Application of Java in UG Based CAD Tool Development

Xuefeng Zhang
GE Global Research
Niskayuna, NY

Outline

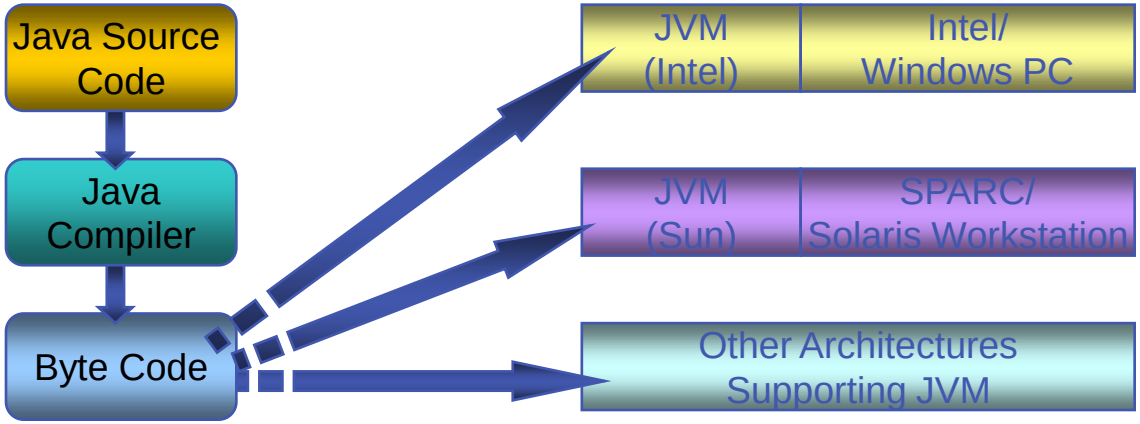
- Overview of Java technology
- UG based CAD tool development using Java
  - Ufunc and common API
  - Development tools
  - Execute Java application in UG
  - Debug Java application in UG
- Practical experience
  - What if Ufunc API/common API has problem
  - Load dynamic library
  - Java GUI development
  - Java thread and UG thread
  - Other practical experience
- Case study: X-section Matching tool.

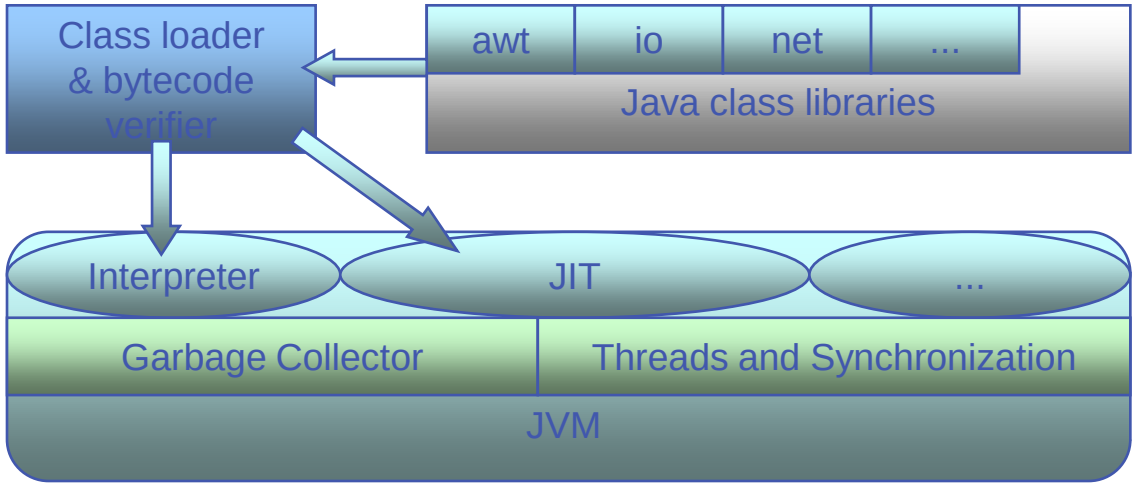
Java Technology

- Java Programming Language

- Java Platform
  - Java 2 platform Standard Edition (J2SE)
  - Java 2 platform Enterprise Edition (J2EE)
  - Java 2 platform Micro Edition (J2ME)
  - Java Card technology

