヤを作成する

引数:

attach "top"もしくは"bottom"に添付

void createLayer (String *layName,*
String *subClass,*
int *layPos,*
String *readable*
)

レイヤを作成する

引数:

layName レイヤ名
subClass レイヤサブクラス
layPos レイヤ位置
readable 読込可能面

void createLayer (int *laynum*)

シグナルレイヤを作成する

引数:

laynum シグナルレイヤの番号

void createQRCode (String *sCode,*
double *dDotSize,*
double *dAngle,*
String *sMirror,*
double *dClr,*
double *dLabelClr,*
String *sLabelPos,*
boolean *bMicroQR,*
boolean *bReverse*
)

Create QRCode as a block aperture and add the aperture like current to the layer. If you want add a flash, you must call insertFlash too.

引数:

sCode	Text of the QRCode
dDotSize	Size of the dot in the QRCode
dAngle	Rotation
sMirror	The mirror setting, either "", "X", "Y" or "XY".
dClr	Clearance
dLabelClr	Label clearance
sLabelPos	label position possible values are "top" or "bottom"
bMicroQR	if true the QRCode is MicroQR
bReverse	Indication whether the block aperture needs to be reversed or not.

void createQRCode (String *sCode,*
double *dDotSize,*
double *dAngle,*
String *sMirror,*
double *dClr,*
String *sLabel,*
double *dLabelClr,*
String *sLabelPos,*
boolean *bMicroQR,*
boolean *bReverse*
)

Create QRCode as a block aperture and add the aperture like current to the layer. If you want add a flash, you must call insertFlash too.

引数:

sCode	Text of the QRCode
dDotSize	Size of the dot in the QRCode

dAngle	Rotation
sMirror	The mirror setting, either "", "X", "Y" or "XY".
dClr	Clearance
sLabel	The QRCode label.
dLabelClr	Label clearance
sLabelPos	label position possible values are "top" or "bottom"
bMicroQR	if true the QRCode is MicroQR
bReverse	Indication whether the block aperture needs to be reversed or not.

void createQRCode (String *sCode,*
double *dDotSize,*
double *dAngle,*
String *sMirror,*
double *dClr,*
boolean *bMicroQR,*
boolean *bReverse*
)

Create QRCode as a block aperture and add the aperture like current to the layer. If you want add a flash, you must call insertFlash too.

引数:

sCode	Text of the QRCode
dDotSize	Size of the dot in the QRCode
dAngle	Rotation
sMirror	The mirror setting, either "", "X", "Y" or "XY".
dClr	Clearance
bMicroQR	if true the QRCode is MicroQR
bReverse	Indication whether the block aperture needs to be reversed or not.

void createSubJob (int *from,*
int *to,*
int *selectedSubJob,*
int *selectedLevel*
)

サブジョブを修正／作成する

引数:

from	新規サブジョブもしくは修正サブジョブが開始するレイヤのインデックス
to	新規サブジョブもしくは修正サブジョブが終了するレイヤのインデックス
selectedSubJob	サブジョブのインデックス 0から始まり -1は新規ジョブを作成する
selectedLevel	サブジョブのレベルインデックス 0から始まり -1は新規ジョブを作成する

void createVoronoiDiagram (int *iEdgeTypes*)

Create a new layer which contains the voronoi diagram of the layer in plane 1. This method works on selections, too. This method is licensed.

引数:

iEdgeTypes bit mask specifying which edge/node types need to be output: INNER_EDGES = 0x01; OUTER_EDGES = 0x02; MAIN_EDGES = 0x04 (net separators); DEGENERATE_EDGES = 0x08; BIG_DEADENDS = 0x10; MAXIMUM_NODES = 0x20; ALL_EDGES = 0x3f.

void createVoronoiDiagram (int *iEdgeTypes,*
boolean *bExpandArcs*
)

Create a new layer which contains the voronoi diagram of the layer in plane 1. This method works on selections, too. This method is licensed.

引数:

iEdgeTypes bit mask specifying which edge/node types need to be output: INNER_EDGES = 0x01; OUTER_EDGES = 0x02; MAIN_EDGES = 0x04 (net separators); DEGENERATE_EDGES = 0x08; BIG_DEADENDS = 0x10; MAXIMUM_NODES = 0x20; ALL_EDGES = 0x3f.

bExpandArcs Expand arcs before calculating the diagram.

void createVoronoiEdgesExtFile (boolean *bExpandArcs,*
String *sFilePath,*
String *sOptions*
)

Create an external file with the voronoi diagram edges of the layer in plane 1. This method works on selections, too. This method is licensed.

引数:

bExpandArcs Expand arcs before calculating the diagram.

sFilePath complete destination file path

sOptions options separated by comma like: binary(default), xml, copper_only, space_only (default is copper and space)

void CU9000ApplyPlotstamps (ObjectList *plotstamps*)

The function takes the Object Aray with plotststmps' definitions and applies the changes to plotstamps linked to given layer (usually layer in plane 1)

引数:

plotstamps Object Array with plotstams' definitions

boolean CU9000CheckPlotstamps ( )

The method checks Level plotstamps if they are place at correct Level (PCB counter at PCB level etc.)

戻り値:

`false` check failed due to invalid conditions (layer is not in plane 1, license is missing etc.)

boolean CU9000DetectAutoAreas (String *resultLayerName,*
String *referenceLayerName,*
double *margin*
)

Automatically generate rectangular areas from defined area marks in reference layer.

引数:

resultLayerName	name of the generated area layer
referenceLayerName	reference layer name where area marks are presented
margin	margin to be added

戻り値:

false if a problem was detected during Area Detection or true otherwise

boolean CU9000DetectExactAreas (String *resultLayerName,*
int *blockMode,*
String *pcbName,*
String *referenceLayerName,*
double *margin,*
double *outline*
)

Automatically generate exact outline areas for uPCB blocks for the red layer

引数:

resultLayerName	name of the generated area layer
blockMode	what to use as a reference for block detection: OUTLINE (1) - use outline layer REFERENCE_LAYER (2) - use specified reference layer SELF (3) - use red layer
pcbName	uPCB name to search (if empty, search for deepest level)
referenceLayerName	reference layer name REFERENCE_LAYER block mode
margin	margin to be added
outline	stroke width for exact outline detection

戻り値:

false if a problem was detected during Area Detection or true otherwise.

int CU9000DetectGlobalAlignment (String *sAPRefLayerName*)

Automatically detects global alignment points in the red layer

引数:

sAPRefLayerName	a layer name containing alignment points

戻り値:

a negative number if a problem was detected or the number of placed global alignment points otherwise

int CU9000DetectGlobalAlignment ( )

Automatically detects global alignment points in the red layer

戻り値:
a negative number if a problem was detected or the number of placed global alignment points otherwise

int CU9000DetectLocalAlignmentPoints (String *sAPRefLayerName*)

Automatically detects local alignment points in the red layer

引数:
sAPRefLayerName a layer name containing alignment points

戻り値:
a negative number if a problem was detected or the number of placed local alignment points otherwise

int CU9000DetectLocalAlignmentPoints ( )

Automatically detects local alignment points in the red layer

戻り値:
a negative number if a problem was detected or the number of placed local alignment points otherwise

**boolean CU9000DetectRectangularAreas (String *resultLayerName,*
int *blockMode,*
String *pcbName,*
String *referenceLayerName,*
double *margin*
)**

Automatically generate rectangular areas for uPCB blocks for the red layer

引数:

resultLayerName	name of the generated area layer
blockMode	what to use as a reference for block detection: OUTLINE (1) - use outline layer REFERENCE_LAYER (2) - use specified reference layer SELF (3) - use red layer
pcbName	uPCB name to search (if empty, search for deepest level)
referenceLayerName	reference layer name REFERENCE_LAYER block mode
margin	margin to be added

戻り値:
false if a problem was detected during Area Detection or true otherwise

ObjectList CU9000GetPlotstamps ( )

The function returns Object Array of the plotstamps' definitions for given (current) layer.

戻り値:
Object Array with plotstamps' definitions related to the given layer. **Example:** [{[{"Code=PP,Polarity=Positive,Mirror=x,Rotation=0.0", "247.3172", "170.7196", "panel=1,array=3,pcb=3"}], [{"Code=PP,Polarity=Positive,Mirror=x,Rotation=0.0", "247.3172", "205.7196", "panel=1,array=3,pcb=1"}], [{"Code=PP,Polarity=Positive,Mirror=x,Rotation=0.0", "282.3172", "170.7196", "panel=1,array=3,pcb=4"}], ...}]}]

void CU9000GUIApply ( )

Method simulates Apply button press.

void CU9000GUILoadAlignment (String *sAlignmentPath*)

loads Alignment points definition file and updates GUI

引数:

sAlignmentPath Alignment points definition file (TXT or Gerber) full path

void CU9000GUILoadBrd (String *sBrdPath*)

loads definition file and updates GUI

引数:

sBrdPath Board Definition file (.brd) full path

void CU9000GUILoadRgi (String *sRgiPath*)

loads definition file and updates GUI

引数:

sRgiPath Board Definition file (.rgi) full path

void CU9000GUISaveAlignment (String *sAlignmentPath*)

saves Alignment points definition file

引数:

sAlignmentPath Alignment points definition file (TXT or Gerber) full path

boolean CU9000LoadBoardSetup (String *path*)

Apply the given Board Setup file to all active layers

引数:

path path to board setup file

戻り値:

false if the board file could not be loaded or true otherwise.

boolean CU9000LoadResistSetup (String *path*)

Apply the given Resist Setup file to all active layers

引数:

path path to resist setup file

戻り値:

false if the resist file could not be loaded or true otherwise.

**boolean CU9000LoadResources (String *sPropertiesPath,*
String *sPropertiesName,*
String *sConversionFileName*
)**

Loads given resource files

引数:

sPropertiesPath	properties directory
sPropertiesName	eg. DS_DI.properties file name
sConversionFileName	eg. odb2Li-Ledia.txt plotstamps conversion definition file

戻り値:

`true` if the files have been correctly loaded; otherwise `false`

**ObjectList CU9000OrderPlotstamps (Object[] *plotstamps,*
String *sLevel,*
String *sAtLevel,*
String *sStart,*
String *sOrder*
)**

Reorder specific Plotstamps at given level.

引数:

plotstamps	the Object Array of the all plotstamps.
sLevel	the level of the plotstamps to be reordered (usually "pcb", "array").
sAtLevel	reordering may be related to given level (usually "panel" or "array").
sStart	may be one of "TL" - for top left, "TR" - top right, "BL" - bottom left, "BR" - bottom right
sOrder	may be one of "XX" - row ordered in the same orientation, "YY" - columns are ordered in the same direction, "XY" - rows are ordered zigzag, "YZ" - columns are ordered zigzag.

boolean CU9000Output (String *machine*)

Perform CU9000 output of active layers.

引数:

machine name of target machine

戻り値:

false if a problem was detected during output or true otherwise

void CU9000SaveBPIs ( )

Saves BPI for current front layer and back layer if exists. Layer activity is also taken into account

void CU9000SaveLocalAlignmentPoints (String *sOutputFilePath*)

Saves local alignment points from current layer. The local alignment points are taken from BPI linked to current layer

引数:

sOutputFilePath A full output file name. Eg. "T:¥¥CU9000job¥¥F_LOCAL_ALN_MARK.TXT"

**void CU9000SaveLocalAlignmentPoints (String *sOutputFilePath,*
boolean *bShareAlignmentMarks*
)**

Saves local alignment points from this layer data structure

引数:

sOutputFilePath A full output file name. Eg."T:¥¥CU9000job¥¥F_LOCAL_ALN_MARK.TXT"
bShareAlignmentMarks the alignment marks are shared with other areas if `true` and areas are set.

boolean CU9000SetParameters (String *xmlFile*)

Apply parameters for CU9000 from XML file to the active layers

引数:

xmlFile full path to XML file containing the settings to be applied. The XML file must conform to the DI_Settings schema

戻り値:

false if the file was found to be invalid or if a problem occurred during applying the settings, or true otherwise

void cutToClipboard ( )

Plane 1 のレイヤにある（選択された）オブジェクト全てを削除する　オブジェクトはクリップボードに保管される

**void dbBoolean (String *dbKey,*
Boolean *bValue*
)**

HOME Ucam.dbファイルを変更する 任意のキーの値を設定する

引数:

dbKey 追加又は変更されたUcam.db キー
bValue Ucam.db キーの値

boolean dbBoolean (String *dbKey*)

Returns boolean Ucam.db value of the given key

引数:

dbKey Ucam.db key

戻り値:

boolean Ucam.db value of the given key or false if the key is not defined

boolean dbBooleanDef (String *dbKey*,
boolean *bDefault*
)

Returns boolean Ucam.db value of the given key

引数:

dbKey Ucam.db key
bDefault default value is returned if given key doesn't exist

戻り値:

boolean Ucam.db value of the given key or false if the key is not defined

void dbDouble (String *dbKey*,
Boolean *dValue*
)

HOME Ucam.dbファイルを変更する 任意のキーの値を設定する

引数:

dbKey 追加又は変更されたUcam.db キー
dValue Ucam.db キーの値

double dbDouble (String *dbKey*)

任意のキーのdouble型Ucam.db値を返す

引数:

dbKey Ucam.db キー

戻り値:

任意のキーのDouble型Ucam.db値

double dbDoubleDef (String *dbKey*,
double *dDefault*
)

Returns double Ucam.db value of the given key

引数:
dbKey Ucam.db key
dDefault default value is returned if given key doesn't exist

戻り値:
double Ucam.db value of the given key

void dbInteger (String *dbKey,*
Integer *iValue*
)

HOME Ucam.dbファイルを変更する 人気のキーの値を設定する

引数:
dbKey 追加又は変更されたUcam.db キー
iValue Ucam.db キーの値

int dbInteger (String *dbKey*)

任意のキーのinteger型Ucam.db値を返す

引数:
dbKey Ucam.dbキー

戻り値:
任意のキーのInteger型Ucam.db値

int dbIntegerDef (String *dbKey,*
int *iDefault*
)

Returns integer Ucam.db value of the given key

引数:
dbKey Ucam.db key
iDefault default value is returned if given key doesn't exist

戻り値:
integer Ucam.db value of the given key

void dbPath (String *dbKey,*
String *sPath*
)

HOME Ucam.dbファイルを変更する 任意のキーの値を設定する

引数:
dbKey 追加又は変更されたUcam.db キー
sPath Ucam.db キーの値

String dbPath (String *dbKey*)

任意のキーのPath Ucam.db値を返す

引数:
dbKey Ucam.dbキー

戻り値:
任意のキーのPath Ucam.db値

String dbPathDef (String *dbKey*,
String *sDefault*
)

Returns Path Ucam.db value of the given key

引数:
dbKey Ucam.db key
sDefault default value is returned if given key doesn't exist

戻り値:
Path Ucam.db value of the given key

void dbString (String *dbKey*,
String *sValue*
)

HOME Ucam.dbファイルを変更する 任意のキーの値を設定する

引数:
dbKey 追加又は変更されたUcam.db キー
sValue Ucam.db キーの値

String dbString (String *dbKey*)

任意のキーのString型Ucam.db値を返す

引数:
dbKey Ucam.dbキー

戻り値:
任意のキーのString型Ucam.db値

String dbStringDef (String *dbKey*,
String *sDefault*
)

Returns String Ucam.db value of the given key

引数:
dbKey Ucam.db key
sDefault default value is returned if given key doesn't exist

戻り値:
String Ucam.db value of the given key

void dbUnitValue (String *dbKey,*
String *sValue*
)

HOME Ucam.dbファイルを変更する 任意のキーの値を設定する

引数:
dbKey 追加又は変更されたUcam.db キー
sValue Ucam.db キーの値（例："1 mil"...）

void dbUnitValue (String *dbKey,*
Double *dValue*
)

HOME Ucam.dbファイルを変更する 任意のキーの値を設定する

引数:
dbKey 追加又は変更されたUcam.db キー
dValue Ucam.db キーの値

double dbUnitValue (String *dbKey*)

任意のキーのUnit Ucam.db 値を返す

引数:
dbKey Ucam.dbキー

戻り値:
任意のキーのUnit Ucam.db 値

double dbUnitValueDef (String *dbKey,*
double *dDefault*
)

Returns Unit Ucam.db value of the given key

引数:
dbKey Ucam.db key
dDefault default value is returned if given key doesn't exist

戻り値:
Unit Ucam.db value of the given key

**double dbUnitValueDef (String *dbKey,*
String *sDefault*
)**

Returns Unit Ucam.db value of the given key

引数:

dbKey Ucam.db key
sDefault default value is returned if given key doesn't exist

戻り値:

Unit Ucam.db value of the given key

void defaultOrder ( )

Displays the current order of the rout path in a separate layer. Each chain of draws and arcs makes up a rout group which gets a sequence number. This number can be changed to change the rout order of the groups.

**void defineFirst (double *p_x,*
double *p_y*
)**

Defines start point of the chain. It is the from point of the closest element to defined point. The point is defined as X and Y world coordinates.

引数:

p_x (X coordinate) the new start poiunt
p_y (Y coordinate) the new start poiunt

void defineFirst (Point *p*)

Defines start point of the chain. It is the from point of the closest element to defined point. The point is defined as X and Y world coordinates.

引数:

p the new start poiunt

**void defineGroup (double *p_x,*
double *p_y,*
int *iGroupNumber*
)**

Defines group at given point with the given index. The point is defined as X and Y world coordinates.

引数:

p_x (X coordinate) the point
p_y (Y coordinate) the point
iGroupNumber the index of the defined group

void defineGroup (Point *p*,
int *iGroupNumber*
)

Defines group at given point with the given index. The point is defined as X and Y world coordinates.

引数:

p	the point
iGroupNumber	the index of the defined group

void defineSelectedGroup ( )

選択したエレメントからグループを定義する

void delete ( )

アクティブレイヤ上の全ての（選択された）オブジェクトを削除する

参照:

deleteWithApe()

void deleteAllCFMEEAlignmentPoints ( )

Remove all alignment points

void deleteAllRefPoints (boolean *bOnAllActiveLay*)

Delete all reference points from layer

引数:

bOnAllActiveLay	if true it work on all active layers otherwise only on active loaded layer in plane 1

void deleteAllYsphotechAlignmentPoints (int *region*)

Remove all alignment point from a region

引数:

region	the regionnumber (0 for global)

void deleteAperture ( )

アパーチャマネジャー： カレントアパーチャを削除

void deleteApertureAttribute (String *sAttributeName*)

アパーチャマネジャー： カレントアパーチャの属性を削除

引数:

sAttributeName 削除する属性の名

void deleteCFMEEAlignmentPoint (int *point*)

Remove all alignment points

引数:

point the alignment point number

void deleteDouble ( )

Deletes overlapping parts of the draws. When two draws partially overlap the overlapping part of the draw is deleted. When two draws completely overlap the shortest draw is deleted.

void deleteLayerByClass (String *className,*
String *subClass,*
int *num,*
String *side*
)

Delete Layer by Class

引数:

className	The class of the layer wanted : "layer", "drill", "core" or "extra".
subClass	The subclass for the layer wanted. The default subclasses offered by Ucam are: "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".
num	The layer number. For "layer" and "drill" the layers are numbered from 1 to the number of layers for that class. For "extra" the layers are numbered per subclass.
side	Specifies the attachment for "extra" layers. Can be "top", "bottom", or "none"

void deleteLayersByActivation (boolean *active*)

アクティブ／非アクティブレイヤを削除する

引数:

active アクティブレイヤの削除はTrue 非アクティブレイヤの削除はfalseを設定する

void deleteLayersByName (String *layName*)

Delete Layers by name

引数:

layName Name (or name part) of any layer (layer, extra or drill)

参照:
deleteLayersByNames(String)

void deleteLayersByNames (String *layNames*)

Delete Layers by names **Example:**

- "test_txt" exact name
- "*_txt" or all layers with name ending "_txt"
- "tmp_*;*_tmp;_tmp_?" or semicolon separated list of wild cards

引数:
layNames semicolon separated list of the wild cards.

参照:
deleteLayersByName(String)

void deleteLayersByPlane (int *plane*)

プレーンによりレイヤ削除する

引数:
plane プレーン番号

void deleteRefPoint (int *iIndex,*
boolean *bOnAllActiveLay*
)

Delete refpoint from layer

引数:
iIndex The reference point number.
bOnAllActiveLay if true it work on all active layers otherwise only on active loaded layer in plane 1

void deleteRefPoints (ObjectList *Indexes,*
boolean *bOnAllActiveLay*
)

Delete reference points from layer

引数:
Indexes list of reference points numbers.
bOnAllActiveLay if true it work on all active layers otherwise only on active loaded layer in plane 1

void deleteSubJob (int *index,*
int *level*

)

サブジョブを削除する

引数:

index index - 任意のインデックスをもつサブジョブに作用する
level index - そのインデックスにおいて サブジョブの定義されたレベルを削除する

void deleteTrueObjects ( )

deletes all True Objects contained in a given job

void deleteWithApe ( )

アクティブレイヤ上の全ての（選択された）オブジェクトを削除し GUIで使用していないアパーチャを削除する GUIで使用していないアパーチャを削除する

参照:

delete()

void deleteWorkspace (String *sWorkspaceName*)

Delete workspace layout file with a given name. NOTE: The same as menu command Workspaces > Delete... > Delete

引数:

sWorkspaceName

void deleteYsphotechAlignmentPoint (int *region*,
int *point*
)

Remove all alignment point from a region

引数:

region the regionnumber (0 for global)
point

void deleteZeroLengthDraws (double *dMaxLength*,
boolean *bFunctional*,
boolean *bNonFunctional*
)

Delete Zero Length Draws and Arcs

引数:

dMaxLength Maximum length of objects to be deleted
bFunctional Delete objects within functional copper if true

bNonFunctional Delete objects within non-functional copper if true

void deselectAll ( )

deselectAll 全てのオブジェクトの非選択にする

void deselectAllApertures ( )

アパーチャマネジャー： アパーチャリストのアパーチャオブジェクトを全て非選択

void deselectAperture (ObjectList *apeIndexArray*)

アパーチャマネジャー: アパーチャオブジェクトの非選択

引数:

apeIndexArray カレントレイヤ上のアパーチャのインデックス配列

void deselectAperture ( )

アパーチャマネジャー: カレントアパーチャのオブジェクトを非選択

**void deselectObjectAttribute (String *sAttrName,*
String *sAttrValue*
)**

非推奨:

deselectObjectAttribute 任意の属性名および属性値を持つオブジェクトを非選択にする

引数:

sAttrName オブジェクト属性名
sAttrValue オブジェクトの属性値

void deselectObjectAttribute (String *sAttrName*)

非推奨:

deselectObjectAttribute 任意の属性名をもつオブジェクトをカレントジョブで非選択にする

引数:

sAttrName オブジェクト属性名

**void deselectObjectByAttribute (String *sAttrName,*
String *sAttrValue*
)**

deselectObjectAttribute deselect objects with attribute with the given name and value from current job.

引数:

sAttrName The object attribute name
sAttrValue The object attribute value

void deselectObjectByAttribute (String *sAttrName*)

deselectObjectAttribute deselect objects with attribute with the given name from current job.

引数:

sAttrName The object attribute name

boolean detectPCBOutlines (String *sLayName,*
String *sParams*
)

Detects PCB Outlines in current job

引数:

sLayName name of a new layer
sParams comma separated paremeters

戻り値:

`true` if OK, otherwise `false` when something fails

boolean DetectPlaceHolders (String *sHandlerName,*
String *sParams*
)

The function detects all placeholders using customer's Handler.

引数:

sHandlerName Handler's name is the same as the name of registered Handler
sParams can be `null` or empty if it has to use default parameters. Given parameters are handled in specific Handler constructor.

戻り値:

`true` when a layer with placeholder has been created and added to the Job, otherwise it returns `false`

boolean DetectPlaceHoldersAtLayer (String *sHandlerName,*
String *sParams*
)

The function detects all placeholders using customer's Handler.

引数:

sHandlerName Handler's name is the same as the name of registered Handler
sParams can be `null` or empty if it has to use default parameters. Given parameters are handled in specific Handler constructor.

戻り値:

`true` when a layer with placeholder has been created and added to the Job, otherwise it returns `false`

```
void dimensioning ( String      sType,
                    double      dApertureSize,
                    ObjectList  oPoints,
                    boolean     bShowErrors,
                    double      dArrowHeadWidth,
                    double      dArrowHeadHeight,
                    double      dRuleToElement,
                    double      dRuleToDimLine,
                    double      dTextToDimLine,
                    double      dTextHeight,
                    double      dTextWidth,
                    double      dTolerancePos,
                    double      dToleranceNeg,
                    double      dToleranceScale,
                    int         iFormat,
                    boolean     bProjectionHorizontal,
                    boolean     bProjectionVertical,
                    String      sFontName,
                    int         iFontStyle,
                    int         iFontSize,
                    String      sLabel
                  )
```

ポイントリストを使用しているレイヤにおいて 任意のタイプのオブジェクトを挿入する

引数:

sType	オブジェクトタイプの寸法
dApertureSize	アパーチャサイズ
oPoints	ポイント配列
bShowErrors	trueの場合はエラー表示 falseの場合はエラー非表示 プレビューモードではFalseを使用
dArrowHeadWidth	矢印先端部の幅
dArrowHeadHeight	矢印先端部の高さ
dRuleToElement	基準となるドロー（draw）と測定ポイント間の距離
dRuleToDimLine	矢印ラインのテキスト配置ルール
dTextToDimLine	テキストから矢印ラインまでの距離
dTextHeight	テキスト高さ
dTextWidth	テキスト幅 フォントが定義されている場合は この値は無視される
dTolerancePos	正の許容値 正および負の許容値がいずれも0.0の場合 寸法公差ラベルは表示されない
dToleranceNeg	負の許容値 正および負の許容値がいずれも0.0の場合 寸法公差ラベルは表示されない
dToleranceScale	寸法公差テキストの比率
iFormat	使用する小数点以下の桁数
bProjectionHorizontal	trueの場合 水平方向の距離が表示されるが 角度距離は表示されない
bProjectionVertical	trueの場合 垂直方向の距離が表示されるが 角度距離は表示されない
sFontName	フォント名 java.awt.Font constructorを参照
iFontStyle	フォントスタイル java.awt.Font constructorを参照
iFontSize	java.awt.Font constructor

フォントサイズ　　を参照
sLabel　　ラベルに表示するテキスト

String direction ( )

Get the Move/Copy direction value

戻り値:
"h" for horizontal, "v" for vertical or "" for free

void direction (String *sDirection*)

Set the Move/Copy direction value

引数:
sDirection "h" for horizontal, "v" for vertical or "" for free

void distance (double *distance*)

距離値を設定する

引数:
distance 距離の値

double distance ( )

距離の数値を取得する

戻り値:
距離値

boolean distort (double *x*,
double *y*,
double *pCenter_x*,
double *pCenter_y*
)

Distortとは: レイヤオブジェクトのXおよび／もしくはY座標で歪み補正値を設定する フラッシュやドロー座標のみに作用し パッドサイズには作用しない ブロックアパーチャに対しては ブロック内のデータおよびブロックフラッシュポイントに対し歪み補正が加えられる ブロックオプションは考慮されないので 回転やミラーなどのブロックオプションを行う場合には注意が必要である ここでの歪み補正は全てのアクティブレイヤおよびレイヤ上の選択されたオブジェクトに作用する

引数:
x - 乗算値 X座標の歪み補正量
y - 乗算値 Y座標の歪み補正量
pCenter_x （X座標） 使用する中心ポイント
pCenter_y （Y座標） 使用する中心ポイント

戻り値:
歪みに問題があればtrue 問題がなければfalse

boolean distort (double *x*,
double *y*,
Point *pCenter*
)

Distortとは: レイヤのオブジェクトのXおよび／もしくはY座標で歪み補正値を設定する フラッシュやドロー座標のみに作用し パッドサイズには作用しない ブロックアパーチャに対しては ブロック内のデータおよびブロックフラッシュポイントに対し歪み補正が加えられる ブロックオプションは考慮されないので 回転やミラーなどのブロックオプションを行う場合には注意が必要である ここでの歪み補正は全てのアクティブレイヤおよびレイヤ上の選択されたオブジェクトに作用する

引数:
x - 乗算値 X座標の歪み補正量
y - 乗算値 Y座標の歪み補正量
pCenter 使用する中心ポイント

戻り値:
問題があればtrue 問題がなければfalse

boolean distort (double *x*,
double *y*
)

Distortとは: レイヤのオブジェクトのXおよび／もしくはY座標で歪み補正値を設定する フラッシュやドロー座標のみに作用し パッドサイズには作用しない ブロックアパーチャに対しては ブロック内のデータおよびブロックフラッシュポイントに対し歪み補正が加えられる ブロックオプションは考慮されないので 回転やミラーなどのブロックオプションを行う場合には注意が必要である ここでの歪み補正は全てのアクティブレイヤおよびレイヤ上の選択されたオブジェクトに作用する

引数:
x - 乗算値 X座標の歪み量
y - 乗算値 Y座標の歪み量

戻り値:
問題があればtrue 問題がなければfalse

void doActiveFunction ( )

DO function Execute current function using values entered in the Numbers **Example:**

```
setInPlane(1,1);
doMove(Point(100,200));
doActiveFunction();
doActiveFunction();
doActiveFunction();
doActiveFunction();
doCancelActiveFunction();
```

参照:
doMove(Upoint)

doCopy(Upoint)

doCancelActiveFunction()

void doCancelActiveFunction ( )

Cancel active function Resets current "Do" button function in the Numbers

参照:

doActiveFunction()

void doCopy (double *offset_x*,
double *offset_y*
)

Copy using parameters. doCopy respects current direction setting. **Example:**

```
setInPlane(1,1);
direction("");
copy(100,200);
doCopy(100,200);
doCopy(100,200);

direction("h");
copy(100,200);
doCopy(100,200);
doCopy(100,200);

direction("v");
copy(100,200);
doCopy(100,200);
doCopy(100,200);
```

引数:

offset_x (X coordinate) offset vector

参照:

copy(Upoint)

direction(String)

doActiveFunction()

doCancelActiveFunction()

引数:

offset_y (Y coordinate) offset vector

参照:

copy(Upoint)

direction(String)

doActiveFunction()

doCancelActiveFunction()

void doCopy (Point *offset*)

Copy using parameters. doCopy respects current direction setting. **Example:**

```
setInPlane(1,1);
direction("");
copy(100,200);
doCopy(100,200);
doCopy(100,200);

direction("h");
copy(100,200);
doCopy(100,200);
doCopy(100,200);

direction("v");
copy(100,200);
doCopy(100,200);
doCopy(100,200);
```

引数:

offset offset vector

参照:

copy(Upoint)

direction(String)

doActiveFunction()

doCancelActiveFunction()

void doMove (double *offset_x,*
double *offset_y*
)

Move using parameters. doMove respects current direction setting. **Example:**

```
setInPlane(1,1);
direction("");
move(100,200,false);
doMove(100,200);
doMove(100,200);

direction("h");
move(100,200,false);
doMove(100,200);
doMove(100,200);

direction("v");
move(100,200,false);
doMove(100,200);
doMove(100,200);
```

引数:

offset_x (X coordinate) offset vector

参照:

move(Upoint, boolean)

direction(String)

doActiveFunction()

doCancelActiveFunction()

引数:

offset_y (Y coordinate) offset vector

参照:

move(Upoint, boolean)

direction(String)

doActiveFunction()

doCancelActiveFunction()

void doMove (Point *offset*)

Move using parameters. doMove respects current direction setting. **Example:**

```
setInPlane(1,1);
direction("");
move(100,200,false);
doMove(100,200);
doMove(100,200);

direction("h");
move(100,200,false);
doMove(100,200);
doMove(100,200);

direction("v");
move(100,200,false);
doMove(100,200);
doMove(100,200);
```

引数:

offset offset vector

参照:

move(Upoint, boolean)

direction(String)

doActiveFunction()

doCancelActiveFunction()

void doOption (String *sOption*)

カレントオペレーションモードを設定

引数:

sOption - "sel" "all" "selall"のいずれか

String doOption ( )

カレントオペレーションモードを取得

戻り値:

モード("sel" "all" "selall"のいずれか)

void doRemoveAttribute (boolean *jobAttr,*
boolean *layAttr,*
boolean *apeAttr,*
boolean *objAttr*

```
                    )
```

remove all attributes from job, layer, aperture, object

引数:

jobAttr	if true remove attributes for job
layAttr	if true remove attributes for layer
apeAttr	if true remove attributes for aperture
objAttr	if true remove attributes for object

```
void drag ( double  clickp_x,
            double  clickp_y,
            double  dRadius,
            double  offset_x,
            double  offset_y,
            double  rect_xmin,
            double  rect_ymin,
            double  rect_xmax,
            double  rect_ymax
          )
```

新ロケーションへパッド／トラックをドラッグする

引数:

clickp_x	（X座標）パッドフラッシュポイント／トラックエンドポイント
clickp_y	（Y座標）パッドフラッシュポイント／トラックエンドポイント
dRadius	オブジェクトがあるべきクリックポイント周りの半径
offset_x	（X座標）新パッドフラッシュポイント／トラックエンドポイントのオフセット
offset_y	（Y座標）新パッドフラッシュポイント／トラックエンドポイントのオフセット
rect_xmin	（長方形の左端 ）選択された長方形−内部の全エレメンツをドラッグする
rect_ymin	（長方形のボトム境界 ）選択された長方形−内部の全エレメンツをドラッグする
rect_xmax	（長方形の右端 ）選択された長方形−内部の全エレメンツをドラッグする
rect_ymax	（長方形のトップ境界）選択された長方形−内部の全エレメンツをドラッグする

```
void drag ( Point      clickp,
            double     dRadius,
            Point      offset,
            Rectangle  rect
          )
```

パッド／トラックを新ロケーションへドラッグする

引数:

clickp	フラッシュポイント／トラックエンドポイント
dRadius	オブジェクトがあるべきクリックポイント周りの半径
offset	新パッドフラッシュポイント／トラックエンドポイントのオフセット
rect	選択された長方形−内部の全エレメンツをドラッグする

```
void drag ( double  clickp_x,
```

```
            double  clickp_y,
            double  dRadius,
            double  offset_x,
            double  offset_y
          )
```

パッド／トラックを新ロケーションへドラッグする

引数:

clickp_x （X座標）パッドフラッシュポイント／トラックエンドポイント
clickp_y （Y座標）パッドフラッシュポイント／トラックエンドポイント
dRadius オブジェクトがあるべきクリックポイント周りの半径
offset_x （X座標）新パッドフラッシュポイント／トラックエンドポイントのオフセット
offset_y （Y座標）新パッドフラッシュポイント／トラックエンドポイントのオフセット

```
void drag ( Point   clickp,
            double  dRadius,
            Point   offset
          )
```

新ロケーションへパッド／トラックをドラッグ

引数:

clickp パッドフラッシュポイント／トラックエンドポイント
dRadius オブジェクトがあるべきクリックポイント周りの半径
offset 新パッドフラッシュポイント／トラックエンドポイントのオフセット

```
void dragAngle ( double   pt_x,
                 double   pt_y,
                 double   dRadius,
                 double   dist,
                 double   mlen,
                 boolean  bUseLimit
               )
```

Drag track angle corresponds to: Transform Objects - BGA Tracks - Drag Angle

引数:

pt_x (X coordinate) Start point of drag
pt_y (Y coordinate) Start point of drag
dRadius radius around click point where must be an object
dist Drag distance
mlen Minimum track length
bUseLimit use limit for move data

```
void dragAngle ( Point   pt,
                 double  dRadius,
                 double  dist,
                 double  mlen,
```

```
                boolean bUseLimit
            )
```

Drag track angle corresponds to: Transform Objects - BGA Tracks - Drag Angle

引数:

pt	Start point of drag
dRadius	radius around click point where must be an object
dist	Drag distance
mlen	Minimum track length
bUseLimit	use limit for move data

```
void dragAngle ( double pt_x,
                 double pt_y,
                 double dRadius,
                 double dist,
                 double mlen
               )
```

このドラッグトラック角度の設定は オブジェクト変換ダイアログ（Transform Objects）- BGA Tracks - Drag Angle の操作に相当する

引数:

pt_x	（X座標）ドラッグのXスタートポイント
pt_y	（Y座標）ドラッグのYスタートポイント
dRadius	オブジェクトがあるべきクリックポイント周りの半径
dist	ドラッグ距離
mlen	最小トラック長さ

```
void dragAngle ( Point  pt,
                 double dRadius,
                 double dist,
                 double mlen
               )
```

このトラックドラッグ角度の設定は オブジェクト変換ダイアログ（Transform Objects） - BGA Tracks - Drag Angle の操作に相当する

引数:

pt	ドラッグのスタートポイント
dRadius	オブジェクトがあるべきクリックポイント周りの半径
dist	ドラッグ距離
mlen	最小トラック長さ

```
void dragLayer ( String  prevClass,
                 String  newClass,
                 int     prevPosition,
                 int     newPosition,
                 boolean duplicate
```

)

Drag Layer, optionally with duplicate

引数:

prevClass	"layer" or "extra" or "drill"
newClass	"layer" or "extra" or "drill"
prevPosition	index of the layer. For the "drill" layers it is from 1 to the number of drill layers. For the "layer" and "extra" it is from 1 to the number of all layers.
newPosition	index of the new layer. For the "drill" layers it is from 1 to the number of drill layers. For the "layer" and "extra" it is from 1 to the number of all layers.
duplicate	true makes exact copy of the original layer

Line dragLine (String *sLabel*)

Wait for user drag **Line**. It needs Ucam GUI. The function pauses script execution and waits for user interaction.

引数:

sLabel Label in information dialog

戻り値:

Line dragged by user.

例外:

AbortException after user abort

Rectangle dragRectangle (String *sLabel*)

Wait for user drag **Rectangle**. It needs Ucam GUI. The function pauses script execution and waits for user interaction.

引数:

sLabel Label in information dialog

戻り値:

Rectangle dragged by user.

例外:

AbortException after user abort

void drawLastPlanesInFront (boolean *bDo*)

Changes the drawing order of the planes. Planes 6-11 are placed in front of planes 1-5, while it is normally otherways.

