PCB-Investigator Interface update V7

New features:

IAutomation

The easylogix\PCB-Investigator of user in application data folder.

```
static string PCBIDataPath
```

Set the internal working directory for temp-data or unzip jobs etc.
(Default is user temp path: Path.GetTempPath())
Value can only be used after creating an IPCBIWindow instance.

```
static bool SetPCBIWorkingDirectory(string WorkingDirectory)
```

ICMPLayer/IODBLayer:

Gets count of Elements on this layer

```
override int GetElementCount()
```

Sets an overlay image for this layer or removes the image (bmp=null)

```
override void SetOverlayImage(Bitmap bmp, System.Drawing.Drawing2D.Matrix matrix,
                              Dictionary<int, Dictionary<int, double>> pixelValues,
                              string overlayLabel, string overlayUnit)
```

Set the alpha value (transparency) for the Overlay image

```
override void SetOverlayAlpha(byte alpha)
```

Returns the value and description of the overlay at a certain world coordinate

```
override bool GetOverlayPixelValue(double worldX, double worldY, out double val,
                                   out string label, out string unit)
```

ICMPObject:

GetComponentAtributes returns a Dictionary of strings with all component attributes.

```
public Dictionary<string,string> GetComponentAttributeDictionary()
```

Calculate the distance between two objects.

```
public PCBI.Automation.IODBObject.DistanceResultClass DistanceTo(IODBObject Object2,
                                   bool includePins)
```

IEdge:

Calculates the Distance between two IEdges

```
static double Distance(IEdge e1, IEdge e2, out PointD p1, out PointD p2)
```

Returns true if the two IEdges intersect

```
static bool GetIntersectingPoints(IEdge e1, IEdge e2, ref PointD p1, ref PointD p2)
```

IFilter:

Adds a special tool definition.

int AddToolDefinitionSpecial(Automation.IODBLayer Layer, string cadConformName, ISurfaceSpecificsD OutlinePad, float diameter, int ToolNr = 0, bool RenameAndCreateIfAlreadyExists = true)

static int AddToolDefinitionSpecial(Automation.IODBLayer Layer, IPCBIWindow pcbi, string cadConformName, List<IODBObject> ObjectList, double offsetX, double offsetY)

Add a definition of oval symbol.

static int AddToolDefinitionOval(Automation.IODBLayer Layer, float diameter, float ovalHeight, float ovalWidth)

Add a fool definiton for butterfly to the layer.

static int AddToolDefinitionButterfly(Automation.IODBLayer Layer, float diameterOrSize, bool rounded)

Create a Rectangle tool with option of corners.

static int AddToolDefinitionRectCorners(Automation.IODBLayer Layer, double diameter, double height, double width, double cornerRadius, bool rounded)

int AddToolDefinitionRect(Automation.IODBLayer Layer, double height, double width, double radiusCorners, bool cornerRightUp, bool cornerLeftUp, bool cornerLeftDown, bool cornerRightDown)

Convertes any Pad to a list of Surfaces without adding it to the layer

List<ISurfaceSpecificsD> CreateSurfacesFromPad(IPadSpecificsD spec, IODBLayer layer)

Get the symbol information for the givcen ShapeIndex and Layer

ToolDefinition GetSymbolByShapeIndex(int shapeIndex, IODBLayer ParentLayer)

Copy existing Symbol of parent layer.

ToolDefinition CopySymbol(int ShapeIndexOfOriginal, IODBLayer ParentLayer)

Insert a list of elements in the existing element list of the ParentLayer on position IndexInLayerElementList.

bool InsertElementsToLayer(int IndexInLayerElementList, IODBLayer ParentLayer, List<IODBObject> InsertElements, PCBI.MathUtils.PointD Offset , bool MirrorX = false, bool MirrorY = false, double Rotation = 0)

IMath:

Calculate length on arc depending on radius and angle of arc.

static double DistanceOnArc(double angle, double radius)

Calculates a point on a Line with a certain distance from the start point.

static PointD GetPointOnLine(PointD from, PointD to, double dist)

IMatrix:

Calculate the distance between layers, depending on the material height of each layer.

double GetDistanceBetweenLayers(int IndexFirstLayer, int IndexSecondLayer)

Return the first silk screen layer after the signal block, there can be a later silk screen layer e.g. if there is a copy at the end of the stackup.

string GetBotSilkScreenLayerName()

Create document layer from compoonent layer top or bottom.

bool MakeDocumentFromComponents(bool Top)

INet:

Gets the area in world coords (mils) where this net is used over all layers
PCBI.MathUtils.RectangleD GetVisualNetBounds()
Net number of this net.
int GetNetNumber()
Select this net in PCB-Investigator.
void SelectNet()

new Class INote!

Everything in this class is new.