Java Runtime Environment





http://system.cmlab.csie.ntu.edu.tw/seminar/javaintr.ppt

Why Java

- Simple
- Object-Oriented
- Write once, run anywhere
- Small byte code
- High performance
- Distributed computing
- Multi-thread
- Garbage collected
- Feature rich
- Popular
- .....
- **Supported by UG!**

UG Based CAD Tool Development Using Java

APIs

- UG/Open for Java since NX3
- Ufunc
  - Wrapped fro Java
  - Mature
  - Well documented
  - Well supported
- Common API
  - Object-Oriented
  - Journal
  - Not complete
  - Not well documented
  - Future trend

Ufunc for Java

- Package nxopen.uf

uf
uf_abort
uf_assem
uf_attr
uf_bound
uf_brep
uf_cam
uf_cfi
uf_cgm
uf_clear
uf_clone
uf_csys
uf_curve
uf_dbc_mld

UF	Contains wrappers for all 'UF' functions and data types.
UFAbort	Contains wrappers for all 'UFAbort' functions and data types.
UFAssem	Contains wrappers for all 'UFAssem' functions and data types.
UFAttr	Contains wrappers for all 'UFAttr' functions and data types.
UFBound	Contains wrappers for all 'UFBound' functions and data types.
UFBrep	Contains wrappers for all 'UFBrep' functions and data types.
UFCam	Contains wrappers for all 'UFCam' functions and data types.
UFCambnd	Contains wrappers for all 'UFCambnd' functions and data types.
UFCamgeom	Contains wrappers for all 'UFCamgeom' functions and data types.
UFCamPref	Contains wrappers for all 'UFCamPref' functions and data types.
UFCamPrepro	Contains wrappers for all 'UFCamPrepro' functions and data types.
UFCamtext	Contains wrappers for all 'UFCamtext' functions and data types.
UFCfi	Contains wrappers for all 'UFCfi' functions and data types.
UFCgm	Contains wrappers for all 'UFCgm' functions and data types.
UFClear	Contains wrappers for all 'UFClear' functions and data types.
UFClone	Contains wrappers for all 'UFClone' functions and data types.

open

```
UFPart.OpenData open(String partName)
                throws NXException,
                       RemoteException
```

Refer to UF_PART_open for documentation

Example Code

```
public static void main(String[] args) throws Exception
{
    uf_session = (UFSession)SessionFactory.get("UFSession");

    UFPart uf_part = uf_session.part() ;
    UFObj uf_obj = uf_session.obj() ;

    UFPart.OpenData open_data = uf_part.open("test.prt") ;
    if(open_data.errorStatus.failed)
        throw new Exception("Failed to open the part file") ;

    Tag prt = open_data.part ;

    String prt_name = uf_part.askPartName(prt) ;

    //do something
    //

    //save the file
    uf_part.save() ;

    //close file
    uf_part.close(prt, 0, 1) ;
```