引数:

bDo A flag indicating that this should happen, or should be undone.

void drawSlots (String *sDPFApeRef,*
double *dTolerance,*

String *sDPFSlotApe*
)

Replaces defined slots according to the given DPF aperture definition and tolerance with new aperture definition

Example:

```
setMode("selall", "mil", "no");
drawSlots("208=REC,110,90", 0.1, "215=CIR,90");
drawSlots("208=REC,110,90", 0.1, "215=CIR,90");
```

引数:

sDPFApeRef the DPF aperture definition - reference for search
dTolerance the tolerance for searching
sDPFSlotApe the DPF aperture definition - the new slot aperture

void drawSlotsSelect (String *sDPFApe,*
double *dTolerance*
)

Selects slots according to the given DPF aperture definition and tolerance

Example:

```
setMode("selall", "mil", "no");
selectSlots("208=REC,110,90", 0.1);
```

引数:

sDPFApe the DPF aperture definition
dTolerance the tolerance for searching

void drillMapReplace (String *sSymbolFilePath,*
ObjectList *oMappingTable*
)

Drill Mapダイアログからの置換メソッド

引数:

sSymbolFilePath Symbol dpfファイルへのパス
oMappingTable 置換されたアパーチャのテーブル順序 例を参照

順序例(2,0,1):

- 1つ目のドリルアパーチャは シンボルdpfファイルの2番目のアパーチャで置換される
- 2つ目のドリルアパーチャは シンボルdpfファイルのインデックスが0であり 置換されないことを示す
- 3つ目のドリルアパーチャは シンボルdpfファイルの1番目のアパーチャで置換される

void DSAOIAlignmentApply ( )

Method simulates Alignment Apply button press

void DSAOIAlignmentDetect (String *sMachineType,*
String *sObjectRestrictions,*
boolean *bPositive,*
boolean *bNegative,*
double *dMinimumSize,*
double *dMaximumSize*
)

Method simulates Detect button press. Detects alignment points in the active layers

引数:

sMachineType	The selected machine type used to determine the parameters
sObjectRestrictions	A specification for the object restrictions
bPositive	Allow positive objects as alignment points. Only used when a shape is defined
bNegative	Allow negative objects as alignment points. Only used when a shape is defined
dMinimumSize	The minimum size of the alignment point. (-1 means not active)
dMaximumSize	The maximum size of the alignment point. (-1 means not active)

boolean DSAOIApply ( )

Method simulates Apply button press

戻り値:

true if the apply succeeded, false if not. The apply will not succeed if the fields Minimum **Line**, Minimum Space or Thickness are <= 0

void DSAOIAreaDetection (boolean *detectImportantLine,*
boolean *detectImportantSpace,*
boolean *detectImportantClearance,*
boolean *detectImportantDrill,*
boolean *detectImportantFutureDrill,*
boolean *detectImportantMaskOpenings,*
boolean *detectProhibitCopper,*
boolean *detectProhibitSpace,*
boolean *detectProhibitNonfunctionalCopper,*
ObjectList *importantLineWidth,*
Object[] *importantSpaceWidth,*
double *importantClearanceWidth,*
double *importantDrillSpreadValue,*
double *prohibitLineWidth,*
double *prohibitSpaceWidth,*
double *minSliverSize,*
String *pathStrategy,*
boolean *mergeOutput,*
boolean *clipProhibitWithImportant,*
boolean *outputNormalAreas,*
boolean *maskPolarity,*
boolean *paintedArea,*
double *paintedAreaValue,*

```
                    String      PCBName,
                    boolean     outputDrillBinary
                )
```

Auto-detect `important' and `prohibit' areas for the current job

引数:

detectImportantLine	whether or not to detect important areas in copper
detectImportantSpace	whether or not to detect important areas in space
detectImportantClearance	whether or not to detect clearance around drill holes
detectImportantDrill	whether or not to add actual drill holes to the important areas
detectImportantFutureDrill	whether or not to add future drill holes to the important areas
detectImportantMaskOpenings	whether or not to add solder mask openings as important areas
detectProhibitCopper	whether or not to detect prohibit areas in copper
detectProhibitSpace	whether or not to detect prohibit areas in space
detectProhibitNonfunctionalCopper	whether or not to add non-functional copper (NWP) as mask areas
importantLineWidth	maximum reported line width classes in important areas
importantSpaceWidth	maximum reported space width in important areas
importantClearanceWidth	maximum reported copper to drill hole clearance value in important areas
importantDrillSpreadValue	add this ring around the drill holes reported as important areas
prohibitLineWidth	maximum reported line width in prohibit areas
prohibitSpaceWidth	maximum reported space width in prohibit areas
minSliverSize	minimum size for slivers
pathStrategy	how to report functional copper where there are multiple paths ("thickest" or "all")
mergeOutput	whether or not to merge all important and all prohibit layers into just two layers
clipProhibitWithImportant	whether or not to clip the prohibit areas with the important areas
outputNormalAreas	whether or not to create a layer containing the areas which are neither important or prohibit
maskPolarity	if true, mask openings are positive; if false, resist is positive
paintedArea	create extra mask from the painted areas
paintedAreaValue	value used by painted area detection if allowed by "paintedArea"
PCBName	the name of the PCB block to use or null if none specified
outputDrillBinary	generate one binary file per copper layer with the drill information

```
void DSAOIAreaDetection ( boolean     detectImportantLine,
                          boolean     detectImportantSpace,
                          boolean     detectImportantClearance,
                          boolean     detectImportantDrill,
                          boolean     detectImportantFutureDrill,
                          boolean     detectImportantMaskOpenings,
                          boolean     detectProhibitCopper,
                          boolean     detectProhibitSpace,
                          boolean     detectProhibitNonfunctionalCopper,
                          ObjectList  importantLineWidth,
                          Object[]    importantSpaceWidth,
                          double      importantClearanceWidth,
                          double      importantDrillSpreadValue,
                          double      prohibitLineWidth,
                          double      prohibitSpaceWidth,
```

```
                        double      minSliverSize,
                        String      pathStrategy,
                        boolean     mergeOutput,
                        boolean     clipProhibitWithImportant,
                        boolean     outputNormalAreas,
                        boolean     maskPolarity,
                        String      PCBName,
                        boolean     outputDrillBinary
                    )
```

Auto-detect `important' and `prohibit' areas for the current job

引数:

detectImportantLine	whether or not to detect important areas in copper
detectImportantSpace	whether or not to detect important areas in space
detectImportantClearance	whether or not to detect clearance around drill holes
detectImportantDrill	whether or not to add actual drill holes to the important areas
detectImportantFutureDrill	whether or not to add future drill holes to the important areas
detectImportantMaskOpenings	whether or not to add solder mask openings as important areas
detectProhibitCopper	whether or not to detect prohibit areas in copper
detectProhibitSpace	whether or not to detect prohibit areas in space
detectProhibitNonfunctionalCopper	whether or not to add non-functional copper (NWP) as mask areas
importantLineWidth	maximum reported line width classes in important areas
importantSpaceWidth	maximum reported space width in important areas
importantClearanceWidth	maximum reported copper to drill hole clearance value in important areas
importantDrillSpreadValue	add this ring around the drill holes reported as important areas
prohibitLineWidth	maximum reported line width in prohibit areas
prohibitSpaceWidth	maximum reported space width in prohibit areas
minSliverSize	minimum size for slivers
pathStrategy	how to report functional copper where there are multiple paths ("thickest" or "all")
mergeOutput	whether or not to merge all important and all prohibit layers into just two layers
clipProhibitWithImportant	whether or not to clip the prohibit areas with the important areas
outputNormalAreas	whether or not to create a layer containing the areas which are neither important or prohibit
maskPolarity	if true, mask openings are positive; if false, resist is positive
PCBName	the name of the PCB block to use or null if none specified
outputDrillBinary	generate one binary file per copper layer with the drill information

```
void DSAOIAreaDetection ( boolean     detectImportantLine,
                          boolean     detectImportantSpace,
                          boolean     detectImportantClearance,
                          boolean     detectImportantDrill,
                          boolean     detectImportantFutureDrill,
                          boolean     detectImportantMaskOpenings,
                          boolean     detectProhibitCopper,
                          boolean     detectProhibitSpace,
                          boolean     detectProhibitNonfunctionalCopper,
                          ObjectList  importantLineWidth,
                          Object[]    importantSpaceWidth,
```

```
                    double      importantClearanceWidth,
                    double      importantDrillSpreadValue,
                    double      prohibitLineWidth,
                    double      prohibitSpaceWidth,
                    double      minSliverSize,
                    String      pathStrategy,
                    boolean     mergeOutput,
                    boolean     clipProhibitWithImportant,
                    boolean     outputNormalAreas,
                    boolean     maskPolarity,
                    String      PCBName
                )
```

Auto-detect `important' and `prohibit' areas for the current job

引数:

detectImportantLine	whether or not to detect important areas in copper
detectImportantSpace	whether or not to detect important areas in space
detectImportantClearance	whether or not to detect clearance around drill holes
detectImportantDrill	whether or not to add actual drill holes to the important areas
detectImportantFutureDrill	whether or not to add future drill holes to the important areas
detectImportantMaskOpenings	whether or not to add solder mask openings as important areas
detectProhibitCopper	whether or not to detect prohibit areas in copper
detectProhibitSpace	whether or not to detect prohibit areas in space
detectProhibitNonfunctionalCopper	whether or not to add non-functional copper (NWP) as mask areas
importantLineWidth	maximum reported line width classes in important areas
importantSpaceWidth	maximum reported space width in important areas
importantClearanceWidth	maximum reported copper to drill hole clearance value in important areas
importantDrillSpreadValue	add this ring around the drill holes reported as important areas
prohibitLineWidth	maximum reported line width in prohibit areas
prohibitSpaceWidth	maximum reported space width in prohibit areas
minSliverSize	minimum size for slivers
pathStrategy	how to report functional copper where there are multiple paths ("thickest" or "all")
mergeOutput	whether or not to merge all important and all prohibit layers into just two layers
clipProhibitWithImportant	whether or not to clip the prohibit areas with the important areas
outputNormalAreas	whether or not to create a layer containing the areas which are neither important or prohibit
maskPolarity	if true, mask openings are positive; if false, resist is positive
PCBName	the name of the PCB block to use or null if none specified

```
void DSAOIAreaDetection ( boolean     detectImportantLine,
                          boolean     detectImportantSpace,
                          boolean     detectImportantClearance,
                          boolean     detectImportantDrill,
                          boolean     detectImportantFutureDrill,
                          boolean     detectImportantMaskOpenings,
                          boolean     detectProhibitCopper,
                          boolean     detectProhibitSpace,
                          boolean     detectProhibitNonfunctionalCopper,
```

```
                    ObjectList  importantLineWidth,
                    double      importantSpaceWidth,
                    double      importantClearanceWidth,
                    double      importantDrillSpreadValue,
                    double      prohibitLineWidth,
                    double      prohibitSpaceWidth,
                    double      minSliverSize,
                    String      pathStrategy,
                    boolean     mergeOutput,
                    boolean     clipProhibitWithImportant,
                    boolean     outputNormalAreas,
                    boolean     maskPolarity,
                    String      PCBName
                )
```

Auto-detect `important' and `prohibit' areas for the current job

引数:

detectImportantLine	whether or not to detect important areas in copper
detectImportantSpace	whether or not to detect important areas in space
detectImportantClearance	whether or not to detect clearance around drill holes
detectImportantDrill	whether or not to add actual drill holes to the important areas
detectImportantFutureDrill	whether or not to add future drill holes to the important areas
detectImportantMaskOpenings	whether or not to add solder mask openings as important areas
detectProhibitCopper	whether or not to detect prohibit areas in copper
detectProhibitSpace	whether or not to detect prohibit areas in space
detectProhibitNonfunctionalCopper	whether or not to add non-functional copper (NWP) as mask areas
importantLineWidth	maximum reported line width classes in important areas
importantSpaceWidth	maximum reported space width in important areas
importantClearanceWidth	maximum reported copper to drill hole clearance value in important areas
importantDrillSpreadValue	add this ring around the drill holes reported as important areas
prohibitLineWidth	maximum reported line width in prohibit areas
prohibitSpaceWidth	maximum reported space width in prohibit areas
minSliverSize	minimum size for slivers
pathStrategy	how to report functional copper where there are multiple paths ("thickest" or "all")
mergeOutput	whether or not to merge all important and all prohibit layers into just two layers
clipProhibitWithImportant	whether or not to clip the prohibit areas with the important areas
outputNormalAreas	whether or not to create a layer containing the areas which are neither important or prohibit
maskPolarity	if true, mask openings are positive; if false, resist is positive
PCBName	the name of the PCB block to use or null if none specified

void DSAOIAreasApply ( )

Method simulates Areas Apply button press

```
void DSAOIDetectRectangularArea ( double  dMargin,
                                  String  sBlockMode,
```

String *sPCBName,*
String *sReferenceLayerName,*
boolean *bIsSingleLevel,*
boolean *bIsInspection,*
boolean *bIsPmiP1,*
boolean *bIsPmiP2,*
boolean *bIsDrcP1,*
boolean *bIsDrcP2*
)

Automatically generate inspection areas for all active layers

引数:

dMargin	The found areas are spread with the given margin value
sBlockMode	Defines the reference layer for block detection. Possible values are BLOCK_OUTLINE, BLOCK_LAYER and BLOCK_REFERENCE_LAYER
sPCBName	The name of the PCB block to use or null if none specified
sReferenceLayerName	The name of the layer to be used if BLOCK_REFERENCE_LAYER mode is selected or null if none specified
bIsSingleLevel	When true, areas are generated for all active layers. When false, job level areas are generated
bIsInspection	When true, inspection areas are generated. When false, mask areas are generated
bIsPmiP1	Defines the PMI P1 setting for the generated areas
bIsPmiP2	Defines the PMI P2 setting for the generated areas
bIsDrcP1	Defines the DRC P1 setting for the generated areas
bIsDrcP2	Defines the DRC P2 setting for the generated areas

void DSAOILayerList ( )

Method simulates LayerList button press

void DSAOILayerListAreaApply ( )

Method simulates LayerList Area Apply button press

void DSAOILayerListAreaOutput ( )

Method simulates LayerList Area Output button press

void DSAOILayerListAreaSelect ( )

Method simulates LayerList Area Select button press

void DSAOILayerListGroupValue (String *sNewValue*)

Specifies Group in LayerList

引数:

sNewValue new value of the group

void DSAOILayerListRowDeselect (int *iIndexFrom,*
int *iIndexTo*
)

Method deselects rows in the LayerList between indexes (the first line has index 1)

引数:

iIndexFrom begin row index
iIndexTo end row index

void DSAOILayerListRowDeselect (int *iRow*)

Method deselects row in the LayerList (the first line has index 1)

引数:

iRow

void DSAOILayerListRowSelect (int *iIndexFrom,*
int *iIndexTo*
)

Method selects rows in the LayerList between indexes (the first line has index 1)

引数:

iIndexFrom begin row index
iIndexTo end row index

void DSAOILayerListRowSelect (int *iRow*)

Method selects row in the LayerList (the first line has index 1)

引数:

iRow

boolean DSAOILoadLayerListProfile (String *pro*)

Load a DS PI layer list profile

引数:

pro profile

戻り値:

false if the profile was not loaded or true otherwise

boolean DSAOIOutput ( )

Perform DS PI output of the red layer and its back layer, if it is active

戻り値:
false if a problem was detected during output or true otherwise

**void DSAOIpinpointDetection (double *dStep,*
double *dInfinity,*
String *sOutputFilePath*
)**

DS107 pinpoint detection

引数:

dStep	step between pinpoint locations
dInfinity	value bigger then specified are not important
sOutputFilePath	the complete file path including the file name

**void DSAOIpinpointDetection (double *dStep,*
double *dInfinity,*
String *sPCBName,*
String *sOutputFilePath,*
String *sLocation*
)**

DS107 pinpoint detection (in red and green layers)

引数:

dStep	step between pinpoint locations
dInfinity	value bigger then specified are not important
sPCBName	PCB name, can be "$" (complete layer), null or "" ("deepest level"), single PCB name or list of PCBs separated by ';'
sOutputFilePath	the complete file path including the file name
sLocation	

void DSAOIPositionApply ( )

Method simulates LayerList Area Apply button press

void DSAOISetApplyToBackLayers (boolean *bValue*)

sets value of the 'Apply To Back Layers' check box

引数:

bValue value can be true (checked) or false (unchecked)

void DSAOISetApplyToFrontLayers (boolean *bValue*)

sets value of the 'Apply To Front Layers' check box

引数:

bValue value can be true (checked) or false (unchecked)

**void DTMCalculate (String *sPlatingType,*
String *sToleranceScript*
)**

Calculates drill sizes according to given plating type and given tolerance VHS script

引数:

sPlatingType plating type, one from the Tooltable name
sToleranceScript tolerance script file name

boolean DTMCreateSymbolDrawing ( )

Creates layer with symbols and symbol table

戻り値:

`true` if the Symbol drawings successfully created, otherwise `false`

boolean DTMLoadData ( )

Loads necessary data from drill layers and sets "uCustomerDiameter" attributes to holes

戻り値:

boolean Activate the SDTMFrame UpdateDpfButton

void DTMLoadToleranceFile (String *sToleranceFileName*)

Loads tolerance file

引数:

sToleranceFileName Smart DrillTool Manager tolerance file name

void DTMRemoveAttributes ( )

Removes all attributes from drills

void DTMSaveDataToAttributes (String *sJobName*)

Saves the table data to attributes

引数:

sJobName

void DTMUpdateDPF (String *sJobName*)

Moves the plated objects to the plated layer and the unplated objects to the unplated layer

引数:

sJobName

void duplicateAperture ( )

アパーチャマネジャー：カレントアパーチャを複製

**void duplicateLayer (String *layName,*
String *newName*
)**

レイヤを複製

引数:

layName いずれかのレイヤ（レイヤ 特殊レイヤ ドリルレイヤ）の名前
newName 複製したレイヤの名前

**void dwAnnotate (String *annotationLayerName,*
String *sChipID*
)**

The annotation layer describes positions of the DW Shots (chips) on panel. Each aperture in annotation layer has one flash point corresponding to one chip flash point in layer in plane 1. The aperture attribute dwShotAnnotation contains chip ID and position output in File A.

引数:

annotationLayerName the layer in job containing annotations
sChipID the chip defining block name

void dwAnnotate (String *annotationLayerName*)

The annotation layer describes positions of the DW Shots (chips) on panel. Each aperture in annotation layer has one flash point corresponding to one chip flash point in layer in plane 1. The aperture attribute dwShotAnnotation contains chip ID and position output in File A.

引数:

annotationLayerName the layer in job containing annotations

void dwApplyTransform (String *sResultFilePath*)

The Result file is the File B with real chip transformations. The method applies the transformations from the File B to layer in plane 1.

引数:

sResultFilePath full result File B path

void editAperture (int *iApeNum,*
String *sApeName,*
String *sApeDef*
)

アパーチャマネジャー: カレントアパーチャの編集

引数:

iApeNum (new) アパーチャ番号
sApeName (new) アパーチャ名
sApeDef (new) DPF形式でのアパーチャのString型定義 例）"REC,1.905,0.3048"

void emptyClipboard ( )

クリップボードを空にする

void enlargePads (String *absRel,*
double *aVal,*
boolean *bExclcon,*
boolean *bUseBGA*
)

絶対値／相対値でパッドを拡大（伸長）させる これは オブジェクト変換ダイアログ（Transform Objects）- BGAパッド - 拡大 に相当する

引数:

absRel Absは絶対値 Relは相対値を示す
aVal 拡大値
bExclcon trueの場合 輪郭も含める
bUseBGA trueの場合 オブジェクト属性がuBGAのオブジェクトのみ 拡大される

String envString (String *name*)

環境変数を返す

引数:

name 例: "TEMP"

戻り値:

任意の環境変数値

void equalizeTrackSpace ( )

トラックスペースを均等にする これは オブジェクト変換ダイアログ（Transform Objects）- BGAトラック - Equalize Spaceの操作に相当する

void etchCompensation (boolean *bUseExcludeAreas,*
boolean *bCreateLayerBackup,*
boolean *bShowLayerBackup,*
int *iOutStyle,*
boolean *bAsymmetricPadTrackComp,*
ObjectList *arrPrefOffset*
)

UseCompensateAreasなしにエッチング補正を行う -> パラメーター設定削減

引数:

bUseExcludeAreas	エッチング補正除外エリアを持つレイヤを使う
bCreateLayerBackup	リファレンスとして全アクティブレイヤをコピーする
bShowLayerBackup	リファレンスレイヤがあれば表示する
iOutStyle	0:デフォルト 1:元のオブジェクトの上に輪郭を出力 2:元のオブジェクトを置換して輪郭を出力
bAsymmetricPadTrackComp	非対称パッドトラックの補正
arrPrefOffset	推奨オフセットのパラメータ配列: • オフセットテーブル（String型） • 最小クリアランス（Double型値） • パッドのDouble型値 • トラックのDouble型値

void etchCompensation (boolean *bUseExcludeAreas,*
boolean *bUseCompensateAreas1,*
boolean *bUseCompensateAreas2,*
boolean *bCreateLayerBackup,*
boolean *bShowLayerBackup,*
int *iOutStyle,*
boolean *bAsymmetricPadTrackComp,*
ObjectList *arrPrefOffset,*
Object[] *arrCompLay1,*
Object[] *arrCompLay2*
)

エッチング補正

引数:

bUseExcludeAreas	エッチング補正除外エリアがあるレイヤを使用する
bUseCompensateAreas1	レイヤ1の補正エリアを使用 -> arrCompLay1にパラメータが必要
bUseCompensateAreas2	レイヤ2の補正エリアを使用 -> arrCompLay2にパラメータが必要
bCreateLayerBackup	リファレンスのために全アクティブレイヤをコピーする
bShowLayerBackup	リファレンスレイヤがあれば表示する
iOutStyle	0:デフォルト 1:元のオブジェクトの上に輪郭を出力 2:元のオブジェクトを置換して輪郭を出力

bAsymmetricPadTrackComp 非対称パッドトラックの補正
arrPrefOffset 推奨オフセットのパラメータ配列:

- オフセットテーブル（String型）
- 最小クリアランス（double型）
- パッド補正値（double型）
- トラック補正値（double型）

arrCompLay1 レイヤ1の補正エリアに対するパラメータ配列

- オフセットテーブル（String型）
- 最小クリアランス（double型）
- パッド補正値（double型）
- トラック補正値（double型）

arrCompLay2 レイヤ2の補正エリアに対するパラメータ配列

- オフセットテーブル（String型）
- 最小クリアランス（double型）
- パッド補正値（double型）
- トラック補正値（double型）

void expandArcs ( )

expandArcs ジョブのドローでアークを置換する アーク自体は削除される

void expandBlock ( )

expandBlock ジョブのブロックアパーチャの全データを展開する ブロックアパーチャ自体はアパーチャリストから削除される

void expandNibble (double *overlapValue,*
double *pitchValue,*
boolean *useOverlap*
)

エレメントをニブルへ展開する

引数:

overlapValue ニブルでのオーバーラップ量
pitchValue ニブルでのピッチ
useOverlap オーバーラップ量を使用する場合はtrue ピッチを使用する場合はfalse

void expandText ( )

expandText 全レイヤの全テキストアパーチャを 文字毎にコンプレックスアパーチャのフラッシュに変更する これは一文字毎の編集を可能にし 文字の拡大・縮小を可能にする

void expandTrueObjects ( )

expandTrueObjects サブオブジェクトを持つフラッシュ ドロー・アーク・リージョンを削除し それらのサブオブジェクトで置き換える

void expandVtx ( )

expandVtx ジョブ中の全ベクトルテキストデータを展開する

void externalLinkManagerCheck ( )

The button Check on External Link Manager press

void filletJoin (double *pt_x,*
double *pt_y,*
double *dis*
)

2つのドローの接合部分を丸める

引数:

pt_x （X座標）2つのドローの接合部分
pt_y （Y座標）2つのドローの接合部分
dis フィレット（丸み）の半径値

void filletJoin (Point *pt,*
double *dis*
)

2つのドローの接合部分を丸める

引数:

pt 2つのドローの接合部分
dis フィレット（丸み）の半径値

void fillPolygon (boolean *bDirection*)

引数:

bDirection direction

void fillPolygonCCW ( )

CCW fill polygon

void fillPolygonCW ( )

CW fill polygon

**void fillWithAngledPattern (String *shape,*
double *size,*
double *pitch,*
double *angle*
)**

Fills selected apertures with angled pattern

引数:

shape Dot shape ("circle", "square" and "diamond")
size Dot size
pitch Pitch between dots
angle Pattern angle

参照:

HyperShell::PATTERN_ANGLED_CIRCLE

HyperShell::PATTERN_ANGLED_SQUARE

HyperShell::PATTERN_ANGLED_DIAMOND

**void fillWithPatternPads (String *sKind,*
boolean *bKeepEdge,*
double *pGridOrigin_x,*
double *pGridOrigin_y,*
double *pGridStep_x,*
double *pGridStep_y*
)**

選択した輪郭を選択したパターンで置き換える — パッドで埋める

引数:

sKind 奇数 - 1つのアクティブレイヤの場合: 輪郭エリアには グリッド上のパッドの間 交互にパターンを発生する 左下のグリッドには輪郭が発生しない

- 2つ以上のアクティブレイヤの場合: パターンがトップからボトムまで交互に発生する 上のアクティブレイヤが奇数としてスタートする アクティブレイヤのみが考慮される 偶数 - 1つのアクティブレイヤの場合: 輪郭エリアには グリッド上のパッドの間 交互にパターンを発生する 左下のグリッドには輪郭が発生する
- 2つ以上のアクティブレイヤの場合: パターンがトップからボトムまで交互に発生する 上のアクティブレイヤが偶数としてスタートする アクティブレイヤのみが考慮される すべて - 全レイヤにパターンを発生

bKeepEdge - trueの場合 縁を保持する
pGridOrigin_x (X 座標) - グリッド原点
pGridOrigin_y (Y 座標) - グリッド原点
pGridStep_x (X 座標) - グリッドステップ
pGridStep_y (Y 座標) - グリッドステップ

```
void fillWithPatternPads ( String    sKind,
                           boolean   bKeepEdge,
                           Point     pGridOrigin,
                           Point     pGridStep
                         )
```

選択した輪郭を選択したパターンで置き換える—パッドで埋める

引数:

sKind 奇数 - 1つのアクティブレイヤの場合: 輪郭エリアには グリッド上のパッドの間 交互にパターンを発生する 左下のグリッドには輪郭が発生しない

- 2つ以上のアクティブレイヤの場合: パターンがトップからボトムまで交互に発生する 上のアクティブレイヤが奇数としてスタートする アクティブレイヤのみが考慮される 偶数 - 1つのアクティブレイヤの場合: 輪郭エリアには グリッド上のパッドの間 交互にパターンを発生する 左下のグリッドには輪郭が発生する
- 2つ以上のアクティブレイヤの場合: パターンがトップからボトムまで交互に発生する 上のアクティブレイヤが偶数としてスタートする アクティブレイヤのみが考慮される すべて - 全レイヤにパターンを発生

bKeepEdge - trueの場合 縁を保持する
pGridOrigin - 原点グリッド
pGridStep - グリッドステップ

```
void fillWithPatternStarburst ( int       iSegments,
                                String    sKind,
                                int       dBlack,
                                boolean   bWithCenter,
                                double    pCenter_x,
                                double    pCenter_y,
                                boolean   bKeepEdge,
                                double    dEdgeWith
                              )
```

選択した輪郭を選択したパターンで置き換える—スターバースト（放射状模様） でventingパターンを埋める

引数:

iSegments - 使用するセグメント(黒のエリア)総数
sKind 奇数 - セグメントのスタート位置を指定する X-軸の正極に第1黒いセグメントを発生する 偶数 - セグメントのスタート位置を指定する X-軸の正極に第1白いセグメントを発生する 交互 - トップからボトムまでの層を交互に発生する 上のアクティブレイヤから偶数のパターンを発生する アクティブレイヤのみが考慮される
dBlack - パターンで埋めるエリアでの黒／白エリア割合指定する
bWithCenter - スターバストの中心点を定義する
pCenter_x (X 座標) - 中心座標
pCenter_y (Y 座標) - 中心座標
bKeepEdge - trueの場合は 縁取りを保持する
dEdgeWith - 縁取りを保持

```
void fillWithPatternStarburst ( int     iSegments,
                                String  sKind,
                                int     dBlack,
                                boolean bWithCenter,
                                Point   pCenter,
                                boolean bKeepEdge,
                                double  dEdgeWith
                              )
```

選択した輪郭を選択したパターンで置き換える―スターバースト（放射状模様） でventingパターンを塗り込む

引数:

iSegments	- 使用するセグメント(黒のエリア)総数
sKind	奇数 - セグメントのスタート位置を指定する　X-軸の正極に第1黒いセグメントを発生する　偶数 - セグメントのスタート位置を指定する　X-軸の正極に第1白いセグメントを発生する　交互 - トップからボトムまでの層を交互に発生する　上のアクティブレイヤから偶数のパターンを発生する　アクティブレイヤのみが考慮される
dBlack	- パターンで埋めるエリアでの黒／白エリア割合指定する
bWithCenter	- Defines the center point of the starburst.
pCenter	- 中心座標
bKeepEdge	- trueの場合は 縁取りを保持する
dEdgeWith	- 縁取りを保持

```
void fillWithPatternTracks ( String  sPattern,
                             double  dStep,
                             double  dWidth,
                             double  dRotation,
                             boolean bKeepEdge
                           )
```

選択した輪郭を選択したパターンで置き換える―トラックでventingパターンを塗り込む

引数:

sPattern	- 必要なパターンを選択する
dStep	- 2ライン間のステップを定義する
dWidth	- ライン幅を定義する （幅<ステップ）
dRotation	- パターンの回転を定義する （CCW）
bKeepEdge	- trueの場合 縁取りを保持する

```
int fillWithVectors ( double dOverlap,
                      double dDiameter,
                      int    iApeCount,
                      int    iApeNum,
                      String sFillOpt
                    )
```

Fills the specified apertures with circular draws.

引数:

dOverlap	Specifies the overlap for the apertures.

dDiameter Specifies the diameter of the fill apertures. The first aperture has a diameter dDiameter. The n-th has a diameter n*dDiameter.

iApeCount The number of fill apertures.

iApeNum The number of the first fill aperture.

sFillOpt Specifies which kind of apertures should be filled up. This can be any combination of "com", "box", "con" and "txt". Boxes, complexes and text apertures are replaced by a block aperture.

戻り値:

number of errors

void findSections (String *szOptions*)

Find job sections

引数:

szOptions options ("sel" or "all" or "selall")

void findSlots ( )

Find "IPCATG" slots according keys that defines them.

int findStandardShape (double *dTolerance,*
String *szOpt,*
String *szAction*
)

Find standard shapes within COM and CON apertures on layer

引数:

dTolerance tolerance

szOpt options "con,com,blo:cir,rec,box,the" con, com or blo means what to consider (blo means go into block definitions), cir and rec means what to replace with, you can also use * as joker -> "*:rec,cir, box" or "*:*" or "con,com:*"

szAction ("select", "replace"), in case of select all regions and/or complexes will be assigned attribute "uStandardShape" with value standard shape defintion

戻り値:

amount of find/replaced apertures, 0 when no, -1 on error

void flashMakerDeleteComplex ( )

検索したペイントオブジェクトリストから 全コンプレックスを削除する フラッシュメーカーのCOM削除

void flashMakerDeselectComplex ( )

検索したペイントオブジェクトリストから 全コンプレックスを非選択にする フラッシュメーカのCOM選択解除

void flashMakerFind ( )

フラッシュを置換するのに適したペイントオブジェクトを全て見つける　フラッシュメーカーの検出ボタン

void flashMakerFindStandardShapes (Uxjob *oJob*)

Finds all REGs and COMs that are suitable to replace with flashes of standard shapes (CIR and REC). FlashMaker Find button for Con&Com tab

引数:

oJob current job

void flashMakerReplace ( )

全ての検索したペイントオブジェクトをフラッシュで置換する　フラッシュメーカーの置換ボタン

void flashMakerReplaceStandardShapes (Uxjob *oJob*)

Replaces all found pianted objects with flashes FlashMaker Replace button

引数:

oJob current job

void flashMakerSetup (double *minCutoff,*
double *minSize,*
double *maxSize,*
boolean *useTol,*
double *tol,*
boolean *useMask,*
boolean *deselNonModel*
)

FlashMaker setup

引数:

minCutoff	Minimum rounding value for box corners If smaller rounding is found, box will be replaced by rectangle
minSize	Minimum x/y size of pads to be replaced. (0 = replace all)
maxSize	Maximum x/y size of pads to be replaced
useTol	Use a tolerance value (if false, optimal tolerance is used)
tol	Tolerance on size of replaced objects
useMask	Only look at pads free of soldermask
deselNonModel	Finds pads in selected data. Deselects all other objects.

void flashMakerSetup (double *minCutoff,*

```
                        double  minSize,
                        double  maxSize,
                        boolean useTol,
                        double  tol,
                        boolean useMask,
                        boolean completelyFree,
                        boolean deselNonModel
                      )
```

FlashMaker setup

引数:

minCutoff	Minimum rounding value for box corners If smaller rounding is found, box will be replaced by rectangle
minSize	Minimum x/y size of pads to be replaced. (0 = replace all)
maxSize	Maximum x/y size of pads to be replaced
useTol	Use a tolerance value (if false, optimal tolerance is used)
tol	Tolerance on size of replaced objects
useMask	Only look at pads free of soldermask
completelyFree	Only look at pads completely free of mask
deselNonModel	Finds pads in selected data. Deselects all other objects.

```
void flipJob ( String  mirror,
               boolean bFlipBuildup,
               boolean bFlipAttachNone,
               String  suffix
             )
```

ジョブのビルドアップ順番の上下を並び替える

引数:

mirror	レイヤにミラーをかける方向 ”x”（x軸に沿う）もしくは”y”（y軸に沿う）を指定する
bFlipBuildup	ジョブのビルドアップ順番の上下を並び替える falseもしくはtrueを指定
bFlipAttachNone	特殊(extra)クラスかつ添付なしのレイヤに対し 上下並び替えを実行
suffix	レイヤ名に追加される接尾語

- Flipはジョブに'uFlipJob'属性を追加する その値は'x', 'y', 'xy', 'yx'のいずれかになる 上下並び替えをすることにより 属性が切り替わる つまり flip x + flip x, results in uFlipJob = '' ということである
- 接尾語は状況に応じえ切り替わる つまり 2度上下並び替えを行った際には元に戻るので 指定した_xxは追加されず 元のレイヤ名に戻る
- PlotParameter Mirror が反転する
- パターン面視が反転する

```
void forEachApe ( String sType )
```

Do the operations on all apertures of the given type.

引数:

sType it must be on from the following

- "cir", "rec", "box", "oct", "con", "com", "the", "txt", "blo", "squ" and "don".

Example:

```
print("JOB NAME " + jobName());
forEachLayer() {
    print("¥n--- Layer " + layName() + " ---");

    forEachApe("cir") {
        String shape = apeShape();

        print("  Aperture " + apeInfo());
        print("    outer: " + apeOuter());
    }
}
```

void forEachApe ( )

全アパーチャを操作する　例:

```
print("JOB NAME " + jobName());
forEachLayer() {
    print("¥n--- Layer " + layName() + " ---");

    forEachApe() {
        String shape = apeShape();

        print("  Aperture " + apeInfo());
        if (isEqual(shape, "cir") || isEqual(shape, "don")  || isEqual(shape,
"the")) {
            print("  outer: " + apeOuter());
            if (!isEqual(shape, "cir")) {
                print("  inner: " + apeInner());
            }
        }
        else if (isEqual(shape, "rec") || isEqual(shape, "box")) {
            print("  X size: " + apeXSize());
            print("  Y size: " + apeYSize());
        }
    }
}
```

void forEachArc ( )

カレントアパーチャの全アークの操作を行う

void forEachDraw ( )

カレントアパーチャの全ドローの操作を行う

void forEachDrill (String *sSubClass*)

Do the operations on all drill layers of the given subclass.

引数:

sSubClass The subclass for the drill layer wanted. If not important specify "". The default subclasses offered by Ucam are "drill", "buried", "blind", "plated", "unplated" and "fixing".

void forEachDrill ( )

全ドリルレイヤを操作する　例:

```
print("--- JOB statistics ---");
print("Name: " + jobName());
print("Spec: " + jobSpec());
print("Path: " + jobPath());
print("Customer: " + jobCustomer());
print("Revision: " + jobRevision());
int layerCount = 1;
forEachDrill() {
    print("¥n  --- Layer " + layerCount++ + " ---");
    print("    Name: " + layName());
    print("    Class: " + layClass());
    print("    Subclass: " + laySubClass());
    print("    Attach: " + layAttach());
    print("    Readable: " + layReadable());
    print("    Reverse: " + layReverse());
}
```

**void forEachExtra (String *sSubClass,*
String *sAttach*
)**

Do the operations on all extra layers of the given subclass and attachment.

引数:

sSubClass The subclass for the layer wanted. If not important specify "". The default subclasses offered by Ucam are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".

sAttach specifies the attachment for "extra" layers. Can be "top", "bottom", "none" or "all".

void forEachExtra (String *sSubClass*)

Do the operations on all Extra layers of the given subclass.

引数:

sSubClass The subclass for the layer wanted. If not important specify "". The default subclasses offered by Ucam are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate" for extra layers. "drill", "buried", "blind", "plated", "unplated" and "fixing" for drill layers. "outer", "inner" and "mixed" for signal layers.

void forEachExtra ( )

全特殊レイヤを操作する　例:

```
print("--- JOB statistics ---");
print("Name: " + jobName());
print("Spec: " + jobSpec());
print("Path: " + jobPath());
print("Customer: " + jobCustomer());
print("Revision: " + jobRevision());
int layerCount = 1;
forEachExtra() {
    print("¥n  --- Layer " + layerCount++ + " ---");
    print("    Name: " + layName());
    print("    Class: " + layClass());
    print("    Subclass: " + laySubClass());
    print("    Attach: " + layAttach());
    print("    Readable: " + layReadable());
    print("    Reverse: " + layReverse());
}
```

void forEachFlash ( )

カレントアパーチャの全フラッシュを操作する

**void forEachI8Job (String *serverName,*
String *dbname,*
String *username,*
String *password,*
String *context,*
String *i8path*
)**

Do the operations on all jobs in the given database context.

引数:

serverName * name of the database server
dbname * name of the database to use (usually 'integr8tor')
username * database login
password * database password
context * integr8tor context to use (usually 'autoflow')
i8path * path to your integr8tor installation (parent of the Integr8tor folder)

**void forEachInRectangle (Rectangle *rect,*
boolean *opt*
)**

Do the operation on the objects whose enclosing rectangle overlap the specified rectangle.

引数:

rect The target rectangle.
opt if true, all the objects which actually overlap the rectangle are found, otherwise all objects whose enclosing rectangle overlaps the specified rectangle are found.

void forEachInRectangle (Rectangle *rect*)

Do the operation on the objects which actually overlap the specified rectangle.

引数:

rect The target rectangle.

Object forEachItem (ObjectList *items*)

任意のオブジェクトリストの全アイテムに対して作業する

例:

```
int count = 0;
fileName = forEachItem(osGetFileList(chooseDirPath())) {
    print(fileName);
```

```
    count++;
}
print(count + " file(s) in directory.");
```

引数:

items オブジェクトリストのアイテムに対し繰り返し処理する

戻り値:

ループ内のオブジェクトリストの各オブジェクトを返す

参照:

osGetFileList(String)

osGetFileList(String, boolean)

osGetFileList(String, String, boolean)

chooseDirPath()

---

void forEachJobNet ( )

Create the net iterator over the all job layers

**void forEachLayer (String *sClass,*
String *sSubClass,*
String *sAttach*
)**

Do the operations on all layers of the given class, subclass and attachment.

引数:

sClass	"layer" or "drill" or "extra"
sSubClass	The subclass for the layer wanted. If not important specify "". The default subclasses offered by Ucam are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".
sAttach	specifies the attachment for "extra" layers. Can be "top", "bottom", "none" or "all".

**void forEachLayer (String *sClass,*
String *sSubClass*
)**

Do the operations on all layers of the given class and subclass.

引数:

sClass	"layer" or "drill" or "extra"
sSubClass	The subclass for the layer wanted. If not important specify "". The default subclasses offered by Ucam are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate" for extra layers. "drill", "buried", "blind", "plated", "unplated" and "fixing" for drill layers. "outer", "inner" and "mixed" for signal layers.

---

void forEachLayer (String *sClass*)

任意のクラスをもつレイヤを操作する

引数:

sClass 下記値を用いる

- "layer" (シグナルレイヤの意)
- "drill"
- "extra"

void forEachLayer ( )

全レイヤを操作する 用例:

```
print("--- JOB statistics ---");
print("Name: " + jobName());
print("Spec: " + jobSpec());
print("Path: " + jobPath());
print("Customer: " + jobCustomer());
print("Revision: " + jobRevision());
int layerCount = 1;
forEachLayer() {
    print("¥n  --- Layer " + layerCount++ + " ---");
    print("    Name: " + layName());
    print("    Class: " + layClass());
    print("    Subclass: " + laySubClass());
    print("    Attach: " + layAttach());
    print("    Readable: " + layReadable());
    print("    Reverse: " + layReverse());
}
```

void forEachLayerNet ( )

Create the net iterator over the layer in plane 1

void forEachNet (int *iNet*)

Do the operations on all objects with the given net number on current layer. It goes over the all apertures in current layer. Example:

```
int objCount = 0;
int net = 20;
forEachNet(net) {
    objSelect("+");
    objCount++;
}
print(objCount + " object(s) with the net number " + net + " selected.");
```

引数:

iNet The net number we look for

void forEachObject (String *sClass*)

Do the operations on all objects of the given class of the current aperture.

引数:

sClass it must be on from the following

- Possible values are "arc", "dra", "fla" or "vtx".

void forEachObject ( )

カレントアパーチャの全オブジェクトを操作する 用例:

```
print("JOB NAME " + jobName());
forEachLayer() {
    print("¥n--- Layer " + layName() + " ---");
    forEachApe("cir") {
        print(" Aperture " + apeInfo());
        forEachObject() {
            print("    " + objInfo());
        }
    }
}
```

void forEachPEInputJob ( )

Do the operations on all PanelEditor input jobs of the current PanelJob.

void forEachPEPanelJob ( )

Do the operations on all proposed PanelEditor PanelJob of the current solution.

void forEachPESolution ( )

Do the operations on all proposed PanelEditor solutions.

void forEachRegion ( )

輪郭（Contour）の全リージョンを操作する

void forEachSignal (String *sSubClass*)

Do the operations on all signal layers of the given subclass.

引数:

sSubClass The subclass for the signal layer wanted. If not important specify "". The default subclasses offered by Ucam are "outer", "inner" and "mixed".

void forEachSignal ( )

全シグナルレイヤを操作する 用例:

```
print("--- JOB statistics ---");
print("Name: " + jobName());
print("Spec: " + jobSpec());
print("Path: " + jobPath());
print("Customer: " + jobCustomer());
```

```
print("Revision: " + jobRevision());
int layerCount = 1;
forEachSignal() {
    print("¥n  --- Layer " + layerCount++ + " ---");
    print("    Name: " + layName());
    print("    Class: " + layClass());
    print("    Subclass: " + laySubClass());
    print("    Attach: " + layAttach());
    print("    Readable: " + layReadable());
    print("    Reverse: " + layReverse());
}
```

void forEachVtxt ( )

カレントアパーチャの全ベクトルテキストを操作する

int GDSII_outLayer (String *filename*)

Generate GDSII from the current layer.

参照:

GDSII_outLayer(String, String)

GDSII_outLayer(String, String)

引数:

filename

**int GDSII_outLayer (String *filename,*
String *options*
)**

Generate GDSII from the current layer.

引数:

filename ... full specification of the GDSII file to be generated

options ... set of options separated by comma (can be empty): 'ARCEXPAND=units' ... provide precision for the expansion of arcs (like ARCEXPAND=0.01mm) 'DPF' ... in addition to the GDSII file, generate corresponding DPF file too (for verification/debugging purposes) 'PRECISE' ... use all post-merge steps in final contourization (typically substantially slower) 'PIECEWISE' ... flatten cells and write them by pieces instead of providing one big contour for each cell(keep array structure) 'NODATE' ... use the same date (1.1.1970 0:00.00) instead of the current date/time) for all items in the GDSII file 'FEEDBACK' ... show data in various steps of the process in feedback layers (for verification/debugging purposes)

戻り値:

status (0 if everything was OK)

**boolean generateContours (double *dGap,*
double *dOverlap*
)**

輪郭（クローズ形）を作成する

引数:

dGap クローズ対象となる最大オープン値
dOverlap 削除対象となる最大オーバーラップ値

戻り値:
ステータス

**boolean generateContoursOnLayer (double *dGap,*
double *dOverlap*
)**

輪郭（クローズ形）を作成する

引数:
dGap クローズ対象となる最大オープン値
dOverlap 削除対象となる最大オーバーラップ値

戻り値:
ステータス

ObjectList getAttrCategories ( )

Returns ObjectList of the all available attribute categories

戻り値:
ObjectList of the all available attribute categories

ObjectList getAttrNames (String *sCategory*)

Return ObjectList of the attribute names in given category

引数:
sCategory Attribute category name

参照:
getAttrCategories()

戻り値:
ObjectList of the attribute names in given category

ObjectList getAttrValues (String *sAttributeName*)

Returns ObjectList of the available values for attribute with given name

引数:
sAttributeName Attribute name

戻り値:
ObjectList of the available values for attribute with given name

int getCount (String *sType*)

Returns number of the faults of the given type

引数:
sType fault type name

戻り値:
Number of the faults of the given type

ObjectList getFaultTypes ( )

Returns fault type names in ObjectList

戻り値:
ObjectList of the fault type names

String getFileLastModified (ObjectList *fileInfo*)

Returns the time that the file denoted by this fileInfo was last modified.

引数:
fileInfo objectlist with the file information

戻り値:
a string representation of the last modification date.

参照:
HSH_base::osFileInfo(String)

String getFileName (ObjectList *fileInfo*)

Returns the name of the file or directory denoted by this fileInfo.

引数:
fileInfo objectlist with the file information

戻り値:
the name of the file or directory denoted by this fileInfo

参照:
HSH_base::osFileInfo(String)

String getFileParent (ObjectList *fileInfo*)

Returns the pathname string of this fileInfo's parent, or null if this fileInfo does not name a parent directory.

引数:
fileInfo objectlist with the file information

戻り値:
the pathname string of this fileInfo's parent, or null if this fileInfo does not name a parent directory

参照:

HSH_base::osFileInfo(String)

long getFileSize (ObjectList *fileInfo*)

Returns the length of the file denoted by this fileInfo. The return value is unspecified if this fileInfo does not denotes a file.

引数:

fileInfo objectlist with the file information

戻り値:

the length, in bytes, of the file denoted by this fileInfo, or 0 otherwise.

参照:

HSH_base::osFileInfo(String)

Ulayer getLayer (ObjectList *layerID*)

Get Layer by the given ID

戻り値:

layer matching to the given ID; or `null`

引数:

layerID

ObjectList getLayerNames ( )

Returns ObjectList containing layer names in ObjectList

Warning: The Object list may contain the same name more then once in case the job has not unique Layer names.

Example: Prints out all layers' name.

```
int counter = 1;
item = forEachItem(getLayerNames()) {
item = forEachItem(getLayerNames()) {
    print("Layer name " + (counter++) + " is " + item);
}
```

戻り値:

ObjectList with all layer names

ObjectList getLayers ( )

Returns structured information about all layers in ObjectList Items in ObjectList are ObjectLists with layer information in fixed order.

Note: The function is used to get information about layers in job without changes in planes and layer activities

Item structure:

info[LAYER_NAME] - Layer name
info[LAYER_CLASS] - Layer class
info[LAYER_SUBCLASS] - Layer subclass

info[LAYER_ATTACH] - Layer attachment
info[LAYER_INDEX] - Layer index
info[LAYER_ACTIVITY] - Layer activity
info[LAYER_APERTURES] - Layer number of apertures or -1 if the layer is not loaded yet.

Example: Prints out all layers' info.

```
int counter = 1;
info = forEachItem(getLayers()) {
info = forEachItem(getLayers()) {
    print("---- Layer " + (counter++) + " ----");
    print("  name " + info[LAYER_NAME]);
    print("  class " + info[LAYER_CLASS]);
    print("  subclass " + info[LAYER_SUBCLASS]);
    print("  attach " + info[LAYER_ATTACH]);
    print("  index " + info[LAYER_INDEX]);
    print("  active " + info[LAYER_ACTIVITY]);
    print("  apertures " + info[LAYER_APERTURES]);
}
```

戻り値:
ObjectList with layers' info (ObjectList)

参照:
LAYER_NAME

LAYER_CLASS

LAYER_SUBCLASS

LAYER_ATTACH

LAYER_INDEX

LAYER_ACTIVITY

LAYER_APERTURES

Rectangle getLocationOnScreen (String *sFrameName*)

Gives screen location and dimension of this dockable frame.

引数:
sFrameName identification frame given by getFrameID() method of CustomFrame class

戻り値:
location and width and height of the dockable frame as an **Rectangle**

ObjectList getMode ( )

現在のオペレーションモード・単位系・スナップモード情報を取得

戻り値:
[{sOption, sUnit, sSnapMode}]の配列

参照:
com.barco.ets.ucam.hypershell.HyperShell::setMode(String, String, String)

ObjectList getNetAttrNames ( )

Return ObjectList of the net attribute names in given Layer

戻り値:
ObjectList of the net attribute names

ObjectList getNetNames ( )

Return ObjectList of the net names in current job

戻り値:
ObjectList of the net names

int getNetNumberByClick (double *pt_x*,
double *pt_y*
)

Found object in layer on plane 1 on the coordinates and returns its net number

引数:
pt_x (X coordinate) coordinates, click point
pt_y (Y coordinate) coordinates, click point

戻り値:
net number a object on the coordinate or 0: no net or -1: no object on the coordinate

int getNetNumberByClick (Point *pt*)

Found object in layer on plane 1 on the coordinates and returns its net number

引数:
pt coordinates, click point

戻り値:
net number a object on the coordinate or 0: no net or -1: no object on the coordinate

Ulayer getNextLayer ( )

Returns next layer after current layer in plane 1

戻り値:
next layer after current layer in plane 1

ObjectList getODBxxSteps (String *sPath*)

getODBxxSteps

引数:
sPath path to ODB job

戻り値:

two item ObjectArray, the first item is Object List of step names and the second item is Object List of layer names.

int getPlotParam (String *sKey,*
int *iDefValue*
)

Sets the plot parameter.