IODBLayer:

Removes a list of objects (no Undo)
void RemoveObjects(List<IODBObject> ObjectsToRemove)
Check the shape index, if it is a special symbol all elements are returned.
List<IODBObject> GetSpecialSymbolElements(int shapeIndex)
Polygonizes all overlapping Objects to one polygon
List<PCBI.MathUtils.IPolyClass> PolygonizeNets(bool onlySelected = false,
IPCBIWindow.ProgressChanged progChanged = null)
Hatch an object with the special pattern.
List<IODBObject> HatchObject(PCBI.MathUtils.IPolyClass outlineElements,
PatternHatch PatternType, double DistaceStep = 10,
double ElementWidth = 5, bool OnlyInside = true)
Reset all free texts on this layer.
void ResetFreeText()

IODBObject:

Returns a Dictionary with FeatureAttributeEnum or user attribute name and value as string.
Dictionary<string, string> GetAttributesDictionaryStringKeys(bool KeysToUpper = true)
Move the object.
void SetOffset(PCBI.MathUtils.PointD offset, bool AddUndo = true)
Update internal fields and values (e.g. arc start angle).
void UpdateInternal()

IPCBIWindow:

Enable last mouse tool (it remembers last disable call).
void MouseToolEnabled()
Disable active mouse tool and change to default cursor (without selection, zoom...).
void MouseToolDisabled()
Handle mouse wheel of layerlist and main form.
event System.Windows.Forms.MouseEventHandler PCBIFormMouseWheel
Occurs when user click in the graphic pane, the location has to be transformed in job coordinates (use ClientToWorld).
event MouseEventHandler PCBIMouseUpInGraphicPane;

The event handler for saving matrix.
event EventHandler PCBIJobMatrixSaved;
Show Component Properties from Component.
void OpenPropertyOfComponent(ICMPObject Component)
ODB ++ v8.1 defines zones (e.g. for specific heights), you can show them with this property.
bool ShowZones

IPicuteLayer:

Fit the image to the given rectangle.
This scales the matrix in such a way it have to be to let the image exactly fit in the rectangle.
void FitToRectangle(RectangleF RectangleToFitIn)

IPin:

Returns the bounds of the pin with double values.
PCBI.MathUtils.RectangleD GetBoundsD(ICMPObject Parent)
Get the pin color.
Color GetPinColor(ICMPObject Parent)
New overrides of Equal NotEqual and opterators.

Interface which defines Methods for Objects with Bounds
public interface **IBoundable**

IPolygon:

Returns the Index of the given edge in the list.
int GetEdgeIndex(IEdge edge)
Returns number of Edges in this Polygon
int GetEdgeCount()
Create a new list of Edges from this polygon with y values in the raster(s) crossing the give Rectangle.
List<IEdge> GetRasterEdges(RectangleD rect)
Scales the Polygon
void Scale(double scaleX, double scaleY)
Returns a list of Intersecting Points of the Poly with the given Edge. Also the Edge-Indices which are intersecting are returned in indexList
List<PointD> GetIntersectingPoints(IEdge e, ref List<int> indexList)
Sortes the Sub-Polygons in single Isles (including their holes)
List<IPolyClass> SplitInIsleAndHoles()
Union combines a list of polygones to a new polygon.
static IPolyClass Union(List<IPolyClass> polys)
Returns true if Polygon is Clockwise
bool IsClockWise()

IObjectSpecificsD

Move Positive to here, Clone and Tag.

IComponentSprecificsD

Rotation of component.

```
double Rotation
```

IPolygonSpecificsD :

Check the point in the polygon.

```
bool IsInPolygon(PointD InPoint)
```

ISurfaceSpecificsD:

Offset for all subelements of this surface specifics.

```
void SetOffset(PointD Offset)
```

Rotate the surface including all subelements

```
void Rotate(double degrees)
```

Mirror the surface including all subelements in X

```
void MirrorX()
```

Mirror the surface including all subelements in Y

```
void MirrorY()
```

Scale the surface including all subelements

```
void Scale(double scale)
```

Create a List of IPolygonSpecificsD, who represent all isle and holes of the surface.

```
List<IPolygonSpecificsD> GetPolygonOutlineD()
```

Returns true if the point is inside of the surface (1 Isle and 0 or more Holes)

```
bool IsPointInsideSurface(PointD point)
```

IStep:

List of iNotes attached to the Step or single Layers

```
List<INote> Notes
```

Get the step header datum point.
This value is only used for step and repeat positions (has no effect for standard root steps).

```
PCBI.MathUtils.PointD GetDatumStepHDR()
```

Gets a bitmap for the chosen rectangle area, expands the choosen area if it is too small.

```
Bitmap GetBitmap(List<ILayer> Layers, RectangleF DetailRectangle, int Width, int Height,
                 bool DrawPCBOutline, bool FillBoardOutline, bool ShowComponentDetails,
                 bool IgnoreSelection, bool DrawOnlySelection,
                 out RectangleF DrawnRectangle)
```

Get Height of layer from layer attributes, this is depending on the matrix layer type. For dielectric layers is a other attribute relevant (.layer_dielectric) and for copper layers it use .copper_weight.
All values are in mils.

```
double GetHeightOfLayer(string Layername)
```

Set height of layer in mils.

void SetHeightOfLayer(string Layername, double HeightOfLayer)

void SetHeightOfLayer(string Layername, MatrixLayerType typeOfLayer, double HeightOfLayer)

Calculate distance between two layers, it use all layers between exclude the start and end layer.
It's important to have start layer before end layer (do use matrix order in correct sequence).

double GetDistanceLayerToLayer(string LayerNameStart, string LayerNameEnd)