Get access to ufunc APIs

Get access to APIs defined in uf_part.h

Same as **UF_PART_open**

Return value usually saved in a data structure

No need to call UF_free etc

Common API for Java

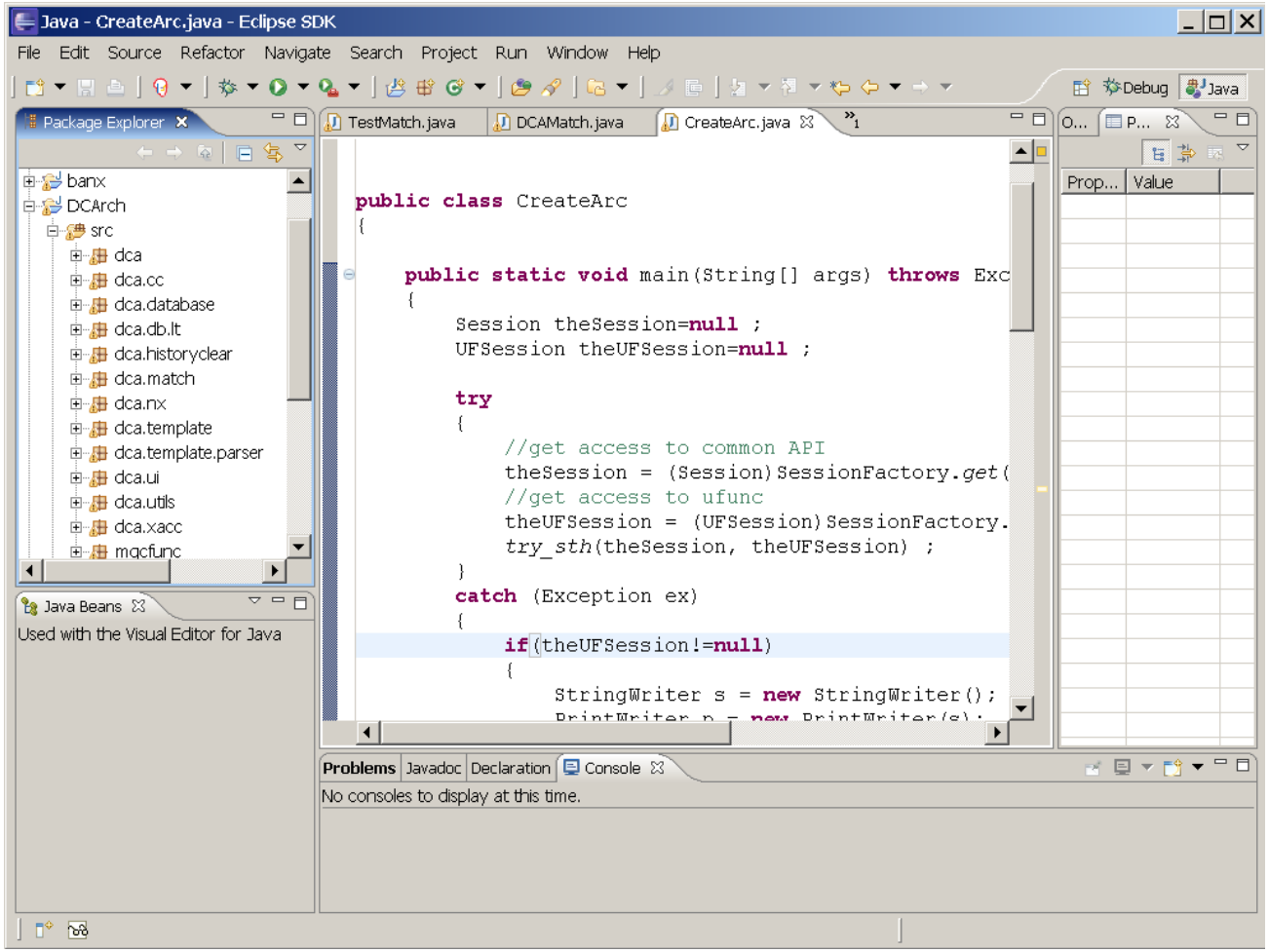
Packages	
nxopen	Provides classes and interfaces for NX Open Common API.
nxopen.annotations	Provides classes and interfaces relating to annoatations.
nxopen.assemblies	Provides classes and interfaces relating to assemblies.
nxopen.cae	Provides classes and interfaces relating to CAE.
nxopen.cam	Provides classes and interfaces relating to CAM.
nxopen.die	Provides classes and interfaces relating to die design.
nxopen.drawings	Provides classes and interfaces relating to drawings.
nxopen.features	Provides classes and interfaces relating to features.
nxopen.features.sheetmetal	Provides classes and interfaces relating to sheetmetal features.
nxopen.fields	Provides classes and interfaces relating to fields.
nxopen.geometricutilities	Provides classes and interfaces relating to geometric utilities.
nxopen.layer	Provides classes and interfaces relating to layers.
nxopen.motion	Provides classes and interfaces relating to motion.
nxopen.options	Provides classes and interfaces relating to options.
nxopen.pdm	Provides classes and interfaces relating to PDM.
nxopen.positioning	Provides classes and interfaces relating to positioning.
nxopen.preferences	Provides classes and interfaces relating to preferences.
nxopen.routing	Provides classes and interfaces relating to routing.