引数:

sKey The key. Possible values are :

- "RESOLUTION"
- "XOFF" (the x offset)
- "YOFF" (the y offset)
- "XSCALE" (the x scale)
- "YSCALE" (the y scale)
- "SXCEN" (the x scale center)
- "SYCEN" (the y scale center)
- "MXCEN" (the x mirror center)
- "MYCEN" (the y mirror center)
- "POSITION":
- - MergeJob.PLOT_POSITION_COMBINE_FILL_PERCENTAGE
- - MergeJob.PLOT_POSITION_COMBINE
- - MergeJob.PLOT_POSITION_SINGLE
- - MergeJob.PLOT_POSITION_SINGLE_LEFT_FIXED
- - MergeJob.PLOT_POSITION_SINGLE_RIGHT_FIXED
- "ENLARGE_VECTORS_MINIMUMSIZE"
- "ENLARGE_VECTORS_AMOUNT"
- "ENLARGE_FLASH_MINIMUMSIZE"
- "ENLARGE_FLASH_AMOUNT"
- "ENLARGE_CONDUCTOR_SIZE" (Adds a size. Use 0 to remove them all.)
- "ENLARGE_CONDUCTOR_AMOUNT"
- "ENLARGE_COMPLEX"
- "APPLY_ENLARGE_TO" (The value must be one of Ulayer.PLOT_ENLARGE_*)
- "VARIABLE_TEXT_HEIGHT"
- "VARIABLE_TEXT_DIRECTION" (The value must be one of Ulayer.PLOT_VARTEXT_DIRECTION_*)

iDefValue The default value

戻り値:

value of the plot parameter or default value

double getPlotParam (String *sKey,*
double *dDefValue*
)

Sets the plot parameter.

引数:

sKey The key. Possible values are :

- "RESOLUTION"
- "XOFF" (the x offset)
- "YOFF" (the y offset)

"XSCALE" (the x scale)
- "YSCALE" (the y scale)
- "SXCEN" (the x scale center)
- "SYCEN" (the y scale center)
- "MXCEN" (the x mirror center)
- "MYCEN" (the y mirror center)
- "POSITION":
- - MergeJob.PLOT_POSITION_COMBINE_FILL_PERCENTAGE
- - MergeJob.PLOT_POSITION_COMBINE
- - MergeJob.PLOT_POSITION_SINGLE
- - MergeJob.PLOT_POSITION_SINGLE_LEFT_FIXED
- - MergeJob.PLOT_POSITION_SINGLE_RIGHT_FIXED
- "ENLARGE_VECTORS_MINIMUMSIZE"
- "ENLARGE_VECTORS_AMOUNT"
- "ENLARGE_FLASH_MINIMUMSIZE"
- "ENLARGE_FLASH_AMOUNT"
- "ENLARGE_CONDUCTOR_SIZE" (Adds a size. Use 0 to remove them all.)
- "ENLARGE_CONDUCTOR_AMOUNT"
- "ENLARGE_COMPLEX"
- "APPLY_ENLARGE_TO" (The value must be one of Ulayer.PLOT_ENLARGE_*)
- "VARIABLE_TEXT_HEIGHT"
- "VARIABLE_TEXT_DIRECTION" (The value must be one of Ulayer.PLOT_VARTEXT_DIRECTION_*)

dDefValue The default value

戻り値:
value of the plot parameter or default value

**String getPlotParam (String *sKey,*
String *sDefValue*
)**

Sets the plot parameter.

引数:
sKey The key. Possible values are :

- "RESOLUTION"
- "XOFF" (the x offset)
- "YOFF" (the y offset)
- "XSCALE" (the x scale)
- "YSCALE" (the y scale)
- "SXCEN" (the x scale center)
- "SYCEN" (the y scale center)
- "MXCEN" (the x mirror center)
- "MYCEN" (the y mirror center)
- "POSITION":
- - MergeJob.PLOT_POSITION_COMBINE_FILL_PERCENTAGE
- - MergeJob.PLOT_POSITION_COMBINE
- - MergeJob.PLOT_POSITION_SINGLE
- - MergeJob.PLOT_POSITION_SINGLE_LEFT_FIXED
- - MergeJob.PLOT_POSITION_SINGLE_RIGHT_FIXED
- "ENLARGE_VECTORS_MINIMUMSIZE"
- "ENLARGE_VECTORS_AMOUNT"
- "ENLARGE_FLASH_MINIMUMSIZE"
- "ENLARGE_FLASH_AMOUNT"
- "ENLARGE_CONDUCTOR_SIZE" (Adds a size. Use 0 to remove them all.)
- "ENLARGE_CONDUCTOR_AMOUNT"

- "ENLARGE_COMPLEX"
- "APPLY_ENLARGE_TO" (The value must be one of Ulayer.PLOT_ENLARGE_*)
- "VARIABLE_TEXT_HEIGHT"
- "VARIABLE_TEXT_DIRECTION" (The value must be one of Ulayer.PLOT_VARTEXT_DIRECTION_*)

sDefValue The default value

戻り値:
value of the plot parameter or default value

void grabWidget ( )

Prints out the name of the widget that is clicked with the mouse.

void gridAlign (double *dStep*)

Align to grid

引数:
dStep grid size

void gridCross (boolean *bCross*)

Sets the grid crosses or lines displayed when the grid is visible

引数:
bCross `true` show the grid crosses; `false` show the grid lines

boolean gridCross ( )

Returns `true` if the grid crosses are displayed or not

戻り値:
`true` if the grid crosses are displayed or `false` if the grid lines are displayed

void gridOrigin (double *ptOrigin_x*,
double *ptOrigin_y*
)

Sets the grid origin to given **Point**

引数:
ptOrigin_x (X coordinate) new grid origin **Point**
ptOrigin_y (Y coordinate) new grid origin **Point**

void gridOrigin (Point *ptOrigin*)

Sets the grid origin to given **Point**

引数:

ptOrigin new grid origin **Point**

Point gridOrigin ( )

Returns the current grid origin **Point**

戻り値:

the current grid origin **Point**

void gridOutline (double *refPoint_x*,
double *refPoint_y*,
double *offset_x*,
double *offset_y*,
int *repeatX*,
int *repeatY*
)

PCB images in a flat data need to be outlined. We can manually construct outline contour in this (usually the layer in plane 1 and outline extra layer) layer according to a PCB image in reference layer. We mark reference point in PCB image and give an offset of the same reference data in direction X and Y. The repeat parameter defines a number of outlines in each direction.

引数:

refPoint_x (X coordinate) A point where the PCB data are taken as an reference (for alignment)
refPoint_y (Y coordinate) A point where the PCB data are taken as an reference (for alignment)
offset_x (X coordinate) of the outline grid in X and Y
offset_y (Y coordinate) of the outline grid in X and Y
repeatX a number of the outlines in X direction
repeatY a number of the outlines in Y direction

戻り値:

Uapeobj can be `null` in case the new outline couldn't be created. The return aperture can be the same in case the target is without rotation. Only the object(flash) is the new. In case there is a rotation the return object is completely new aperture with the (one) new flash.

void gridOutline (Point *refPoint*,
Point *offset*,
int *repeatX*,
int *repeatY*
)

PCB images in a flat data need to be outlined. We can manually construct outline contour in this (usually the layer in plane 1 and outline extra layer) layer according to a PCB image in reference layer. We mark reference point in PCB image and give an offset of the same reference data in direction X and Y. The repeat parameter defines a number of outlines in each direction.

引数:

refPoint A point where the PCB data are taken as an reference (for alignment)

offset of the outline grid in X and Y
repeatX a number of the outlines in X direction
repeatY a number of the outlines in Y direction

戻り値:
Uapeobj can be `null` in case the new outline couldn't be created. The return aperture can be the same in case the target is without rotation. Only the object(flash) is the new. In case there is a rotation the return object is completely new aperture with the (one) new flash.

void gridStep (double *dStepX,*
double *dStepY*
)

Sets the X and Y distances between grid crosses or lines

引数:
dStepX the X distance between grid crosses or lines
dStepY the Y distance between grid crosses or lines

Point gridStep ( )

Returns the **Point** with coordinates presenting the X and Y distance between grid crosses or lines

戻り値:
the **Point** with coordinates presenting the X and Y distance between grid crosses or lines

double gridStepX ( )

Returns the X distance between grid crosses or lines

戻り値:
the X distance between grid crosses or lines

double gridStepY ( )

Returns the Y distance between grid crosses or lines

戻り値:
the Y distance between grid crosses or lines

void gridVisible (boolean *bVisible*)

Sets the grid visible or hidden

引数:
bVisible `true` to make the grid visible, `false` to make it hidden

boolean gridVisible ( )

returns `true` if the grid is currently visible

戻り値:
`true` if the grid is currently visible

void groupApeBy (String *spec*)

全アクティブレイヤにおいて 共通のスペックを持つアパーチャをグループ化する

引数:
spec スペック: "number", "definition" , "plane"のいずれか

void groupApertureDefinitions ( )

アパーチャマネジャー: プレーン1のアクティブレイヤにおける定義によるアパーチャのグループ化

void groupApertureNumbers ( )

アパーチャマネジャー: プレーン1のアクティブレイヤにおける番号によるアパーチャのグループ化

void groupAperturesByPolarity ( )

アパーチャマネジャー: プレーン1のアクティブレイヤにおけるネガポジ極性によるアパーチャのグループ化

double groupUFD (double *dDistance*)

Group faults in a fault list

引数:
dDistance - neighborhood

戻り値:
true value of grouping distance

void groupUFD ( )

Group faults in a fault list

void helpOnContext ( )

Get help on context.

void hideAll ( )

全レイヤを非表示にする

void hideBlockStructure ( )

hide Block Structure Information dialog

void HitachiSpotDiameterCompensation (double *dOffset,*
double *dArcExpandMarginOverrideMicrons,*
int *iMode,*
int *iFastMode,*
boolean *bAmSkipFlag,*
boolean *bChangePolarity*
)

Performs Hitachi SPP

引数:

dOffset	compensation value in microns (positive = normal (thin); negative = reverse (thicken))
dArcExpandMarginOverrideMicrons	if >= 0, override ucam.db arc expand margin (in micron)
iMode	SPP mode (can be 1 or 2)
iFastMode	SPP fast mode flag (can be 0 or 1 or 2)
bAmSkipFlag	whether or not to skip processing of Aperture Macros (ignored)
bChangePolarity	nonzero indicates that the layer will be reversed by the DE system; therefore, the sign of the offset should swapped when working in Mode 2

void HitachiSpotDiameterCompensation (double *dOffset,*
double *dArcExpandMarginOverrideMicrons,*
int *iMode,*
boolean *bFastMode,*
boolean *bAmSkipFlag,*
boolean *bChangePolarity*
)

Performs Hitachi SPP

引数:

dOffset	compensation value in microns (positive = normal (thin); negative = reverse (thicken))
dArcExpandMarginOverrideMicrons	if >= 0, override ucam.db arc expand margin (in microns)
iMode	SPP mode (can be 1 or 2)
bFastMode	SPP fast mode flag
bAmSkipFlag	whether or not to skip processing of Aperture Macros (ignored)
bChangePolarity	nonzero indicates that the layer will be reversed by the DE system; therefore, the sign of the offset should swapped when working in Mode 2

String i8Jobarticleid ( )

カレントジョブのarticleid（アーティクルのID）を返す

戻り値:
カレントジョブのarticleid（アーティクルID）を返すか カレントジョブがなければ null を返す

String i8JobBoardid ( )

カレントジョブのboardid(基板のID)を返す

戻り値:
カレントジョブの基板IDを返すか カレントジョブがなければ null を返す

String i8JobCustomer ( )

カレントジョブの顧客を返す

戻り値:
カレントジョブの顧客を返すか カレントジョブがなければ null を返す

boolean i8Jobdelete ( )

カレントジョブを削除する

戻り値:
ジョブが正常に削除できれば true それ以外は false

int i8JobDuration ( )

カレントジョブのduration(処理時間)を返す

戻り値:
カレントジョブのduration(処理時間)を返すか カレントジョブがなければ -1 を返す

Date i8JobFinishtime ( )

カレントジョブの終了時間を返す

戻り値:
カレントジョブの終了時間を返すか カレントジョブがなければ null を返す

int i8JobFullduration ( )

カレントジョブのfullduration(全体処理時間）を返す

戻り値:
カレントジョブのfullduration(全体処理時間）を返すか カレントジョブがなければ -1 を返す

int i8JobId ( )

カレントジョブのIDを返す

戻り値:
カレントジョブのIDを返すか カレントジョブがなければ -1 を返す

String i8JobLocation ( )

カレントジョブのロケーションを返す

戻り値:
カレントジョブのロケーションを返すか カレントジョブがなければ null を返す

int i8JobPriority ( )

カレントジョブの優先度を返す

戻り値:
カレントジョブの優先度を返すか カレントジョブがなければ -1 を返す

String i8JobProgress ( )

カレントジョブの進捗状況を返す

戻り値:
カレントジョブの進捗状況を返すか カレントジョブがなければ null を返す

int i8JobQueueposition ( )

カレントジョブのqueueposition（キュー内の現在位置）を返す

戻り値:
カレントジョブのqueueposition（キュー内での現在位置）を返すか カレントジョブがなければ -1 を返す

Date i8JobStarttime ( )

カレントジョブの開始時間を返す

戻り値:
カレントジョブの開始時間を返すか カレントジョブがなければ null を返す

Date i8JobSubmittime ( )

カレントジョブのsubmittime（サブミットタイム：ジョブがキューに追加された時間）を返す

戻り値:
カレントジョブのsubmittime（サブミットタイム：ジョブがキューに追加された時間）を返すか カレントジョブがなければ null を返す

void importEpc (String *sPath*)

import Epc CAR job

引数:
sPath - full path to 'job'.car file inside EPC files directory

```
int importExternal ( String  sExtName,
                     String  sWheName,
                     String  sLanguage,
                     boolean bKeepExtension,
                     String  sLayClass,
                     String  sLayAtt,
                     String  sStatus,
                     String  sWheLang,
                     boolean bAnalyzed,
                     String  sWheFile,
                     int     iLocale
                   )
```

外部ファイルをインポートする

引数:

sExtName	外部ファイルの完全パス
sWheName	ホイールファイルの完全パス
sLanguage	言語
bKeepExtension	拡張子の保持
sLayClass	レイヤクラス
sLayAtt	レイヤ属性
sStatus	ステータス
sWheLang	ホイール言語
bAnalyzed	解析フラグ
sWheFile	ホイールファイルに必要な情報（"---" もしくは"..."）
iLocale	ファイルロケーション：1=ローカル 0=リモート

戻り値:
ステータス

```
int importExternal ( String  sExtName,
                     String  sWheName,
                     String  sLanguage,
                     boolean bKeepExtension
```

)

外部ファイルをインポートする

引数:

sExtName 外部ファイルの完全パス
sWheName ホイールファイルの完全パス
sLanguage 言語
bKeepExtension 拡張子の保持

戻り値:

ステータス

int importExternal (String *sExtName*)

外部ファイルをインポートする フォーマット解析を呼び出すする

引数:

sExtName 外部ファイルの完全パス

戻り値:

ステータス

void importFile (String *sScriptPath*)

ファイルからスクリプトを読み込む

引数:

sScriptPath - スクリプトファイルの完全パス

void importGwk (String *sPath*)

GWKファイルを読み込む

引数:

sPath - GWKファイルの完全パス

String importHeptaCSV (String *sCSVFile,*
String *sDXFFile,*
String *sOptions*
)

引数:

sCSVFile csv file
sDXFFile dxf file
sOptions options for hepta csv

戻り値:

`null` if OK otherwise return s error message as a string

void importHouei (String *sPath*)

import Houei job

引数:

sPath - full path to Houei kend_xxx.par file inside Houei job directory

void importIpc (String *sPath,*
String *sVersion*
)

MET・IPC356・IPC356bファイルを読み込む

引数:

sPath - METもしくはIPCファイルの完全パス
sVersion - IPCフォーマットのバージョン（正しい値は "ipc356" "ipc356b" "met"のいずれかになる ）

void importIPC2581 (String *sPath,*
String *sStep,*
String *sLayer*
)

import IPC2581

引数:

sPath path to IPC2581 file
sStep step name
sLayer layer name

void importODBxx (String *sPath,*
String *sStep,*
String *sLayer,*
ObjectList *oReplaceCodeMap*
)

import ODBxx

引数:

sPath path to ODB job
sStep step name
sLayer layer name
oReplaceCodeMap the replace code map ObjectList

Example: [{ [{"$$CODE","Result text1"}], [{"$$CODE1","Result text2"}] }]

void importODBxx (String *sPath,*
String *sStep,*

String ***sLayer***
)

import ODBxx

引数:

sPath path to ODB job
sStep step name
sLayer layer name

void importODBxx (String ***sPath,***
String ***sStep,***
ObjectList ***oReplaceCodeMap***
)

import ODBxx

引数:

sPath path to ODB job
sStep ODB step
oReplaceCodeMap the replace code map ObjectList

Example: [{ [{"$$CODE","Result text1"}], [{"$$CODE1","Result text2"}] }]

void importODBxx (String ***sPath,***
String ***sStep***
)

ODBxxをインポートする

引数:

sPath ODBジョブのパス
sStep ODBステップ

void importPolarBuildup (String ***sPolarFilePath*** **)**

Polarアプリケーションによって生成されたstkxファイルから ビルドアップ情報を読み込む

引数:

sPolarFilePath Polar stkxファイルのファイルパス仕様

void importScript (String ***sScript*** **)**

任意のスクリプトを読み込む

引数:

sScript - スクリプトテキスト

void importWf (String *sPath*)

WFファイルを読み込む

引数:

sPath - WFファイルの完全パス

void innerCopperCount (boolean *bUseMask*,
boolean *bConfirmMaskUsage*
)

Calculates the copper surface in active inner layers.

引数:

bUseMask	When true, active mask layers are taken into account: The "free of mask" area's are then calculated. The mask with attachment top is used for the top outer layer. The mask with attachment bottom is used for the bottom outer layer. The mask with attachment none is used for inner layers.
bConfirmMaskUsage	When true and active mask layers exist, then asks for confirmation if active mask layers should be used, or not.

void innerCopperCount ( )

Calculates the copper surface in active inner layers.

void insertAperture (boolean *bBefore*,
String *sSrcLayer*,
ObjectList *srcApeIndex*
)

アパーチャマネジャー: カレントアパーチャ近くに （他の）レイヤのアパーチャを挿入する

引数:

bBefore	true: apeIndexの前に挿入する false: apeIndexの後に挿入する
sSrcLayer	アパーチャを取得するレイヤの名前
srcApeIndex	ソースとなるレイヤにあるアパーチャのインデックス配列

int insertArc (double *arc_fx*,
double *arc_fy*,
double *arc_tx*,
double *arc_ty*,
double *arc_cx*,
double *arc_cy*,
String *arc_sense*,
int *iNet*,
String *sSelection*
)

カレントアパーチャでアークを挿入する

引数:

arc_fx （始点のX座標）アーク
arc_fy （始点のY座標）アーク
arc_tx （終点のX座標）アーク
arc_ty （終点のY座標）アーク
arc_cx （中心点のX座標）アーク
arc_cy （中心点のY座標）アーク
arc_sense （アークの描画方向）アーク
iNet オブジェクトのネット番号
sSelection 選択オプション "all"/ "sel"のいずれかを設定 "sel"が指定された場合 selが強調表示される

戻り値:

エラーステータス （0は成功 0以外は失敗）

```
int insertArc ( Arc    arc,
                int    iNet,
                String sSelection
              )
```

カレントアパーチャでアークを挿入する

引数:

arc アーク
iNet オブジェクトのネット番号
sSelection 選択オプション "all"/ "sel"のいずれかを設定 "sel"が指定された場合 selが強調表示される

戻り値:

エラーステータス（0は成功 0以外は失敗）

```
void insertArc3Point ( double pnt1_x,
                       double pnt1_y,
                       double pnt2_x,
                       double pnt2_y,
                       double pnt3_x,
                       double pnt3_y
                     )
```

カレントアパーチャを使用しアークを挿入する 3つの外形ポイントからアークを計算する

引数:

pnt1_x （X座標）外形ポイント（始点）
pnt1_y （Y座標）外形ポイント（始点）
pnt2_x （X座標）外形ポイント
pnt2_y （Y座標）外形ポイント
pnt3_x （X座標）外形ポイント（終点）
pnt3_y （Y座標）外形ポイント（終点）

```
void insertArc3Point ( Point pnt1,
```

```
                    Point  pnt2,
                    Point  pnt3
                  )
```

カレントアパーチャを使用しアークを挿入する　3つの外形ポイントからアークを計算する

引数:

pnt1 外形ポイント（始点）
pnt2 外形ポイント
pnt3 外形ポイント（終点）

```
void insertArc3Point ( double  pnt1_x,
                       double  pnt1_y,
                       double  pnt2_x,
                       double  pnt2_y,
                       double  pnt3_x,
                       double  pnt3_y,
                       int     iNet,
                       String  sSelection
                     )
```

カレントアパーチャを使用しアークを挿入する　3つの外形ポイントからアークを計算する

引数:

pnt1_x （X座標）外形ポイント（始点）
pnt1_y （Y座標）外形ポイント（始点）
pnt2_x （X座標）外形ポイント
pnt2_y （Y座標）外形ポイント
pnt3_x （X座標）外形ポイント（終点）
pnt3_y （Y座標）外形ポイント（終点）
iNet オブジェクトのネット番号
sSelection 選択オプション ”sel”を指定すると オブジェクトは選択されたことがわかるように表示される

```
void insertArc3Point ( Point   pnt1,
                       Point   pnt2,
                       Point   pnt3,
                       int     iNet,
                       String  sSelection
                     )
```

カレントアパーチャを使用しアークを挿入する　3つの外形ポイントからアークを計算する

引数:

pnt1 外形ポイント（始点）
pnt2 外形ポイント
pnt3 外形ポイント（終点）
iNet オブジェクトのネット番号
sSelection 選択オプション ”sel”を指定すると オブジェクトは選択されたことがわかるように表示される

```
void insertArcCenterStart ( double  pntCenter_x,
                            double  pntCenter_y,
                            double  pntFrom_x,
                            double  pntFrom_y,
                            double  pntTo_x,
                            double  pntTo_y,
                            String  sDirection
                          )
```

カレントアパーチャを使用してアークを挿入する　2つの外形ポイントとセンターポイントからアークを計算する

引数:

pntCenter_x	（X座標）センターポイント
pntCenter_y	（Y座標）センターポイント
pntFrom_x	（X座標）外形ポイント
pntFrom_y	（Y座標）外形ポイント
pntTo_x	（X座標）外形ポイント
pntTo_y	（Y座標）外形ポイント
sDirection	アークの方法

```
void insertArcCenterStart ( Point   pntCenter,
                            Point   pntFrom,
                            Point   pntTo,
                            String  sDirection
                          )
```

カレントアパーチャを使用してアークを挿入する　2つの外形ポイントとセンターポイントからアークを計算する

引数:

pntCenter	センターポイント
pntFrom	外形ポイント
pntTo	外形ポイント
sDirection	アークの方向

```
void insertArcCenterStart ( double  pntCenter_x,
                            double  pntCenter_y,
                            double  pntFrom_x,
                            double  pntFrom_y,
                            double  pntTo_x,
                            double  pntTo_y,
                            String  sDirection,
                            int     iNet,
                            String  sSelection
                          )
```

カレントアパーチャを使用してアークを挿入する　2つの外形ポイントとセンターポイントからアークを計算する

引数:

pntCenter_x	（X座標）センターポイント
pntCenter_y	（Y座標）センターポイント
pntFrom_x	（X座標）外形ポイント
pntFrom_y	（Y座標）外形ポイント
pntTo_x	（X座標）外形ポイント
pntTo_y	（Y座標）外形ポイント
sDirection	アークの方向
iNet	オブジェクトのネット番号
sSelection	選択オプション ”sel”を指定すると オブジェクトは選択されたことがわかるように表示される

void insertArcCenterStart (Point *pntCenter,*
Point *pntFrom,*
Point *pntTo,*
String *sDirection,*
int *iNet,*
String *sSelection*
)

カレントアパーチャを使用してアークを挿入する 2つの外形ポイントとセンターポイントからアークを計算する

引数:

pntCenter	センターポイント
pntFrom	外形ポイント
pntTo	外形ポイント
sDirection	アークの方向
iNet	オブジェクトのネット番号
sSelection	選択オプション ”sel”を指定すると オブジェクトは選択されたことがわかるように表示される

boolean insertArcConcentric (boolean *bSelection,*
ObjectList *oArcs*
)

カレントアパーチャで同心アークを挿入する

引数:

bSelection	選択 true指定の場合 オブジェクトは選択されていることが分かるように表示される
oArcs	アークの配列

戻り値:

エラーステータス（trueの場合は 何か追加されたことを示す）

void insertBreak (Point *line_fp,*
Point *line_tp*
)

分割点（break）をライン上のアークやドローに追加する

引数:
line_fp （始点）分割点はライン上におく
line_tp （終点）分割点はライン上におく

void insertBreak (double *line_fx*,
double *line_fy*,
double *line_tx*,
double *line_ty*
)

分割点（break）をライン上のアークやドローに追加する

引数:
line_fx （始点X座標）分割点はライン上におく
line_fy （始点Y座標）分割点はライン上におく
line_tx （終点X座標）分割点はライン上におく
line_ty （終点Y座標）分割点はライン上におく

void insertBreak (Line *line*)

分割点（break）をライン上のアークやドローに追加する

引数:
line 分割点はライン上におく

void insertContourText (double *rect_xmin*,
double *rect_ymin*,
double *rect_xmax*,
double *rect_ymax*,
String *sText*,
String *sFontName*,
int *iFontStyle*,
String *sMirror*,
boolean *bReverse*,
boolean *bAllowDistortion*,
String *sSelection*
)

指定テキストを持つ輪郭アパーチャを作成し レイヤに追加する

引数:
rect_xmin （長方形の左境界）テキストを長方形状に囲う
rect_ymin （長方形の下境界）テキストを長方形状に囲う
rect_xmax （長方形の右境界）テキストを長方形状に囲う
rect_ymax （長方形の上境界）テキストを長方形状に囲う
sText 画像化したいテキスト
sFontName フォント名 java.awt.Fontコンストラクターを参照
iFontStyle フォントスタイル java.awt.Fontコンストラクターを参照
sMirror ミラー設定 "", "X", "Y", "XY"のいずれか

bReverse	輪郭アパーチャをリバースする否かを示す
bAllowDistortion	trueを設定すると 文字の変形が許可される
sSelection	“sel”を設定すると 輪郭テキストが選択される

```
void insertContourText ( Rectangle  rect,
                         String     sText,
                         String     sFontName,
                         int        iFontStyle,
                         String     sMirror,
                         boolean    bReverse,
                         boolean    bAllowDistortion,
                         String     sSelection
                       )
```

指定テキストを持つ輪郭アパーチャを作成し レイヤに追加する

引数:

rect	テキストを長方形状に囲う
sText	画像化したいテキスト
sFontName	フォント名 java.awt.Fontコンストラクターを参照
iFontStyle	フォントスタイル java.awt.Fontコンストラクターを参照
sMirror	ミラー設定 "", "X", "Y", "XY"のいずれか
bReverse	輪郭アパーチャをリバースする否かを示す
bAllowDistortion	trueを設定すると 文字の変形が許可される
sSelection	“sel”を設定すると 輪郭テキストが選択される

```
void insertCopper ( int     number,
                    String  attach,
                    String  material,
                    double  thickness,
                    String  reference,
                    double  tolerance,
                    String  supplier
                  )
```

特定のレイヤに任意の材料仕様を指定して銅を追加する

引数:

number	シグナルレイヤインデックス
attach	“top”もしくは”bottom”に添付
material	材料名
thickness	材料厚み
reference	材料リファレンス
tolerance	材料の許容量
supplier	材料供給者

```
void insertCore ( int     layNum,
                  boolean matTop,
```

```
                boolean    matBot,
                String     material,
                String     topMaterial,
                String     botMaterial,
                double     thickness,
                double     topThickness,
                double     botThickness,
                String     reference,
                double     tolerance,
                double     erConstant,
                String     supplier,
                ObjectList attrNames,
                Object[]   attrValues,
                boolean    revInsert
              )
```

Inserts core with a given material specification between the specified Layers.

引数:

layNum	int Top layer index
matTop	boolean true if the core is attached to the top layer
matBot	boolean true if the core is attached to the bottom layer
material	String Material name
topMaterial	String material on the top of the core
botMaterial	String material on the bot of the core
thickness	double Material thickness
topThickness	double thickness of the top material
botThickness	double thickness of the bot material
reference	String Material reference
tolerance	double Material tolerance
erConstant	double Material ER constant
supplier	String Material supplier
attrNames	String list of the Material attribute names
attrValues	String the same sized list of the Material attribute values. If the Material attribute has no value put the empty string "", no `null` value!
revInsert	boolean toggles top and bottom material.

```
void insertCore ( int        layNum,
                  boolean    matTop,
                  boolean    matBot,
                  String     material,
                  String     topMaterial,
                  String     botMaterial,
                  double     thickness,
                  double     topThickness,
                  double     botThickness,
                  String     reference,
                  double     tolerance,
                  double     erConstant,
                  String     supplier,
                  ObjectList attrNames,
```

```
                Object[]    attrValues
            )
```

Inserts core with a given material specification between the specified Layers.

引数:

layNum	int Top layer index
matTop	boolean true if the core is attached to the top layer
matBot	boolean true if the core is attached to the bottom layer
material	String Material name
topMaterial	String material on the top of the core
botMaterial	String material on the bot of the core
thickness	double Material thickness
topThickness	double thickness of the top material
botThickness	double thickness of the bot material
reference	String Material reference
tolerance	double Material tolerance
erConstant	double Material ER constant
supplier	String Material supplier
attrNames	String list of the Material attribute names
attrValues	String the same sized list of the Material attribute values. If the Material attribute has no value put the empty string "", no `null` value!

```
void insertCore ( int       layTop,
                  int       layBot,
                  boolean   matTop,
                  boolean   matBot,
                  String    material,
                  String    topMaterial,
                  String    botMaterial,
                  double    thickness,
                  double    topThickness,
                  double    botThickness,
                  String    reference,
                  double    tolerance,
                  double    erConstant,
                  String    supplier,
                  ObjectList attrNames,
                  Object[]  attrValues,
                  boolean   revInsert
                )
```

Inserts core with a given material specification between the specified Layers.

引数:

layTop	int Top layer index
layBot	int Bottom layer index
matTop	boolean true if the core is attached to the top layer
matBot	boolean true if the core is attached to the bottom layer
material	String Material name
topMaterial	String material on the top of the core
botMaterial	String material on the bot of the core

thickness	double Material thickness
topThickness	double thickness of the top material
botThickness	double thickness of the bot material
reference	String Material reference
tolerance	double Material tolerance
erConstant	double Material ER constant
supplier	String Material supplier
attrNames	String list of the Material attribute names
attrValues	String the same sized list of the Material attribute values. If the Material attribute has no value put the empty string "", no `null` value!
revInsert	boolean toggles top and bottom material.

void insertCore (int *layTop,*
int *layBot,*
boolean *matTop,*
boolean *matBot,*
String *material,*
String *topMaterial,*
String *botMaterial,*
double *thickness,*
double *topThickness,*
double *botThickness,*
String *reference,*
double *tolerance,*
double *erConstant,*
String *supplier,*
ObjectList *attrNames,*
Object[] *attrValues*
)

Inserts core with a given material specification between the specified Layers.

引数:

layTop	int Top layer index
layBot	int Bottom layer index
matTop	boolean true if the core is attached to the top layer
matBot	boolean true if the core is attached to the bottom layer
material	String Material name
topMaterial	
botMaterial	
thickness	double Material thickness
topThickness	
botThickness	
reference	String Material reference
tolerance	double Material tolerance
erConstant	double Material ER constant
supplier	String Material supplier
attrNames	String list of the Material attribute names
attrValues	String the same sized list of the Material attribute values. If the Material attribute has no value put the empty string "", no `null` value!

```
int insertDraw ( double  line_fx,
                 double  line_fy,
                 double  line_tx,
                 double  line_ty,
                 int     iNet,
                 String  sSelection
               )
```

カレントアパーチャを用いてドローを追加する

引数:

line_fx （始点X座標）ライン
line_fy （始点Y座標）ライン
line_tx （終点X座標）ライン
line_ty （終点Y座標）ライン
iNet オブジェクトのネット番号
sSelection 選択オプション ”all”もしくは”sel”を設定 ”sel“を指定すると selが強調表示される

戻り値:

エラーステータス（0は成功 0以外は失敗）

```
int insertDraw ( Line    line,
                 int     iNet,
                 String  sSelection
               )
```

カレントアパーチャを用いてドローを追加する

引数:

line ライン
iNet オブジェクトのネット番号
sSelection 選択オプション ”all”もしくは”sel”を設定 ”sel“を指定すると selが強調表示される

戻り値:

エラーステータス（0は成功 0以外は失敗）

```
int insertDraw ( double  line_fx,
                 double  line_fy,
                 double  line_tx,
                 double  line_ty
               )
```

カレントアパーチャを用いてドローを追加する

引数:

line_fx （始点X座標）ライン
line_fy （始点Y座標）ライン
line_tx （終点X座標）ライン
line_ty （終点Y座標）ライン

戻り値:

エラーステータス（0は成功 0以外は失敗）

int insertDraw (Line *line*)

カレントアパーチャを用いてドローを追加する

引数:
line ライン

戻り値:
エラーステータス（0は成功 0以外は失敗）

int insertFlash (double *pt_x,*
double *pt_y,*
int *iNet,*
String *sSelection*
)

カレントアパーチャを用いてフラッシュを追加する ネット番号と選択箇所を設定する

引数:

pt_x	(X 座標) フラッシュポイント
pt_y	(Y 座標) フラッシュポイント
iNet	オブジェクトのネット番号
sSelection	選択オプション "all" 又は "sel" "sel" 指定の場合は オブジェクトが選択されたとしてマークされる

戻り値:
エラーステータス（0は成功 0以外は失敗）

int insertFlash (Point *pt,*
int *iNet,*
String *sSelection*
)

カレントアパーチャを用いてフラッシュを追加する ネット番号と選択箇所を設定する

引数:

pt	フラッシュポイント
iNet	オブジェクトのネット番号
sSelection	選択オプション "all" 又は "sel" "sel" 指定の場合は オブジェクトが選択されたとしてマークされる

戻り値:
エラーステータス（0は成功 0以外は失敗）

int insertFlash (double *pt_x,*
double *pt_y*
)

カレントアパーチャを用いてフラッシュを追加する

引数:

pt_x (X 座標) フラッシュポイント
pt_y (Y 座標) フラッシュポイント

戻り値:

エラーステータス（0は成功 0以外は失敗）

int insertFlash (Point *pt*)

カレントアパーチャを用いてフラッシュを追加する

引数:

pt フラッシュポイント

戻り値:

エラーステータス（0は成功 0以外は失敗）

void insertFlashRepeat (double *pt_x*,
double *pt_y*,
int *iNx*,
double *dXstep*,
int *iNy*,
double *dYstep*,
String *sSelection*
)

Generates a step and repeat of flashes

引数:

pt_x	(X coordinate) The offset (start) point.
pt_y	(Y coordinate) The offset (start) point.
iNx	The number of repetitions in x direction.
dXstep	The step in x direction.
iNy	The number of repetitions in y direction.
dYstep	The step in y direction.
sSelection	The selection option. Either "all" or "sel". If "sel" is specified, the flashes are marked as selected.

void insertFlashRepeat (Point *pt*,
int *iNx*,
double *dXstep*,
int *iNy*,
double *dYstep*,
String *sSelection*
)

Generates a step and repeat of flashes

引数:

pt	The offset (start) point.

iNx	The number of repetitions in x direction.
dXstep	The step in x direction.
iNy	The number of repetitions in y direction.
dYstep	The step in y direction.
sSelection	The selection option. Either "all" or "sel". If "sel" is specified, the flashes are marked as selected.

```
void insertFlashRepeat ( double  pt_x,
                         double  pt_y,
                         int     iNx,
                         double  dXstep,
                         int     iNy,
                         double  dYstep
                       )
```

フラッシュのステップ&リピートを発生させる

引数:

pt_x	（X座標）オフセット（開始）ポイント
pt_y	（Y座標）オフセット（開始）ポイント
iNx	x方向のリピート回数
dXstep	x方向のステップ
iNy	y方向のリピート回数
dYstep	y方向のステップ

```
void insertFlashRepeat ( Point   pt,
                         int     iNx,
                         double  dXstep,
                         int     iNy,
                         double  dYstep
                       )
```

フラッシュのステップ&リピートを発生させる

引数:

pt	オフセット（開始）ポイント
iNx	x方向のリピート回数
dXstep	x方向のステップ
iNy	y方向のリピート回数
dYstep	y方向のステップ

```
void insertFullArc3Point ( double  pnt1_x,
                           double  pnt1_y,
                           double  pnt2_x,
                           double  pnt2_y,
                           double  pnt3_x,
                           double  pnt3_y
                         )
```

カレントアパーチャを使用し 完全なアークを挿入する 3つの外形ポイントからアークを計算する

引数:

pnt1_x （X座標）外形ポイント
pnt1_y （Y座標）外形ポイント
pnt2_x （X座標）外形ポイント
pnt2_y （Y座標）外形ポイント
pnt3_x （X座標）外形ポイント
pnt3_y （Y座標）外形ポイント

```
void insertFullArc3Point ( Point  pnt1,
                           Point  pnt2,
                           Point  pnt3
                         )
```

カレントアパーチャを使用し 完全なアークを挿入する 3つの外形ポイントからアークを計算する

引数:

pnt1 外形ポイント
pnt2 外形ポイント
pnt3 外形ポイント

```
void insertFullArc3Point ( double  pnt1_x,
                           double  pnt1_y,
                           double  pnt2_x,
                           double  pnt2_y,
                           double  pnt3_x,
                           double  pnt3_y,
                           int     iNet,
                           String  sSelection
                         )
```

カレントアパーチャを使用し 完全なアークを挿入する 3つの外形ポイントからアークを計算する

引数:

pnt1_x （X座標）外形ポイント
pnt1_y （Y座標）外形ポイント
pnt2_x （X座標）外形ポイント
pnt2_y （Y座標）外形ポイント
pnt3_x （X座標）外形ポイント
pnt3_y （Y座標）外形ポイント
iNet オブジェクトのネット番号
sSelection 選択オプション ”all”もしくは”sel”を設定 ”sel”を指定すると オブジェクトは選択されたことがわかるように表示される

```
void insertFullArc3Point ( Point  pnt1,
                           Point  pnt2,
                           Point  pnt3,
                           int    iNet,
```

```
                    String  sSelection
                )
```

カレントアパーチャを使用し 完全なアークを挿入する 3つの外形ポイントからアークを計算する

引数:

pnt1	外形ポイント
pnt2	外形ポイント
pnt3	外形ポイント
iNet	オブジェクトのネット番号
sSelection	選択オプション "all"もしくは"sel"を設定 "sel"を指定すると オブジェクトは選択されたことがわかるように表示される

```
void insertFullArcCenterRadius ( double  pntCenter_x,
                                 double  pntCenter_y,
                                 double  dRadius,
                                 String  sDirection
                               )
```

カレントアパーチャを使用し完全なアークを挿入する 半径と中心点からアークを計算する

引数:

pntCenter_x	（X座標）中心点
pntCenter_y	（Y座標）中心点
dRadius	アークの半径
sDirection	アークの方向

```
void insertFullArcCenterRadius ( Point   pntCenter,
                                 double  dRadius,
                                 String  sDirection
                               )
```

カレントアパーチャを使用し完全なアークを挿入する 半径と中心点からアークを計算する

引数:

pntCenter	中心点
dRadius	アークの半径
sDirection	アークの方向

```
void insertFullArcCenterRadius ( double  pntCenter_x,
                                 double  pntCenter_y,
                                 double  dRadius,
                                 String  sDirection,
                                 int     iNet,
                                 String  sSelection
                               )
```

カレントアパーチャを使用し完全なアークを挿入する 半径と中心点からアークを計算する

引数:

pntCenter_x （X座標）中心点
pntCenter_y （Y座標）中心点
dRadius アークの半径
sDirection アークの方向
iNet オブジェクトのネット番号
sSelection 選択オプション ”all”もしくは”sel”を設定 ”sel”を指定すると オブジェクトは選択されたことがわかるように表示される

void insertFullArcCenterRadius (Point *pntCenter,*
double *dRadius,*
String *sDirection,*
int *iNet,*
String *sSelection*
)

カレントアパーチャを使用し完全なアークを挿入する 半径と中心点からアークを計算する

引数:
pntCenter 中心点
dRadius アークの半径
sDirection アークの方向
iNet オブジェクトのネット番号
sSelection 選択オプション ”all”もしくは”sel”を設定 ”sel”を指定すると オブジェクトは選択されたことがわかるように表示される

boolean insertParallel (boolean *bSelection,*
ObjectList *oLines*
)

カレントアパーチャを用いて水平ドローを挿入する

引数:
bSelection 選択 trueが設定されると オブジェクトが選択されていることが分かるように表示される
oLines ドロー配列

戻り値:
エラーステータス（trueの場合は 何か追加されたことを示す）

void insertPolydrawRect (double *rect_xmin,*
double *rect_ymin,*
double *rect_xmax,*
double *rect_ymax,*
boolean *bSel*
)

Insert polydraw rectangle using current aperture

引数:
rect_xmin (left boundary of rectangle) rectangle to draw
rect_ymin (bottom boundary of rectangle) rectangle to draw

rect_xmax (right boundary of rectangle) rectangle to draw
rect_ymax (top boundary of rectangle) rectangle to draw
bSel if `true` selects the new created objects

void insertPolydrawRect (Rectangle *rect*,
boolean *bSel*
)

Insert polydraw rectangle using current aperture

引数:

rect rectangle to draw
bSel if `true` selects the new created objects

void insertPolydrawRect (double *rect_xmin*,
double *rect_ymin*,
double *rect_xmax*,
double *rect_ymax*
)

カレントアパーチャを用いて複数のドローからなる長方形を挿入する

引数:

rect_xmin （長方形の左側境界）描画したい長方形
rect_ymin （長方形のボトム境界）描画したい長方形
rect_xmax （長方形の右側境界）描画したい長方形
rect_ymax （長方形のトップ境界）描画したい長方形

void insertPolydrawRect (Rectangle *rect*)

カレントアパーチャを用いて複数のドローからなる長方形を挿入する

引数:

rect 描画したい長方形

void insertPolydrawRect (double *drawRectangle_xmin*,
double *drawRectangle_ymin*,
double *drawRectangle_xmax*,
double *drawRectangle_ymax*,
boolean *bRectCW*,
boolean *bSel*
)

カレントアパーチャを用いて複数のドローからなる長方形を挿入する

引数:

drawRectangle_xmin （長方形の左側境界）描画したい長方形
drawRectangle_ymin （長方形のボトム境界）描画したい長方形

drawRectangle_xmax （長方形の右側境界）描画したい長方形
drawRectangle_ymax （長方形のトップ境界）描画したい長方形
bRectCW 長方形をCW（右回り）にする場合 trueを設定
bSel 長方形を選択状態にす場合 trueを設定

void insertPolydrawRect (Rectangle *drawRectangle,*
boolean *bRectCW,*
boolean *bSel*
)

カレントアパーチャを用いて複数のドローからなる長方形を挿入する

引数:

drawRectangle 描画したい長方形
bRectCW 長方形をCW（右回り）にする場合 trueを設定
bSel 長方形を選択状態にす場合 trueを設定

void insertPolydrawRect (Point *p1,*
Point *p2,*
boolean *bRectCW,*
boolean *bSel*
)

非推奨:
カレントアパーチャを用いて複数のドローからなる長方形を挿入する

引数:

p1 長方形の左下ポイント
p2 長方形の右上ポイント
bRectCW 長方形をCW（右回り）にする場合 trueを設定
bSel 長方形を選択状態にす場合 trueを設定

void insertPolygon (boolean *bSelection,*
ObjectList *polygon*
)

カレントアパーチャを用いて多角形を挿入する

引数:

bSelection 選択 trueが指定されると オブジェクトは選択されていることが分かるように表示される
polygon ライン配列としての多角形

void insertPrePreg (int *topLayer,*
int *bottomLayer,*
String *sPosition,*
String *material,*
double *thickness,*

```
                String    reference,
                double    tolerance,
                double    erConstant,
                String    supplier,
                ObjectList attrNames,
                Object[]  attrValues
              )
```

Inserts prepreg with a given specification between the specified Layers.