Example Code

```
// NX 4.0.4.2
// Journal created by 200012060 on Thu May 22 17:30:54 2008 Eastern Daylight Time
//
import nxopen.*;

public class journal
{
  public static void main(String [] args) throws NXException, java.rmi.RemoteException
  {
    Session theSession = (Session)SessionFactory.get("Session");
    Part workPart = theSession.parts().work();
    Part displayPart = theSession.parts().display();
    // ----------------------------------------------
    //   Menu: Insert->Design Feature->Extrude...
    // ----------------------------------------------
    int markId1;
    markId1 = theSession.setUndoMark(nxopen.Session.MarkVisibility.VISIBLE, "Start");

    Section section1;
    section1 = workPart.sections().createSection(0.02413, 0.0254, 0.5);

    nxopen.features.Feature nullFeatures_Feature = null;
    nxopen.features.ExtrudeBuilder extrudeBuilder1;
    extrudeBuilder1 = workPart.features().createExtrudeBuilder(nullFeatures_Feature);

    extrudeBuilder1.setSection(section1);

    nxopen.geometricutilities.Limits limits1;
    limits1 = extrudeBuilder1.limits();

    nxopen.geometricutilities.FeatureOptions featureOptions1;
    featureOptions1 = extrudeBuilder1.featureOptions();

    nxopen.geometricutilities.LinearLimits linearLimits1 = ((nxopen.geometricutilities.LinearLimits)limits1);
    nxopen.geometricutilities.Extend extend1;
    extend1 = linearLimits1.startExtend();

    nxopen.geometricutilities.Extend extend2;
    extend2 = linearLimits1.endExtend();

    nxopen.geometricutilities.BooleanOperation booleanOperation1;
    booleanOperation1 = extrudeBuilder1.booleanOperation();

    Direction direction1;
```

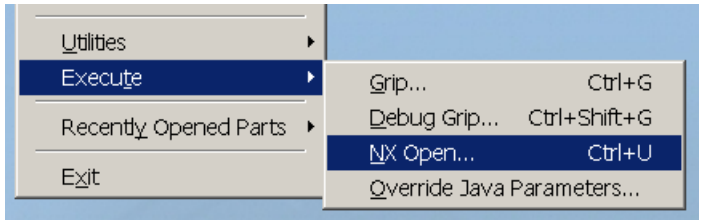
Development Tools

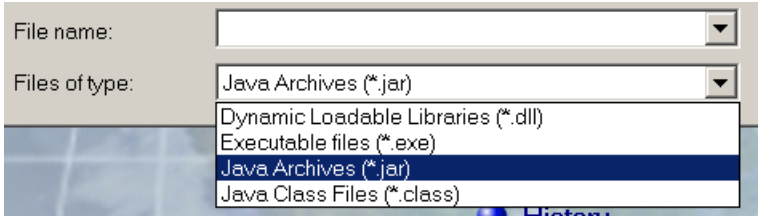
- Compiler: JDK
- IDE: Eclipse, NetBeans, etc



Execute Java Application in UG

- Execute an External UG application written in Java
  - Include UG system jars in CLASSPATH
- Execute an internal UG application written in Java





- Associate toolbar button to jar/class file

```
...
BUTTON DCA_cc
LABEL &DCA_cc
ACTIONS dca.jar
...
```

XXX.men

```
...
BUTTON DCA_cc
LABEL &DCA_cc
BITMAP cc.bmp
...
```

XXX.tbr

Execute Java Application in UG

- Execute Java application with a different version of JRE
  - JRE coming with NX 4.0.4.2 is version 1.4.2
  - Some Java features are not available in early version
  - IR 5818247: The NXOpen jar files were inadvertently built with the wrong version of Java. Refer to development as PR 1611708

```
set UGII_JVM_LIBRARY_DIR=C:\Program Files\java\jre1.5.0_08\bin
```

Debug External UG Application with IDE

- Debug it like a normal Java application

Debug - TestMatch.java - Eclipse SDK

File Edit Source Refactor Navigate Search Project Run Window Help

Debug | Java

Debug

TestMatch [Java Application]
dca.match.TestMatch at localhost:3989
Thread [AWT-Shutdown] (Running)
Thread [AWT-Windows] (Running)
Thread [DestroyJavaVM] (Running)
Thread [AWT-EventQueue-0] (Suspended (breakpoint at line 272
TestMatch.test_move_constraints0() line: 272

Variables | Breakpoints

Name	Value
prt	Tag (id=20)

FeatureName.java | TestMatch.java

```
public static void test_move_constraints0() throws Excepti
{
    Tag prt = DCANxUtils.uf_session().part().open("h:\\avi
    //Tag from = DCANxUtils.find_named_object("ex1") ;
    //Tag to = DCANxUtils.find_named_object("ex2") ;
    Tag from = DCANxUtils.find_named_object("ax0") ;
    Tag to = DCANxUtils.find_named_object("ex00") ;
    Tag gc_sket = DCANxUtils.find_named_object("SKETCH_000
    Tag gc_sket2 = DCANxUtils.find_named_object("SKETCH_00
    DCAMatch.move_constraints(from, to, new Tag[] {gc_sket
}
```