引数:

topLayer	top Layer for prepreg
bottomLayer	bottom Layer for prepreg
sPosition	"Up" or "Down" value
material	String Material name
thickness	double Material thickness
reference	String Material reference
tolerance	double Material tolerance
erConstant	double Material ER constant
supplier	String Material supplier
attrNames	String list of the Material attribute names
attrValues	String the same sized list of the Material attribute values. If the Material attribute has no value put the empty string "", no `null` value!

```
void insertTab ( double  p_x,
                 double  p_y,
                 double  dis,
                 String  pat
               )
```

Replaces a part of a track or a part of a corner formed by two tracks by a predefined pattern. This pattern is actually a predefined DPF-job. You must position the pattern in the required position using the Numbers dialog box or by inserting it as a flash. The use on corners is limited to corners of 90 degrees, formed by a horizontal and a vertical tracks. The patterns are flashed on the corner point (for corners) or on the click point for single tracks.

引数:

p_x	(X coordinate) tab location
p_y	(Y coordinate) tab location
dis	the distance of the opening in the rout where the tab will be placed
pat	the current pattern

```
void insertTab ( Point   p,
                 double  dis,
                 String  pat
               )
```

Replaces a part of a track or a part of a corner formed by two tracks by a predefined pattern. This pattern is actually a predefined DPF-job. You must position the pattern in the required position using the Numbers dialog box or by inserting it as a flash. The use on corners is limited to corners of 90 degrees, formed by a horizontal and a vertical tracks. The patterns are flashed on the corner point (for corners) or on the click point for single tracks.

引数:

p tab location
dis the distance of the opening in the rout where the tab will be placed
pat the current pattern

void insertVectorText (double *pt_x*,
double *pt_y*,
String *sText*,
String *sFont*,
double *dWidth*,
double *dSpacing*,
String *sMirror*,
double *dRotation*,
double *dScale*
)

カレントアパーチャを用いてベクトルテキストを追加する

引数:

pt_x （X座標）フラッシュポイント
pt_y （Y座標）フラッシュポイント
sText String型テキスト
sFont ベクトルテキストフォント
dWidth (ODB標準 のような固定幅フォントの)文字幅
dSpacing (vtx のような可変幅フォントの)文字ピッチ
sMirror ミラー（設定しないか X Y XYのいずれかを設定）
dRotation 回転
dScale 両方向のスケール

void insertVectorText (Point *pt*,
String *sText*,
String *sFont*,
double *dWidth*,
double *dSpacing*,
String *sMirror*,
double *dRotation*,
double *dScale*
)

カレントアパーチャを用いてベクトルテキストを追加する

引数:

pt フラッシュポイント
sText String型テキスト
sFont ベクトルテキストフォント
dWidth （ ODB標準 のような固定幅フォントの）文字幅
dSpacing （ vtx のような可変幅フォントの）文字ピッチ
sMirror ミラー（設定しないか X Y XYのいずれかを設定）
dRotation 回転
dScale 両方向のスケール

```
void insertVectorText ( double  pt_x,
                        double  pt_y,
                        String  sText,
                        String  sFont,
                        double  dWidth,
                        double  dSpacing,
                        String  sMirror,
                        double  dRotation,
                        double  dScaleX,
                        double  dScaleY
                      )
```

カレントアパーチャを用いてベクトルテキストを追加する

引数:

pt_x	（X座標）フラッシュポイント
pt_y	（Y座標）フラッシュポイント
sText	String型テキスト
sFont	ベクトルテキストフォント
dWidth	（ ODB標準 のような固定幅フォントの）文字幅
dSpacing	（ vtx のような可変幅フォントの）文字ピッチ
sMirror	ミラー（設定しないか X Y XYのいずれかを設定）
dRotation	回転
dScaleX	X方向のスケール
dScaleY	Y方向のスケール

```
void insertVectorText ( Point   pt,
                        String  sText,
                        String  sFont,
                        double  dWidth,
                        double  dSpacing,
                        String  sMirror,
                        double  dRotation,
                        double  dScaleX,
                        double  dScaleY
                      )
```

カレントアパーチャを用いてベクトルテキストを追加する

引数:

pt	フラッシュポイント
sText	String型テキスト
sFont	ベクトルテキストフォント
dWidth	（ ODB標準 のような固定幅フォントの）文字幅
dSpacing	（ vtx のような可変幅フォントの）文字ピッチ
sMirror	ミラー（設定しないか X Y XYのいずれかを設定）
dRotation	回転
dScaleX	X方向のスケール
dScaleY	Y方向のスケール

**void intersectDraws (double *pt_x,*
double *pt_y*
)**

2つの交わるドロー上に交点を追加する

引数:
pt_x （X座標）交差位置
pt_y （Y座標）交差位置

void intersectDraws (Point *pt*)

2つの交わるドロー上に交点を追加する

引数:
pt 交差位置

boolean isDirectory (ObjectList *fileInfo*)

Tests whether the file denoted by this fileInfo is a directory.

引数:
fileInfo objectlist with the file information

戻り値:
true if and only if the file denoted by this fileInfo is a directory; false otherwise

参照:
HSH_base::osFileInfo(String)

**boolean isEqual (Object *oParam1,*
Object *oParam2*
)**

Compare 2 given objects

引数:
oParam1 it can be any of supported object
oParam2 it can be any of supported object

戻り値:
true if both are the same

boolean isFile (ObjectList *fileInfo*)

Tests whether the file denoted by this abstract is a normal file. A file is normal if it is not a directory and, in addition, satisfies other system-dependent criteria.

引数:

fileInfo objectlist with the file information

戻り値:
`true` if the file is normal file; `false` otherwise

参照:
HSH_base::osFileInfo(String)

boolean isHidden (ObjectList *fileInfo*)

Tests whether the file denoted by this fileInfo is a hidden file. The exact definition of hidden is system-dependent. On UNIX systems, a file is considered to be hidden if its name begins with a period character ('.'). On Microsoft Windows systems, a file is considered to be hidden if it has been marked as such in the filesystem.

引数:
fileInfo objectlist with the file information

戻り値:
`true` if and only if the file denoted by this fileInfo is hidden according to the conventions of the underlying platform

参照:
HSH_base::osFileInfo(String)

boolean isLayerInPlane (int *iPlaneNumber*)

Returns `true` when the layer in given plane exists, otherwise returns `false`

引数:
iPlaneNumber plane number

戻り値:
`true` when the layer in given plane exists, otherwise returns `false`

void job_save_shm_and_release (String *sShmName*)

save current job to shared memory

引数:
sShmName

int jobApeMaxNumber ( )

アパーチャ番号のうち最大番号を取得する

戻り値:
その時点までに使用されているアパーチャ番号の最大値 アパーチャが見つからない場合は -1を返す

String jobATEMachine ( )

ATE検査機用ファイルの仕様を返す

戻り値:
ATE検査機用ファイルの仕様を返す

**void jobAttribute (String *name,*
String *value*
)**

任意のジョブ属性に対し 任意の値を設定する 属性が存在すれば 既存の値は新しい値に変更される これに該当しなければ 属性は新規作成される 値がnull値の場合は その任意の名前に該当する属性は削除される

引数:
name ジョブ属性名
value ジョブ属性値

String jobAttribute (String *name*)

Returns the value of the Job attribute with given name.

引数:
name the job attribute name

戻り値:
the value of the Job attribute with given name or null if the attribute is not defined in the job.

String jobAttribute ( )

Returns comma separated list of the all Job attributes. **Example:** "uJobSize=mil,5840,2048,**adjacency.distance**=0,adjacency.blindnetsubst=1,uSessionPaiFile=D:¥data¥jobs¥_session.pai "

戻り値:
comma separated list of the all Job attributes.

**void jobCopperCount (boolean *bUseMask,*
boolean *bConfirmMaskUsage*
)**

Calculates the copper surface in job (all layers).

引数:

bUseMask	When true, active mask layers are taken into account: The "free of mask" area's are then calculated. The mask with attachment top is used for the top outer layer. The mask with attachment bottom is used for the bottom outer layer. The mask with attachment none is used for inner layers.
bConfirmMaskUsage	When true and active mask layers exist, then asks for confirmation if active mask layers should be used, or not.

void jobCopperCount ( )

Calculates the copper surface in job (all layers).

void jobCustomer (String *sCustomer*)

どの顧客向けのジョブなのか 顧客を設定する

引数:

sCustomer このジョブの対象顧客

String jobCustomer ( )

どの顧客向けのジョブなのか 顧客を取得する

戻り値:

どの顧客向けのジョブなのか 顧客を返す

void jobDRCParameters (String *sDrc*)

このジョブにおいて デザインルールチェックパラメータのファイルエントリーを設定する

引数:

sDrc ジョブにおけるデザインルールチェックパラメータのファイルエントリー

String jobDRCParameters ( )

このジョブにおいて デザインルールチェックパラメータのファイルエントリーを返す

戻り値:

ジョブにおけるデザインルールチェックパラメータのファイルエントリー

Rectangle jobEnclosingBox ( )

Gets the enclosing rectangle of the job.

戻り値:

Job enclosing rectangle.

void jobExtension (String *sExtension*)

ジョブの拡張子を設定する

引数:

sExtension

void jobFixture (String *sFixture*)

電気検査に使用する治具タイプを設定する

引数:

sFixture 治具タイプ "top", "bot", "ssa" , "dsa"のいずれか

String jobFixture ( )

電気検査に使用される治具タイプを返す 可能値は "top", "bot", "ssa" , "dsa"である

戻り値:

電気検査に使用される治具タイプ 可能値は "top", "bot", "ssa" , "dsa"である

boolean jobHasPattern (boolean *bUsed*)

Returns `true` if the job has an aperture with a pattern in active layers.

引数:

bUsed pattern is used when it affects the image.

戻り値:

`true` if the job has an aperture with a pattern in active layers.

void jobInfo (String[] *sInfo*)

ジョブ関連情報を設定する

引数:

sInfo ジョブに関する新情報

void jobInfo (String *sInfo*)

ジョブ関連情報を設定する

引数:

sInfo ジョブに関する新情報

String jobInfo ( )

オブジェクト関連情報を返す

戻り値:

オブジェクト関連情報

void jobLayMask (String *sLayMask*)

Sets the layer mask parameter for the Job. If the layer mask is already set in the Job, the function takes all active layer having current layer mask and replaces the layer name with the new layer mask.

Example:

```
String cadDir = "HOME:¥¥Data¥¥Cad";


openJob(cadDir + "¥¥cad.job");


String layMask = jobLayMask();
String layMask = jobLayMask();


activateAllLayers();


jobLayMask("cad_");
jobLayMask("cad_");

layMask = jobLayMask();
layMask = jobLayMask();



jobLayMask("Lepa_");
jobLayMask("Lepa_");

String trgDir = cadDir + "¥¥copy";
osMkDir(trgDir);

saveJobAs(trgDir + "¥¥lepa.job");
```

引数:

sLayMask the layer mask that will be set in the Job.

String jobLayMask ( )

Gets the layer mask currently set in the Job parameters.

戻り値:

the layer mask currently set in the Job parameters.

参照:

jobLayMask(String)

jobLayMask(String)

int jobMaxNetnumer ( )

ジョブにおけるネット番号の最大値を取得する

戻り値:

ジョブにおけるネット番号の最大値

void jobName (String *sName*)

ジョブ名を設定する

引数:

sName 新規名前

String jobName ( )

ジョブ名を返す

戻り値:
ジョブ名

boolean jobNetlist ( )

全てのポジデータがネット番号を持っているかをチェックする クラスがJL_EXTRAのレイヤに対しては行わない このチェックには レイヤをアクティブにする必要はない

戻り値:
ポジデータがネット番号を持つ場合は trueを返す

int jobNumApes ( )

ジョブ内のアパーチャ総数を返す

戻り値:
ジョブ内のアパーチャ総数

int jobNumBothExtras ( )

トップ・ボトムの両レイヤに添付のあるジョブにおいて 特殊レイヤ総数を返す

戻り値:
トップ・ボトム両レイヤに添付のあるジョブにおいて 特殊レイヤ総数を返す

int jobNumBothExtras (String *subClass*)

Returns the number of the extra layers in the job with the both attach.

引数:
subClass The subclass for the layers wanted. If not important specify "". The default offered subclasses are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".

戻り値:
the number of the extra layers in the job with the both attach.

int jobNumBottomExtras ( )

ボトムレイヤに添付のあるジョブにおいて 特殊レイヤ総数を返す

戻り値:

ボトムレイヤに添付のあるジョブにおける特殊レイヤ総数

int jobNumBottomExtras (String *subClass*)

Returns the number of the extra layers in the job with the bottom attach.

引数:

subClass The subclass for the layers wanted. If not important specify "". The default offered subclasses are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".

戻り値:

the number of the extra layers in the job with the bottom attach.

int jobNumCores ( )

ジョブのコア総数を返す

戻り値:

ジョブのコア総数

int jobNumDrills ( )

ジョブのドリルレイヤ総数を返す

戻り値:

ジョブのドリルレイヤ総数

int jobNumDrills (String *subClass*)

Returns the number of the drill layers in the job.

引数:

subClass The subclass for the layer wanted. If not important specify "". The default offered subclasses are "drill", "buried", "blind", "plated", "unplated" and "fixing".

戻り値:

the number of the drill layers in the job.

int jobNumExtras ( )

ジョブの特殊レイヤ総数を返す

戻り値:

ジョブの特殊レイヤ総数を返す

int jobNumExtras (String *subClass*)

Returns the number of the extra layers in the job.

引数:

subClass The subclass for the layers wanted. If not important specify "". The default offered subclasses are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".

戻り値:

the number of the extra layers in the job.

int jobNumLayers ( )

ジョブの全レイヤ総数を返す

戻り値:

ジョブの全レイヤ総数

int jobNumNoneExtras ( )

添付がないジョブにおける特殊レイヤの総数を返す

戻り値:

添付がないジョブにおける特殊レイヤ総数

int jobNumNoneExtras (String *subClass*)

Returns the number of the extra layers in the job with the none attach.

引数:

subClass The subclass for the layers wanted. If not important specify "". The default offered subclasses are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".

戻り値:

the number of the extra layers in the job with the none attach.

int jobNumPrepregs (int *start*)

任意のレイヤとそれに続くレイヤとの間にあるジョブにおいて プリプレグ総数を返す

引数:

start プリプレグのカウントを始めるレイヤインデックス

戻り値:

任意のレイヤとそれに続くレイヤとの間にあるジョブでのプリプレグ総数

int jobNumPrepregs ( )

ジョブのプリプレグ総数を返す

戻り値:
ジョブのプリプレグ総数

int jobNumSignals ( )

ジョブのシグナルレイヤ総数を返す

戻り値:
ジョブのシグナルレイヤ総数

int jobNumSignals (String *subClass*)

Returns the number of the signal layers in the job.

引数:
subClass The subclass for the layer wanted. If not important specify "". The default offered subclasses are "outer", "inner" and "mixed".

戻り値:
the number of the signal layers in the job.

int jobNumTopExtras ( )

トップレイヤに添付があるジョブにおける特殊レイヤ総数を返す

戻り値:
トップレイヤに添付があるジョブでの特殊レイヤ総数

int jobNumTopExtras (String *subClass*)

Returns the number of the extra layers in the job with the top attach.

引数:
subClass The subclass for the layers wanted. If not important specify "". The default offered subclasses are "rout", "silk", "mask", "paste", "exclusion", "netref", "testpoints", "probe" and "guideplate".

戻り値:
the number of the extra layers in the job with the top attach.

void jobPath (String *sPath*)

ジョブのパスを設定する

引数:
sPath ジョブのパス

String jobPath ( )

ジョブのパスを返す

戻り値:
ジョブのパス

void jobRevision (String *sRevision*)

ジョブのリビジョン番号を設定する

引数:
sRevision ジョブリビジョン番号

String jobRevision ( )

ジョブのリビジョン番号を返す

戻り値:
ジョブのリビジョン番号

int jobSelectCount (String *sOption*)

Returns the number of selected objects in a job.

引数:
sOption Specifies the type of the objects to count. "a" for arcs, "f" for flashes, "d" for draws, "r" for regions and "v" for vector text.

戻り値:
the number of selected objects.

int jobSelectCount ( )

ジョブの選択オブジェクト数を返す

戻り値:
選択オブジェクト数

boolean jobSelection ( )

このジョブが 選択状態のアイテムを含んでいるかを調べる

戻り値:
選択されているものがあればtrue なければfalseを返す

Rectangle jobSelectionEnclosingBox ( )

Gets the enclosing rectangle of the job selection.

戻り値:
Job selection enclosing rectangle.

**void jobSize (double *pntSize_x,*
double *pntSize_y*
)**

ジョブサイズを設定する

引数:
pntSize_x （X座標）ジョブの新規サイズ
pntSize_y （Y座標）ジョブの新規サイズ

void jobSize (Point *pntSize*)

ジョブサイズを設定する

引数:
pntSize ジョブの新規サイズ

**void jobSize (String *sUnit,*
double *pntSize_x,*
double *pntSize_y*
)**

ジョブサイズを設定する

引数:
sUnit サイズの単位
pntSize_x （X座標）ジョブの新規サイズ
pntSize_y （Y座標）ジョブの新規サイズ

**void jobSize (String *sUnit,*
Point *pntSize*
)**

ジョブサイズを設定する

引数:
sUnit サイズの単位
pntSize ジョブの新規サイズ

void jobSize (String *sSize*)

ジョブサイズを設定する

引数:

sSize String型で "mil,0,0"（単位, Xサイズ, Yサイズ）のように記述する

Point jobSize ( )

Returns job size using uJobSize attribute value.

戻り値:
Point representing job size, x and y size.

void jobSpec (String *sSpec*)

ジョブファイルのフル仕様を設定する

引数:
sSpec ジョブファイルのフル仕様

String jobSpec ( )

ジョブファイルのフル仕様を返す

戻り値:
ジョブファイルのフル仕様

void jobUserData (String *sUserData*)

ジョブのユーザーデータを設定する

引数:
sUserData ジョブのユーザーデータ

String jobUserData ( )

ジョブのユーザーデータを返す

戻り値:
ジョブのユーザーデータ

void lajCleanLegendLayer (boolean *DoMask*,
double *MaskClearance*,
boolean *DoCu*,
double *CuClearance*,
boolean *DoCuPads*,
double *CuPadClearance*,
boolean *bDoCuFOM*,
double *iCuFOMClearance*,
boolean *bDoCuPadsFOM*,

```
                        double   iCuPadsFOMClearance,
                        boolean  doPlatedDrills,
                        double   platedDrillClearance,
                        boolean  doUnplatedDrills,
                        double   UnplatedDrillClearance,
                        boolean  DoSmallDraws,
                        double   MinDrawSize
                      )
```

Cleans all active silk layers against all active matching copper + drill + mask layers with the presented clearances

引数:

DoMask	Clip around mask
MaskClearance	Min. mask clearance
DoCu	Clip around outer layer
CuClearance	Min. outer clearance
DoCuPads	Clip around copper pads
CuPadClearance	Min. copper pad clearance
bDoCuFOM	Clip around copper not covered by mask
iCuFOMClearance	Min. unmasked copper clearance
bDoCuPadsFOM	Clip around copper pads not covered by mask
iCuPadsFOMClearance	Min. unmasked copper pad clearance
doPlatedDrills	Clip around plated drills
platedDrillClearance	Min. plated drill clearance
doUnplatedDrills	Clip around unplated drills
UnplatedDrillClearance	Min. unplated drill clearance
DoSmallDraws	Remove small objects
MinDrawSize	Min. object size

```
void lajDefineWord (  )
```

Turns selected objects on all active layers into a word

```
void lajDeselectAllWords (  )
```

Deselect all textboxes on all active layers

```
void lajDragWord ( double   pt_x,
                   double   pt_y,
                   double   radius,
                   double   offset_x,
                   double   offset_y,
                   double   limit,
                   boolean  enforcelimit
                 )
```

Drag move a word, used in mousetool

引数:

pt_x	(X coordinate) **Point** where word is
pt_y	(Y coordinate) **Point** where word is
radius	Radius in which to search for a word
offset_x	(X coordinate) Contains horizontal and vertical distance to move
offset_y	(Y coordinate) Contains horizontal and vertical distance to move
limit	maximal distance word can be moved
enforcelimit	whether to enforce the distance limit

void lajDragWord (Point *pt*,
double *radius*,
Point *offset*,
double *limit*,
boolean *enforcelimit*
)

Drag move a word, used in mousetool

引数:

pt	**Point** where word is
radius	Radius in which to search for a word
offset	Contains horizontal and vertical distance to move
limit	maximal distance word can be moved
enforcelimit	whether to enforce the distance limit

void lajLegendDRC (boolean *bDoLineWidth*,
double *dMinLineWidth*,
boolean *bDoMask*,
double *dMaskClearance*,
boolean *bDoCu*,
double *dCuClearance*,
boolean *bDoCuPads*,
double *dCuPadClearance*,
boolean *bDoCuFOM*,
double *dCuFOMClearance*,
boolean *bDoCuPadsFOM*,
double *dCuPadsFOMClearance*,
boolean *bDoPlatedDrills*,
double *dPlatedDrillClearance*,
boolean *bDoUnplatedDrills*,
double *dUnplatedDrillClearance*,
boolean *bDoSmallDraws*,
double *dMinDrawSize*
)

Does DRC check on legend layers against other active layers

引数:

bDoLineWidth	Check line width
dMinLineWidth	Min. line width

bDoMask	Check mask clearance
dMaskClearance	Min. mask clearance
bDoCu	Check outer clearance
dCuClearance	Min. outer clearance
bDoCuPads	Check copper pad clearance
dCuPadClearance	Min. copper pad clearance
bDoCuFOM	Check clearance to copper free of mask
dCuFOMClearance	Min. unmasked copper clearance
bDoCuPadsFOM	Check clearance to copper pads free of mask
dCuPadsFOMClearance	Min. unmasked copper pad clearance
bDoPlatedDrills	Check clearance to plated drills
dPlatedDrillClearance	Min. plated drill clearance
bDoUnplatedDrills	Check clearance to unplated drills
dUnplatedDrillClearance	Min. unplated drill clearance
bDoSmallDraws	Check for small objects
dMinDrawSize	Min. object size

**void lajLegendTextToWords (double *dMaxSize,*
int *iMaxSpacing*
)**

Converts text to rectangles, splitting vertical and horizontal text, on all active silk layers

引数:

dMaxSize	maximal text size
iMaxSpacing	maximal letter spacing, in percent of average letter width

**void lajMoveWord (String *value,*
double *dx,*
double *dy,*
double *limit*
)**

Move word with given uText value over distance dx, dy, unless the distance is larger than limit and enforcelimit is true

引数:

value	uText attribute value of the word to be moved
dx	horizontal distance
dy	vertical distance
limit	maximal distance word can be moved

**void lajMoveWord (String *value,*
double *dx,*
double *dy*
)**

Move word with given uText value over distance dx, dy

引数:

value uText attribute value of the word to be moved
dx horizontal distance
dy vertical distance

void lajScaleWord (String *value,*
double *factor,*
double *limit*
)

Scale the word with the give uText value with the given factor, but don't make apertures smaller than limit

引数:

value uText attribute value of the word to be scaled
factor scale factor
limit min. aperture size after scaling

void lajScaleWord (String *value,*
double *factor*
)

Scale the word with the give uText value with the given factor

引数:

value uText attribute value of the word to be scaled
factor scale factor

void lajScaleWordOnPt (double *pt_x,*
double *pt_y,*
double *radius,*
double *scale,*
double *limit,*
boolean *enforcelimit*
)

Scale word nearest to point with given factor, keeping aperture larger than min. size. Used in mousetool.

引数:

pt_x (X coordinate) **Point** where word is
pt_y (Y coordinate) **Point** where word is
radius Radius in which to search for a word
scale factor by which to scale word
limit min. aperture width after scaling
enforcelimit whether to enforce the aperture size limit

void lajScaleWordOnPt (Point *pt,*
double *radius,*
double *scale,*

double *limit,*
boolean *enforcelimit*
)

Scale word nearest to point with given factor, keeping aperture larger than min. size. Used in mousetool.

引数:

pt	**Point** where word is
radius	Radius in which to search for a word
scale	factor by which to scale word
limit	min. aperture width after scaling
enforcelimit	whether to enforce the aperture size limit

void lajSelectAllWords ( )

Select all textboxes on all active layers

void lajUndefineWord ( )

Removes uText attribute from selected objects on active layers

void layActive (ObjectList *layerID,*
boolean *bActive*
)

Sets activity on the Layer defined by layer ID

引数:

layerID	the layer ID e.g. from **getLayers()** function
bActive	`true` if the layer defined by layer ID will be set as active; `false` for remove activity on a layer defined by the layer ID

参照:

getLayers()

boolean layActive (ObjectList *layerID* **)**

Returns activity of the Layer with the given layer ID

引数:

layerID	the layer ID e.g. from **getLayers()** function

戻り値:

`true`/`false` activity on a layer defined by the layer ID

参照:

getLayers()

void layActive (boolean *bActive*)

Sets activity on the current Layer

引数:

bActive `true` if the current layer will be set as active; `false` for remove activity on the current layer

boolean layActive ( )

Returns activity of the current Layer

戻り値:

`true`/`false` activity of the current Layer

void layAlias (String *sAlias*)

Sets new alias to the current Layer.

引数:

sAlias String new alias of the current Layer.

String layAlias ( )

Returns current Layer alias.

戻り値:

current Layer alias

int layApeCount ( )

Return numbers of aperture on current layer

戻り値:

Number of apertures on current layer. If the layer does not exist return -1

void layAttach (String *sAttach*)

特殊レイヤの添付を設定する

引数:

sAttach 可能値は"top", "bottom", "none", "both"

String layAttach ( )

特殊レイヤの添付を取得する

戻り値:

"top", "bottom", "none", "both"のいずれか

**void layAttribute (String *name,*
String *value*
)**

任意のレイヤ属性に任意の値を設定する　属性が存在していれば その値は新しい値に変更される そうでなければ属性が新規作成される　値がnullの場合は 任意の名前に該当する属性が削除される

引数:
name レイヤ属性名
value レイヤ属性値

String layAttribute (String *name*)

Returns the value of the layer attribute with given name.

引数:
name the layer attribute name

戻り値:
the value of the layer attribute with given name or null if the attribute is not defined in the layer.

String layAttribute ( )

Returns comma separated list of the all layer attributes. **Example:** "uMatReference=R-5715-4,uMatTolerance=0.0076,uMatSupplier=Ucamco,form="

戻り値:
comma separated list of the all layer attributes.

void layClass (String *sNewClass*)

カレントレイヤのクラスを新しいクラスに変更する

引数:
sNewClass レイヤクラス 可能値は"layer", "drill", "extra"のいずれか

String layClass ( )

カレントレイヤのクラスをString型データとして取得する

戻り値:
"layer", "drill", "extra"のいずれか

**void layCopperCount (boolean *bUseMask,*
boolean *bConfirmMaskUsage*
)**

Calculates the copper surface in active layers.

引数:

bUseMask	When true, active mask layers are taken into account: The "free of mask" area's are then calculated. The mask with attachment top is used for the top outer layer. The mask with attachment bottom is used for the bottom outer layer. The mask with attachment none is used for inner layers.
bConfirmMaskUsage	When true and active mask layers exist, then asks for confirmation if active mask layers should be used, or not.

void layCopperCount ( )

Calculates the copper surface in active layers.

Rectangle layEnclosingBox ( )

カレントレイヤが含んでいる長方形囲み領域を取得する

戻り値:

レイヤが含む長方形囲み領域

void layerViewSplit (boolean *on*)

Set variable to enable/diable displaying split lines in the Job View dialog Call repaint for the dialog Job View

引数:

on if true will be Job View will be set to Split View

int layExtractPlotStamps (String *dstLayName,*
String *sOptions,*
ObjectList *sFilters*
)

extract plotstamps into another layer, keep aperture an object attributes

引数:

dstLayName	destination layer name (if null or "" or layer with given name does not exist, returns only count)
sOptions	"TEXT" or "VTXT" or "BLO" or combination like "TEXT,VTX,BLO", also flag "DELETE" to delete original objects and "KEEPLINK" to keep link for crosslinked blocks
sFilters	list of strings to match when extracting (if no match, no extraction)(null=empty=no Filter=extract all) **Example:** [{"%Level", "%BatchNo"}]

戻り値:

nr. of found objects

void layFrom (int *layFrom*)

ドリルホールのスタートレイヤの番号を設定する

引数:

layFrom ドリルホールのスタートレイヤの番号

int layFrom ( )

ドリルホールのスタートレイヤの番号を取得する

戻り値:

ドリルホールのスタートレイヤの番号

boolean layHasPattern (boolean *bUsed*)

Returns `true` if the layer has an aperture with a pattern.

引数:

bUsed pattern is used when it affects the image.

戻り値:

`true` if the layer has an aperture with a pattern.

ObjectList layID ( )

Returns ObjectArray with the current layer specification

戻り値:

ObjectArray with layer specification (name, class, subclass, attachment, index, activity, number of apertures) **Example:** [{"cad_m1", "extra", "mask", "bottom", 1, true, -1}]

参照:

getLayers()

void layIndex (int *iIndex*)

Sets the index of the current layer in the array of layers in the job this layer belongs to.

引数:

iIndex the new index of the current layer in the array of layers in the job this layer belongs to.

int layIndex ( )

Gets the index of the current layer in the array of layers in the job this layer belongs to.

戻り値:

the index of the current layer in the array of layers in the job this layer belongs to. -1 if the current layer is not set.

void layInfo (String *sText*)

カレントレイヤに情報を設定する

引数:

sText String型 レイヤ関連情報

String layInfo ( )

Gets the information associated with the current layer.

戻り値:

the information associated with the current layer. `null` if the current layer is not set.

void layMaterial (String *sMaterial*)

Sets the layer material (extra or layer class).

引数:

sMaterial the new layer material.

String layMaterial ( )

Gets the layer material (extra or layer class).

戻り値:

the layer material (extra or layer class). null if the current Layer is not set.

void layName (String *sName*)

カレントレイヤに新しい名前を設定する

引数:

sName カレントレイヤの新しい名前（String型）

String layName ( )

カレントレイヤ名を返す

戻り値:

カレントレイヤ名

void layNumber (int *iNumber*)

Sets the layer number in the group of extra layers with the same class, subclass and attach.

引数:

iNumber the new layer number in the group of extra layers with the same class, subclass and attach.

int layNumber ( )

Gets the layer number in the group of extra Layers with the same class, subclass and attach.

戻り値:
the layer number in the group of extra Layers with the same class, subclass and attach. -1 if the current Layer is not set.

void layReadable (String *sSide*)

レイヤの面視を設定する

引数:
sSide “top”もしくは”bottom”を設定 （String型）

String layReadable ( )

レイヤの面視を取得する

戻り値:
“top”もしくは”bottom”を返す

void layReverse (boolean *bReverse*)

Sets the reverse flag for the current Layer.

引数:
bReverse boolean true if the layer has negative data. (Its polarity is opposite compared to the produced copper image.)

boolean layReverse ( )

Has the layer negative data?

戻り値:
true if the layer has negative data. (Its polarity is opposite compared to the produced copper image.)

boolean laySelection ( )

Checks if the current Layer contains selected items.

戻り値:
`true` if there are selections, `false` otherwise.

Rectangle laySelectionEnclosingBox ( )

Gets the enclosing rectangle of the selection of the current layer.

戻り値:
Layer selection enclosing rectangle.

void laySubClass (String *sSubClass*)

Sets the subclass of current layer.

引数:
sSubClass String the default subclasses offered by Ucam are "rout", "silk", "mask", "paste", "exclusion", "netref", "frame", "testpoints", "guideplate" and "probe".

String laySubClass ( )

Gets the subclass of current layer. The default subclasses offered by Ucam are "rout", "silk", "mask", "paste", "exclusion", "netref", "frame", "testpoints", "guideplate" and "probe".

戻り値:
the subclass of current layer eg. "rout", "silk", "mask", "paste", "exclusion", "netref", "frame", "testpoints", "guideplate" and "probe".

void layThickness (double *dThickness*)

カレントレイヤの厚みを設定する

引数:
dThickness カレントレイヤの厚み（double型）

double layThickness ( )

カレントレイヤの厚みを取得する

戻り値:
カレントレイヤの厚み

void layTo (int *layTo*)

ドリルホールが終了するレイヤの番号を設定する

引数:
layTo ドリルホールが終了するレイヤの番号

int layTo ( )

ドリルホールが終了するレイヤの番号を取得する

戻り値:
ドリルホールが終了するレイヤの番号

double layZPos ( )

Return the x position of the current layer.

戻り値:
the x position of the current layer.

void liftUpUpcbBlocks ( )

Lift up blocks with uPcb attribute to the begin of the layer. Image is kept.

Line Line (double *ptFromX*,
double *ptFromY*,
double *ptToX*,
double *ptToY*,
String *units*
)

4座標からなるラインを作成する

引数:
ptFromX 始点X座標
ptFromY 始点Y座標
ptToX ライン終点X座標
ptToY ライン終点Y座標
units Ucam単位

戻り値:
ライン

Line Line (double *ptFromX*,
double *ptFromY*,
double *ptToX*,
double *ptToY*
)

4座標からなるラインを作成する

引数:
ptFromX 始点X座標
ptFromY 始点Y座標
ptToX ライン終点X座標
ptToY ライン終点Y座標

戻り値:

ライン

Line Line (Line *line*)

ラインのコピーを作成する

引数:
line コピー元になるライン

戻り値:
ライン

Line Line (Point *ptFrom*,
Point *ptTo*
)

2ポイントからなるラインを作成する

引数:
ptFrom ライン始点
ptTo ライン終点

戻り値:
ライン

ObjectList listFrames ( )

Lists names of all custom dialogs

戻り値:
Object Array with custom dialogs names. Name can be used in openFrame command.

参照:
openFrame(String)

void loadApertures (String *sDpfFile*)

アパーチャマネジャー: DPFファイルからアパーチャをロードする

引数:
sDpfFile アパーチャを読み込むDPFファイル

void loadBuildup (String *buildupSpec*)

Loads the buildup from the given .jot file into the current job. The layer names of the current job are overwritten.

引数:
buildupSpec the .jot file path with the buildup content to be loaded.

**void loadFrames (boolean *bVerbose,*
boolean *bLoadOnce*
)**

Loads all custom dialogs (dialogs extending CustomFrame class).

引数:

bVerbose detailed information about loading is visible if set to `true`

bLoadOnce Load dialog only once if set to `true`. A `false` value is useful when a dialog is under development and needs to be loaded every time the command is used in script.

参照:

CustomFrame

void loadFrames ( )

Loads all custom dialogs (dialogs extending CustomFrame class).

参照:

loadFrames(boolean, boolean)

loadFrames(boolean, boolean)

void loadSplitConfig (String *sConfigName*)

Load split configuration for the given name

引数:

sConfigName name of configuration

void loadUFD (String *sUFDName*)

Loads all the faults from the given file into the current fault database.

引数:

sUFDName UFD file specification

void loadWorkspace (String *sWorkspaceName*)

Discard modifications and reload current layout. NOTE: The same as menu command Workspaces > *workspace_name*

引数:

sWorkspaceName

void loadWorkspace ( )

Discard modifications and reload current layout. NOTE: The same as menu command Workspaces > Reload

double maxInvalidArcsDeviation ( )

Find and return maximal deviation of all (selected) invalid arcs The deviation is from distance Center point to To point or Center point to From point Current units will be used

戻り値:
maximal deviation in current units of all (selected) invalid arcs

int measureFingers (String *szOption*)

Measure gold fingers (ASE)

引数:
szOption The given options: "attribute" or "selection"

戻り値:
0: ok, 1: an error occurred

void measureLayers ( )

アクティブ選択の寸法を測定する

void measureObjects (double *p1_x*,
double *p1_y*,
double *p2_x*,
double *p2_y*
)

オブジェクト間のクリアランスを測定する 結果はナンバーズダイアログに表示される

引数:
p1_x （X座標）1つ目のオブジェクトのフラッシュポイント
p1_y （Y座標）1つ目のオブジェクトのフラッシュポイント
p2_x （X座標）2つ目のオブジェクトのフラッシュポイント
p2_y （Y座標）2つ目のオブジェクトのフラッシュポイント

void measureObjects (Point *p1*,
Point *p2*
)

オブジェクト間のクリアランスを測定する 結果はナンバーズダイアログに表示する

引数:
p1 1つ目のオブジェクトのフラッシュポイント
p2 2つ目のオブジェクトのフラッシュポイント

**void measurePoints (double *pt_x,*
double *pt_y*
)**

Sets the point1 and point2 in Numbers dialog as the same point

引数:

pt_x (X coordinate) first and the same second point
pt_y (Y coordinate) first and the same second point

void measurePoints (Point *pt*)

Sets the point1 and point2 in Numbers dialog as the same point

引数:

pt first and the same second point

**void measurePoints (double *p1_x,*
double *p1_y,*
double *p2_x,*
double *p2_y*
)**

ポイント間のクリアランスを測定する　結果はナンバーズダイアログに表示される

引数:

p1_x （X座標）1つ目オブジェクト
p1_y （Y座標）1つ目オブジェクト
p2_x （X座標）2つ目オブジェクト
p2_y （Y座標）2つ目オブジェクト

**void measurePoints (Point *p1,*
Point *p2*
)**

ポイント間のクリアランスを測定する　結果はナンバーズダイアログに表示される

引数:

p1 1つ目オブジェクト
p2 2つ目オブジェクト

void mergeContours ( )

輪郭を合成する

void mergeContoursSingle ( )

Merge Contours Single

void mergeContoursSingleAdd ( )

Merge Contours Single Add

void mergeLayers (String *posNegAlt,*
boolean *delLay*
)

レイヤを合成する

引数:

posNegAlt 合成オプション("Positive", "Negative", "Alternate")
delLay trueの場合 結合レイヤを削除する

void mirror (String *axis,*
boolean *bUseCenter,*
boolean *bOnRefPoints*
)

軸を中心に選択にミラーをかける

引数:

axis 軸（XもしくはY）の値
bUseCenter trueの場合 センター中心にミラーがかかる
bOnRefPoints trueの場合 リファレンスポイントにもミラーが適用される

void models (String *sModelShape,*
double *dTolerance*
)

Models replace selected painted shape on active layers by standard or complex shape

引数:

sModelShape Shape of selected model: standard or complex
dTolerance tolerance to be used when searching for model instances

参照:

models(String)

models(String)

void models (String *sModelShape*)

Models replace selected painted shape on active layers by standard or complex shape try count the best

tolerance to be used when searching for model instances.

引数:

sModelShape Shape of selected model: standard or complex

参照:

models(String, double)

models(String, double)

boolean modelsCreateComplex ( )

modelsCreateComplex 選択されたオブジェクトと同形状のコンプレックスを作成する 事前にmodelsDefineSelections()がファンクションコールされていなければならない

戻り値:

ステータス コンプレックスアパーチャが作成されれば ステータスはtrueである

参照:

models(String)

models(String, double)

boolean modelsCreateStandard (double *dTolerance*)

modelsCreateStandard 選択オブジェクトに類似する標準アパーチャを探す 事前にmodelsDefineSelections()がファンクションコールされていなければならない

引数:

dTolerance 許容量

戻り値:

ステータス アパーチャが見つかれば ステータスはtrueである

参照:

models(String)

models(String, double)

Rectangle modelsDefineSelections ( )

modelsDefineSelections プレーン1（plane1）のアクティブレイヤの選択箇所を定義する このファンクションは モデルダイアログの他のファンクションに先立って実行しなくてはならない

戻り値:

一時レイヤにあるボックス囲み領域（長方形囲み領域）

参照:

models(String)

models(String, double)

int modelsReplace (double *pntTolerance_x*, double *pntTolerance_y*

)

modelsReplace プレーン1のアクティブレイヤで 選択オブジェクトを作成したモデルに置換する 事前に`modelsDefineSelections()`がファンクションコールされていなければならない

引数:
pntTolerance_x （X座標）許容量
pntTolerance_y （Y座標）許容量

戻り値:
置換したアパーチャ数を返す -1はエラーを示す

参照:
models(String)

models(String, double)

int modelsReplace (Point *pntTolerance*)

modelsReplace プレーン1のアクティブレイヤで 選択オブジェクトを作成したモデルに置換する 事前に`modelsDefineSelections()`がファンクションコールされていなければならない

引数:
pntTolerance 許容量

戻り値:
置換したアパーチャ数を返す -1はエラーを示す

参照:
models(String)

models(String, double)

**int modelsSelect (double *pntTolerance_x*,
double *pntTolerance_y*
)**

modelsSelect オブジェクトを選択する 事前にmodelsDefineSelections()がファンクションコールされていなければならない

引数:
pntTolerance_x （X座標）許容量
pntTolerance_y （Y座標）許容量

戻り値:
選択されたオブジェクト数

参照:
models(String)

models(String, double)

int modelsSelect (Point *pntTolerance*)

modelsSelect オブジェクトを選択する 事前にmodelsDefineSelections()がファンクションコールされていな

ければならない

引数:

pntTolerance 許容量

戻り値:

選択されたオブジェクト数

参照:

models(String)

models(String, double)

void modifyCore (int *iTopLay*,
String *sAtt*,
int *iNewTopLay*,
int *iNewBotLay*,
String *sNewAtt*,
double *dThickness*,
String *sMaterial*,
String *sInfo*
)

コア属性を修正する

引数:

iTopLay	コアが接しているトップレイヤのインデックス
sAtt	コアの添付
iNewTopLay	コアが接する新しいトップレイヤのインデックス
iNewBotLay	コアが接する新しいボトムレイヤのインデックス
sNewAtt	"top", "bottom", "both","none"のいずれか　コアの添付を定義する
dThickness	コア厚みを定義する
sMaterial	コア材料を定義する
sInfo	コアの説明/情報を定義する

void modifyDrill (String *sName*,
String *sAlias*,
String *sClass*,
String *sSubClass*,
int *iFrom*,
int *iTo*,
double *dThickness*
)

Modify Drill

引数:

sName	new layer name
sAlias	new alias
sClass	class of the changed layer - "Layer", "Drill" or "Extra"
sSubClass	subclass of the changed layer

- **for class "Layer"** - "outer", "inner", "mixed"

- **for class "Drill"** - "drill", "buried", "blind", "plated", "unplated", "fixing"
- **for class "Extra"** - "rout", "score", "outline", "mask", "silk", "clipping", "coupon", "netref", "testpoints", "midpoints", "probe", "guideplate", "frame", "bolref", "bolinsp", "resistor", "dro", "snapshot", "help", "adjacency"

iFrom	index of the first drilled layer from top
iTo	index of the last drilled layer from top
dThickness	thickness of the modified layer

```
void modifyExtra ( String  sName,
                   String  sAlias,
                   String  sClass,
                   String  sSubClass,
                   String  sAttach,
                   int     iNumber,
                   boolean bReverse,
                   String  sMaterial
                 )
```

Modify Extra layer

引数:

sName	new layer name
sAlias	new alias
sClass	class of the changed layer - "Layer", "Drill" or "Extra"
sSubClass	subclass of the changed layer

- **for class "Layer"** - "outer", "inner", "mixed"
- **for class "Drill"** - "drill", "buried", "blind", "plated", "unplated", "fixing"
- **for class "Extra"** - "rout", "score", "outline", "mask", "silk", "clipping", "coupon", "netref", "testpoints", "midpoints", "probe", "guideplate", "frame", "bolref", "bolinsp", "resistor", "dro", "snapshot", "help", "adjacency"

sAttach	Attach - "top" or "bottom", "none"
iNumber	new position of the modified layer
bReverse	true - modified layer will be marked as reverse
sMaterial	Name of the material

```
void modifyFeedback ( String  sName,
                      String  sAlias,
                      String  sClass,
                      String  sSubClass,
                      String  sAttach
                    )
```

Modify Feedback layer

引数:

sName	new layer name
sAlias	new alias
sClass	class of the changed layer - "Layer", "Drill" or "Extra"
sSubClass	subclass of the changed layer

- **for class "Layer"** - "outer", "inner", "mixed"
- **for class "Drill"** - "drill", "buried", "blind", "plated", "unplated", "fixing"
- **for class "Extra"** - "rout", "score", "outline", "mask", "silk", "clipping", "coupon", "netref", "testpoints", "midpoints", "probe", "guideplate", "frame", "bolref", "bolinsp", "resistor", "dro", "snapshot", "help", "adjacency"

sAttach Attach - "top" or "bottom", "none"

```
void modifyLayer ( String  sName,
                   String  sAlias,
                   String  sClass,
                   String  sSubClass,
                   int     iNumber,
                   boolean bReverse,
                   double  dZPosition,
                   String  sReadable,
                   String  sMaterial,
                   double  dThickness
                 )
```

Modify signal Layer

引数:

sName new layer name
sAlias new alias
sClass class of the changed layer - "Layer", "Drill" or "Extra"
sSubClass subclass of the changed layer

- **for class "Layer"** - "outer", "inner", "mixed"
- **for class "Drill"** - "drill", "buried", "blind", "plated", "unplated", "fixing"
- **for class "Extra"** - "rout", "score", "outline", "mask", "silk", "clipping", "coupon", "netref", "testpoints", "midpoints", "probe", "guideplate", "frame", "bolref", "bolinsp", "resistor", "dro", "snapshot", "help", "adjacency"

iNumber new position of the modified layer
bReverse true - modified layer will be marked as reverse
dZPosition ZPozition
sReadable Readable side "top" or "bottom"
sMaterial Name of the material
dThickness thickness of the modified layer

```
void modifyPrePreg ( int    iTopLay,
                     int    iIndex,
                     int    iNewTopLay,
                     int    iNewBotLay,
                     int    iNewIndex,
                     double dThickness,
                     String sMaterial,
                     String sInfo
                   )
```

プリプレグ属性を修正する

引数:

iTopLay	プリプレグが接するトップレイヤのインデックス
iIndex	プリプレグ積層におけるプリプレグインデックス
iNewTopLay	プリプレグが接する新たなトップレイヤインデックス
iNewBotLay	プリプレグが接する新たなボトムレイヤのインデックス
iNewIndex	プリプレグ積層における新たなプリプレグインデックス
dThickness	プリプレグの厚みを定義する
sMaterial	プリプレグ材料を定義する
sInfo	プリプレグの説明/情報を定義する

void move (double *pt_x,*
double *pt_y,*
boolean *bOnRefPoints*
)

Move (selected) object(s) using board coordinates

Example:

```
setInPlane(1,1);
direction("");
move(100,200,false);
move(100,200,false);
doMove(100,200);

direction("h");
move(100,200,false);
move(100,200,false);
doMove(100,200);

direction("v");
move(100,200,false);
move(100,200,false);
doMove(100,200);
```

引数:

pt_x (X coordinate) Offset (vector) where to create the copy of the source objects

参照:

com.barco.ets.ucam.hypershell.HyperShell::doMove(Upoint)

com.barco.ets.ucam.hypershell.HyperShell::direction(String)

引数:

pt_y (Y coordinate) Offset (vector) where to create the copy of the source objects

参照:

com.barco.ets.ucam.hypershell.HyperShell::doMove(Upoint)

com.barco.ets.ucam.hypershell.HyperShell::direction(String)

引数:

bOnRefPoints If true, move is also applied to reference points

void move (Point *pt,*
boolean *bOnRefPoints*
)

Move (selected) object(s) using board coordinates

Example:

```
setInPlane(1,1);
direction("");
move(100,200,false);
move(100,200,false);
doMove(100,200);

direction("h");
move(100,200,false);
move(100,200,false);
doMove(100,200);

direction("v");
move(100,200,false);
move(100,200,false);
doMove(100,200);
```

引数:

pt Offset (vector) where to create the copy of the source objects

参照:

com.barco.ets.ucam.hypershell.HyperShell::doMove(Upoint)

com.barco.ets.ucam.hypershell.HyperShell::direction(String)

引数:

bOnRefPoints If true, move is also applied to reference points

void netlistBuild (String *target*)

ネットリストを構築する

引数:

target ネットリスト対象（jobもしくはlayer） "job" - ジョブ全体のネットリスト情報を構築する アクティブ/非アクティブに関係なく 全レイヤが使用される まだメモリーにないレイヤは 自動的にロードされる "layer" - ジョブのネットリスト情報を構築する アクティブレイヤのみが使用される

void netlistClear ( )

ネットリストを削除する ジョブのネットリスト情報を全て削除する

void netlistReference (String *target*)

ネットリストのリファレンスを作成する

引数:

target ネットリストのリファレンス対象（jobもしくはlayer）

**void newJob (String *jobPath,*
String *jobName*
)**

任意の名で新規ジョブを作成する 必要であれば ディレクトリが自動作成される ジョブが既に存在する場合 警告が表示され 既存のジョブがロードされる

引数:

jobPath ジョブファイルのパス名
jobName ジョブ名 拡張子.jobは省略される

void notImplemented (String *sFuncName*)

Method warns that the HSH command does not exist

引数:

sFuncName function name

**void objAttribute (String *name,*
String *value*
)**

Sets the given value to the given object attribute. If the attribute exists, its value is changed to the new value. Otherwise the attribute is created. If the value is null the attribute with the given name is removed.

引数:

name attribute name
value attribute value

String objAttribute (String *name*)

Returns the value of the object attribute with given name.

引数:

name the object attribute name

戻り値:

the value of the object attribute with given name or null if the attribute is not defined in the object or empty string if the value is not defined.

String objAttribute ( )

Returns comma separated list of the all object attributes. **Example:** ".out_scale=,nominal_y=yscale,.edgeline="

戻り値:

comma separated list of the all object attributes.

Point objCenterPoint ( )

アークの中心ポイントを返す

戻り値:

Point アークの中心ポイント

double objClearance ( )

2つのオブジェクト間のクリアランスを測定する

戻り値:
2つのオブジェクト間のクリアランス（double型）

Rectangle objEnclosingBox ( )

Returns enclosing rectangle of the current object

戻り値:
Rectangle enclosing rectangle of the current object

Point objFlash ( )

フラッシュもしくはベクトルテキストのフラッシュポイントを返す

戻り値:
Point オブジェクトのフラッシュポイント

Point objFromPoint ( )

ドローもしくはアークの第1ポイントを返す

戻り値:
Point ドロー／アークの第1ポイント

String objInfo ( )

オブジェクト情報を印刷する

戻り値:
カレントオブジェクト情報を返す

int objNet ( )

オブジェクトネット番号を返す

戻り値:
ネット番号（int型）

Point objPoint ( )

フラッシュもしくはベクトル文字オブジェクトのフラッシュポイントを返すか ドローもしくはアークの第1ポイントを返す

戻り値:
Point オブジェクトのフラッシュポイントもしくはドロー／アークの第1ポイント

double objRing ( )

Measures inner clearance (ring) between two objects.

戻り値:
A constant Not-a-Number (NaN) value when error, double value is clearance between two objects. Negative value means that second object is inside the first object.

void objSelect (String *sel*)

任意のパラメーターに基づいて オブジェクトを選択/非選択する

引数:
sel "+"はオブジェクトの選択 ”-”はオブジェクト非選択

boolean objSelect ( )

オブジェクトが選択されていればtrueを返す

戻り値:
boolean型 オブジェクトが選択されていればtrueを返す

String objSense ( )

アークの描画方向を返す

戻り値:
String型 次のいずれかを返す: "cw"（右回り） ”ccw”（左回り） null（オブジェクトがアークでない）

String objShape ( )

オブジェクトアパーチャの形状を返す

戻り値:
String型 可能値: "cir", "rec", "box", "oct", "con", "com", "the", "txt", "blo", "squ" ,"don"

void objString (String *vtxString*)

Sets the String value of the VTX object

引数:
vtxString new string value in current VTX object

String objString ( )

Returns the String of the VTX object

戻り値:
the String of the VTX object

Point objToPoint ( )

ドローもしくはアークの終了ポイントを返す

戻り値:
Point ドロー／アーク終了ポイント

String objType ( )

オブジェクトタイプを返す

戻り値:
可能値: "arc", "dra", "reg", "fla", "vtx".

void offset (double *offset_x,*
double *offset_y*
)

ナンバーズダイアログで使用するオフセットを設定する

引数:
offset_x （X座標）オフセット
offset_y （Y座標）オフセット

void offset (Point *offset*)

ナンバーズダイアログで使用するオフセットを設定する

引数:
offset オフセット

Point offset ( )

ナンバーズダイアログで使用されるオフセットを取得する

戻り値:
オフセット

void offsetX (double *offsetX*)

ナンバーズダイアログで使用するオフセットのX座標を設定する

引数:

offsetX オフセットX座標

void offsetY (double *offsetY*)

ナンバーズダイアログで使用するオフセットのY座標を設定する

引数:

offsetY オフセットY座標

void openAboutUcamco ( )

Opens ucamco website. (There is no closing function, since it is not strictly a dialog)

void openAboutUcamX ( )

Opens Ucam X product website. (There is no closing function, since it is not strictly a dialog)

void openAdvantools ( )

show Advantools

void openAMLIJobManager ( )

open AMLI Job Manager

void openAnamorphicScale ( )

show AnamorphicScale dialog

void openApeCreator ( )

open Aperture Creator

void openApeEditor ( )

open Aperture Editor

void openApertureAttributes ( )

show Aperture Attributes dialog

void openApertureManager ( )

アパーチャマネージャダイアログを表示する

void openAttributeEditor ( )

show Attribute Editor dialog

void openAttributeManager ( )

show Attribute Manager dialog

void openAutoDrill ( )

自動ドリルダイアログを表示する

void openAutoDrillEditor ( )

Show AutoDrill Editor

void openAutoFixture ( )

Show AutoFixture dialog

void openBarcode ( )

show Barcode dialog

void openBarcode128 ( )

show Barcode 128 dialog

void openBoardAnalyzer ( )

基板解析ダイアログを表示する

void openBoardSnapshot ( )

ボードスナップショットダイアログを表示する

void openCalculatorSetup ( )

演算設定ダイアログを表示する

void openCamtek (String *sMachineCfg*)

Camtekダイアログを表示する

引数:

sMachineCfg

void openCFMEEOutput ( )

Open the CFMEE output dialog

void openCheckList ( )

show CheckList Dialog

void openCheckListDefineChecklist ( )

Show "CheckList: Define Checklist" Dialog

void openCheckListDefineSteps ( )

Show "CheckList: Define Steps" Dialog

void openClipping ( )

クリッピングダイアログを表示する

void openColor ( )

Show Color dialog

void openConnect ( )

接続ダイアログを表示する

void openContourHandling ( )

輪郭ハンドリングダイアログを表示する

void openConvertAttributes ( )

show Attribute Converter dialog

void openCopperBalance ( )

open Copper Balance Dialog

void openCopperRepair ( )

銅修正ダイアログを表示する

void openCoverlayOptimizer ( )

show Coverlay Optimizer dialog

void openCU9000Dialog ( )

Open DS DI output dialog

void openDatums ( )

データムダイアログを表示する

void openDistort ( )

スケール補正ダイアログを表示する

void openDraw (double *pt_x,*

double *pt_y*,
double *dis*
)

ドローをオープンにする ブレイクを挿入する

引数:

pt_x (X座標) オープンの場所
pt_y (Y座標) オープンの場所
dis オープンの2エンドポイント間のクリアランス

void openDraw (Point *pt*,
double *dis*
)

ドローをオープンにする ブレイクを挿入する

引数:

pt オープンの場所
dis オープンの2エンドポイント間のクリアランス

void openDrawSlots ( )

show Draw Slots Dialog

void openDRC ( )

DRCダイアログを表示する

void openDrillInfo ( )

ドリル情報ダイアログを表示する

void openDrillMap ( )

open Drill Map Dialog

void openDrillOptimizer ( )

open Smart Drill Optimizer

void openDrillRoutSetups ( )

Show Drill/Rout Setups dialog

void openDrillTolerance ( )

Show Drill Tolerance dialog

void openDrillToolManager ( )

ドリルツールマネージャを表示する

void openDsAoi ( )

Open DS AOI dialog

void openDsAoiAdvanced ( )

Open DS AOI Advanced dialog (no closing function yet)

void openDSAOIialog ( )

Open DS AOI output dialog

void openDsAoiPreview ( )

Open DS AOI dialog

void openDsAoiQueue ( )

Open DS AOI Queue dialog (no closing function yet)

void openEditingToolbox ( )

ツールボックス編集ダイアログを表示する

void openEditVectorText (double *pickPoint_x*,
double *pickPoint_y*
)

ベクトル文字編集ダイアログを表示する

引数:

pickPoint_x (X座標)
pickPoint_y (Y座標)

void openEditVectorText (Point *pickPoint*)

ベクトル文字編集ダイアログを表示する

引数:

pickPoint

void openErrors ( )

エラーダイアログを表示する

void openEtchCompensation ( )

show Etch Compensation dialog

void openExpand ( )

展開ダイアログを表示する

void openExternalLinkManager ( )

show External Link Manager

void openFiducials ( )

show Fiducials dialog

void openFillAngledPattern ( )

open Fill Angled Pattern Dialog

void openFillPattern ( )

open Fill Pattern Dialog

void openFillVector ( )

open Fill Vector Dialog

void openFlashMaker ( )

フラッシュメーカーダイアログを表示する

void openFlexManager ( )

open uFlex Manager

void openFlipJob ( )

show Flip Job dialog

void openFrame (String *sFrameName*)

Opens custom dockable frame with given identification

引数:

sFrameName identification frame given by getFrameID() method of CustomFrame class

void openGridParameters ( )

グリッドパラメータダイアログを表示する

void openHelp ( )

Opens the application's documentation. (There is no closing function, since it is not strictly a dialog)

void openHelpOnHelp ( )

Opens help documentation. (There is no closing function, since it is not strictly a dialog)

void openHelpOnHypertool ( )

Opens hypertool documentation. (There is no closing function, since it is not strictly a dialog)

void openHelpOnResources ( )

Opens resource's documentation. (There is no closing function, since it is not strictly a dialog)

void openHelpOnVersion ( )

Opens version documentation. (There is no closing function, since it is not strictly a dialog)

void openHiPot ( )

show HiPot dialog

void openImageCompare ( )

show Image Compare dialog

void openImpedanceControl ( )

show Impedance Control dialog

void openImportIPC356 ( )

show Import IPC356 dialog (no closing command)

void openImportMET ( )

show Import MET dialog (no closing command)

void openImportODBxx ( )

インポートODBxx ステップダイアログを表示する

void openImportWF ( )

show Import WF dialog (no closing command)

void openInsertContourText ( )

show Insert Contour Text dialog

void openInsertVectorText ( )

ベクトル文字ダイアログを表示する

void openJob (String *jobName*)

任意のジョブ名のジョブを開く　ジョブが存在しない場合は 警告が表示され 空の新規ジョブがロードされる

引数:

jobName パスを含めた完全なジョブ名

void openJob_shm (String *sShmName*)

read job from shared memory

引数:

sShmName - Name of the shared memory

void openJobCreate ( )

Show create job dialog (no close)

void openJobDefinition ( )

ジョブ定義ダイアログを表示する

void openJobEdit ( )

ジョブディタダイアログを表示する

void openJobEditor ( )

Show Job Editor dialog

void openJobEditorOptions ( )

Show Job Editor Options dialog

void openJobLoad ( )

Show open job dialog (no close)

void openJobMerge ( )

ジョブ合成ダイアログを表示する

void openJobPlaneSetup ( )

Show Job Plane Setup dialog

void openJobPrint ( )

Show Job Print dialog

void openJobView ( )

show Job View dialog

void openLayerEdit ( )

opens Layer Modify dialog

void openLegendOptimizer ( )

show Legend Optimizer dialog

void openLicenseHelp ( )

Opens license agreement help. (There is no closing function, since it is not strictly a dialog)

void openLoadCheckList ( )

show Load CheckList Dialog

void openMagnifier ( )

拡大レンズウィンドウをを表示する

void openMarkupAssistant ( )

open Markup Assistant

void openMessages ( )

メッセージログウィンドウを表示する

void openMLIOutput ( )

open MLI Output dialog

void openModels ( )

モデルダイアログを表示する

void openNetCompare ( )

show Net Compare dialog

void openNetfixSetup ( )

検査ダイアログを表示する

void openNonFunctionalPad ( )

show Non-Functional pads dialog

void openNumbers ( )

ナンバーズダイアログを表示する

void openObjectAttributes ( )

オブジェクト属性ダイアログを表示する

void openObjectCompare ( )

show Object Compare dialog

void openOutputAccumatch ( )

Open Accumatch Output dialog

void openOutputAOI ( )

Open AOI Output dialog

void openOutputCAD ( )

Output CADダイアログを表示する

void openOutputCamtek ( )

Open Camtek Output dialog

void openOutputDrillRout (String *sDrillMachine*)

Output Drill/Rout（ドリル/ルータのアウトプット）ダイアログを表示する

引数:

sDrillMachine

void openOutputDsDi ( )

Open DS DI output dialog

void openOutputDsDiPreview ( )

Open DS DI Preview dialog

void openOutputDsDiQueue ( )

Open DS DI queue dialog (no closing function yet)

void openOutputNetlist ( )

Output Netlistダイアログを表示する

void openOutputOrbot ( )

Open Orbot Output dialog

void openOutputSapphire ( )

Open Sapphire Output dialog

void openOutputScoring ( )

Output Scoringダイアログを表示する

void openOutputSmartArgos ( )

Open SmartArgos Output dialog

void openOutputTrackscan ( )

Open Trackscan Output dialog

void openOutputUxpAutomanager ( )

Open Output UXP Automanager dialog

void openOutputUxpEtec ( )

Open Output UXP Etec dialog

void openOutputUxpLocal ( )

Open Output UXP Local dialog (no closing function yet)

void openPanelFramesCoupons ( )

Show Panel Frames Coupons dialog

void openPanelLinks ( )

Show Panel Links dialog

void openPanelPlus ( )

パネルプラスダイアログを表示する

void openPanelReproduce ( )

パネル再構築ダイアログを表示する

void openPanelSetup ( )

Show Panel Setup dialog

void openPanelStepRepeat ( )

Panel Step Repeatダイアログを表示する

void openPlaneAdjuster ( )

show Plane Adjuster dialog

void openPlotParameters ( )

show Plot Parameters dialog

void openPPMonitor ( )

show PPMonitor dialog

void openProductionStagesEditor ( )

show Production Stages Editor dialog

void openQueryNet ( )

show Query net dialog

void openQueryObject ( )

オブジェクト解析ダイアログを表示する

void openReferencePoints ( )

リファレンスポイントダイアログを表示する

void openRegister ( )

Registerダイアログを表示する

void openRemoveAttributes ( )

show Remove Attributes dialog

void openRepair (String *szLabname*)

Show Repair dialog

引数:

szLabname

void openRoutManager ( )

ルータ編集ダイアログを表示する

void openRoutManagerCleanUp ( )

クリーンアップページにルータ編集ダイアログを表示する

void openRoutManagerDimensioning ( )

ルータ編集ダイアログの寸法ページを表示する

void openRoutManagerEditor ( )

ルータ編集ダイアログの作成/編集ページを表示する

void openRoutManagerTools ( )

ルータ編集ダイアログのツールページを表示する

void openSaveJobAs ( )

Open Save Current job as...

void openSaveLayout ( )

レイアウトの保存ダイアログを表示する

void openSecureEtchCompensation ( )

show Secure Etch Compensation dialog

void openSelections ( )

選択ダイアログを表示する

void openSetupOptions ( )

Show Setup Options dialog

void openSetupSave ( )

Show Save dialog

void openShavePads ( )

シェービングダイアログを表示する

void openSignalLayerAdjuster ( )

show Signal Layer Adjuster dialog

void openSignalLayerAdjusterAssistant ( )

show Signal Layer Adjuster Assistant dialog

void openSilkOptimizer ( )

open Silk Optimizer

void openSmartCamtek (String *sMachineCfg*)

SmartCamtekダイアログを表示する

引数:

sMachineCfg

void openSmartDRC ( )

Smart DRCダイアログを表示する

void openSmartFix ( )

open SmartFix dialog

void openSmartplot ( )

Open Smartplot dialog

void openSmartSR ( )

Show Smart S&R dialog

void openSmartStart ( )

ファイル入力ダイアログを表示する

void openSoldermask ( )

open Soldermask Dialog

void openSoldermaskOptimizer ( )

open Soldermask Optimizer Dialog

void openTearDrop ( )

open Tear Drop Dialog

void openTechnicalAnalyzer ( )

show Technical Analyzer dialog

void openTestpointEdit ( )

show Testpoint edit dialog

void openToolbarManager ( )

show Toolbar Manager Dialog

void openToolbars ( )

show Toolbars Dialog

void openTransformObjects ( )

オブジェクト変換ダイアログを表示する

void openTransformObjectsBGAPads ( )

オブジェクト変換のBGA Padsダイアログを表示する

void openTransformObjectsBGATracks ( )

オブジェクト変換のBGA Tracksダイアログを表示する

void openTransformObjectsEdit ( )

オブジェクト変換のEditダイアログを表示する

void openTransformObjectsRescale ( )

オブジェクト変換のRescaleダイアログを表示する

void openUcamDbEditor ( )

Show Ucamdb Editor dialog

void openUndoRedoDetails ( )

show Undo/Redo Details

void openUTest ( )

検査ダイアログを表示する

void openUtestUtilities ( )

Show Utest Utilities dialog

void openValidateLayer ( )

show Layer Validation dialog (or not, if everything is fine)

void openVectorTextFont ( )

ベクトル文字フォント（Vector Text Font）ダイアログを表示する

void openVerifyArcsDraws ( )

アーク・ドロー検証（Verify Arcs and Draws）ダイアログを表示する

void openViewGuide ( )

ビューガイドダイアログを表示する

void openYsphotechOutput ( )

Open the Ysphotech output dialog

```
void optimizeDrill ( int    nPasses,
                     int    optMode,
                     double yxTime
                   )
```

ツール->ツーリング->ドリル放熱最適化（SmartDrill Optimize）のアクション

引数:

nPasses - パスの数
optMode - 最適化方式: Path Length（経路長）もしくはDrill Time（ドリル時間）
yxTime - ドリル時間最適化方式の時間比率Y/X

void optimizeMaskLayer (double *dMinRing,*
double *dMaxRing,*
double *dMaskToCopper,*
double *dMaskToMask,*
double *dBigRing*
)

Optimize Soldermask layer

引数:

dMinRing - The minimum value accepted by the ring for adding mask to the copper pad.
dMaxRing - The value of the ring around the copper pad which is used to start searching for solder mask pad.
dMaskToCopper - The minimum clearance that is still legal between a pad of the mask layer and copper on the copper layer.
dMaskToMask - The minimum clearance that is still permissible between two pads of the mask layer.
dBigRing - Big Pad Ring

String osChDir ( )

カレントディレクトリを返す

戻り値:

ファンクションが失敗すればnull 成功すればカレントディレクトリの完全パスを返す

String osChDir (String *sDir*)

カレントディレクトリを設定する

引数:

sDir ディレクトリ指定

戻り値:

ファンクションが失敗すればnull 成功すれば新しいカレントディレクトリの完全パスを返す

int osCopy (String *sSrcName,*
String *sDstName*
)

他ファイルへファイルをコピーする

引数:

sSrcName コピーするファイルの指定

sDstName コピー先ファイルの指定

戻り値:
コピーがOKならステータス0を返す

String osCreateTmpDir (String *sBasePath*)

Function that creates tmp directory under given path and returns its name **Warning:** All temporary files and files placed into temporary directories created during a script are deleted at the end of the script run.

引数:
sBasePath the base path where the temporary directory will be created.

戻り値:
null if the function failed; otherwise the full path to the new temporary directory.

例外:
IOException

String osCreateTmpDir ( )

Function creates tmp directory under system $TEMP directory and returns its name **Warning:** All temporary files and files placed into temporary directories created during a script are deleted at the end of the script run.

戻り値:
null if the function failed; otherwise the full path to the new temporary directory.

例外:
IOException

int osDelete (String *sFileName*)

指定したファイルを削除する

引数:
sFileName 削除するファイル名

戻り値:
削除が正常に行われればステータス0

ObjectList osFileInfo (String *sPath*)

Returns objectlist of the file information

引数:
sPath The full path of the file we need get the information

戻り値:
objectlist with the file information

Example:

```
list = osGetFileList("D:¥¥ucam¥¥ucam¥¥custom", true);
```

```
item = forEachItem(list) {
    fileInfo = osFileInfo(item);
    fileInfo = osFileInfo(item);

    if (isFile(fileInfo)) {
        print("FILE: " + getFileName(fileInfo) + " (size " + getFileSize(fileInfo)
+ "b)");
    }
    else if (isDirectory(fileInfo)) {
        print("DIRECTORY: " + getFileName(fileInfo));
    }
    print("¥tat location " + getFileParent(fileInfo));
    if (canRead(fileInfo) && !canWrite(fileInfo)) {
        print("¥tis READ-ONLY");
    }
    if (isHidden(fileInfo)) {
        print("¥tis HIDDEN");
    }
    print("¥twas last modified on " + getFileLastModified(fileInfo));
}
```

Output:

```
FILE: Udualstrip.java (size 4808b)
    at location D:¥¥ucam¥¥ucam¥¥custom¥¥impedance
    is HIDDEN
    was last modified on Thu Aug 11 12:39:19 CEST 2011
FILE: Uembmstrip.java (size 3951b)
    at location D:¥¥ucam¥¥ucam¥¥custom¥¥impedance
    was last modified on Tue Nov 01 12:20:51 CET 2005
FILE: Usrfmstrip.java (size 3993b)
    at location D:¥¥ucam¥¥ucam¥¥custom¥¥impedance
    was last modified on Tue Nov 01 12:20:51 CET 2005
FILE: Usymstrip.java (size 4036b)
    at location D:¥¥ucam¥¥ucam¥¥custom¥¥impedance
    is READ-ONLY
    is HIDDEN
    was last modified on Tue Nov 01 12:20:51 CET 2005
```

参照:

- **canRead(ObjectList)**
- **canWrite(ObjectList)**
- **getFileLastModified(ObjectList)**
- **getFileName(ObjectList)**
- **getFileParent(ObjectList)**
- **getFileSize(ObjectList)**
- **isDirectory(ObjectList)**
- **isFile(ObjectList)**
- **isHidden(ObjectList)**

ObjectList osGetFileList (**String** ***sDir,***
String ***sFileMask,***
boolean ***bRecurse,***
boolean ***bFullPath,***
boolean ***bWithDirs***
)

Enumerates directory and returns ObjectList of filenames (string)

引数:

sDir The directory path

sFileMask file mask filter supports WildCards (e.g. "*.vhs", "?_script.*")
bRecurse if `true` the function goes to subdirectories as well
bFullPath if `true` the function returns the list of the full pathes
bWithDirs if `true` the OjectList contains also directories

戻り値:
Object List containing all file names in given directory and subdirectories if bRecurse is set to true and matching given sFileMask. The ObjectList includes the names of the directories if bWithDirs is set to `true`.

例外:
IOException If an I/O error occurs, which is possible because the construction of the file pathname list may require filesystem queries

ObjectList osGetFileList (String *sDir,*
String *sFileMask,*
boolean *bRecurse,*
boolean *bFullPath*
)

Enumerates directory and returns ObjectList of filenames (string)

引数:
sDir The directory path
sFileMask file mask filter supports WildCards (e.g. "*.vhs", "?_script.*")
bRecurse if `true` the function goes to subdirectories as well
bFullPath if `true` the function returns the list of the full paths

戻り値:
Object List containing all file names in given directory and subdirectories if bRecurse is set to true and matching given sFileMask.

例外:
IOException If an I/O error occurs, which is possible because the construction of the file pathname list may require filesystem queries

ObjectList osGetFileList (String *sDir,*
String *sFileMask,*
boolean *bRecurse*
)

Enumerates directory and returns ObjectList of filenames (string)

引数:
sDir The directory path
sFileMask file mask filter supports WildCards (e.g. "*.vhs", "?_script.*")
bRecurse if true the function goes to subdirectories as well

戻り値:
Object List containing all file and directory full paths in given directory and subdirectories if bRecurse is set to `true` and matching given sFileMask.

例外:
IOException If an I/O error occurs, which is possible because the construction of the file pathname list

may require filesystem queries

**ObjectList osGetFileList (String *sDir,*
String *sFileMask*
)**

Enumerates directory and returns ObjectList of filenames (string)

引数:

sDir The directory path
sFileMask file mask filter supports WildCards (e.g. "*.vhs", "?_script.*")

戻り値:

Object List containing all file and directory full paths in given directory matching the sFileMask. It is not recursive.

例外:

IOException If an I/O error occurs, which is possible because the construction of the file pathname list may require filesystem queries

**ObjectList osGetFileList (String *sDir,*
boolean *bRecurse*
)**

Enumerates directory and returns ObjectList of filenames (string)

引数:

sDir The directory path
bRecurse if `true` the function goes to subdirectories as well

戻り値:

Object List containing all file names in given directory and subdirectories if bRecurse is set to true.

例外:

IOException If an I/O error occurs, which is possible because the construction of the file pathname list may require filesystem queries

**ObjectList osGetFileList (String *sDir,*
boolean *bRecurse,*
boolean *bFullPath,*
boolean *bWithDirs*
)**

Enumerates directory and returns ObjectList of filenames (string)

引数:

sDir The directory path
bRecurse if `true` the function goes to subdirectories as well
bFullPath if `true` the function returns the list of the full pathes
bWithDirs if `true` the OjectList contains also directories

戻り値:

Object List containing all file names in given directory

例外:

IOException If an I/O error occurs, which is possible because the construction of the file pathname list may require filesystem queries

ObjectList osGetFileList (String *sDir*)

Enumerates directory and returns ObjectList of filenames (string)

引数:

sDir The directory path

戻り値:

Object List containing all file and directory full paths in given directory. It is not recursive.

例外:

IOException If an I/O error occurs, which is possible because the construction of the file pathname list may require filesystem queries

int osMarkAsTmp (String *sName*)

ファイル（もしくはディレクトリ）を一時ファイル（一時ディレクトリ）としてマークする マークされた各ファイル／ディレクトリは スクリプトの最後で自動的に削除される

引数:

sName ここで指定した名前のファイル／ディレクトリは 一時ファイル／一時ディレクトリとしてマークされる

戻り値:

ファイルが一時ファイルとして正常にマークされれば0 それ以外は1を返す

例外:

EvalError

int osMkDir (String *sDirName*)

Creates the directory named by the given pathname.

引数:

sDirName Directory path that will be created

戻り値:

0 if and only if the directory was created; 1 otherwise

int osMove (String *sSrcName,*
String *sDstName*
)

ファイルの名前を変更し ファイルを移動する

引数:

sSrcName 名前を変更したいファイル名
sDstName 新しいファイル名

戻り値:
名前変更／移動が正常に行われれば ステータス0を返す

void osRmDir (String *sDirName*)

空のディレクトリを削除する

引数:
sDirName ディレクトリ名

void osRmTree (String *sDirName*)

空でないディレクトリ／サブツリーを削除する

引数:
sDirName ディレクトリ名

int osUnTgz (String *sTgzArchive,*
String *sDstDir*
)

This function creates a subdirectory with the basename of the tgz-file and will gunzip and untar the given tgz-archive under this subdirectory.

引数:
sTgzArchive TGZ archive
sDstDir Destination directory

戻り値:
0 if the given file is successfuly decompressed or 1 if it failed

int osUnZip (String *sZipArchive,*
String *sDstDir*
)

.zipファイルを定義した解凍先ディレクトリへ解凍する

引数:
sZipArchive ZIPアーカイブ
sDstDir 解凍先ディレクトリ

戻り値:
任意のファイルが正常に解凍されれば0 解凍に失敗すれば1を返す

void outAtgFixture (String *key,*

```
                String  sTool,
                String  iRes,
                int     iSession
              )
```

ジョブ内の全アクティブレイヤをAtg治具フォーマットへ変換する .

引数:

key - 言語: anf, anfref, anf_a2000, anfref_a2000
sTool - ツールファイル
iRes - リソースファイル
iSession - セッションID

void output274x (String *sRes*)

ジョブ内の全アクティブレイヤをGerber 274xフォーマットに変換する

引数:

sRes リソースファイルのパス

void outputAft (String *res*)

ジョブ内の全アクティブレイヤをAFTフォーマットへ変換する

引数:

res リソースファイルのパス

```
void outputAoi ( boolean  bCadData,
                 boolean  bReference
               )
```

AOIアウトプットを生成する

引数:

bCadData - CAD情報（LPファイル）を生成する
bReference - リファレンスデータ（GERファイル）を生成する

void outputAtf ( )

ジョブ内の全アクティブレイヤをATFフォーマットへ変換する

ObjectList outputAutoDrill (String *sDrjFile*)

Output AutoDrill

引数:

sDrjFile AutoDrill configuration file path.

戻り値:
ObjectArray, with the names of the output file names.

void outputCFMEE (String *outputPath*,
boolean *reverse*,
double *marginx*,
double *marginy*,
boolean *distort*,
double *distortx*,
double *distorty*,
double *resizex*,
double *resizey*,
boolean *deleteOutside*
)

Generate CFMEE

引数:

outputPath	the location where the gdsii output will be written
reverse	whether the image should be reversed
marginx	margin x when reversing the image
marginy	margin y when reversing the image
distort	
distortx	
distorty	
resizex	anamorphic resize x value
resizey	anamorphic resize y value
deleteOutside	whether to delete objects outside the outline

void outputCli ( )

ジョブ内の全アクティブレイヤをCLIフォーマットに変換する

void outputColorPDF (String *sPdfFullPath*)

Create a pdf file from given layer (active in plane 1 - red layer) Apertures will be coloring by attribute uColor

引数:

sPdfFullPath	full path to new PDF file, include file name

void outputDp40 (double *pt_x*,
double *pt_y*,
boolean *bPositive*,
boolean *bMirrorx*,
boolean *bMirrory*,
double *dLaserPower*,
int *iPolygonSpeed*,

```
                int     iPcbFormat,
                String  unit
            )
```

引数:

pt_x (X座標)
pt_y (Y座標)
bPositive
bMirrorx
bMirrory
dLaserPower
iPolygonSpeed
iPcbFormat
unit

```
void outputDp40 ( Point   pt,
                  boolean bPositive,
                  boolean bMirrorx,
                  boolean bMirrory,
                  double  dLaserPower,
                  int     iPolygonSpeed,
                  int     iPcbFormat,
                  String  unit
                )
```

引数:

pt
bPositive
bMirrorx
bMirrory
dLaserPower
iPolygonSpeed
iPcbFormat
unit

```
void outputDxf ( String  unit,
                 int     iConturize,
                 int     iKeepTXT,
                 double  dExpandArcs,
                 int     iCenterLine,
                 int     iAllInOne
               )
```

ジョブ内の全アクティブレイヤをDXFフォーマットへ変換する

引数:

unit - 単位
iConturize - 輪郭化する
iKeepTXT - テキストを保持する
dExpandArcs - アークを展開する

iCenterLine - センターライン
iAllInOne - オールインワン

void outputDxfV6 (String *unit*)

ジョブ内の全アクティブレイヤをDXF v6フォーマットに変換する

引数:
unit - 単位

void outputEie (String *sRes*,
ObjectList *par*
)

ジョブ内の全アクティブレイヤをEIEフォーマットに変換する

引数:
sRes リソースファイルのパス
par パラメータ配列

void outputEtec (String *sResistOuter*,
String *sResistInner*,
double *mediaX*,
double *mediaY*,
int *iAlignType*,
int *iLevelType*,
int *iCycles*,
String *sDate*,
String *sTime*,
boolean *bAuto*,
double *dScaleX*,
double *dScaleY*,
double *dScaleOriX*,
double *dScaleOriY*,
String *sMDFfile*,
String *sResource*
)

ETECアウトプットを生成する

引数:
sResistOuter
sResistInner
mediaX
mediaY
iAlignType
iLevelType
iCycles
sDate

sTime
bAuto
dScaleX
dScaleY
dScaleOriX
dScaleOriY
sMDFfile
sResource

```
void outputExt ( String     lan,
                 String     too,
                 String     res,
                 ObjectList resdb,
                 String     inc1,
                 String     inc2,
                 int        session,
                 Object[]   pre,
                 Object[]   pos,
                 Object     notUsed
               )
```

Converts all active layers of this job to many format types.

引数:

lan - output language
too - tool file
res - resource file
resdb - resource database
inc1 - part of Outpar.inc1
inc2 - part of Outpar.inc2
session - part of Outpar.session
pre - part of Outpar.pre
pos - part of Outpar.pos
notUsed - the parameter is not used at all must be set to `null`

```
void outputExt ( String     lan,
                 String     too,
                 String     res,
                 ObjectList resdb,
                 String     inc1,
                 String     inc2,
                 int        session,
                 Object[]   pre,
                 Object[]   pos
               )
```

Converts all active layers of this job to many format types.

引数:

lan - output language

too - tool file
res - resource file
resdb - resource database
inc1 - part of Outpar.inc1
inc2 - part of Outpar.inc2
session - part of Outpar.session
pre - part of Outpar.pre
pos - part of Outpar.pos

void outputHimt (double *datum_x*,
double *datum_y*,
double *offset_x*,
double *offset_y*,
String *mirror*,
String *rotation*
)

ジョブ内の全アクティブレイヤをHIMTフォーマットに変換する

引数:
datum_x (X座標) - データポイント
datum_y (Y座標) - データポイント
offset_x (X座標) - オフセットポイント
offset_y (Y座標) - オフセットポイント
mirror - ミラー
rotation - 回転

void outputHimt (Point *datum*,
Point *offset*,
String *mirror*,
String *rotation*
)

ジョブ内の全アクティブレイヤをHIMTフォーマットに変換する

引数:
datum - データポイント
offset - オフセットポイント
mirror - ミラー
rotation - 回転

void outputIpc2581 ( )

ジョブ内の全アクティブレイヤをIPC2581フォーマットに変換する

void outputIpcUfd (String *key*,
String *res*,

```
                String  version
            )
```

ジョブ内の全アクティブレイヤをIPC350, IPC356, MET*, MNF*フォーマットへ変換する

引数:

key - 言語
res - リソースファイル
version - Ucamのバージョン

```
void outputLpg ( int     iPpi,
                 int     iChoke,
                 double  offset_x,
                 double  offset_y
               )
```

引数:

iPpi - 解像度
iChoke - 細らせ
offset_x (X座標)
offset_y (Y座標)

```
void outputLpg ( int    iPpi,
                 int    iChoke,
                 Point  offset
               )
```

引数:

iPpi - 解像度
iChoke - 細らせ
offset

```
int outputManiaSapphire ( String   sOutputPath,
                          String   sDescription,
                          String   sGeometryfile,
                          boolean  bStatistics,
                          boolean  bDrill
                        )
```

マニアSapphireアウトプットを生成する

引数:

sOutputPath - アウトプットパス
sDescription - 説明
sGeometryfile - 形状ファイル
bStatistics - 統計
bDrill - ドリル

戻り値:

エラーステータス 0はOKの意

void outputMda (String *sPath,*
ObjectList *par,*
Object[] *subpar,*
int *iApr,*
int *iSubfig,*
int *iRenum*
)

ジョブ内の全アクティブレイヤをMDAフォーマットに変換する

引数:

sPath パス
par メインMDAファイルのG04パラメータ
subpar サブMDAファイルのG04パラメータ
iApr アパーチャスケールファクター
iSubfig 1の場合 サブフィギュアが許容される
iRenum 1の場合 アパーチャの再度番号割り当てを行う （デフォルトは0）

void outputNec (String *too*)

ジョブ内の全アクティブレイヤをNECフォーマットに変換する

引数:

too - ツールファイル

void outputOdbxx (String *res*)

ジョブ内の全アクティブレイヤをODB++フォーマットへ変換する

引数:

res - リソースファイル

void outputOdbxxv7 (String *res*)

Converts all active layers of this job to the ODB++ v7 format.

引数:

res - resource file

void outputOif (String *oifVersion,*
int *byJob,*
int *pan,*
int *fillin*
)

ジョブ内の全アクティブレイヤをOI2002もしくはOI5000フォーマットへ変換する

引数:

oifVersion - oi2002もしくはoi5000
byJob - "uout_oifbac_tog" の値
pan - "uout_oifpan_tog"の値
fillin - "uout_oif_fil"の値

boolean outputOrbot ( )

Output Orbot - 拡張もしくはキャンセルされる

戻り値:

ステータス

void outputPdf ( )

ジョブの全アクティブレイヤをPDFフォーマットに変換する

void outputProbe (String *sLang,*
int *iSession,*
int *iAccuracy*
)

ジョブ内の全アクティブレイヤをフライングプローブフォーマットprobotやrislangに変換する

引数:

sLang 言語: "probot" もしくは"rislang"
iSession セッション番号
iAccuracy プロボットの座標精度

int outputRaid ( )

このジョブの全てのアクティブレイヤをRAIDフォーマットに変換する

戻り値:

ステータス

void outputRpd (int *iPpi,*
double *datum_x,*
double *datum_y,*
double *offset_x,*
double *offset_y,*
String *sMirror,*
String *sRotation*
)

ジョブ内の全アクティブレイヤをRPDフォーマットに変換する

引数:

iPpi
datum_x (X座標)
datum_y (Y座標)
offset_x (X座標)
offset_y (Y座標)
sMirror
sRotation

```
void outputRpd ( int    iPpi,
                 Point  datum,
                 Point  offset,
                 String sMirror,
                 String sRotation
               )
```

ジョブ内の全アクティブレイヤをRPDフォーマットへ変換する

引数:

iPpi
datum
offset
sMirror
sRotation

```
boolean outputSchmid ( int        resolution,
                       double     maskRectangle_xmin,
                       double     maskRectangle_ymin,
                       double     maskRectangle_xmax,
                       double     maskRectangle_ymax,
                       int        maskRotation,
                       String     maskMirror,
                       String     maskPolarity,
                       int        equipmentRotation,
                       String     equipmentMirror,
                       double     offsetX,
                       double     offsetY,
                       ObjectList fiducials,
                       String     imagePath,
                       String     configPath,
                       String     batchFile
                     )
```

カレントジョブの全アクティブレイヤをSchmidアウトプット（tiffやxmlファイル）する

引数:

resolution tiffファイルの解像度
maskRectangle_xmin (長方形の左境界) tiffファイル内の長方形囲み領域を表す

maskRectangle_ymin (長方形のボトム境界) tiffファイル内の長方形囲み領域を表す
maskRectangle_xmax (長方形の右境界) tiffファイル内の長方形囲み領域を表す
maskRectangle_ymax (長方形のトップ境界) tiffファイル内の長方形囲み領域を表す
maskRotation tiffファイルの回転（回転度）
maskMirror tiffファイルのミラー設定 ("", "X", "Y", "XY"のいずれか)
maskPolarity tiffファイルの極性（"P" もしくは"N")
equipmentRotation インクジェット機器におけるtiffファイルの回転
equipmentMirror インクジェット機器におけるtiffファイルのミラー
offsetX インクジェット機器におけるtiff画像のXオフセット
offsetY インクジェット機器におけるtiff画像のYオフセット
fiducials 基準点リスト（CAD座標）
imagePath 画像ファイルの保管場所ディレクトリ
configPath xmlファイルの保管場所ディレクトリ
batchFile アウトプットが作成される際に走らせる後処理.batファイルを指定する　指定するには 画像ファイルであることを示すためのiや xmlファイルであることを示すためのxを含める

戻り値:
output return value

```
boolean outputSchmid ( int          resolution,
                       Rectangle    maskRectangle,
                       int          maskRotation,
                       String       maskMirror,
                       String       maskPolarity,
                       int          equipmentRotation,
                       String       equipmentMirror,
                       double       offsetX,
                       double       offsetY,
                       ObjectList   fiducials,
                       String       imagePath,
                       String       configPath,
                       String       batchFile
                     )
```

カレントジョブの全アクティブレイヤをSchmidアウトプット（tiffやxmlファイル）する

引数:
resolution tiffファイルの解像度
maskRectangle tiffファイル内の長方形囲み領域を表す
maskRotation tiffファイルの回転（回転度）
maskMirror tiffファイルのミラー設定 ("", "X", "Y", "XY"のいずれか)
maskPolarity tiffファイルの極性（"P" もしくは"N")
equipmentRotation インクジェット機器におけるtiffファイルの回転
equipmentMirror インクジェット機器におけるtiffファイルのミラー
offsetX インクジェット機器におけるtiff画像のXオフセット
offsetY インクジェット機器におけるtiff画像のYオフセット
fiducials 基準点リスト（CAD座標）
imagePath 画像ファイルの保管場所ディレクトリ
configPath xmlファイルの保管場所ディレクトリ
batchFile アウトプットが作成される際に走らせる後処理.batファイルを指定する　指定するには 画像ファイルであることを示すためのiや xmlファイルであることを示すためのxを含める

戻り値:
output return value

void outputSl13 (String *sResources,*
String *sKey*
)

ジョブ内の全アクティブレイヤをSL3.7 もしくは SL13.9フォーマットに変換する

引数:
sResources - リソースファイルのパス
sKey - 言語

void outputSprint (String *sOutputFolder,*
boolean *bStandardMark,*
String *sCopperName,*
String *sRefName,*
String *sAttributeZero,*
int *iZeroPointNumber,*
String *sAttributeCamera,*
int *iCameraNumber,*
int *iRotation0,*
int *iRotation90,*
int *iRotation180,*
int *iRotation270,*
String *sText1,*
String *sText2,*
String *sText3,*
String *sCleanOption*
)

Generates Sprint output for the current job.