Outline

- dca.match
- import declarations
- TestMatch

Console | Tasks

TestMatch [Java Application] C:\Program Files\Java\jre1.5.0_08\bin\javaw.exe (May 22, 2008 10:49:03 AM)

```
DCA [Thu May 22 10:49:23 EDT 2008]
System dir : h:\aviation\dcahome
```

Writable | Smart Insert | 272 : 1

Debug Internal UG Application with IDE

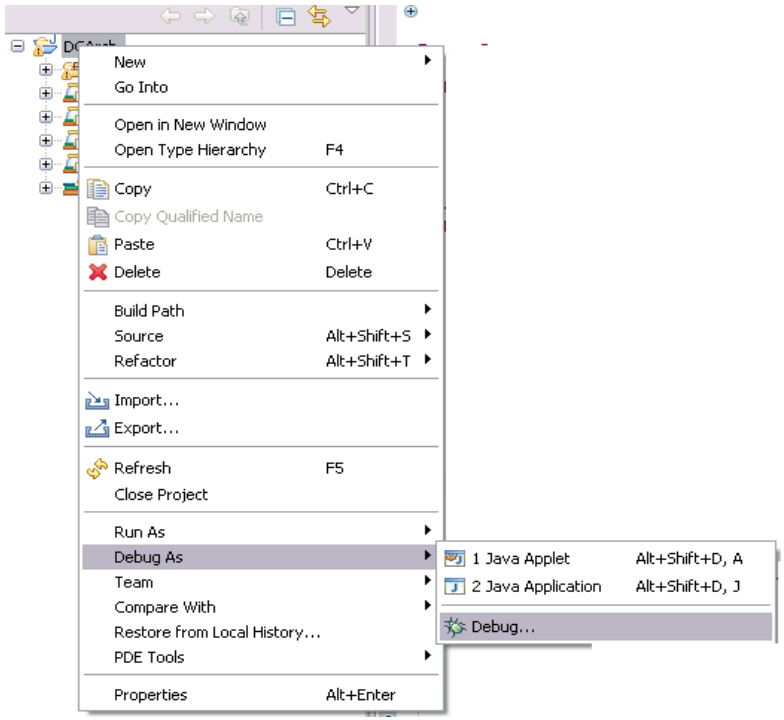
- Can not load UG from Eclipse debugger
- Debug it as a remote Java application in Eclipse
- Step 1: Launch UG with following env variable

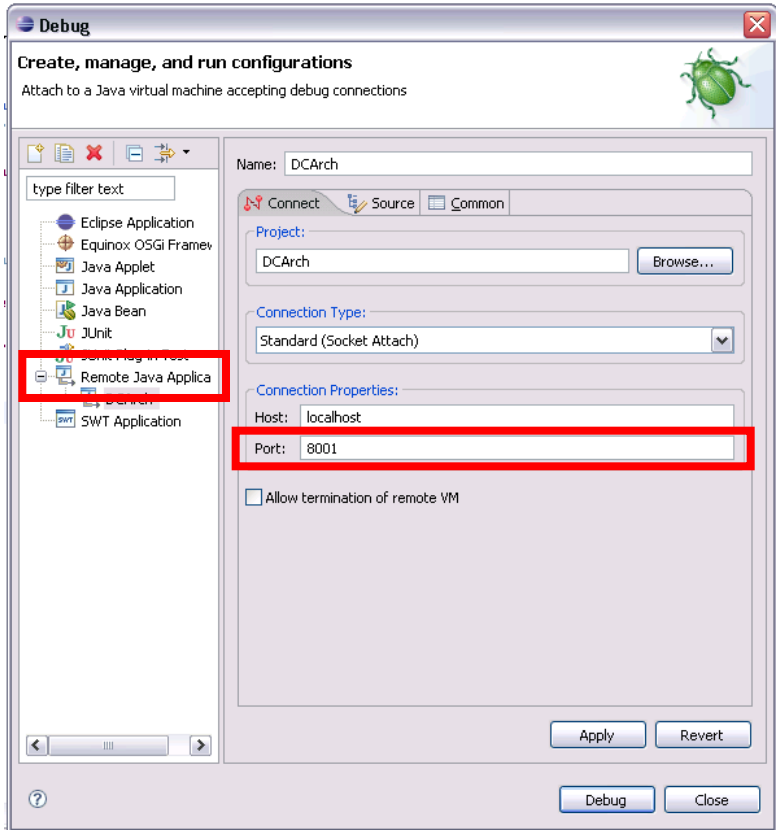
```
set UGII_JVM_OPTIONS=-Xdebug -Xrunjdwp:transport=dt_socket,
address=8001,server=y,suspend=n
```