引数:
sOutputFolder The output folder
bStandardMark
sCopperName
sRefName
sAttributeZero
iZeroPointNumber
sAttributeCamera
iCameraNumber
iRotation0
iRotation90
iRotation180
iRotation270
sText1
sText2
sText3
sCleanOption

void outputSys ( )

ジョブの全アクティブレイヤをSystronicフォーマットに変換する

void outputTiff (String *sPath,*
String *sExt,*
String *sOptions,*
int *iResolution*
)

Makes a pixel file of the job/layer/aperture.

引数:

sPath	The path of the file.
sExt	The extension of the file.
sOptions	
iResolution	The output resolution in ppi.

int outputTs3 (boolean *bDrilledBoards,*
double *dSpace*
)

引数:

bDrilledBoards	true：ドリルレイヤもアウトプットする false：ドリルレイヤはアウトプットしない
dSpace	スペースの公称値

戻り値:
戻り値をアウトプット

void outputWf2 (ObjectList *par*)

ジョブ内の全アクティブレイヤをWF2フォーマットに変換する

引数:
par - 全パラメータを持つ配列

void outputXdpf (String *sPath*)

XDPFジョブを生成する

引数:
sPath - 送り先パス

void outputYsphotech (boolean *imagecomp,*

```
                    String   outputPath,
                    String   reverse,
                    double   marginx,
                    double   marginy,
                    boolean  mirrorx,
                    boolean  mirrory,
                    double   rotate,
                    double   distortx,
                    double   distorty,
                    double   resize,
                    boolean  keepArrays,
                    boolean  deleteOutside,
                    boolean  autoDetected,
                    String   layername
                )
```

Generate ysphotech output on layer

引数:

imagecomp	
outputPath	the location where the gdsii output will be written
reverse	Yes, No or False. Yes will reverse the image, No won't reverse the image. And False will prepare to reverse later: add $$ to the name and inverse the resize value
marginx	margin x when reversing the image
marginy	margin y when reversing the image
mirrorx	whether the image should be mirrored along the x-axis
mirrory	whether the image should be mirrored along the y-axis
rotate	rotation angle in degrees
distortx	distort x value
distorty	distort y value
resize	resize value
keepArrays	whether to keep the array structure
deleteOutside	whether to delete objects outside the outline
autoDetected	whether or not alignment points are automatically detected or selected by the user
layername	the name of the layer to be outputted, GDSII output will only happen when the layer is active

```
void outputYsphotech ( String   outputPath,
                       String   reverse,
                       double   marginx,
                       double   marginy,
                       boolean  mirrorx,
                       boolean  mirrory,
                       double   rotate,
                       double   distortx,
                       double   distorty,
                       double   resize,
                       boolean  keepArrays,
                       boolean  deleteOutside,
                       boolean  autoDetected,
                       String   layername
```

)

Generate ysphotech output on all active layers

引数:

outputPath	the location where the gdsii output will be written
reverse	
marginx	margin x when reversing the image
marginy	margin y when reversing the image
mirrorx	whether the image should be mirrored along the x-axis
mirrory	whether the image should be mirrored along the y-axis
rotate	rotation angle in degrees
distortx	distort x value
distorty	distort y value
resize	resize value
keepArrays	whether to keep the array structure
deleteOutside	whether to delete objects outside the outline
autoDetected	whether or not alignment points are automatically detected or selected by the user
layername	the name of the layer to be outputted, GDSII output will only happen when the layer is active

void pajPlaneAdjust (**double** ***dPlatedClearance,***
double ***dUnplatedClearance,***
double ***dRingSize,***
double ***dRingClearance,***
double ***dLineWidth,***
double ***dCuClearance,***
boolean ***bDoCut,***
double ***dOutlineCLearance***
)

Move word with given uText value over distance dx, dy, unless the distance is larger than limit and enforcelimit is true

引数:

dPlatedClearance
dUnplatedClearance
dRingSize
dRingClearance
dLineWidth
dCuClearance
bDoCut
dOutlineCLearance

void pajPlaneAdjust (**double** ***dPlatedClearance,***
double ***dUnplatedClearance,***
double ***dRingSize,***
double ***dRingClearance,***
double ***dLineWidth,***
double ***dCuClearance,***
boolean ***bDoCut,***

```
                    double  dOutlineClearance,
                    boolean bOutputAsContour,
                    boolean bSaveBackup,
                    boolean bErrorPopups
                )
```

Move word with given uText value over distance dx, dy, unless the distance is larger than limit and enforcelimit is true

引数:

dPlatedClearance
dUnplatedClearance
dRingSize
dRingClearance
dLineWidth
dCuClearance
bDoCut
dOutlineClearance
bOutputAsContour
bSaveBackup
bErrorPopups

```
void pajPlaneAdjust ( double  dPlatedClearance,
                      double  dUnplatedClearance,
                      double  dRingSize,
                      double  dRingClearance,
                      double  dLineWidth,
                      double  dCuClearance,
                      boolean bDoCut,
                      double  dOutlineClearance,
                      boolean bOutputAsContour,
                      boolean bSaveBackup
                    )
```

Move word with given uText value over distance dx, dy, unless the distance is larger than limit and enforcelimit is true

引数:

dPlatedClearance
dUnplatedClearance
dRingSize
dRingClearance
dLineWidth
dCuClearance
bDoCut
dOutlineClearance
bOutputAsContour
bSaveBackup

```
void panelStepRepeat ( double pStart_x,
                       double pStart_y,
```

```
                        int     iRepeatX,
                        int     iRepeatY,
                        double  dStepX,
                        double  dStepY,
                        String  sFlashPoint
                      )
```

ステップ&リピートパネル

引数:

pStart_x	(X座標) 1つ目フラッシュポイントのポジション
pStart_y	(Y座標) 1つ目フラッシュポイントのポジション
iRepeatX	X方向でのリピート数
iRepeatY	Y方向でのリピート数
dStepX	X方向のステップサイズ
dStepY	Y方向のステップサイズ
sFlashPoint	フラッシュポイント： "middle", "zero", "center"のいずれか

```
void panelStepRepeat ( Point   pStart,
                        int     iRepeatX,
                        int     iRepeatY,
                        double  dStepX,
                        double  dStepY,
                        String  sFlashPoint
                      )
```

ステップ&リピートパネル

引数:

pStart	1つ目フラッシュポイントのポジション
iRepeatX	X方向でのリピート数
iRepeatY	Y方向でのリピート数
dStepX	X方向のステップサイズ
dStepY	Y方向のステップサイズ
sFlashPoint	フラッシュポイント： "middle", "zero", "center"のいずれか

```
void panelStepRepeatCenter ( double  pStart_x,
                             double  pStart_y,
                             int     iRepeatX,
                             int     iRepeatY,
                             double  dStepX,
                             double  dStepY
                           )
```

ステップ&リピートパネル ＞ フラッシュポイント＝Center（センター）

引数:

pStart_x	(X座標) 1つ目フラッシュポイントのポジション
pStart_y	(Y座標) 1つ目フラッシュポイントのポジション
iRepeatX	X方向でのリピート数
iRepeatY	Y方向でのリピート数

dStepX x方向でのステップサイズ
dStepY y方向でのステップサイズ

void panelStepRepeatCenter (Point *pStart*,
int *iRepeatX*,
int *iRepeatY*,
double *dStepX*,
double *dStepY*
)

ステップ&リピートパネル ＞ フラッシュポイント＝Center（センター）

引数:

pStart 1つ目フラッシュポイントのポジション
iRepeatX X方向でのリピート数
iRepeatY Y方向でのリピート数
dStepX x方向でのステップサイズ
dStepY y方向でのステップサイズ

void panelStepRepeatJobZero (double *pStart_x*,
double *pStart_y*,
int *iRepeatX*,
int *iRepeatY*,
double *dStepX*,
double *dStepY*
)

ステップ&リピートパネル ＞ フラッシュポイント＝Job Zero

引数:

pStart_x (X座標) 1つ目フラッシュポイントのポジション
pStart_y (Y座標) 1つ目フラッシュポイントのポジション
iRepeatX X方向のリピート数
iRepeatY Y方向のリピート数
dStepX X方向のステップサイズ
dStepY Y方向のステップサイズ

void panelStepRepeatJobZero (Point *pStart*,
int *iRepeatX*,
int *iRepeatY*,
double *dStepX*,
double *dStepY*
)

ステップ&リピートパネル ＞ フラッシュポイント＝Job Zero

引数:

pStart 1つ目フラッシュポイントのポジション
iRepeatX X方向のリピート数

iRepeatY Y方向のリピート数
dStepX X方向のステップサイズ
dStepY Y方向のステップサイズ

void panelStepRepeatMiddle (double *pStart_x*,
double *pStart_y*,
int *iRepeatX*,
int *iRepeatY*,
double *dStepX*,
double *dStepY*
)

ステップ&リピートパネル ＞ フラッシュポイント＝Middle（ミドル）

引数:

pStart_x (X座標) 1つ目フラッシュポイントのポジション
pStart_y (Y座標) 1つ目フラッシュポイントのポジション
iRepeatX X方向のリピート数
iRepeatY Y方向のリピート数
dStepX X方向のステップサイズ
dStepY Y方向のステップサイズ

void panelStepRepeatMiddle (Point *pStart*,
int *iRepeatX*,
int *iRepeatY*,
double *dStepX*,
double *dStepY*
)

ステップ&リピートパネル ＞ フラッシュポイント＝Middle（ミドル）

引数:

pStart 1つ目フラッシュポイントのポジション
iRepeatX X方向のリピート数
iRepeatY Y方向のリピート数
dStepX X方向のステップサイズ
dStepY Y方向のステップサイズ

int panelStepRepeatValidate ( )

Checks, whether blocked jobs are valid for use with panel iterator Some problems are solved automatically

戻り値:

0 if it is no stepRepeat job; 1: problems were found and solved; 2: if problems were found, but not (all) solved

void pasteFromClipboard ( )

クリップボードから全アクティブレイヤ上に 全オブジェクトを貼り付ける

String peGetInputFile ( )

Gets the file used as input for the server mode.

戻り値:
The file used as input for the server mode.

boolean peGetInputJobBooleanProperty (String *name*)

Gets the value as a boolean of the property with the given name for the current Panel Editor input job.

引数:
name * The name of the property

戻り値:
The corresponding value or true if the property is not defined.

double peGetInputJobDoubleProperty (String *name*)

Gets the value as a double of the property with the given name for the current Panel Editor input job.

引数:
name * The name of the property

戻り値:
The corresponding value or 0.0 if the property is not defined or the current job is not set.

int peGetInputJobIntegerProperty (String *name*)

Gets the value as an int of the property with the given name for the current Panel Editor input job.

引数:
name * The name of the property

戻り値:
The corresponding value.

String peGetInputJobProperty (String *name*)

Gets the value as a String of the property with the given name for the current Panel Editor input job.

引数:
name * The name of the property

戻り値:
The corresponding value or null if the property is not defined.

String peGetJobList ( )

Gets the value as a boolean of the property with the given name for the current Panel Editor input job.

戻り値:
The corresponding value or true if the property is not defined.

int peGetOptionalQuantity ( )

Gets the optional quantity for the current PanelEditor input job.

戻り値:
the optional quantity for the current PanelEditor input job.

boolean peGetPanelJobBooleanProperty (String *name*)

Gets the value as a boolean of the property with the given name for the current Panel Editor PanelJob.

引数:
name * The name of the property

戻り値:
The corresponding value or true if the property is not defined.

double peGetPanelJobDoubleProperty (String *name*)

Gets the value as a double of the property with the given name for the current Panel Editor PanelJob.

引数:
name * The name of the property

戻り値:
The corresponding value or 0.0 if the property is not defined or the current job is not set.

int peGetPanelJobIntegerProperty (String *name*)

Gets the value as an int of the property with the given name for the current Panel Editor PanelJob.

引数:
name * The name of the property

戻り値:
The corresponding value.

String peGetPanelJobProperty (String *name*)

Gets the value as a String of the property with the given name for the current Panel Editor PanelJob.

引数:
name * The name of the property

戻り値:
The corresponding value or null if the property is not defined.

int peGetPCBQuantity ( )

Gets the number of PCBs of PanelEditor input job on the current Panel job.

戻り値:
the number of PCBs of PanelEditor input job on the current Panel job.

boolean peGetSingleOptimization ( )

Gets the optimization method for the server mode.

戻り値:
True when single panels are generated.False when combinations are generated.

boolean peGetSolutionBooleanProperty (String *name*)

Gets the value as a boolean of the property with the given name for the current Panel Editor solution.

引数:
name * The name of the property

戻り値:
The corresponding value or true if the property is not defined.

double peGetSolutionDoubleProperty (String *name*)

Gets the value as a double of the property with the given name for the current Panel Editor solution.

引数:
name * The name of the property

戻り値:
The corresponding value or 0.0 if the property is not defined or the current job is not set.

int peGetSolutionIntegerProperty (String *name*)

Gets the value as an int of the property with the given name for the current Panel Editor solution.

引数:
name * The name of the property

戻り値:
The corresponding value.

String peGetSolutionProperty (String *name*)

Gets the value as a String of the property with the given name for the current Panel Editor solution.

引数:

name * The name of the property

戻り値:

The corresponding value or null if the property is not defined.

boolean peGetUseFrameSet ( )

Gets the useframeset method for the server mode.

戻り値:

True when all frames from the set are used. False when only one frame is used.

void peSetInputFile (String *fileName*)

Sets the optional quantity for the current PanelEditor input job.

引数:

fileName

void peSetInputJobProperty (String *name*,
boolean *value*
)

Sets the value of the property with the given name to the given value for the current Input Job.

引数:

name * The name of the property
value * The value to assign to the property

void peSetInputJobProperty (String *name*,
double *value*
)

Sets the value of the property with the given name to the given value for the current Input Job.

引数:

name * The name of the property
value * The value to assign to the property

void peSetInputJobProperty (String *name*,
int *value*
)

Sets the value of the property with the given name to the given value for the current Input Job.

引数:

name * The name of the property
value * The value to assign to the property

**void peSetInputJobProperty (String *name,*
String *value*
)**

Sets the value of the property with the given name to the given value for the current Input Job.

引数:

name * The name of the property
value * The value to assign to the property

void peSetOptionalQuantity (int *quantity*)

Sets the optional quantity for the current PanelEditor input job.

引数:

quantity The value to set.

**void peSetPanelJobProperty (String *name,*
boolean *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor PanelJob.

引数:

name * The name of the property
value * The value to assign to the property

**void peSetPanelJobProperty (String *name,*
double *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor PanelJob.

引数:

name * The name of the property
value * The value to assign to the property

**void peSetPanelJobProperty (String *name,*
int *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor PanelJob.

引数:

name * The name of the property
value * The value to assign to the property

**void peSetPanelJobProperty (String *name,*
String *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor PanelJob.

引数:

name * The name of the property
value * The value to assign to the property

void peSetSingleOptimization (boolean *set*)

Sets the optimization method for the server mode.

引数:

set * When true, single panels are generated. When false, combinations are generated.

**void peSetSolutionProperty (String *name,*
boolean *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor solution.

引数:

name * The name of the property
value * The value to assign to the property

**void peSetSolutionProperty (String *name,*
double *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor solution.

引数:

name * The name of the property
value * The value to assign to the property

**void peSetSolutionProperty (String *name,*
int *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor solution.

引数:

name * The name of the property
value * The value to assign to the property

**void peSetSolutionProperty (String *name,*
String *value*
)**

Sets the value of the property with the given name to the given value for the current Panel Editor solution.

引数:

name * The name of the property
value * The value to assign to the property

void peSetUseFrameSet (boolean *set*)

Sets the frame set mode method for the server mode.

引数:

set * When true, all frames from the set are used. When false, only one frame is used.

**void pickAperture (double *pt_x,*
double *pt_y,*
double *radius*
)**

アパーチャマネジャー: 任意のポイント周辺でアパーチャをピック

引数:

pt_x (X座標) クリックポイント
pt_y (Y座標) クリックポイント
radius 任意のポイント周辺の検索半径

**void pickAperture (Point *pt,*
double *radius*
)**

アパーチャマネジャー: 任意のポイント周辺でアパーチャをピック

引数:

pt クリックポイント
radius 任意のポイント周辺の検索半径

Point pickPoint (String *sLabel*)

Wait for user pick point. It needs Ucam GUI. The function pauses script execution and waits for user interaction.

引数:

sLabel Label in information dialog

戻り値:

Point picked by user

例外:

AbortException after user abort

**void plotAddLayerToMerge (String *sJobName,*
String *sPath*
)**

Add layer to merge - to plot

引数:

sJobName name of job
sPath path to a dpf (layer) file, include ".dpf"

例外:

AbortException

void plotAddLayerToMerge ( )

Add layer in plane 1 to merge - to plot

**boolean plotLayer (String *sPath,*
int *iFillPercentage,*
boolean *bSeparator,*
boolean *bClean*
)**

Sent layer from the path to RIP

引数:

sPath path to dpf (layer) file, include .dpf
iFillPercentage is a percentage between 0% and 100%.
bSeparator value true/false
bClean If true, removes the succesfully merged MergeJob instances.

戻り値:

true if all is ok, false if was a problem

例外:

AbortException

boolean plotLayer (int *iFillPercentage,*

```
                    boolean  bSeparator,
                    boolean  bClean
                )
```

Sent layer in plane 1 to RIP

引数:

iFillPercentage	is a percentage between 0% and 100%.
bSeparator	value true/false
bClean	If true, removes the succesfully merged MergeJob instances.

戻り値:

true if all is ok, false if was a problem

例外:

AbortException

```
boolean plotMergedLayers ( int      iFillPercentage,
                           boolean  bSeparator,
                           boolean  bClean
                         )
```

Send all prepare layers to RIP

引数:

iFillPercentage	is a percentage between 0% and 100%.
bSeparator	value true/false
bClean	If true, removes the succesfully merged MergeJob instances.

戻り値:

true if all is ok, false if was a problem

例外:

AbortException

```
void plotResetParams (  )
```

Set plot parameters to default

```
void plotSetAttribute ( ObjectList  oLayID,
                        String      sName,
                        String      sValue
                      )
```

Set plot layer attribute

引数:

oLayID	layer identification
sName	name of attribute
sValue	value of attribute

**void plotSetAttribute (String *sName,*
String *sValue*
)**

Set plot layer attribute

引数:

sName name of attribute
sValue value of attribute

例外:

AbortException

void plotSetAttribute (String *sName*)

Set plot layer attribute

引数:

sName name of attribute

例外:

AbortException

**void plotSetParam (ObjectList *oLayID,*
String *sKey,*
String *sName,*
double *dValue*
)**

Set plot parameter to layer with the index

引数:

oLayID layer identification
sKey key of parameter
sName name of parameter
dValue value of parameter

**void plotSetParam (String *sKey,*
String *sName,*
double *dValue*
)**

Set plot parameter

引数:

sKey key of parameter
sName name of parameter
dValue value of parameter

例外:

AbortException

void plotSetParam (ObjectList *oLayID*,
String *sKey*,
boolean *bValue*
)

Set plot parameter to layer with the index

引数:

oLayID layer identification
sKey key name of parameter
bValue value of parameter

void plotSetParam (String *sKey*,
boolean *bValue*
)

Set plot parameter

引数:

sKey key name of parameter
bValue value of parameter

例外:

AbortException

void plotSetParam (ObjectList *oLayID*,
String *sKey*,
int *iValue*
)

Set plot parameter to layer with the index

引数:

oLayID layer identification
sKey key name of parameter
iValue value of parameter

void plotSetParam (String *sKey*,
int *iValue*
)

Set plot parameter

引数:

sKey key name of parameter
iValue value of parameter

例外:

AbortException

void plotSetParam (ObjectList *oLayID*,
String *sKey*,
double *dValue*
)

Set plot parameter to layer with the index

引数:

oLayID layer identification
sKey key name of parameter
dValue value of parameter

void plotSetParam (String *sKey*,
double *dValue*
)

Set plot parameter

引数:

sKey key name of parameter
dValue value of parameter

例外:

AbortException

void plotSetParam (ObjectList *oLayID*,
String *sKey*,
String *sValue*
)

Set plot parameter to layer with the index

引数:

oLayID layer identification
sKey key name of parameter
sValue value of parameter

void plotSetParam (String *sKey*,
String *sValue*
)

Set plot parameter

引数:

sKey key name of parameter

sValue value of parameter

例外:
AbortException

void plotSetRipHost (String *sRIP*)

set Rip Host name

引数:
sRIP - Rip Host name

void plotStartNew ( )

Initialization Set plot parameters to default values

Point Point (Point *point*)

ポイントのコピーを作成する

引数:
point 元になるポイント

戻り値:
ポイント

Point Point (double *x*,
double *y*,
String *units*
)

2つの座標（double型値）からポイントを作成する

引数:
x 座標
y 座標
units Ucam単位

戻り値:
ポイント

Point Point (double *x*,
double *y*
)

2つの座標（double型値）からポイントを作成する

引数:

x 座標
y 座標

戻り値:
the point

void point1 (double *point1_x,*
double *point1_y*
)

ナンバーズダイアログのポイント1を設定する

引数:
point1_x (X座標) ポイント1
point1_y (Y座標) ポイント1

void point1 (Point *point1*)

ナンバーズダイアログのポイント1を設定する

引数:
point1 ポイント1

Point point1 ( )

ナンバーズダイアログのポイント1を取得する

戻り値:
ポイント1

void point1Active (boolean *bActivate*)

ナンバーズダイアログのポイント1に対し アクティブ/非アクティブを設定する

引数:
bActivate ポイント1をアクティブする場合はtrue

boolean point1Active ( )

ナンバーズダイアログでポイント1がアクティブの場合に trueを返す

戻り値:
ポイント1がアクティブならばtrue

void point1X (double *pt1X*)

ナンバーズダイアログのポイント1のX座標を設定する

引数:
pt1X ポイント1のX座標

void point1Y (double *pt1Y*)

ナンバーズダイアログのポイント1のY座標を設定する

引数:
pt1Y ポイント1のY座標

void point2 (double *point2_x*,
double *point2_y*
)

ナンバーズダイアログのポイント2を設定する

引数:
point2_x (X座標) ポイント2
point2_y (Y座標) ポイント2

void point2 (Point *point2*)

ナンバーズダイアログのポイント2を設定する

引数:
point2 ポイント2

Point point2 ( )

ナンバーズダイアログのポイント2を取得する

戻り値:
ポイント2

void point2Active (boolean *bActivate*)

ナンバーズダイアログのポイント2のアクティブ/非アクティブを設定する

引数:
bActivate ポイント2をアクティブにするならばtrue

boolean point2Active ( )

ナンバーズダイアログのポイント2がアクティブであればtrueを返す

戻り値:

ポイント2がアクティブであればtrue

void point2X (double *pt2X*)

ナンバーズダイアログのポイント2のX座標を設定する

引数:

pt2X ポイント2のX座標

void point2Y (double *pt2Y*)

ナンバーズダイアログのポイント2のY座標を設定する

引数:

pt2Y ポイント2のY座標

void printListRefPoints (boolean *bOnAllActiveLay*)

Print list od reference points

引数:

bOnAllActiveLay if true it work on all active layers other way only on active loaded layer in plane 1

void printListRefPoints ( )

Print list od reference points on active loaded layer in plane 1

**boolean promptBoolean (String *optName,*
boolean *def*
)**

Show a checkbox field in prompt dialog

引数:

optName String name, will be visible in prompt dialog as a label
def boolean, default value **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

boolean a default value

**double promptDouble (String *doubleName,*
double *def*
)**

Show a double type field in prompt dialog

引数:

doubleName String name, will be visible in prompt dialog as a label

def double, default value **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

double a default value

void promptEnd ( )

プロンプトシーケンスを閉じ プロンプトダイアログを表示する

String promptFileName (String *strLabel,*
String *def*
)

Show a text field with browse button in prompt dialog

引数:

strLabel variable name, will be visible in prompt dialog as a label

def String, default value for file Name **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

fileName a default value

int promptInteger (String *intName,*
int *def*
)

Show an integer type field in prompt dialog

引数:

intName String name, will be visible in prompt dialog as a label

def int, default value **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

integer a default value

void promptLabel (String *labelText*)

Show label with given message

引数:

labelText String label text **Note:** if the parameter is variable, it must be defined globally for script.

Line promptLine (String *lineName,*
double *fromX,*

```
                double  fromY,
                double  toX,
                double  toY
              )
```

Show four unit fields to define **Line** from **Point** X and Y and to **Point** X and Y coordinates

引数:

lineName	String name, will be visible in prompt dialog as a label
fromX	default X coordinate of the from **Point**
fromY	default Y coordinate of the from **Point**
toX	default X coordinate of the to **Point**
toY	default Y coordinate of the to **Point** **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

Line object

```
Line promptLine ( String  lineName,
                  Line    defLine
                )
```

Show four unit fields to define **Line** from **Point** X and Y and to **Point** X and Y coordinates

引数:

lineName	String name, will be visible in prompt dialog as a label
defLine	default **Line** **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

Line object

```
String promptOption ( String      optName,
                      ObjectList  options,
                      String      def
                    )
```

Show a combobox field in prompt dialog

引数:

optName	String name, will be visible in prompt dialog as a label
options	ObjectList with items shown in combobox
def	one of the string defined on options, default (selected) value **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

String a default text

```
Point promptPoint ( String  pointName,
                    double  ptX,
                    double  ptY
                  )
```

Show two unit field to define **Point** coordinate X and Y

引数:

pointName	String name, will be visible in prompt dialog as a label
ptX	default X coordinate of the **Point**
ptY	default Y coordinate of the **Point** **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

Point object

Point promptPoint (String *pointName,*
Point *point*
)

Show two unit field to define **Point** coordinate X and Y

引数:

pointName	String name, will be visible in prompt dialog as a label
point	default **Point** **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

Point object

Rectangle promptRectangle (String *rectangleName,*
double *xmin,*
double *xmax,*
double *ymin,*
double *ymax*
)

Show four unit fields to define **Rectangle** X min, X max, Y min and Y max coordinates

引数:

rectangleName	String name, will be visible in prompt dialog as a label
xmin	default minimal rectangle X coordinate
xmax	default maximal rectangle X coordinate
ymin	default minimal rectangle Y coordinate
ymax	default maximal rectangle Y coordinate **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

Rectangle object

Rectangle promptRectangle (String *rectangleName,*
Rectangle *rectangle*
)

Show four unit fields to define **Rectangle** X min, X max, Y min and Y max coordinates

引数:

rectangleName String name, will be visible in prompt dialog as a label
rectangle default **Rectangle** **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:
Rectangle object

**void promptStart (String *sSetName,*
String *sTitle*
)**

Initiates prompt dialog for users input and sets the name of the variables set Must be the first command before a prompt commands sequence

引数:
sSetName the name of the variable set. If it is `null` or is empty it is ignored.
sTitle the PromptDialog title. If it is `null` or is empty it is ignored **Note:** if the parameter is variable, it must be defined globally for script.

void promptStart (String *sSetName*)

Initiates prompt dialog for users input and sets the name of the variables set Must be the first command before a prompt commands sequence

引数:
sSetName the name of the variable set. If it is `null` or is empty it has the same meaning as **promptStart()**. **Note:** if the parameter is variable, it must be defined globally for script.

void promptStart ( )

ユーザー入力のためのプロンプトダイアログを起動する 複数プロンプトコマンドのシーケンスに先立って 最初のコマンドとして設定されなければならない

**String promptString (String *strName,*
String *def*
)**

Show a text field in prompt dialog

引数:
strName String name, will be visible in prompt dialog as a label
def String, default value **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:
String a default value

**double promptUnit (String *unitName,*
double *def,***

String *units*
)

Show an unit field in prompt dialog

引数:

unitName String name, will be visible in prompt dialog as a label
def double, default value
units "mm", "mil" or "inch" to determine units of the default value **Note:** if the parameter is variable, it must be defined globally for script.

戻り値:

a double value in current units

void qmerge (String *sOptions* **)**

qmerge description

引数:

sOptions qmerge options

void quitBlockEdit (boolean *bSave,*
boolean *bKeepLink*
)

Aperture Manager: Quits the Block Definition Edit Mode

引数:

bSave true: save changes; false: discard changes
bKeepLink true: keeps block links

void quitBlockEdit (boolean *bSave* **)**

Aperture Manager: Quits the Block Definition Edit Mode. The saved (if bSave is `true`) block keeps link when Ucam.db key `uiape_block_keeplink` is set to 1

引数:

bSave true: save changes; false: discard changes

void quitBlockMultiEdit (boolean *bSave,*
boolean *bKeepLink*
)

Aperture Manager: Quits the Block Definition Multi Edit Mode

引数:

bSave true: save changes; false: discard changes
bKeepLink true: keeps block links

void quitBlockMultiEdit (boolean *bSave*)

Aperture Manager: Quits the Block Definition Multi Edit Mode. The saved (if bSave is `true`) block keeps link when Ucam.db key `uiape_block_keeplink` is set to 1

参照:
quitBlockEdit(boolean, boolean)

引数:
bSave true: save changes; false: discard changes

void quitComplexEdit (boolean *bSave*)

アパーチャマネジャー： コンプレックス定義編集モードから出る

引数:
bSave true：変更を保存する false：変更を保存しない

void quitConfirm ( )

Quit application after asking for confirmation.

void readAmli (String *sPath*)

read AMLI files

引数:
sPath - path to the root directory

例外:
IOException

void recognizeContours (String *sConName*)

輪郭を認識する

引数:
sConName 変換ファイル／輪郭名

Rectangle Rectangle (Rectangle *rect*)

長方形のコピーを作成する

引数:
rect 元になる長方形

戻り値:
長方形

```
Rectangle Rectangle ( double  xmin,
                      double  ymin,
                      double  xmax,
                      double  ymax,
                      String  units
                    )
```

4つの座標から長方形を作成する

引数:

xmin 長方形の左境界
ymin 長方形の下境界
xmax 長方形の右境界
ymax 長方形の上境界
units Ucam単位

戻り値:

長方形

```
Rectangle Rectangle ( double  xmin,
                      double  ymin,
                      double  xmax,
                      double  ymax
                    )
```

4つの座標から長方形を作成する

引数:

xmin 長方形の左境界
ymin 長方形の下境界
xmax 長方形の右境界
ymax 長方形の上境界

戻り値:

長方形

```
Rectangle Rectangle ( Point  ptPoint1,
                      Point  ptPoint2
                    )
```

2つのポイントからなる長方形囲み領域を作成する

引数:

ptPoint1 1つ目ポイント
ptPoint2 2つ目ポイント

戻り値:

長方形

void redo ( )

Redo an action

void registerJobOnPoints ( )

Does a 3 point transformation on a job/layer to correct distortion of scanned artwork. In the layer(s) point 1 to 6 must be defined and the registration places point 1 on 4, 2 on 5 and 3 on 6. All objects are moved with the average x and y movement of those points.

void registerLayers ( )

Performs automatic layer registration on the job. All active layers are registered on a reference layer. The layer in plane 3 is taken as a reference layer The selected pads in the reference layer are taken as a reference for registration.

void registerOnGrid (double *GridStep_x*,
double *GridStep_y*,
double *GridOri_x*,
double *GridOri_y*,
double *dXRadius*,
double *dYRadius*
)

全ての（選択された）オブジェクトを指定グリッドにスナップする 選択オブジェクトに作用する

引数:

GridStep_x (X座標) グリッドステップ
GridStep_y (Y座標) グリッドステップ
GridOri_x (X座標) グリッド原点
GridOri_y (Y座標) グリッド原点
dXRadius X方向の指定距離内にあるオブジェクトのみを移動させる
dYRadius Y方向の指定距離内にあるオブジェクトのみを移動させる

void registerOnGrid (Point *GridStep*,
Point *GridOri*,
double *dXRadius*,
double *dYRadius*
)

全ての（選択された）オブジェクトを指定グリッドへスナップする 選択オブジェクトに作用する

引数:

GridStep グリッドステップ
GridOri グリッド原点
dXRadius X方向の指定距離内にあるオブジェクトのみを移動させる
dYRadius Y方向の指定距離内にあるオブジェクトのみを移動させる

void registerOnPads (double *dRadius,*
boolean *bOnFlashPoint*
)

Registers a job/layer on a reference layer. All pads that are not registered and all tracks are offset with the average offset of the pads. The layer in plane 2 is taken as a reference layer. Works on selected objects.

引数:

dRadius	Maximum distance between pads. If the distance between a pad and a reference pad is bigger than the radius, the pad is not registered.
bOnFlashPoint	If true register only on pad flashpoint (not the edges)

void removeApeAttr ( )

removes all aperture attributes on layer lay

void removeJobAttr ( )

removes all job attributes

void removeLayAttr ( )

removes all layer attributes on layer lay

void removeNetAttr (int *iNetNumber,*
String *sAttrName,*
String *sAttrValue*
)

Removes all net attributes with given name on job and given net

引数:

iNetNumber	Net number
sAttrName	Net attribute name
sAttrValue	Net attribute value

void removeNetAttr (int *iNetNumber,*
String *sAttrName*
)

Removes all net attributes with given name on job and given net

引数:

iNetNumber	Net number
sAttrName	Net attribute name

**void removeNetAttr (String *sAttrName,*
String *sNetName*
)**

Removes all net attributes with given name and value on job

引数:

sAttrName Net attribute name
sNetName

void removeNetAttr (String *sAttrName*)

Removes all net attributes with given name on job

引数:

sAttrName Net attribute name

void removeObjAttr ( )

removes all object attributes on layer lay

**void removeObjectAttribute (String *sAttrName,*
String *sAttrValue*
)**

removeObjectAttribute Sets the object attribute with the given name and value. Takes objects from active layers from current job by Ucam options. (e.g. all selected objects)

引数:

sAttrName The object attribute name
sAttrValue The object attribute value

void removeObjectAttribute (String *sAttrName*)

removeObjectAttribute remove the object attribute with the given name. Takes objects from active layers from current job by Ucam options. (e.g. all selected objects)

引数:

sAttrName The object attribute name

void removeYsphotechPlotstamp (int *plotstampID*)

removes the plotstamp with a certain id

引数:

plotstampID the ID of the plotstamp to be removed *

void replaceApeByCurrent ( )

全アクティブレイヤ上の選択オブジェクトをカレントアパーチャで置換する

void replaceApertures ( )

アパーチャマネジャー： カレントレイヤの選択アパーチャをカレントアパーチャで置換する

void replaceBitmapByContours ( )

replaceBitmapByContours

void replaceDrawsWithArcs (double *tolerance*)

Tries to replace chains of minimum 3 consecutive vectors by an arc within given tolerance. If the operation succeeds, all the vectors are deleted and a new arc is created. The start/end points of the vector chain become the start and end points of the new arc.

引数:

tolerance The tolerance is the maximum distance from the arc to any of the vectors being replaced.

void replaceInnersByOuters ( )

インナー輪郭をアウター輪郭に置換する

**void replaceZeroLengthDraws (double *dMaxLength,*
boolean *bFunctional,*
boolean *bNonFunctional*
)**

Replace Zero Length Draws and Arcs with flashes

引数:

dMaxLength	Maximum length of objects to be replaced
bFunctional	Replace objects within functional copper if true
bNonFunctional	Replace objects within non-functional copper if true

void reproducePanel (String *report*)

パネル面付け

引数:

report レポートファイル(.prf)

void reset ( )

ナンバーズダイアログの全ての値をリセットする

void resetCFMEE ( )

Resets CFMEE settings: removes all alignment points

**void resetCores (int *iTop,*
String *sAttach*
)**

任意のレイヤ間にあるコアをリセットする

引数:

iTop トップレイヤインデックス
sAttach "top", "bottom", "none" ,"both"のいずれか

void resetCores ( )

全てのコアをリセットする

void resetWorkspace ( )

resets current workspace layout.

void resetYsphotech ( )

Resets Ysphotech settings: removes all alignment points and plotstamps

void restoreArcs (boolean *bPreferFullArc*)

Restore Arcs Repair Invalid Arcs

引数:

bPreferFullArc The parameter influence image for arcs where it does not clear if they are full arcs or zero arcs. If the value is set to true they will be full arcs, other way they will be zero arcs.

void restoreContours (boolean *bPreferFullArc*)

Restore Contours Repair Contours and Complexes

引数:

bPreferFullArc The parameter influence image for arcs where it does not clear if they are full arcs or zero arcs. If the value is set to true they will be full arcs, other way they will be zero arcs.

void returnVariables (ObjectList *returnVariables*)

Terminates called script and returns ObjectList of variables

引数:

returnVariables ObjectList of the return variables

例外:

AbortException after user abort

参照:

runScriptWithReturn(String)

void reverse ( )

選択エレメントを反転させる

void reverseLayer ( )

アクティブレイヤをイメージ反転させる

void reverseLayers ( )

アクティブレイヤをイメージ反転させる

void rotate (double *angle,*
boolean *bUseCenter,*
boolean *bOnRefPoints*
)

原点を中心に選択対象を回転させる この設定は次の操作に相当する: Transform Objects - Edit - Rot 90/Rot270/Rot angle

引数:

angle 角度値（正:左回り もしくは 負:右回り）
bUseCenter tureの場合 回転は中心座標を利用する
bOnRefPoints trueの場合 回転はリファレンスポイントにも適用される

void roundDraw (double *pt_x,*

```
                    double  pt_y,
                    double  dis
                  )
```

ドローの接続点をラウンド形状に置き換える

引数:

pt_x (X座標) ドロー上の **Point** (ポイント)
pt_y (Y座標) ドロー上の **Point** (ポイント)
dis 丸みの半径（フィレットを参照） 半径＝ドローの1/2の長さにするには0を入れる

```
void roundDraw ( Point   pt,
                 double  dis
               )
```

ドローの接続点をラウンド形状に置き換える

引数:

pt ドロー上の **Point** (ポイント)
dis 丸みの半径（フィレットを参照） 半径＝ドローの1/2の長さにするには0を入れる

```
void routStatistics ( String  doOption )
```

カレントジョブのルートレポートを作成する

引数:

doOption “all”もしくは”sel”

```
void routStatistics (  )
```

Create the rout report for the current job.

```
void runDRC ( String   sCfgFile,
              boolean  bBuildNetlist,
              String   sUseNetlist,
              boolean  bSelErrors
            )
```

Smart DRC チェックを実行する

引数:

sCfgFile 設定ファイル
bBuildNetlist ネットリストを構築する（true, false）
sUseNetlist ネットリストを使用する（"none", "layer", "job"）
bSelErrors 選択エラー

```
String runFile ( String  sScriptPath,
```

ObjectList ***argv***
)

Run script from file(s). Uses variables according to `argv` parameter. if `argv` is `null` the call is the same as **`runFile(String sScript)`** and uses all variables defined in parent interpreter if exists if `argv` is `null` the call is the same as **`runFile(String sScript)`** and uses all variables defined in parent interpreter if exists if `argv` is empty `[{}]` the script is called completelly without input variable if `argv` is eg. `[{"name", 1, true}]` the called script knows one ObjectList variable with the name "args". The items are accessable by the index. Eg. `name = args[0];` and `isAvailable = args[2];`

引数:

sScriptPath file name which is either full path, wildcard mask or just base name, in this case the file is searched in current directory or using

```
hyperscript.script.path
```

ucam.db key.

argv ObjectList with input arguments [{arg1, arg2, ..., argn}] or `null` or empty ObjectList [{}]

戻り値:

String RUN_STATUS_OK if run is OK, otherwise returns message about issue encountered.

例外:

AbortException if any problem encountered during script call

参照:

runFile(String)

runFile(String)

String runFile (String ***sScriptPath*** **)**

Run script from file(s)

引数:

sScriptPath file name which is either full path, wildcard mask or just base name, in this case the file is searched in current directory or using

```
hyperscript.script.path
```

ucam.db key.

戻り値:

String RUN_STATUS_OK if run is OK, otherwise returns message about issue encountered.

例外:

AbortException if any problem encountered during script call

String runScript (String ***sScript,***
ObjectList ***argv***
)

Run (interprete) given text. Uses variables according to `argv` parameter. if `argv` is `null` the call is the same as **`runScript(String sScript)`** and uses all variables defined in parent interpreter if exists if `argv` is `null` the call is the same as **`runScript(String sScript)`** and uses all variables defined in parent interpreter if exists if `argv` is empty `[{}]` the script is called completelly without input variable if `argv` is eg. `[{"name", 1, true}]` the called script knows one ObjectList variable with the name "args". The items are accessable by the index. Eg. `name = args[0];` and `isAvailable = args[2];`

引数:

sScript text of the script.

argv ObjectList with input arguments [{arg1, arg2, ..., argn}] or `null` or empty ObjectList [{}]

戻り値:

String RUN_STATUS_OK if run is OK, otherwise returns message about issue encountered.

例外:

AbortException if any problem encountered during script call

参照:

runScript(String)

runScript(String)

String runScript (String *sScript*)

Run (interprete) given text **Example:**

```
promptStart();
    script = promptString("String to execute", "saveJob();");
promptEnd;
runScript(script);
runScript(script);
```

引数:

sScript text of the script.

戻り値:

String RUN_STATUS_OK if run is OK, otherwise returns message about issue encountered.

例外:

AbortException if any problem encountered during script call

ObjectList runScriptWithReturn (String *sScript,*
Object[] *argv*
)

Run (interprete) given text. Uses variables according to `argv` parameter. if `argv` is `null` the call is the same as **`runScriptWithReturn(String sScript)`** and uses all variables defined in parent interpreter if exists if `argv` is `null` the call is the same as **`runScriptWithReturn(String sScript)`** and uses all variables defined in parent interpreter if exists if `argv` is empty `[{}]` the script is called completelly without input variable if `argv` is eg. `[{"name", 1, true}]` the called script knows one ObjectList variable with the name "args". The items are accessable by the index. Eg. `name = args[0];` and `isAvailable = args[2];`

引数:

sScript text of the script.

argv ObjectList with input arguments [{arg1, arg2, ..., argn}] or `null` or empty ObjectList [{}]

戻り値:

An ObjectList with returning values.

例外:

AbortException if any problem encountered during script call

ObjectList runScriptWithReturn (String *sScript*)

Run (interprete) given text. Uses all variables defined in parent interpreter if exists.

引数:

sScript text of the script.

戻り値:

An ObjectList with returning values.

例外:

AbortException if any problem encountered during script call

void saveAmli (String *sPath*)

save AMLI files

引数:

sPath - path to the root directory

void saveBuildup (String *sSpec*,
String *sCustomer*,
String *sDrcPar*,
String *sCoreRef*,
String *sPrePregMat*,
String *sCopperMat*,
String *sJobFlow*,
String *sTechCheck*,
String *sAttrSet*,
String *sDatumList*,
ObjectList *layList*
)

カレントジョブのビルドアップを保存する

引数:

sSpec	ビルドアップファイルのファイル仕様
sCustomer	顧客
sDrcPar	drcパラメータファイルの仕様
sCoreRef	uCoreMaterial属性値
sPrePregMat	uPrePregMaterial属性値
sCopperMat	uCopperMaterial属性値
sJobFlow	uJobFlow 属性値
sTechCheck	uTechnologycheck属性値
sAttrSet	uAttributeset属性値
sDatumList	コンマ区切りのデータリスト
layList	レイヤの名前マッピング配列

int saveJob ( )

カレントジョブを保存する

int saveJobAs (String *fullPath,*
String *sVersion*
)

Save Current job as...

引数:

fullPath The full pathname for the current job The target directory must already exist
sVersion version "3" or "6" or "9"

int saveJobAs (String *fullPath*)

Save Current job as...

引数:

fullPath The full pathname for the current job The target directory must already exist

void saveJobAsV3 ( )

Save Current job as version 3. A separate function is needed for the button, to handle the Licensed case well.

void saveJobAsV6 ( )

Save Current job as version 6. A separate function is needed for the button, to handle the Licensed case well.

void saveJobAsV9 ( )

Save Current job as version 9. A separate function is needed for the button, to handle the Licensed case well.

void saveLayer (String *sClass,*
String *sSubClass,*
int *iLayIndex,*
String *sFullPath*
)

レイヤを名前を付けて保存する

引数:

sClass レイヤクラス: "layer", "drill", "extra"のいずれか
sSubClass レイヤサブクラス 例）"outline", "mask", "silk" など
iLayIndex 任意のクラスもしくは任意のサブクラスにおけるレイヤインデックス
sFullPath 新しいファイルのフルパス名（String型）

**void saveLayer (String *layName,*
String *fullPath*
)**

Save Layer with name

引数:

layName Name of layer to be saved
fullPath The full pathname for the dpf layer The target directory must already exist

void saveMessagesAs (String *sFilePath*)

メッセージウィンドウこのコンテンツを任意のファイルパスで保存する

引数:

sFilePath メッセージは 指定のパスのファイルに保存される

void saveOrder ( )

ルート順序（rout order）に対する修正が全て行われてから 定義された順序を保存する

void saveSplitConfig (String *sConfigName*)

Save split configuration for the given name

引数:

sConfigName name of configuration

void saveUFD (String *sUFDName*)

Saves the contents of the current fault database to the file with the given specification.

引数:

sUFDName UFD file specification

void saveWorkspace ( )

Save currently set workspace layout. NOTE: The same as menu command Workspaces > Save

void saveWorkspace (String *sWorkspaceName*)

Save workspace layout as file with a given name in XML format.

引数:

sWorkspaceName

void saveWorkspaceAs (String *sWorkspaceName*)

Save currently set workspace layout as file with a given name. NOTE: The same as menu command Workspaces > Save as...

引数:

sWorkspaceName

void scale (double *dScaleValue,*
boolean *bUseCenter*
)

Scale selections

引数:

dScaleValue	Scale factor (greater than 1 for up and less than 1 for down)
bUseCenter	If true, Scale is done around center

void scaleObjectOnAttribute (String *sName,*
double *dScaleFactor,*
double *dMinClearance*
)

Scales all objects (flash, draw, arc, txt) that have attribute 'name'. Scale 'factor' defines margin between objects which may define together scale zone. All objects in zone are scaled with factor. The center is used as center of the enclosing rectangle of the objects in a zone.

引数:

sName	an attribute name.
dScaleFactor	a scale factor.
dMinClearance	a minimal clearance between characters that should be respected after enlarging the text.

void screendump ( )

Makes a screendump of the selected area and gives you the ability to print it.

void secureEtchCompensation (double *dPadSpread,*
double *dSmdSpread,*
double *dTrackSpread,*
double *dAreaSpread,*
double *dPadPadClearance,*
double *dPadSmdClearance,*
double *dPadTrackClearance,*
double *dPadAreaClearance,*

```
                                double  dSmdSmdClearance,
                                double  dSmdTrackClearance,
                                double  dSmdAreaClearance,
                                double  dTrackTrackClearance,
                                double  dTrackAreaClearance,
                                double  dAreaAreaClearance,
                                String  sContourMethod,
                                boolean bProcessSameNetSpacing,
                                boolean bBackupOriginalLayer,
                                boolean bCheckMissingCopper,
                                boolean bFastMode,
                                int     iShiftMode,
                                double  dMinCopper
                               )
```

Secure Etch Compensation

引数:

dPadSpread	The spread value to apply to circular pads
dSmdSpread	The spread value to apply to non circular pads
dTrackSpread	The spread value to apply to draws
dAreaSpread	The spread value to apply to areas
dPadPadClearance	The clearance to keep between circular pads
dPadSmdClearance	The clearance to keep between circular pads and non circular pads
dPadTrackClearance	The clearance to keep between circular pads and draws
dPadAreaClearance	The clearance to keep between circular pads and areas
dSmdSmdClearance	The clearance to keep between non circular pads
dSmdTrackClearance	The clearance to keep between non circular pads and draws
dSmdAreaClearance	The clearance to keep between non circular pads and areas
dTrackTrackClearance	The clearance to keep between draws
dTrackAreaClearance	The clearance to keep between draws and areas
dAreaAreaClearance	The clearance to keep between areas
sContourMethod	The contour compensation method ("spread", "stroke", or "circle")
bProcessSameNetSpacing	Whether to process spacing between objects of the same net as well
bBackupOriginalLayer	If true, make backups of the original layer(s)
bCheckMissingCopper	If true, report errors when original copper was removed
bFastMode	If true, skip slow "select embedded" step to make SEC faster but sometimes less correct
iShiftMode	Mode to shift clearances, 0: no shift, 1: shift to cut only areas, 2: like 1, but keep original copper of areas
dMinCopper	The minimum copper width to keep

```
void secureEtchCompensation ( double  dPadSpread,
                              double  dSmdSpread,
                              double  dTrackSpread,
                              double  dAreaSpread,
                              double  dPadPadClearance,
                              double  dPadSmdClearance,
                              double  dPadTrackClearance,
                              double  dPadAreaClearance,
                              double  dSmdSmdClearance,
```

```
                                double  dSmdTrackClearance,
                                double  dSmdAreaClearance,
                                double  dTrackTrackClearance,
                                double  dTrackAreaClearance,
                                double  dAreaAreaClearance,
                                String  sContourMethod,
                                boolean bProcessSameNetSpacing,
                                boolean bBackupOriginalLayer,
                                boolean bCheckMissingCopper,
                                boolean bFastMode,
                                int     iShiftMode
                               )
```

Secure Etch Compensation

引数:

dPadSpread	The spread value to apply to circular pads
dSmdSpread	The spread value to apply to non circular pads
dTrackSpread	The spread value to apply to draws
dAreaSpread	The spread value to apply to areas
dPadPadClearance	The clearance to keep between circular pads
dPadSmdClearance	The clearance to keep between circular pads and non circular pads
dPadTrackClearance	The clearance to keep between circular pads and draws
dPadAreaClearance	The clearance to keep between circular pads and areas
dSmdSmdClearance	The clearance to keep between non circular pads
dSmdTrackClearance	The clearance to keep between non circular pads and draws
dSmdAreaClearance	The clearance to keep between non circular pads and areas
dTrackTrackClearance	The clearance to keep between draws
dTrackAreaClearance	The clearance to keep between draws and areas
dAreaAreaClearance	The clearance to keep between areas
sContourMethod	The contour compensation method ("spread", "stroke", or "circle")
bProcessSameNetSpacing	Whether to process spacing between objects of the same net as well
bBackupOriginalLayer	If true, make backups of the original layer(s)
bCheckMissingCopper	If true, report errors when original copper was removed
bFastMode	If true, skip slow "select embedded" step to make SEC faster but sometimes less correct
iShiftMode	Mode to shift clearances, 0: no shift, 1: shift to cut only areas, 2: like 1, but keep original copper of areas

```
void secureEtchCompensation ( double  dPadSpread,
                              double  dSmdSpread,
                              double  dTrackSpread,
                              double  dAreaSpread,
                              double  dPadPadClearance,
                              double  dPadSmdClearance,
                              double  dPadTrackClearance,
                              double  dPadAreaClearance,
                              double  dSmdSmdClearance,
                              double  dSmdTrackClearance,
                              double  dSmdAreaClearance,
                              double  dTrackTrackClearance,
```

```
                                double   dTrackAreaClearance,
                                double   dAreaAreaClearance,
                                String   sContourMethod,
                                boolean  bProcessSameNetSpacing,
                                boolean  bBackupOriginalLayer,
                                boolean  bCheckMissingCopper,
                                boolean  bFastMode
                              )
```

Secure Etch Compensation

引数:

dPadSpread	The spread value to apply to circular pads
dSmdSpread	The spread value to apply to non circular pads
dTrackSpread	The spread value to apply to draws
dAreaSpread	The spread value to apply to areas
dPadPadClearance	The clearance to keep between circular pads
dPadSmdClearance	The clearance to keep between circular pads and non circular pads
dPadTrackClearance	The clearance to keep between circular pads and draws
dPadAreaClearance	The clearance to keep between circular pads and areas
dSmdSmdClearance	The clearance to keep between non circular pads
dSmdTrackClearance	The clearance to keep between non circular pads and draws
dSmdAreaClearance	The clearance to keep between non circular pads and areas
dTrackTrackClearance	The clearance to keep between draws
dTrackAreaClearance	The clearance to keep between draws and areas
dAreaAreaClearance	The clearance to keep between areas
sContourMethod	The contour compensation method ("spread", "stroke", or "circle")
bProcessSameNetSpacing	Whether to process spacing between objects of the same net as well
bBackupOriginalLayer	If true, make backups of the original layer(s)
bCheckMissingCopper	If true, report errors when original copper was removed
bFastMode	If true, skip slow "select embedded" step to make SEC faster but sometimes less correct

void secureEtchCompensationUndo ( )

Secure Etch Compensation Undo Undoes any previous SEC. Deletes the negative clearance apertures and shrinks any objects carrying the spread value in the attribute named by ucam.db key "ClearanceManager.spreadAttribute".

void selectAll ( )

全てのオブジェクトを選択する

void selectAll (String *selectMode*)

全てのオブジェクトを選択／非選択する

引数:

selectMode + (選択する)もしくは - (非選択する)のいずれかを設定

void selectAllApertures ( )

アパーチャマネジャー: アパーチャリストの全アパーチャの全オブジェクトを選択する

**void selectAllContours (String *selectMode,*
String *conMode*
)**

Select all contours.

引数:

selectMode Either + (select) or - (deselect)
conMode Defines what should be selected. "i" for inner contours, "o" for outer contours and "r" for regions.

**void selectAllContours (String *selectMode,*
double *xSize,*
double *ySize,*
String *conMode*
)**

Select or deselect all contours.

引数:

selectMode Either + (select) or - (deselect)
xSize The maximum width (x size) for the contour enclosing box.
ySize The maximum height (y size) for the contour enclosing box.
conMode Defines what should be selected. "i" for inner contours, "o" for outer contours and "r" for regions.

void selectAmbiguousContours (String *selectMode*)

曖昧な輪郭を選択する

引数:

selectMode + (選択する)もしくは - (非選択する)のいずれかを設定

void selectAperture (ObjectList *apeIndexArray*)

アパーチャマネジャー: アパーチャのオブジェクトを選択する

引数:

apeIndexArray カレントレイヤのアパーチャインデックス配列

void selectAperture ( )

アパーチャマネジャー： カレントアパーチャのオブジェクトを選択する

void selectAperturesBiggerThan (String *selectMode,*
double *dx,*
double *dy*
)

特定の値より大きなアパーチャを選択する

引数:

selectMode	Either + (選択する)もしくは - (非選択する)のいずれかを設定
dx	長方形囲み領域のXサイズ値
dy	長方形囲み領域のYサイズ値

void selectAperturesSmallerThan (String *selectMode,*
double *dx,*
double *dy*
)

特定の値と同等もしくはそれより小さいアパーチャを選択する

引数:

selectMode	+ (選択する)もしくは - (非選択する)のいずれかを設定
dx	長方形囲み領域のXサイズ値
dy	長方形囲み領域のYサイズ値

void selectByApeAttributeNames (String *selectMode,*
String[] *sName*
)

Select or deselect all objects of the specified attribute names.

引数:

selectMode	Either + (select) or - (deselect)
sName	

void selectByApertureShape (String *selectMode,*
String *apertureShapes*
)

非推奨:

Select or deselect all objects of the specified object type.

引数:

selectMode	Either + (select) or - (deselect)
apertureShapes	Comma separated list of cir, don, rec, squ, box, oct, com, the, con, tex, blo

void selectByAttributeName (String *selectMode,*
String *sName*
)

非推奨:
指定した属性名の全オブジェクトを選択／非選択する

引数:
selectMode + (選択する)もしくは - (非選択する)のいずれかを設定
sName - 属性名

void selectByAttributeValue (String *selectMode,*
String *sName,*
String *sValue*
)

非推奨:
指定の属性値の全オブジェクトを選択／非選択する

引数:
selectMode + (選択する)もしくは - (非選択する)のいずれかを設定
sName - 属性名
sValue - 属性値

void selectByObjectType (String *selectMode,*
String *objectTypes*
)

非推奨:
Select or deselect all objects of the specified object type.

引数:
selectMode Either + (select) or - (deselect)
objectTypes Comma separated list of f (flash), d (draw), a (arc) or v (vector text)

void selectChained (String *selectMode,*
double *pt_x,*
double *pt_y*
)

連鎖されたドローを選択する

引数:
selectMode + (選択する)もしくは - (非選択する)のいずれかを設定
pt_x (X 座標) スタートポイント
pt_y (Y 座標) スタートポイント

void selectChained (String *selectMode,*

```
                    Point  pt
                  )
```

連鎖されたドローを選択する

引数:

selectMode	+ (選択する)もしくは - (非選択する)のいずれかを設定
pt	スタートポイント

```
void selectChained ( String  selectMode,
                     double  pt_x,
                     double  pt_y,
                     double  dTolerance
                   )
```

連鎖されたドローを選択する

引数:

selectMode	+ (選択する)もしくは - (非選択する)のいずれかを設定
pt_x	(X 座標) スタートポイント
pt_y	(Y 座標) スタートポイント
dTolerance	クリックの半径

```
void selectChained ( String  selectMode,
                     Point   pt,
                     double  dTolerance
                   )
```

連鎖されたドローを選択する

引数:

selectMode	+ (選択する)もしくは - (非選択する)のいずれかを設定
pt	スタートポイント
dTolerance	クリックの半径

```
void selectChainedObjects ( String   selectMode,
                            double   pnt_x,
                            double   pnt_y,
                            double   pixelRadius,
                            double   dOffCenter,
                            boolean  bSameApe,
                            boolean  bSameOrientation
                          )
```

引数:

selectMode	Either "+" or "-" to define type of selection
pnt_x	(X coordinate) clicked point
pnt_y	(Y coordinate) clicked point
pixelRadius	radius around pnt, tolerance to find an object

dOffCenter	Tolerance between end point of reference object and start point of the following object
bSameApe	If true the chain may only constist of objects of the same aperture
bSameOrientation	If true the object must have the same orientation as the reference

void selectChainedObjects (String *selectMode,*
Point *pnt,*
double *pixelRadius,*
double *dOffCenter,*
boolean *bSameApe,*
boolean *bSameOrientation*
)

引数:

selectMode	Either "+" or "-" to define type of selection
pnt	clicked point
pixelRadius	radius around pnt, tolerance to find an object
dOffCenter	Tolerance between end point of reference object and start point of the following object
bSameApe	If true the chain may only constist of objects of the same aperture
bSameOrientation	If true the object must have the same orientation as the reference

void selectContourByClick (String *selectMode,*
double *pt_x,*
double *pt_y,*
String *conMode*
)

Select contour by click (De)selects a region in a job/layer. A region is a copper area defined by 1 outer contour and 1 or more inner contours. The region should enclose target point.

引数:

selectMode	Either + (select) or - (deselect)
pt_x	(X coordinate) The target point.
pt_y	(Y coordinate) The target point.
conMode	Defines what should be selected. "i" for inner contours, "o" for outer contours and "r" for regions.

void selectContourByClick (String *selectMode,*
Point *pt,*
String *conMode*
)

Select contour by click (De)selects a region in a job/layer. A region is a copper area defined by 1 outer contour and 1 or more inner contours. The region should enclose target point.

引数:

selectMode	Either + (select) or - (deselect)
pt	The target point.

conMode Defines what should be selected. "i" for inner contours, "o" for outer contours and "r" for regions.

void selectContoursInWindow (String *selectMode,*
double *rect_xmin,*
double *rect_ymin,*
double *rect_xmax,*
double *rect_ymax,*
String *conMode*
)

Select contours inside selection window (De)selects all regions in a job/layer inside a rectangular area. A region is a copper area defined by 1 outer contour and 1 or more inner contours.

引数:

selectMode Either "+" (select) or "-" (deselect)
rect_xmin (left boundary of rectangle) The target rectangle.
rect_ymin (bottom boundary of rectangle) The target rectangle.
rect_xmax (right boundary of rectangle) The target rectangle.
rect_ymax (top boundary of rectangle) The target rectangle.
conMode Defines what should be selected. "i" for inner contours, "o" for outer contours and "r" for regions.

void selectContoursInWindow (String *selectMode,*
Rectangle *rect,*
String *conMode*
)

Select contours inside selection window (De)selects all regions in a job/layer inside a rectangular area. A region is a copper area defined by 1 outer contour and 1 or more inner contours.

引数:

selectMode Either "+" (select) or "-" (deselect)
rect The target rectangle.
conMode Defines what should be selected. "i" for inner contours, "o" for outer contours and "r" for regions.

void selectContoursWithThinRegion (double *dThin,*
String *selectMode*
)

Select contours with thin region.

引数:

dThin value in current units
selectMode Either + (select) or - (deselect)

void selectCurrentAperture (String *selectMode*)

カレントアパーチャを使用している全オブジェクトを選択する

引数:

selectMode + (選択する)もしくは - (非選択する)のいずれかを設定

void selectCurrentApertureDefinition (String *selectMode*)

Select all objects using current aperture definition and the current aperture number.

引数:

selectMode Either + (select) or - (deselect)

void selectCurrentObject (String *selectMode*)

カレントオブジェクトを選択する

引数:

selectMode + (選択する)もしくは - (非選択する)のいずれかを設定

void selectDoubles (String *selectMode,*
double *tolerance*
)

Select doubles. (De)selects all objects that are duplicated.

引数:

selectMode Either "+" or "-". Selects when "+", deselects when "-".
tolerance The tolerance on the aperture definition.

void selectEmbedded (String *selectMode,*
double *tolerance*
)

Select embedded. (De)selects all embedded objects.

引数:

selectMode Either "+" or "-". Selects when "+", deselects when "-".
tolerance Tolerance on aperture definition

void selectFlashesLongerThan (String *selectMode,*
double *rRefRatio*
)

ジョブにおいて フラッシュの長手と短手を比較し 長手が短手より dRefRatioで指定する倍率以上に長ければ そのフラッシュを選択／非選択する

引数:

selectMode - + (選択する)もしくは - (非選択する)のいずれかを設定
rRefRatio - 指定倍率

int selectHornablePads (String *selectMode,*
String *apertureShapes*
)

Add hornable pads (rec and/or box) to selection or remove them from selection. Shows an error to the user if some parameters where invalid

引数:
selectMode Either "+" (select) or "-" (deselect), default value is "+"
apertureShapes options are "rec" or "box" default value is "rec,box".

戻り値:
Number of (de)selected hornable pads

boolean selectInvalidArcs ( )

Select Invalid Arcs

戻り値:
false

int selectInvalidArcs (String *sSelectMode,*
double *dDeviation,*
String *sLimit*
)

Select/Deselect invalid arcs by parameters

引数:
sSelectMode "+" for select invalid arcs or "-" deselect deselect arcs
dDeviation is difference from distance Center point to To point or Center point to From point
sLimit ">" for select invalid arcs with the deviation bigger than the value dDeviation "<" for select invalid arcs with the deviation smaller than the value dDeviation

戻り値:
number of selected invalid arcs

void selectIsolatedFlashes (String *selectMode*)

孤立フラッシュ（ライン接続のない単独のフラッシュ）を選択／非選択する

引数:
selectMode オブジェクトを+ (選択する)もしくは - (非選択する)のいずれかを設定

void selectMesh (String *selectMode*)

Select mesh.

引数:

selectMode Either + (select) or - (deselect)

```
int selectNetByClick ( String  selectMode,
                       double  pt_x,
                       double  pt_y
                     )
```

プレーン1 プレーン2 プレーン3の順に レイヤ上で クリックによってネットを選択する

引数:

selectMode + (選択する)オブジェクトもしくは- (非選択する)オブジェクトのいずれかを設定
pt_x (X 座標) クリック場所
pt_y (Y 座標) クリック場所

戻り値:

クリックポイントのネット番号

```
int selectNetByClick ( String  selectMode,
                       Point   pt
                     )
```

プレーン1 プレーン2 プレーン3の順に レイヤ上で クリックによってネットを選択する

引数:

selectMode + (選択する)オブジェクトもしくは - (非選択する)オブジェクトのいずれかを設定
pt クリック場所

戻り値:

クリックポイントのネット番号

```
int selectNetByName ( String  selectMode,
                      String  sNetName
                    )
```

Select net by name in current job (active layers).

引数:

selectMode Either "+" (select) or "-" (deselect) objects
sNetName Net name

```
void selectNetByNumber ( String   selectMode,
                         int      net,
                         boolean  bSelectShaved,
                         boolean  bSelectBroken
                       )
```

Select net by number in current job (active layers).

引数:

selectMode	Either + (select) or - (deselect) objects
net	Net number
bSelectShaved	Specifies if shaved objects should be selected, or not.
bSelectBroken	Specifies if broken objects should be selected, or not.

**void selectNetByNumber (String *selectMode,*
int *net*
)**

カレントジョブ（アクティブレイヤ）において ネット番号を指定してネット選択する

引数:

selectMode	オブジェクトを+ (選択する)もしくは- (非選択する)のいずれかを設定
net	ネット番号

**void selectNetByTestpoints (String *selectMode,*
int *nbt*
)**

テストポイント数により ネットを選択する

引数:

selectMode	オブジェクトを + (選択する)もしくは - (非選択する)のいずれかを設定
nbt	テストポイント数

void selectNetsWithoutPads (String *selectMode*)

パッドのないネットを選択する

引数:

selectMode	オブジェクトを + (選択する)もしくは - (非選択する)のいずれかを設定

String selectNonFunctionalPads ( )

selectNonFunctionalPads カレントジョブで非機能的なパッドを選択する

**void selectObjectAttribute (String *sAttrName,*
String *sAttrValue*
)**

非推奨:
selectObjectAttribute は カレントジョブにおいて 任意の属性名および属性値の属性が設定された オブジェクトを選択する

引数:

sAttrName オブジェクト属性名
sAttrValue オブジェクト属性値

void selectObjectAttribute (String *sAttrName*)

非推奨:
selectObjectAttribute は カレントジョブにおいて 任意の属性名の属性が設定された オブジェクトを選択する

引数:
sAttrName オブジェクト属性名

void selectObjectByAttribute (String *sAttrName,*
String *sAttrValue*
)

selectObjectAttribute は カレントジョブにおいて 任意の属性名および属性値の属性が設定された オブジェクトを選択する

引数:
sAttrName オブジェクト属性名
sAttrValue オブジェクト属性値

void selectObjectByAttribute (String *sAttrName*)

selectObjectAttribute は カレントジョブにおいて 任意の属性名の属性が設定された オブジェクトを選択する

引数:
sAttrName オブジェクト属性名

void selectObjectByAttributeName (String *selectMode,*
String *sName*
)

指定した属性名の全オブジェクトを選択／非選択する

引数:
selectMode + (選択する)もしくは - (非選択する)のいずれかを設定
sName - 属性名

void selectObjectByAttributeValue (String *selectMode,*
String *sName,*
String *sValue*
)

指定の属性値の全オブジェクトを選択／非選択する

引数:

selectMode + (選択する)もしくは - (非選択する)のいずれかを設定
sName - 属性名
sValue - 属性値

**void selectObjectByShape (String *selectMode,*
String *apertureShapes*
)**

Select or deselect all objects of the specified object type.

引数:

selectMode Either + (select) or - (deselect)
apertureShapes Comma separated list of cir, don, rec, squ, box, oct, com, the, con, tex, blo

**void selectObjectByType (String *selectMode,*
String *objectTypes*
)**

Select or deselect all objects of the specified object type.

引数:

selectMode Either + (select) or - (deselect)
objectTypes Comma separated list of f (flash), d (draw), a (arc) or v (vector text)

void selectOpenContours (String *selectMode*)

オープンしている輪郭を選択する

引数:

selectMode + (選択する)もしくは - (非選択する)

void selectOverlappingContours (String *selectMode*)

オーバーラップしている輪郭を選択する

引数:

selectMode + (選択する)もしくは - (非選択する)

void selectOverlaps (String *selectMode*)

オーバーラップしているオブジェクトを選択する

引数:

selectMode + (選択する)もしくは - (非選択する)

void selectPainted (String *selectMode*)

Select painted objects. Painted objects are a collection of tracks that touch and/or overlap.

引数:

selectMode Either "+" or "-". Selects when "+", deselects when "-".

int selectPaintedAreas (boolean *bUseLoops*,
boolean *bExcludeChains*
)

Selects all painted data in a current layer. Painted data are a collection of tracks that are covered on one side by other tracks.

引数:

bUseLoops Use a loop with a number of iterations
bExcludeChains Exclude chains from the painted area

int selectPaintedAreas ( )

Selects all painted data in a current layer. Painted data are a collection of tracks that are covered on one side by other tracks.

void selectPlotStamps (String *selectMode*)

Select or deselect all plot stamps.

引数:

selectMode Either + (select) or - (deselect)

void selectPolygon (boolean *bInside*,
boolean *bOutside*,
boolean *bCrossing*,
String *sShapes*,
String *sObjects*,
String *selectMode*,
ObjectList *polygonPoints*
)

Select Polygon.

引数:

bInside if true, select objects inside the polygon
bOutside if true, select objects outside the polygon
bCrossing if true, select object crossing the polygon outline
sShapes Possible values are "cir", "rec", "squ", "box", "oct", "con", "com", "the", "blo" and "txt", "don", "und". Multiple shapes can be specified in a comma separated list. Specify null, or an empty string to select all shapes.

Example: "cir,box,com".

引数:

sObjects The object name to (de)select. Possible values are "f", "d", "a" and "v" for flash, draw, arc and vtext. Multiple objects can be specified. Specify null, or an empty string to select all objects.

Example: "da", or "d,a".

引数:

selectMode Either + (select) or - (deselect) objects
polygonPoints outline of selections

void selectReferenceLayer (String *selectMode*)

リファレンスレイヤ内において フラッシュポイントをもつオブジェクトを選択する

引数:

selectMode オブジェクトを+ (選択する)もしくは- (非選択する)のいずれかを設定

void selectReverse (String *selectMode*)

リバース極性のオブジェクトを選択する

引数:

selectMode オブジェクトを+ (選択する)もしくは- (非選択する)のいずれかを設定

void selectSmallContours (String *selectMode,*
double *xSize,*
double *ySize,*
String *conMode*
)

Select small contours. (De)selects contours which size is smaller than the given dimensions.

引数:

selectMode Either "+" or "-". Selects when "+", deselects when "-".
xSize The maximum width (x size) for the contour enclosing box.
ySize The maximum height (y size) for the contour enclosing box.
conMode Defines what should be selected. "i" for inner contours, "o" for outer contours and "r" for regions.

void selectSmallSurface (String *selectMode,*
double *surface,*
String *conMode*
)

微少面積の輪郭領域を選択する

引数:

selectMode + (選択する)もしくは - (非選択する)

surface 四角ユニットの領域
conMode Region（リージョン）, Outer（外部）,Inner（内部）のいずれか

void selectSmallTracks (String *selectMode,*
String *lenMode,*
String *dstMode,*
double *maxLength*
)

微少サイズのトラックを選択する

引数:

selectMode + (選択する)もしくは- (非選択する)
lenMode “abs”（長さ）もしくは”rel”（比率）を選択する
dstMode "ctc"（センターからセンター） ”oto”（外側から外側）
maxLength 最大長

void selectTouchingObjects (String *selectMode,*
double *pnt_x,*
double *pnt_y,*
double *pixelRadius*
)

引数:

selectMode Either "+" or "-" to define type of selection
pnt_x (X coordinate) clicked point
pnt_y (Y coordinate) clicked point
pixelRadius radius around pnt, tolerance to find an object

void selectTouchingObjects (String *selectMode,*
Point *pnt,*
double *pixelRadius*
)

引数:

selectMode Either "+" or "-" to define type of selection
pnt clicked point
pixelRadius radius around pnt, tolerance to find an object

void selectWindow (String *selectMode,*
double *rect_xmin,*
double *rect_ymin,*
double *rect_xmax,*
double *rect_ymax,*
String *winopt,*
String *sShapes,*

```
                String  sObjects
            )
```

Select window using board coordinates (De)selects all data in a rectangular area.

引数:

selectMode	Either "+" or "-" to select, or deselect.
rect_xmin	(left boundary of rectangle) rectangular select area
rect_ymin	(bottom boundary of rectangle) rectangular select area
rect_xmax	(right boundary of rectangle) rectangular select area
rect_ymax	(top boundary of rectangle) rectangular select area
winopt	Add "i", "o", and/or "c" to select inside, outside, and/or crossing.

Example: "ic" select all object in rectangle and all crossing objects. objects.

引数:

sShapes	The shape name to (de)select. Possible values are "cir", "rec", "squ", "box", "oct", "con", "com", "the", "blo", "txt", "don" and "und". Multiple shapes can be specified in a comma separated list. Specify null, or an empty string to select all shapes. Example: "cir,box,com".
sObjects	The object name to (de)select. Possible values are "f", "d", "a" and "v" for flash, draw, arc and vtext. Multiple objects can be specified. Specify null, or an empty string to select all objects.

Example: "da", or "d,a".

```
void selectWindow ( String      selectMode,
                    Rectangle   rect,
                    String      winopt,
                    String      sShapes,
                    String      sObjects
                  )
```

Select window using board coordinates (De)selects all data in a rectangular area.

引数:

selectMode	Either "+" or "-" to select, or deselect.
rect	rectangular select area
winopt	Add "i", "o", and/or "c" to select inside, outside, and/or crossing.

Example: "ic" select all object in rectangle and all crossing objects. objects.

引数:

sShapes	The shape name to (de)select. Possible values are "cir", "rec", "squ", "box", "oct", "con", "com", "the", "blo", "txt", "don" and "und". Multiple shapes can be specified in a comma separated list. Specify null, or an empty string to select all shapes. Example: "cir,box,com".
sObjects	The object name to (de)select. Possible values are "f", "d", "a" and "v" for flash, draw, arc and vtext. Multiple objects can be specified. Specify null, or an empty string to select all objects.

Example: "da", or "d,a".

```
void selectZeroLengthDraws ( double   dMaxLength,
                             boolean  bFunctional,
                             boolean  bNonFunctional
                           )
```

Select Zero Length Draws and Arcs

引数:

dMaxLength	Maximum length of objects to be selected
bFunctional	Select objects within functional copper if true
bNonFunctional	Select objects within non-functional copper if true

boolean setApe (int *index*)

Sets current aperture

引数:

index - index of the aperture in the aperture list. Starts by 1 and index must be smaller or equal to aperture count in the aperture list. If the index is incorrect the current aperture is set to `null`

戻り値:

`true` if the index is correct and current aperture is set, `false` if the index is incorrect and current aperture is NOT set.

**void setApertureAttribute (String *sAttributeName,*
String *sAttributeValue*
)**

アパーチャマネジャー： カレントアパーチャの属性を追加／修正する

引数:

sAttributeName	新たな属性名
sAttributeValue	新たな属性値

**void setAttributeOnObject (String *attrName,*
String *attrValue*
)**

非推奨:

This function is GUI ONLY, replaced by addObjectAttribute(sAttrName, sAttrValue)

参照:

addObjectAttribute(String sAttrName, String sAttrValue) Insert attribute on objects

引数:

attrName	Name of attribute
attrValue	Value of attribute

boolean setCurrentAperture (int *iIndex*)

カレントアパーチャを設定する

引数:

iIndex アパーチャリストにおけるアパーチャインデックス 1で始まり インデックスはアパーチャリストのアパーチャ数以下でなければならない インデックスが正しくなければ カレントアパーチャはnullが設定される

戻り値:

インデックスが正しくカレントアパーチャが設定されれば `true`を返す　インデックスに不備がありカレントアパーチャが設定されない場合は`false`を返す

**void setInPlane (int *newPlane,*
int *iIndex*
)**

レイヤにプレーンカラーを設定する

引数:

newPlane ターゲットプレーンカラー
iIndex ジョブのビルドアップにおけるレイヤインデックス

**void setInPlane (int *newPlane,*
String *layName*
)**

レイヤにプレーンカラーを設定する

引数:

newPlane ターゲットプレーンカラー
layName レイヤ名

**void setInPlane (int *newPlane,*
String *layClass,*
String *laySubclass,*
String *attach,*
int *index*
)**

Set layer in plane color and activate/deactivate according to plane action setup in function **VHS.setInPlane**(...). setup in function **VHS.setInPlane**(...).

引数:

newPlane Target plane color
layClass Layer class
laySubclass Layer subclass
attach Attachment top or bottom (only for extra layers)
index Index within the specified class or subclass e.g for a 6 layer job To set the bottom copper layer to plane 1:

```
setInPlane(1, "layer", null, "", 6);
```


```
setInPlane(1, "layer", null, "", 6);
```


The value 6 refers to the 6th layer or

```
setInPlane(1, "layer", "outer", "", 2);
```


```
setInPlane(1, "layer", "outer", "", 2);
```


The value 2 refers to the second layer of subclass "outer". For layers of subclass "extra" the attach mode is also taken into account. To set a bottom mask with index 2 to plane 1:

```
setInPlane(1, "extra", "mask", "bottom", 2);
```

```
setInPlane(1, "extra", "mask", "bottom", 2);
```

Any extra layer can be set to any plane by using its index within the extra layer class. All "extra" layers attached to "top" are numbered from top to bottom, for "extra" layers attached to "bottom" the numbering continues from bottom to top. To set the extra layer with index 4 to plane 1:

```
setInPlane(1, "extra", null, "", 4);
```


```
setInPlane(1, "extra", null, "", 4);
```


Index within the specified drill class or subclass Drill layers are numbered from left to right To set the first plated drill layer of a job with 1 unplated and 2 plated layers to plane 1:

```
setInPlane(1, "drill", null , "", 2);
```


```
setInPlane(1, "drill", null , "", 2);
```


The value 2 refers to the second drill layer or

```
setInPlane (1, "drill", "plated", "", 1);
```


```
setInPlane (1, "drill", "plated", "", 1);
```


The value 1 refers to the first plated drill layer. **Note:** The **`extra.order`** ucam.db key can change the order of the extra layers. Scripts relying on the index of extra layers could fail to execute correctly on other systems.

void setInPlane (int *newPlane,*
String *layClass,*
String *laySubclass,*
String *attach,*
int *index,*
boolean *activate*
)

Set layer in plane color and activate/deactivate.

引数:

newPlane	Target plane color
layClass	Layer class
laySubclass	Layer subclass
attach	Attachment top or bottom (only for extra layers)
index	Index within the specified class or subclass e.g for a 6 layer job To set the bottom copper layer to plane 1:

```
setInPlane(1, "layer", null, "", 6, true);
```


```
setInPlane(1, "layer", null, "", 6, true);
```


The value 6 refers to the 6th layer or

```
setInPlane(1, "layer", "outer", "", 2, true);
```


```
setInPlane(1, "layer", "outer", "", 2, true);
```


The value 2 refers to the second layer of subclass "outer" For layers of subclass "extra" the attach mode is also taken into account. To set a bottom mask with index 2 to plane 2 and deactivate:

```
setInPlane(2, "extra", "mask", "bottom", 2, false);
```


```
setInPlane(2, "extra", "mask", "bottom", 2, false);
```


Any extra layer can be set to any plane by using its index within the extra layer class. All "extra" layers attached to "top" are numbered from top to bottom, for "extra" layers attached to "bottom" the numbering continues from bottom to top. To set the extra layer with index 4 to plane 2 and activate:

```
setInPlane(2, "extra", null, "", 4, true);
```


```
setInPlane(2, "extra", null, "", 4, true);
```


Index within the specified drill class or subclass Drill layers are numbered from left to right To set the first plated drill layer of a job with 1 unplated and 2 plated layers to plane 3 and activate:

```
setInPlane(3, "drill", null, "", 2, true);
```


```
setInPlane(3, "drill", null, "", 2, true);
```


The value 2 refers to the second drill layer or

```
setInPlane (3, "drill", "plated", "", 1, true);
```


```
setInPlane (3, "drill", "plated", "", 1, true);
```


The value 1 refers to the first plated drill layer **Note:** The **`extra.order`** ucam.db key can change the order of the extra layers. Scripts relying on the index of extra layers could fail to execute correctly on other systems.

activate

activate Activation of target plane

void setInPlane (int *newPlane,*
ObjectList *layerID,*
boolean *activate*
)

レイヤにプレーンカラーを設定し アクティブ／非アクティブを切り替える

引数:

newPlane 対象となるプレーンカラー
layerID 提供レイヤID 例: `getLayerID()` 機能によって
activate 対象となるプレーンのアクティブ化

参照:

HSH_base::getLayerID(Ulayer)

HSH_base::getLayerID(Ulayer, String)

void setInPlaneByName (int *newPlane,*
String *layName,*
boolean *activate*
)

プレーンカラーにレイヤを設定し アクティブ／非アクティブを切り替える

引数:

newPlane 対象のプレーンカラー
layName レイヤ名
activate 対象プレーンのアクティブ化

void setLayerViewBottom (ObjectList *nameArray*)

Sets layer names to view in the dialog Job View in the part Bottom View

引数:

nameArray array of layer names

void setLayerViewDrill (ObjectList *nameArray*)

Sets layer names to view in the dialog Job View in the part Drill View

引数:

nameArray array of layer names

void setLayerViewMain (ObjectList *nameArray*)

Sets layer names to view in the dialog Job View in the part Main View

引数:

nameArray array of layer names

void setLayerViewTop (ObjectList *nameArray*)

Sets layer names to view in the dialog Job View in the part Top View

引数:

nameArray array of layer names

void setMode (ObjectList *oMode*)

Sets the current operation mode, units and snapmode

引数:

oMode - array of [sOption, sUnit, sSnapMode]

参照:

setMode(String, String, String)

setMode(String, String, String)

void setMode (String *sParams*)

Sets the current operation mode, units and snapmode parameter can contain all value combinations, they should be separated by comma or space

引数:

sParams - ("sel" or "all" or "selall") and/or ("mm" or "mil" or "inch") and/or ("no", "closest", "midpoint", "intersection", "closestline", "virtualintersection", "layerintersection", "grid")

void setMode (String *sOption,*
String *sUnit,*

String ***sSnapMode***
)

Sets the current operation mode, units and snapmode

引数:

sOption	- "sel" or "all" or "selall"
sUnit	- "mm" or "mil" or "inch"
sSnapMode	- snapmode : "no", "closest", "midpoint", "intersection", "closestline", "virtualintersection", "layerintersection", "grid"

Ulayer setNextLayerToPlane1 ( )

Change layer in plane 1 by default rules (class and index).

戻り値:

new layer in plane 1

void setOrigin (double *p_x,*
double *p_y*
)

パラメータを用いて原点を設定する

引数:

p_x (X座標) 原点に設定するポイント
p_y (Y座標) 原点に設定するポイント

void setOrigin (Point *p*)

パラメータを用いて原点を設定する

引数:

p 原点に設定するポイント

void setOrigin (double *pt_x,*
double *pt_y,*
boolean *bOnRefPoints*
)

基板座標を用いて アクティブレイヤのジョブ原点を設定する

引数:

pt_x	(X座標) 現行のDPFゼロ原点に対し 新しいDPFゼロ原点
pt_y	(Y座標) 現行のDPFゼロ原点に対し 新しいDPFゼロ原点
bOnRefPoints	trueの場合 移動はリファレンスポイントにも適用される

void setOrigin (Point *pt,*

```
                boolean bOnRefPoints
              )
```

基板座標を用いて アクティブレイヤのジョブ原点を設定する

引数:

pt	現行のDPFゼロ原点に対し 新しいDPFゼロ原点
bOnRefPoints	trueの場合 移動はリファレンスポイントにも適用される

```
void setOriginCenter ( double  p_x,
                       double  p_y,
                       boolean useOutline
                     )
```

パラメータを用いて 原点を中心に設定する

引数:

p_x	(X座標) 原点を中心に設定するポイント
p_y	(Y座標) 原点を中心に設定するポイント
useOutline	アウトラインを使用する

```
void setOriginCenter ( Point   p,
                       boolean useOutline
                     )
```

パラメータを用いて 原点を中心に設定する

引数:

p	原点を中心に設定するポイント
useOutline	アウトラインを使用する

```
void setOriginToCenter ( boolean bUseOutline,
                         boolean bOnRefPoints
                       )
```

アクティブレイヤのジョブ原点をジョブの中心に設定する

引数:

bUseOutline	trueの場合 アウトラインレイヤの中心が用いられる
bOnRefPoints	trueの場合 リファレンスレイヤにも移動が適用される

```
void setPlotParam ( String sKey,
                    int    iValue
                  )
```

プロットパラメータを設定する

引数:

sKey キー 使用可能値は次の通り:

- "RESOLUTION"
- "XOFF" (x オフセット)
- "YOFF" (y オフセット)
- "XSCALE" (x スケール)
- "YSCALE" (y スケール)
- "SXCEN" (x スケール中心)
- "SYCEN" (y スケール中心)
- "MXCEN" (x ミラー中心)
- "MYCEN" (y ミラー中心)
- "POSITION":
  - - MergeJob.PLOT_POSITION_COMBINE_FILL_PERCENTAGE
  - - MergeJob.PLOT_POSITION_COMBINE
  - - MergeJob.PLOT_POSITION_SINGLE
  - - MergeJob.PLOT_POSITION_SINGLE_LEFT_FIXED
  - - MergeJob.PLOT_POSITION_SINGLE_RIGHT_FIXED
- "ENLARGE_VECTORS_MINIMUMSIZE"
- "ENLARGE_VECTORS_AMOUNT"
- "ENLARGE_FLASH_MINIMUMSIZE"
- "ENLARGE_FLASH_AMOUNT"
- "ENLARGE_CONDUCTOR_SIZE" (サイズを追加する 全て削除するには0を使う)
- "ENLARGE_CONDUCTOR_AMOUNT"
- "ENLARGE_COMPLEX"
- "APPLY_ENLARGE_TO" (値はUlayer.PLOT_ENLARGE_*のいずれかでなければならない)
- "VARIABLE_TEXT_HEIGHT"
- "VARIABLE_TEXT_DIRECTION" (値はUlayer.PLOT_VARTEXT_DIRECTION_のいずれかでなければならない)

iValue integer型値

**void setPlotParam (String *sKey,*
double *dValue*
)**

プロットパラメータを設定する

引数:

sKey キー 使用可能値は次の通り:

- "XOFF" (x オフセット)
- "YOFF" (y オフセット)
- "XSCALE" (x スケール)
- "YSCALE" (y スケール)
- "SXCEN" (x スケール中心)
- "SYCEN" (y スケール中心)
- "MXCEN" (x ミラー中心)
- "MYCEN" (y ミラー中心)
- "ENLARGE_VECTORS_MINIMUMSIZE"
- "ENLARGE_VECTORS_AMOUNT"
- "ENLARGE_FLASH_MINIMUMSIZE"
- "ENLARGE_FLASH_AMOUNT"
- "ENLARGE_CONDUCTOR_SIZE" (サイズを追加するか サイズが0であれば全てを削除する)
- "ENLARGE_CONDUCTOR_AMOUNT"
- "ENLARGE_COMPLEX"
- "VARIABLE_TEXT_HEIGHT"

dValue double型値

void setPlotParam (String *sKey*,
String *sValue*
)

プロットパラメータを設定する

引数:

sKey キー 使用可能値は次の通り:

- "POLARITY" ("POSITIVE" or "NEGATIVE")
- "EMULSION" ("DOWN" or "UP")
- "MIR" ("N", "X", "Y" or "XY")
- "ROT" ("Y", "N", "F")
- "FILM"
- "VARIABLE_TEXT"

sValue String型値

void setResolution (int *resolution*)

Set resolution used by Ucam

引数:

resolution - resolution in number of internal units per mil

void setSnap (String *sMode*)

このメソッドで設定できるのは シングルスナップモードのみである

引数:

sMode - スナップモード : "no", "closest", "midpoint", "intersection", "closestline", "virtualintersection", "layerintersection", "grid"

void setSnapOnContour (boolean *on*)

turn on/off the snapping on contour

引数:

on true - turn on the snapping on contour, false - turn off the snapping on contour

void setUnit (String *unit*)

Ucamの単位を設定する

引数:

unit - "mm", "mil", "inch"のいずれか

void setYsphotechAlignmentPointType (int *region*,
int *point*,
String *type*
)

set the type of the alignment point

引数:

region	the regionnumber of the point (0 for global)
point	the alignmentpoint number (1 to 4)
type	the type of the alignment point (can be "matrix" or "default")

void setYsphotechPlotstamp (int *plotstampID*,
double *rec_xmin*,
double *rec_ymin*,
double *rec_xmax*,
double *rec_ymax*,
String *type*
)

adds or changes the plotstamp with a certain id

引数:

plotstampID	the ID of the plotstamp. (should be unique)
rec_xmin	(left boundary of rectangle) the enclosing rectangle of the plotstamp
rec_ymin	(bottom boundary of rectangle) the enclosing rectangle of the plotstamp
rec_xmax	(right boundary of rectangle) the enclosing rectangle of the plotstamp
rec_ymax	(top boundary of rectangle) the enclosing rectangle of the plotstamp
type	the type of the plotstamp

void setYsphotechPlotstamp (int *plotstampID*,
Rectangle *rec*,
String *type*
)

adds or changes the plotstamp with a certain id

引数:

plotstampID	the ID of the plotstamp. (should be unique)
rec	the enclosing rectangle of the plotstamp
type	the type of the plotstamp

void shavePads (double *dPadTraClr*,
double *dPadPadClr*,
int *iClip*,
boolean *bShaveInsideCom*
)

Shave Pads

引数:

dPadTraClr	- the pad to track clearance value. If the value is negative or zero then the clearance between pad and track is ignored.
dPadPadClr	- the pad to pad clearance value. If the value is negative or zero then the clearance between pads is ignored.
iClip	- If 1, objects are clipped. If 0, reverse objects are inserted.
bShaveInsideCom	true - shave COM regions between each other, false - shave only with foreign objects.