- 8001 is a port number. You can use any port number that is not in use.
- You can use dt_shmem instead of dt_socket if it is available for your JRE, but dt_shmem is not available on all platforms. For more information
- More details: Java Platform Debugger Architecture

Debug Internal UG Application with IDE

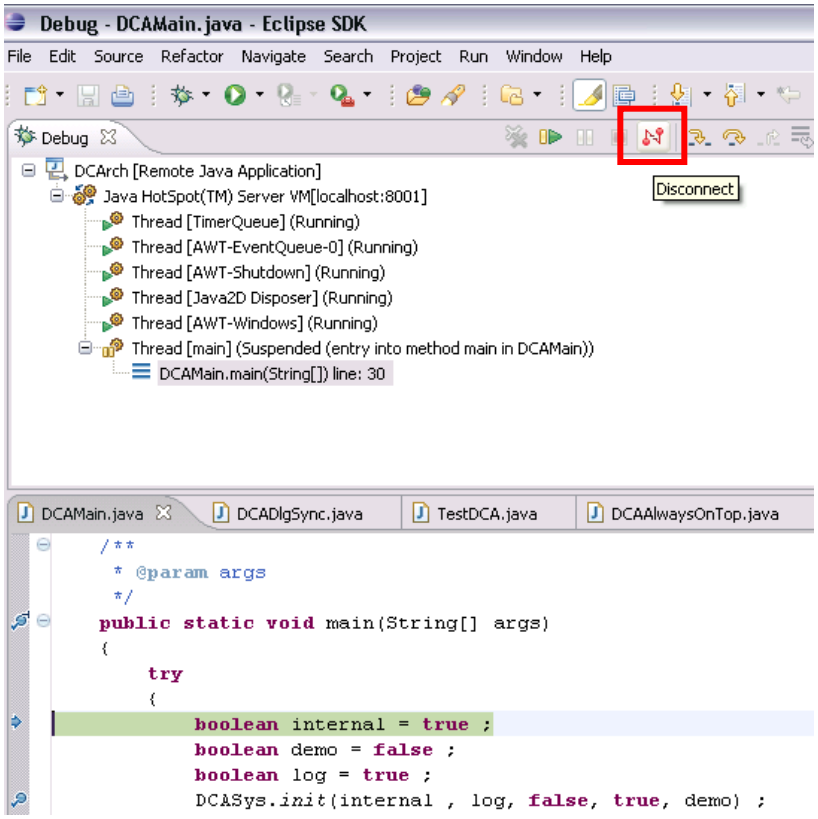
- Step 2: Execute any simple Java application (such as HelloWorld) from UG to activate JVM
- Step 3: Attach Eclipse Debugger to UG





Debug Internal UG Application with IDE

- Step 4. Load Java application in UG



you can stop the debugging by using disconnect and the program will run in normal mode.

Practical Experience

What If Ufunc Doesn't Work

- Possible reasons
  - Feature missing
  - Bug in original ufunc
  - Bug in wrapping ufunc for Java
- First choice: Use Common API to replace it
- Switch between Ufunc and Common APIs
  - Ufunc use Tag as object identifier
  - Common API use Object
- Get object tag: use method tag().

```
Part myPart= theSession.parts().newDisplay("MyPart",Part.Units.MILLIMETERS);
theUFSession.part().askPartName(myPart.tag());
```

- Get object from tag: use TaggedObjectManager

```
Arc arc = (Arc)theSession.taggedObjectManager().get(arc2Tag);
```

Examples

- Example 1:UF_UI_select_with_class_dialog
  - Find a replacement from common API
  - Multiple selectObject and selectObjects methods in nxopen.Selection
- Example 2:

```
/**
 * Used to replace ufunc askDatumAxisParms
 * Take object tag instead of feature tag as input
 */
public static UFModlFeatures.AskDatumAxisParmsData askDatumAxisParms(Tag objId)throws
NXException, RemoteException
{
UFModlFeatures.AskDatumAxisParmsData ret = new UFModlFeatures.AskDatumAxisParmsData() ;
DatumAxis da = (DatumAxis)DCANxUtils.nx_session().taggedObjectManager().get(objId) ;
Vector3d v = da.direction() ;
Point3d p = da.origin() ;
ret.normal = new double[]{v.x, v.y, v.z} ;
ret.origin = new double[]{p.x, p.y, p.z} ;
return ret ;
}
```