**void shavePads (double *dPadTraClr,*
double *dPadPadClr,*
int *iClip*
)**

Shave Pads

引数:

dPadTraClr	- the pad to track clearance value. If the value is negative or zero then the clearance between pad and track is ignored.
dPadPadClr	- the pad to pad clearance value. If the value is negative or zero then the clearance between pads is ignored.
iClip	- If 1, objects are clipped. If 0, reverse objects are inserted.

**void shavePadsOnMaskLayer (double *dPadToTrack,*
double *dPadToPad*
)**

Shave Soldermask layer

引数:

dPadToTrack	- Enter the pad to track clearance value. If the value is negative or zero then the clearance between pad and track is ignored.
dPadToPad	- Enter the pad to pad clearance value. If the value is negative or zero then the clearance between pads is ignored.

void showBlockStructure ( )

show Block Structure Information dialog

**void showMeasureValues (double *p1_x,*
double *p1_y,*
double *dx,*
double *dy,*
double *clr,*
double *rng*
)**

有効な値があれば ナンバーズダイアログに測定値を表示する そうしない場合でも 値は今後の利用のために保存される

引数:

p1_x (X座標) ポイント1
p1_y (Y座標) ポイント1
dx X方向でのポイント2のオフセット
dy Y方向でのポイント2のオフセット
clr クリアランス値
rng リング値

void showMeasureValues (Point *p1*,
double *dx*,
double *dy*,
double *clr*,
double *rng*
)

有効な値があれば ナンバーズダイアログに測定値を表示する そうしない場合でも 値は今後の利用のために保存される

引数:

p1 ポイント1
dx X方向でのポイント2のオフセット
dy Y方向でのポイント2のオフセット
clr クリアランス値
rng リング値

void showNetlistProfile ( )

ネットリストプロファイルをターミナルウィンドウに表示する

void silkOptimize (int *iReference*,
double *dClearanceToReference*,
boolean *bCompensateBumps*,
double *dBumpClearance*,
int *iMethod*,
double *dMinimumDrawLength*
)

Clips all active layers of the current job using the given parameters.

引数:

iReference	Defines the reference data used to perform the clipping against. Either REFERENCE_PLANE2, REFERENCE_COPPER, REFERENCE_COPPER_PADS, REFERENCE_MASK, REFERENCE_COPPER_FREE_OF_MASK or REFERENCE_COPPER_PADS_FREE_OF_MASK
dClearanceToReference	The clearance value to be added to the reference before clipping.
bCompensateBumps	When true, the bumpClearance value is used to clip all data that touches

	copper which is covered by mask. Only applies when the reference is REFERENCE_COPPER_FREE_OF_MASK or REFERENCE_COPPER_PADS_FREE_OF_MASK
dBumpClearance	The clearance value to be added to clip data the touches copper which is covered by mask. Only applies when compensateBumps is true and bumpClearance is bigger than clearanceToReference.
iMethod	Defines the clipping method. Either METHOD_EXACT, METHOD_REVERSE or METHOD_SPLIT_DRAWS.
dMinimumDrawLength	The minimum length of draws that can remain after splitting draws. Only applies when the METHOD_SPLIT_DRAWS is specified.

int smoothen (String *mode,*
double *dMaxDeviation*
)

参照:

smoothen(Ulayer, String, double, int, boolean, int)

smoothen(Ulayer, String, double, int, boolean, int)

引数:

mode
dMaxDeviation

int smoothen (String *mode,*
double *dMaxDeviation,*
int *iMinReplacePoints*
)

Smoothens contours in the currently active layer by approximating groups of draws and arcs by larger arcs or longer draws.

引数:

mode	Determines wheter we can use arcs, draws or both to smoothen a contour. Can be "arc" or "both", "line" is planned.
dMaxDeviation	Maximal allowed deviation between the original draw and its replacement
iMinReplacePoints	The minimal amount of points to consider for an approximation Default is 4 (which is equivalent to three tracks in default mode)

戻り値:

amount of contours that where smoothened or an error code if something went wrong

void spawn_func (String *sCommand*)

Spawn function in separate thread

引数:

sCommand - the command to spawn

int splitContour (double *dOverlapX,*

double *dOverlapY*,
double *dMinX*,
double *dMinY*
)

Function split contour. Input is current aperture from active layer in plane 1. Output is contour or contours without inners. Overlap has to be smaller or equal than Min. Also clean all smaller inners according MinX or MinY.

引数:

dOverlapX	distance overlaping between new results contoures in X coordinate, iOverlapX =< iMinX
dOverlapY	distance overlaping between new results contoures in Y coordinate, iOverlapY =< iMinY
dMinX	all chains must be bigger than this
dMinY	all chains must be bigger than this

戻り値:

Count of the split contours or 0 if no contours have to be split (no inners) or -1 if function failed

void splitContours ( )

Split countours

boolean stackupByGerAttr ( )

change order of layers by Gerber X2 layer attributes

戻り値:

true if OK, false if fault was detected

void standardizeBoxes ( )

Standardize boxes on all active layers

void testVDPathfinder (int *iStart*,
int *iEnd*
)

Test default VDPathfinder between node iStart and iEnd

引数:

iStart
iEnd

void testVDPathfinder2 (double *dStartX*,
double *dStartY*,
double *dEndX*,
double *dEndY*

)

Test default VDPathfinder between coordinates given

引数:

dStartX
dStartY
dEndX
dEndY

void toggleApertureSelections ( )

アパーチャマネジャー: カレントレイヤでアパーチャの選択を切り替える

void toggleSelections ( )

選択状態を切り替える

void toggleViewInBlocks ( )

Toggle Show In Blocks

void toggleViewMode ( )

Toggle View Mode Filled->Skeleton->Outline->Filled...

void toggleViewObjects ( )

Toggle Show Objects

void toggleViewRefPoints ( )

Toggle Show Reference Points

void toggleViewZero ( )

Toggle Show DPF Zero

void trimDraws (double *p1_x,*
double *p1_y,*
double *p2_x,*

double *p2_y*
)

実質的に2つの交差するドローをトリムする（接続する）

引数:

p1_x (X 座標) ドロー1上の **Point** (ポイント)
p1_y (Y 座標) ドロー1上の **Point** (ポイント)
p2_x (X 座標) ドロー2上の **Point** (ポイント)
p2_y (Y 座標) ドロー2上の **Point** (ポイント)

void trimDraws (Point *p1*,
Point *p2*
)

実質的に2つの交差するドローをトリムする（接続する）

引数:

p1 ドロー1上の **Point** (ポイント)
p2 ドロー2上の **Point** (ポイント)

void undo ( )

Undo an action

void undoClear ( )

Clear undo history

void unload (String *sClass*,
String *sSubClass*,
int *iLayIndex*,
boolean *bSave*
)

レイヤをアンロードする

引数:

sClass レイヤクラス: "layer"（”レイヤ層”）, "drill"（”ドリル層”）, "extra"（”特殊層”）
sSubClass レイヤサブクラス: 例） "outline", "mask", "silk", ...
iLayIndex 任意のクラスもしくは任意のサブクラスにおけるレイヤインデックス
bSave アンロードする前にレイヤを保存するにはtrueを設定

void updateProbes ( )

プローブのネット情報を更新する

void updateTestPoints ( )

テストポイントのネット情報を更新する

void updateTestPointsAndProbes ( )

テストポイントとプローブのネット情報を更新する

void updateZPosition ( )

ビルドアップのZポジションを再計算する

void utestCheck3DProbeClearance (boolean *bCheck3DProbeClearance*,
double *dClearance*
)

検査ダイアログにパラメータを設定する

引数:

bCheck3DProbeClearance	クリアランスをチェックする
dClearance	クリアランス値

void utestCreateEtmComponentLayers (boolean *bCreateCompLay*)

Sets parameter to dialog Utest for Create Component Layers

引数:

bCreateCompLay	sets to create the etm component layers

void utestDedicatedFixtures (boolean *bDedicatedFixtures*,
boolean *bFirstProbeNumber0*,
double *dFontScale*,
int *iShowProbeNumberEvery*,
boolean *bShowConnectorNumberAlways*,
boolean *bContinousNumbering*,
boolean *bContinousNumberingOnBottom*
)

Sets parameters to dialog Utest and to subdialog Dedicated Fixtures

引数:

bDedicatedFixtures	enable dialog box to display the Dedicated Fixture-screen
bFirstProbeNumber0	Indicate whether numbering should start with 0
dFontScale	Specify the fontsize you want to use for the probe-numbers.
iShowProbeNumberEvery	Specify the interval between two probe-numbers.

bShowConnectorNumberAlways	Indicate whether or not the connector-number should be included in the probe-number.
bContinousNumbering	if set true, numbering will increment the number with one for each probe.if set false, numbering will reset the probe number for every new connector.
bContinousNumberingOnBottom	Indicate whether or not numbering should continue or restart numbering on bottom.

void utestDo ( )

電気検査装置に必要な全データを生成する

void utestFiducals (boolean *bFiducals*)

検査ダイアログにパラメータを設定する

引数:

bFiducals 設定した内容でダイアログを有効にする

void utestFixtureSizeSplit (boolean *bFixtureSizeSplit,*
int *iSplitSet*
)

Sets parameters to dialog Utest and to subdialog Fixture Size Split

引数:

bFixtureSizeSplit	New layers sessions probe layers are added to the top and the bottom of the job. The new layers are of class EXTRA and subclass probe.
iSplitSet	

void utestGuidePlates (boolean *bGuidePlates*)

検査ダイアログにパラメータを設定する

引数:

bGuidePlates ガイドプレートレイヤの生成を設定

void utestKelvin4WireTest (boolean *bKelvin4WireTest,*
boolean *bUsedMidPoints,*
boolean *bTestOnBlindHoles,*
boolean *bTestOnlyThroughHoles,*
boolean *bTestAllPads,*
double *dMinDrillDia,*
double *dMaxDrillDia,*
int *iSearchDepthLimit*
)

Sets parameters to dialog Utest and to subdialog Kelvin4WireTest

引数:

bKelvin4WireTest	enable settings dialog
bUsedMidPoints	Set true if will midpoints used
bTestOnBlindHoles	Set true if will blind holes used
bTestOnlyThroughHoles	Set true if will Through Holes used
bTestAllPads	test all pads connected to an existing kelvin tested hole
dMinDrillDia	min. drill diameter value
dMaxDrillDia	max. drill diameter value
iSearchDepthLimit	max. search depth value

```
void utestKelvin4WireTest ( boolean bKelvin4WireTest,
                            boolean bUsedMidPoints,
                            boolean bTestOnBlindHoles,
                            boolean bTestOnlyThroughHoles,
                            double  dMinDrillDia,
                            double  dMaxDrillDia,
                            int     iSearchDepthLimit
                          )
```

Sets parameters to dialog Utest and to subdialog Kelvin4WireTest

引数:

bKelvin4WireTest	enable settings dialog
bUsedMidPoints	Set true if will midpoints used
bTestOnBlindHoles	Set true if will blind holes used
bTestOnlyThroughHoles	Set true if will Through Holes used
dMinDrillDia	min. drill diameter value
dMaxDrillDia	max. drill diameter value
iSearchDepthLimit	max. search depth value

```
void utestMachine ( int    iSession,
                    String sMachName,
                    String sAccesstype
                  )
```

検査ダイアログのマシンセクションに値を設定する

引数:

iSession	セッション数
sMachName	マシン名
sAccesstype	アクセスタイプ−サイド（側）

```
void utestMicroAdjustment ( boolean bMicroAdjustment,
                            int     iNbrOfTestPoints,
                            double  dTestPointDiameter,
                            double  dTestPointShiftEdge,
                            double  dTestPointShiftValue,
```

```
                    double  dTestPointPitch,
                    double  dClearanceFactor,
                    double  dCenterDiameter
                )
```

検査ダイアログおよび微調整サブダイアログにパラメータを設定する

引数:

bMicroAdjustment	微調整セットアップを有効にする
iNbrOfTestPoints	選択した全テストポイントに対し 以下のアライメントポイントのパラメータを使用する
dTestPointDiameter	パッドのアパーチャ径 アライメントポイントとして 使われる （現在の単位で）
dTestPointShiftEdge	テストポイントと最初のアライメントポイントの 中心との距離 （現在の単位で）
dTestPointShiftValue	最短側の座標軸における 各アライメントポイント中心間の距離 （現在の単位で）
dTestPointPitch	最長側の座標軸における 各アライメントポイント 中心間の距離 （現在の単位で）
dClearanceFactor	他の銅エリアとテストポイントエッジ間に必要な 最小クリアランス （最長側において）
dCenterDiameter	中心点のアパーチャ径 （現在の単位で）

```
void utestNetlist ( boolean  bNetlist,
                    boolean  bNetlistBuild,
                    boolean  bNetlistExpand
                  )
```

Sets parameters to dialog Utest to section Netlist

引数:

bNetlist	if sets true, generate the netlist for the current job.
bNetlistBuild	if sets true, generates or regenerates a netlist of the job.
bNetlistExpand	if sets true, generates the netlist for a panelized job

```
void utestOutput ( boolean  bOutput,
                   boolean  bDrillFixture,
                   String   sDrillFixture,
                   boolean  bNetlist,
                   String   sNetlist,
                   boolean  bElectTest,
                   boolean  bPinInserter,
                   boolean  bRepairAid,
                   String   sRepairAid
                 )
```

検査ダイアログおよびアウトプットサブダイアログにパラメータを設定する

引数:

bOutput	設定内容でダイアログを有効にする
bDrillFixture	ドリルジグを有効にする
sDrillFixture	ドリルジグマシンを設定する

bNetlist ネットリストアウトプットファイルを出力する
sNetlist IPC-D-356フォーマットで 電機検査ファイルを生成する
bElectTest 電機検査ファイルを生成する
bPinInserter ピンインサーター向けのアウトプットファイルを生成する 現在 PL-2000（ATGマシン）用のみに利用可能
bRepairAid リペアマシン向けの修復支援ファイルを生成する
sRepairAid RAID／EPC／Circuitestへ設定

例外:
AbortException

void utestProbeAssignment (boolean *bProbeAssignment,*
boolean *bStagger,*
String *sStagpnt,*
boolean *bStagopttrian,*
boolean *bStagoptlined,*
double *dPitch,*
double *dTolerance,*
double *dSetBack,*
boolean *bReverse,*
boolean *bAxis*
)

Sets parameters to dialog Utest and to subdialog Probe Assignment

引数:
bProbeAssignment New layers are added to the top and the bottom of the job. The new layers are of class EXTRA and subclass probe.
bStagger Moves SMD test points in the test point layers so as to avoid probe clashes.
sStagpnt This menu allows you to select 2-points, 3-points or 4-points staggering. Depending on your choice 2, 3 or 4 positions are used on the SMD for assigning the probe.
bStagopttrian determine the pattern of the 3-points or 4-points stagger.
bStagoptlined determine the pattern of the 3-points or 4-points stagger.
dPitch Defines the maximum center to center distance between two SMD pads. If the SMD pitch between two SMD pads is smaller than the maximum pitch then these SMD pads are staggered.
dTolerance Enter a value to define the tolerance for finding SMD rows and columns. If the distance between the SMD pads (=SMD pitch) varies more than the tolerance value, the SMD pads no longer belong to the same row or column (for staggering).
dSetBack Moves the staggered probes over a defined distance back towards the flash point of the SMD. This causes a border of the defined distance at the SMD edge where no probes are assigned. The Setback value is the distance between the test pin and the edge of its corresponding pad.
bReverse Uses the reverse orientation of all test points.
bAxis Checks if through holes are present. This is useful when a test point needs to be moved from top to bottom or vice versa, enabling the test point to be swapped top to bottom or vice versa.

例外:
AbortException

void utestProbeMapping (boolean *bProbeMapping*)

Sets parameter to dialog Utest

引数:

bProbeMapping sets to map the test points to the grid of the electrical test machine.

void utestTestpoints (boolean *bTestPoints,*
int *iLoop,*
boolean *bUseMasks,*
boolean *bProbeSwaping,*
boolean *bHandlePaintedPads,*
boolean *bCircuitryCheck,*
boolean *bFilterCopperAreas,*
boolean *bViaOfSMDs,*
boolean *bDrillsWithoutPad*
)

Sets parameters to dialog Utest to section Testpoints

引数:

bTestPoints	if true, calculate the test points of a job and to create one or two test point layers for the job.
iLoop	sets how to test pads in a loop
bUseMasks	if true, takes all solder mask layers into account for test point calculation.
bProbeSwaping	if true, marks all test points that can be technically tested on the other side of the pcb.
bHandlePaintedPads	if thue, handle painted pads
bCircuitryCheck	if true,enables the generation of test points according to electrical test Optimization Rules.
bFilterCopperAreas	if true, reduces the number of test points generated in large coppers by removing all unnecessary test points that are satisfactorily surrounded by copper.
bViaOfSMDs	if true, generates the test points only on the via holes of SMD's, according to the attribute settings for uVia.
bDrillsWithoutPad	if true, generates test points on drill holes without pad.

void utestTestpointsBOT (boolean *bPointsBot1,*
boolean *bPointsBot2,*
boolean *bPointsBot3,*
boolean *bPointsBot4,*
boolean *bPointsBot5,*
boolean *bPointsBot6,*
boolean *bPointsBot7*
)

Sets parameters to dialog Utest to section Testpoints - Optimization Rule bottom side

引数:

bPointsBot1	Sets a test point on a pad that is a drilled pad and that is not connected with any track (on the top or bottom layer).
bPointsBot2	Sets a test point on a pad that is a drilled pad and that is connected with only one track on

the test point side and not connected with any track on the opposite layer.

bPointsBot3 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side and connected with only one track on the opposite layer.

bPointsBot4 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side or the opposite layer, but is connected to one inner layer.

bPointsBot5 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side and connected with more than one track on the opposite layer.

bPointsBot6 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side or the opposite layer, but is connected to two or more inner layers.

bPointsBot7 Sets a test point on a pad that is not a drilled pad and that is connected with more than one track.

void utestTestpointsTOP (boolean *bPointsTop1,*
boolean *bPointsTop2,*
boolean *bPointsTop3,*
boolean *bPointsTop4,*
boolean *bPointsTop5,*
boolean *bPointsTop6,*
boolean *bPointsTop7*
)

Sets parameters to dialog Utest to section Testpoints - Optimization Rule top side

引数:

bPointsTop1 Sets a test point on a pad that is a drilled pad and that is not connected with any track (on the top or bottom layer).

bPointsTop2 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side and connected with only one track on the opposite layer.

bPointsTop3 Sets a test point on a pad that is a drilled pad and that is connected with only one track on the test point side and not connected with any track on the opposite layer.

bPointsTop4 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side or the opposite layer, but is connected to one inner layer.

bPointsTop5 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side and connected with more than one track on the opposite layer.

bPointsTop6 Sets a test point on a pad that is a drilled pad and that is not connected with any track on the test point side or the opposite layer, but is connected to two or more inner layers.

bPointsTop7 Sets a test point on a pad that is not a drilled pad and that is connected with more than one track.

void validateInvalidArcs ( )

無効なアークを有効にする

void viewAmbiguous ( )

曖昧な輪郭に対するエラーを表示する

void viewGrid (boolean *bVisible,*
double *ptOrigin_x,*

```
                double  ptOrigin_y,
                double  dXStep,
                double  dYStep,
                boolean bCross
              )
```

与えられたパラメータによってグリッドを表示又は非表示する

引数:

bVisible `true` は グリッドを表示; `false` は グリッドを非表示にする
ptOrigin_x (X 座標) グリッド原点
ptOrigin_y (Y 座標) グリッド原点
dXStep グリッドクロス又はグリッドライン間Ｘ距離
dYStep グリッドクロス又はグリッドライン間距離
bCross `true` は グリッドクロス; `false` は グリッドラインを表示する

```
void viewGrid ( boolean bVisible,
                Point   ptOrigin,
                double  dXStep,
                double  dYStep,
                boolean bCross
              )
```

与えられたパラメータによってグリッドを表示又は非表示する

引数:

bVisible `true` は グリッドを表示; `false` は グリッドを非表示にする
ptOrigin グリッド原点
dXStep グリッドクロス又はグリッドライン間Ｘ距離
dYStep グリッドクロス又はグリッドライン間距離
bCross `true` は グリッドクロス; `false` は グリッドラインを表示する

```
void viewGrid ( boolean bVisible,
                Point   ptOrigin,
                Point   ptStep,
                boolean bCross
              )
```

与えられたパラメータによってグリッドを表示又は非表示する

引数:

bVisible `true` は グリッドを表示; `false` は グリッドを非表示にする
ptOrigin グリッド原点
ptStep the グリッドクロス又はグリッドライン間距離をＸとＹ座標で表示したポイント
bCross `true` は グリッドクロス; `false` は グリッドラインを表示する

```
void viewGrid ( boolean bVisible )
```

グリッドを表示又は非表示する

引数:

bVisible `true` は グリッドを表示; `false` は グリッドを非表示にする

void viewGrid ( )

グリッドを表示する

void viewGuide ( )

ビューガイドを表示する

void viewHistory ( )

前回の閲覧履歴を見る

void viewInBlocks (boolean *trueFalse*)

ブロックの内外切り替え

引数:

trueFalse トグルの値

void viewMessages ( )

メッセージを表示する

void viewMode (String *sMode*)

モードを表示する

引数:

sMode filled（塗り潰し）, skeleton（スケルトン）, outline（アウトライン）のいずれか

void viewModeFilled ( )

塗り潰しモードを表示する

void viewModeOutline ( )

アウトラインモードを表示する

void viewModeSkeleton ( )

スケルトンモードを表示する

void viewNumbers ( )

ナンバーズを表示する

void viewObjects (boolean *trueFalse*)

オブジェクトのトグル表示を設定／リセットする

引数:

trueFalse トグルの値

void viewPan (double *p1_x*,
double *p1_y*,
double *p2_x*,
double *p2_y*
)

基板座標を用いて メインウィンドウをパンする

引数:

p1_x (X 座標) 移動オフセットの始点
p1_y (Y 座標) 移動オフセットの始点
p2_x (X 座標) 移動オフセットの終点
p2_y (Y 座標) 移動オフセットの終点

void viewPan (Point *p1*,
Point *p2*
)

基板座標を用いて メインウィンドウをパンする

引数:

p1 移動オフセットの始点
p2 移動オフセットの終点

void viewRefPoints (boolean *trueFalse*)

リファレンスポイントのトグル表示を設定／リセットする

引数:

trueFalse トグルの値

void viewRepaint ( )

塗り直しのスクリーンビューを表示する

void viewWarning (String *message*)

警告メッセージを表示する

引数:

message 表示したいメッセージテキスト

void viewZero (boolean *trueFalse*)

DPFゼロののトグル表示を設定／リセットする

引数:

trueFalse トグルの値

void viewZoom (String *sZoom*)

所定のズームレベルを用いて メインウィンドウをズームする

引数:

sZoom "total"（全画面をウィンドウ内に表示）, "in2x"（2倍にズームイン）, or "out2x"（1/2倍にズームアウト）

void viewZoomIn ( )

2倍のズームレベルで メインウィンドウをズームイン（拡大）する

void viewZoomOut ( )

1/2倍のズームレベルで メインウィンドウをズームアウト（縮小）する

void viewZoomSelections ( )

選択エリア上にメインウィンドウをズームする

void viewZoomTotal ( )

メインウィンドウを全画面表示する

void viewZoomWindow ( )

The command starts drag rubberband for definition of zoom window. If the window is defined the view is zoomed according to the window.

void viewZoomWindow (double *rect_xmin,*
double *rect_ymin,*
double *rect_xmax,*
double *rect_ymax,*
boolean *bScreenCenter*
)

Zoom main window using board coordinates

引数:

rect_xmin	(left boundary of rectangle) zoom area
rect_ymin	(bottom boundary of rectangle) zoom area
rect_xmax	(right boundary of rectangle) zoom area
rect_ymax	(top boundary of rectangle) zoom area
bScreenCenter	- true means Screen Center, false means "Center" coordinates from the Numbers Dialog

void viewZoomWindow (Rectangle *rect,*
boolean *bScreenCenter*
)

Zoom main window using board coordinates

引数:

rect	zoom area
bScreenCenter	- true means Screen Center, false means "Center" coordinates from the Numbers Dialog

void viewZoomWindow (double *rect_xmin,*
double *rect_ymin,*
double *rect_xmax,*
double *rect_ymax*
)

基板座標を用いて メインウィンドウをズームする

引数:

rect_xmin	(長方形の左境界) ズームエリア
rect_ymin	(長方形の下境界) ズームエリア
rect_xmax	(長方形の右境界) ズームエリア
rect_ymax	(長方形の上境界) ズームエリア

void viewZoomWindow (Rectangle *rect*)

基板座標を用いて メインウィンドウをズームする

引数:

rect ズームエリア

**void xmlAdd (String *sDataName,*
String *sElementName,*
String *sContent,*
ObjectList *attrArray*
)**

Adds simple element to the data section.

引数:

sDataName	A data section name
sElementName	Simple element name
sContent	A value (content) of the element
attrArray	Object List with pairs (attribute name, value) Eg. [{[{"name","name_1"}], [{"company","name_2"}]}]

**void xmlAdd (String *sDataName,*
String *sElementName,*
String *sContent*
)**

Adds simple element to the data section.

引数:

sDataName	A data section name
sElementName	Simple element name
sContent	A value (content) of the element

**void xmlAdd (String *sElementName,*
String *sContent*
)**

Adds Element to the root of the document. It is a simple element in XML document

引数:

sElementName	Simple element name
sContent	A value (content) of the element

**void xmlAddData (String *sParentDataName,*
String *sDataName*
)**

Adds data section to the given parent data section

引数:

sParentDataName A parent data section name
sDataName A data section name to be added to the data section

void xmlAddData (String *sDataName*)

Adds data section to the root of the XML document

引数:

sDataName A data section name to be added to the XML document

void xmlCreateData (String *sDataName*)

Creates data section with a given name. Each section must be created first, otherwise it is not possible to ad any data(elements) to it

引数:

sDataName New data section name

void xmlDocument (String *rootName*)

Creates new XML document with given root name

引数:

rootName A root element (section) name

void xmlSave (String *sDestFilePath*)

Saves current XML document. It is usually the last command in a script.

引数:

sDestFilePath destination full file path (with file name and extension) where the current XML file will be stored.

例外:

FileNotFoundException When XML file couldn't be created
IOException When I/O issue turns up during the XML file output

void YachiyoAOI_clearOutput (String *name*)

Delete generated data for the given job This simply removes the whole "name" sub-folder from the output folder Ex: YachiyoAOI_clearOutput("job1"); Ex: YachiyoAOI_clearOutput("job1");

引数:

name

```
void YachiyoAOI_defineArea ( int    grpIndex,
                             int    areaIndex,
                             double pos_x,
                             double pos_y
                           )
```

(Re-)define area of a group The group (given by grpIndex) must already exist. Areas must be defined in increasing order (and each defined group shall finally have at least one area). Position specifies the final location of the area, i.e. where the rectangle defined in the corresponding group is placed If the area with the given areaIndex already exists, it is repositioned accordingly Ex: YachiyoAOI_defineArea(1, 1, **Point**(24.5, 14.1)); Ex: YachiyoAOI_defineArea(1, 1, **Point**(24.5, 14.1));

引数:

grpIndex
areaIndex
pos_x (X coordinate)
pos_y (Y coordinate)

```
void YachiyoAOI_defineArea ( int   grpIndex,
                             int   areaIndex,
                             Point pos
                           )
```

(Re-)define area of a group The group (given by grpIndex) must already exist. Areas must be defined in increasing order (and each defined group shall finally have at least one area). Position specifies the final location of the area, i.e. where the rectangle defined in the corresponding group is placed If the area with the given areaIndex already exists, it is repositioned accordingly Ex: YachiyoAOI_defineArea(1, 1, **Point**(24.5, 14.1)); Ex: YachiyoAOI_defineArea(1, 1, **Point**(24.5, 14.1));

引数:

grpIndex
areaIndex
pos

```
void YachiyoAOI_defineGroup ( int    index,
                              double area_xmin,
                              double area_ymin,
                              double area_xmax,
                              double area_ymax,
                              String types
                            )
```

Start definition of a group The group will cover the provided area and allow masks of given types The types string encode both allowed types of masks (DRC, D2D, and/or CRF) and also their parameters (paraX) and optional integer value (used for passing values of additional toggles for D2D and CRF types) Group must be defined in increasing order, i.e. group 2 after group 1 etc. This call can be also used for re-definition of existing group, in that case the given group is re-set to newly provided parameters and all its masks and areas are deleted Ex: YachiyoAOI_defineGroup(1, **Rectangle**(0.0, 0.0, 115.5, 118.0), "DRC=para1,D2D=para2:1,CRF=para1:0"); Ex: YachiyoAOI_defineGroup(1, **Rectangle**(0.0, 0.0, 115.5, 118.0), "DRC=para1,D2D=para2:1,CRF=para1:0"); YachiyoAOI_defineGroup(2, **Rectangle**(0.0, 0.0, 10.3, 20.3), "D2D=para3:0,CRF=para2:1"); YachiyoAOI_defineGroup(2, **Rectangle**(0.0, 0.0, 10.3, 20.3), "D2D=para3:0,CRF=para2:1");

引数:

index
area_xmin (left boundary of rectangle)
area_ymin (bottom boundary of rectangle)
area_xmax (right boundary of rectangle)
area_ymax (top boundary of rectangle)
types

void YachiyoAOI_defineGroup (int *index*,
Rectangle *area*,
String *types*
)

Start definition of a group The group will cover the provided area and allow masks of given types The types string encode both allowed types of masks (DRC, D2D, and/or CRF) and also their parameters (paraX) and optional integer value (used for passing values of additional toggles for D2D and CRF types) Group must be defined in increasing order, i.e. group 2 after group 1 etc. This call can be also used for re-definition of existing group, in that case the given group is re-set to newly provided parameters and all its masks and areas are deleted Ex: YachiyoAOI_defineGroup(1, **Rectangle**(0.0, 0.0, 115.5, 118.0), "DRC=para1,D2D=para2:1,CRF=para1:0"); Ex: YachiyoAOI_defineGroup(1, **Rectangle**(0.0, 0.0, 115.5, 118.0), "DRC=para1,D2D=para2:1,CRF=para1:0"); YachiyoAOI_defineGroup(2, **Rectangle**(0.0, 0.0, 10.3, 20.3), "D2D=para3:0,CRF=para2:1"); YachiyoAOI_defineGroup(2, **Rectangle**(0.0, 0.0, 10.3, 20.3), "D2D=para3:0,CRF=para2:1");

引数:

index
area
types

void YachiyoAOI_defineMask (int *grpIndex*,
int *maskIndex*,
double *area_xmin*,
double *area_ymin*,
double *area_xmax*,
double *area_ymax*,
String *type*
)

(Re-)define mask of a group The group (given by grpIndex) must already exist. Masks must be defined in increasing order **Rectangle** of the mask must be inside rectange of the specified group and also its type must be allowed by the group. Mask can have multiple types, their names are then concatenated by '+' If the mask with the given maskIndex already exists, it is replaced according to the new definition Ex: YachiyoAOI_defineMask(1, 3, **Rectangle**(36.54, 56.23, 43.21, 63.93), "CRF+DRC"); Ex: YachiyoAOI_defineMask(1, 3, **Rectangle**(36.54, 56.23, 43.21, 63.93), "CRF+DRC");

引数:

grpIndex
maskIndex
area_xmin (left boundary of rectangle)
area_ymin (bottom boundary of rectangle)
area_xmax (right boundary of rectangle)
area_ymax (top boundary of rectangle)

type

void YachiyoAOI_defineMask (int *grpIndex,*
int *maskIndex,*
Rectangle *area,*
String *type*
)

(Re-)define mask of a group The group (given by grpIndex) must already exist. Masks must be defined in increasing order **Rectangle** of the mask must be inside rectange of the specified group and also its type must be allowed by the group. Mask can have multiple types, their names are then concatenated by '+' If the mask with the given maskIndex already exists, it is replaced according to the new definition Ex: YachiyoAOI_defineMask(1, 3, **Rectangle**(36.54, 56.23, 43.21, 63.93), "CRF+DRC"); Ex: YachiyoAOI_defineMask(1, 3, **Rectangle**(36.54, 56.23, 43.21, 63.93), "CRF+DRC");

引数:

grpIndex
maskIndex
area
type

void YachiyoAOI_finish ( )

Finish Yachiyo AOI, clean-up related data structures Should be called at the end of processing. Note that generated data are not removed (use **YachiyoAOI_clearOutput()** for that) Ex: **YachiyoAOI_finish()**; Ex: **YachiyoAOI_finish()**;

boolean YachiyoAOI_generateCalibration (String *name,*
double *pos_x,*
double *pos_y,*
double *startRes,*
double *endRes,*
double *step,*
String *options*
)

Generate set of images for calibration The images will be generated into the folder OUTPUTFOLDER/name, centered at given pos and with sizes given by RESOLUTION_BMP_SIZE. The first image will have resolution startRes, the next one startRes + step, and so on until the resolution endRes is met. All resolutions are in um/px and they should be multiplies of 10nm (i.e. 0.01), otherwise rounding is applied. For list of available options see **YachiyoAOI_generateOutput()** above Ex: YachiyoAOI_generateCalibration("job1-calibration1", **Point**(11.65, 12.65), 1.5, 1.8, 0.01, ""); Ex: YachiyoAOI_generateCalibration("job1-calibration1", **Point**(11.65, 12.65), 1.5, 1.8, 0.01, ""); YachiyoAOI_generateCalibration("job1-calibration2", **Point**(11.65, 12.65), 1.0, 20.0, 1.0, "MIRRORX,MIRRORY,SCALEY=0.998,REV,RLE"); YachiyoAOI_generateCalibration("job1-calibration2", **Point**(11.65, 12.65), 1.0, 20.0, 1.0, "MIRRORX,MIRRORY,SCALEY=0.998,REV,RLE");

引数:

name
pos_x (X coordinate)
pos_y (Y coordinate)
startRes

endRes
step
options

boolean YachiyoAOI_generateCalibration (String *name,*
Point *pos,*
double *startRes,*
double *endRes,*
double *step,*
String *options*
)

Generate set of images for calibration The images will be generated into the folder OUTPUTFOLDER/name, centered at given pos and with sizes given by RESOLUTION_BMP_SIZE. The first image will have resolution startRes, the next one startRes + step, and so on until the resolution endRes is met. All resolutions are in um/px and they should be multiplies of 10nm (i.e. 0.01), otherwise rounding is applied. For list of available options see **YachiyoAOI_generateOutput()** above Ex: YachiyoAOI_generateCalibration("job1-calibration1", **Point**(11.65, 12.65), 1.5, 1.8, 0.01, ""); Ex: YachiyoAOI_generateCalibration("job1-calibration1", **Point**(11.65, 12.65), 1.5, 1.8, 0.01, ""); YachiyoAOI_generateCalibration("job1-calibration2", **Point**(11.65, 12.65), 1.0, 20.0, 1.0, "MIRRORX,MIRRORY,SCALEY=0.998,REV,RLE"); YachiyoAOI_generateCalibration("job1-calibration2", **Point**(11.65, 12.65), 1.0, 20.0, 1.0, "MIRRORX,MIRRORY,SCALEY=0.998,REV,RLE");

引数:

name
pos
startRes
endRes
step
options

boolean YachiyoAOI_generateOutput (String *name,*
String *lens,*
String *fixture,*
String *options*
)

Generate complete data (set of inf files and bitmaps) for AOI as specified by Yachiyo All data (i.e. yachiyo.inf file, cadrefpointX.bmp bitmaps for reference points and group-related data (yachiyo_rip.inf and many XXXXYYYY.raw tiles in sub-folders 1, 2, etc.)) are generated into the folder OUTPUTFOLDER/name based on provided lens and fixture parameters (strings choosing appropriate LENSx and FIXTUREx parameters from settings.ini) Available options (case sensitive, multiple options can be specified in any order, but they must be separated by comma and with no spaces): REV ... generate reverse output (i.e. copper as white and background as black) MIRRORX ... mirror the layer horizontally MIRRORY ... mirror the layer vertically SCALEX=float ... scale (distort) the layer in horizontal direction by the given factor (which should be near 1.0) SCALEY=float ... same for vertical direction BMP ... generate images as uncompressed 8-bit BMPs (https: RLE ... generate images as BMPs with RLE8 compression (https: RAW ... generate images in Yachiyo specific format (currently head-less stream of data with RLE8 encoding) PBM ... generate images as "ASCII" PBMs (type P1, http: Ex: YachiyoAOI_generateOutput("job1", "x10.0", "9inch", ""); Ex: YachiyoAOI_generateOutput("job1", "x10.0", "9inch", ""); YachiyoAOI_generateOutput("job2", "x5.0", "5inch", "SCALEX=1.01,MIRRORY,REV,RLE"); YachiyoAOI_generateOutput("job2", "x5.0", "5inch", "SCALEX=1.01,MIRRORY,REV,RLE");

引数:

name
lens
fixture
options

String YachiyoAOI_getStrings (String *kind*)

Return all possible values for the given GUI component (typically labels for items of comboboxes) The values are returned in one string, separated by "|" Kind must be one of the following strings: DRC, D2D, CRF, LENS, RESOLUTION, FIXTURE Ex: YachiyoAOI_getStrings("FIXTURE"); -> "5inch|9inch|13inch|125mm" Ex: YachiyoAOI_getStrings("FIXTURE"); -> "5inch|9inch|13inch|125mm"

引数:

kind

boolean YachiyoAOI_init (String *iniFile*)

Initialize Yachiyo AOI, parse provided settings.ini Must be called first before any other YachiyoAOI command is used Ex: YachiyoAOI_init("H:/UcamBugs/Yachiyo/WORK/YAOI_params/settings.ini"); Ex: YachiyoAOI_init("H:/UcamBugs/Yachiyo/WORK/YAOI_params/settings.ini");

引数:

iniFile

void YachiyoAOI_reset ( )

Delete generated data for the given job This simply removes the whole "name" sub-folder from the output folder Ex: YachiyoAOI_clearOutput("job1");

void YachiyoAOI_setRefPoint (int *index,*
double *pos_x,*
double *pos_y*
)

Set reference point to given pos Points can be set in random order This call can be also used for re-position of any already defined point Ex: YachiyoAOI_setRefPoint(1, **Point**(563.35, 12.65)); Ex: YachiyoAOI_setRefPoint(1, **Point**(563.35, 12.65));

引数:

index
pos_x (X coordinate)
pos_y (Y coordinate)

void YachiyoAOI_setRefPoint (int *index,*
Point *pos*
)

Set reference point to given pos Points can be set in random order This call can be also used for re-position of

any already defined point Ex: YachiyoAOI_setRefPoint(1, **Point**(563.35, 12.65)); Ex: YachiyoAOI_setRefPoint(1, **Point**(563.35, 12.65));

引数:

index

pos

变数

int FILE_ATTRIBUTES = 2

File info field containing file/dir attributes

int FILE_MODIFICATION_DATE = 4

File info field containing last modification time

int FILE_NAME = 5

File info field containing file/dir name

int FILE_PARENT = 1

File info field containing file/dir parent directory

int FILE_SIZE = 3

File info field containing file/dir size

int FILE_TYPE = 0

File info field containing type information

int LAYER_ACTIVITY = 5

Layer info field containing layer's activity

int LAYER_APERTURES = 6

Layer info field containing layer's activity

int LAYER_ATTACH = 3

Layer info field containing layer's attachment

int LAYER_CLASS = 1

Layer info field containing layer's class

int LAYER_INDEX = 4

Layer info field containing layer's index

int LAYER_NAME = 0

Layer info field containing layer's name

int LAYER_SUBCLASS = 2

Layer info field containing layer's subclass

非推奨一覧

メンバ **compareNet**

use **compareNet(boolean, boolean, boolean, boolean, boolean, boolean, boolean, double, boolean, double, String, boolean, boolean)**

use **compareNet(boolean, boolean, boolean, boolean, boolean, boolean, boolean, double, boolean, double, String, boolean, boolean)**

メンバ **copperCount**

Copper count without mask layer usage

引数:

sOpt Set to "job", "layer", or "inner"

メンバ **deselectObjectAttribute**

deselectObjectAttribute 任意の属性名および属性値を持つオブジェクトを非選択にする。

メンバ **deselectObjectAttribute**

deselectObjectAttribute 任意の属性名をもつオブジェクトをカレントジョブで非選択にする。

メンバ **insertPolydrawRect**

カレントアパーチャを用いて複数のドローからなる長方形を挿入する。

引数:

p1	長方形の左下ポイント。
p2	長方形の右上ポイント。
bRectCW	長方形をCW（右回り）にする場合、trueを設定。
bSel	長方形を選択状態にす場合、trueを設定。

メンバ **selectByApertureShape**

Select or deselect all objects of the specified object type.

引数:

selectMode	Either + (select) or - (deselect)
apertureShapes	Comma separated list of cir, don, rec, squ, box, oct, com, the, con, tex, blo

メンバ **selectByAttributeName**

指定した属性名の全オブジェクトを選択／非選択する。

引数:

selectMode	+ (選択する)もしくは - (非選択する)のいずれかを設定。
sName	- 属性名。

メンバ **selectByAttributeValue**

指定の属性値の全オブジェクトを選択／非選択する。

引数:

selectMode	+ (選択する)もしくは - (非選択する)のいずれかを設定。
sName	- 属性名。
sValue	- 属性値。

メンバ **selectByObjectType**

Select or deselect all objects of the specified object type.

引数:

selectMode Either + (select) or - (deselect)
objectTypes Comma separated list of f (flash), d (draw), a (arc) or v (vector text)

メンバ **selectObjectAttribute**

selectObjectAttribute は、カレントジョブにおいて、任意の属性名および属性値の属性が設定された オブジェクトを選択する。

引数:

sAttrName オブジェクト属性名。
sAttrValue オブジェクト属性値。

メンバ **selectObjectAttribute**

selectObjectAttribute は、カレントジョブにおいて、任意の属性名の属性が設定された オブジェクトを選択する。

引数:

sAttrName オブジェクト属性名。

メンバ **setAttributeOnObject**

This function is GUI ONLY, replaced by addObjectAttribute(sAttrName, sAttrValue)

参照:

addObjectAttribute(String sAttrName, String sAttrValue) Insert attribute on objects

引数:

attrName Name of attribute
attrValue Value of attribute