What if Common API Doesn't Work Neither

- If C version of Ufunc works properly, wrap ufunc with JNI (Java Native Interface) for Java by yourself

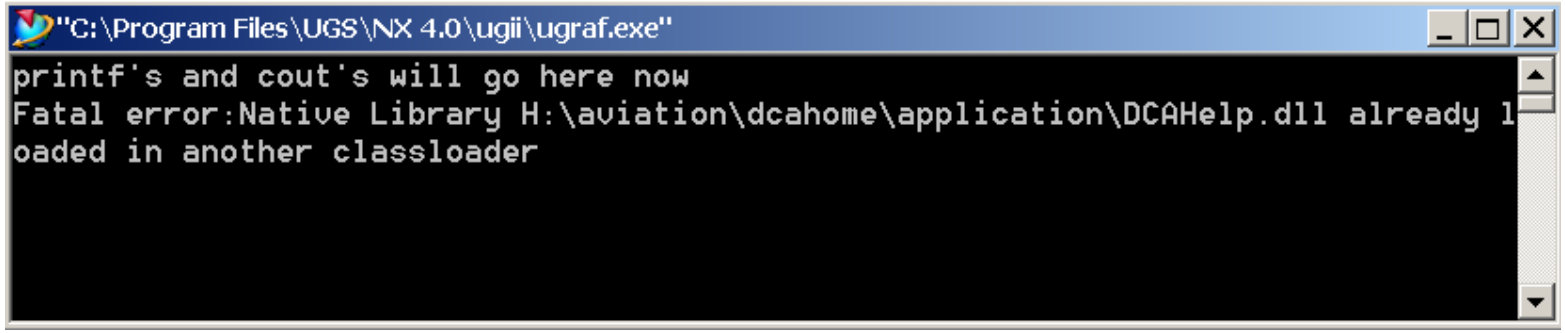
```
static{
System.loadLibrary("DCAHelp") ;
}

private static native int
AskOccsOfEntity0(int obj, int[] occs);

/**
 * This is a replacement of function
 * askOccsOfEntity in class UFAssem.
 * The one comes with UG does not work
 * properly.
*/

public static Tag[] askOccsOfEntity(Tag obj)

{
int[] tags = new int[128] ;
int n = AskOccsOfEntity0(obj.value, tags) ;
if(n <= 0)
    return null ;
Tag[] ret = new Tag[n] ;
for(int i=0 ; i<ret.length ; i++)
    ret[i] = new Tag(tags[i]) ;
return ret ;

}
```

```
JNIEXPORT jint JNICALL
Java_dca_nx_DCANxUtils_AskOccsOfEntity0 (
                    JNIEnv *env,
                    jclass obj,
                    jint obj_tag,
                    jintArray occs)
{
          tag_t *ret ;
          int n ;
          n =
UF_ASSEM_ask_occs_of_entity(obj_tag, &ret) ;

          (*env)->SetIntArrayRegion(env,
occs, 0, n, ret) ;
          UF_free(ret) ;
          return n;
}
```

What if None of Above Works

- File IR
- Find a workaround

Load Dynamic Linked Library



- Reasons
  - BaseSession.LibraryUnloadOption.IMMEDIATELY
  - DLL get loaded by URLClassLoader, which ug people internally subclassed and used for loading the classes and DLL for JVM.
  - If all the classes uses the native method of that DLL, they will also try to load the DLL using The new instance of URLClassLoader .
  - JVM imposes the restriction that DLL can be loaded only once in any class loader.

Load Dynamic Linked Library

- Solution 1: avoid using option: BaseSession.LibraryUnloadOption.IMMEDIATELY
  - Classes won't get re-loaded in second execution

```
//public static final int getUnloadOption()
//{
//    return BaseSession.LibraryUnloadOption.IMMEDIATELY ;
//}
```

- Solution 2: Use UGII_CLASSPATH_PRELOAD
  - Classes/jars listed in UGII_CLASSPATH_PRELOAD will be loaded while JVM is created, and will not be re-loaded

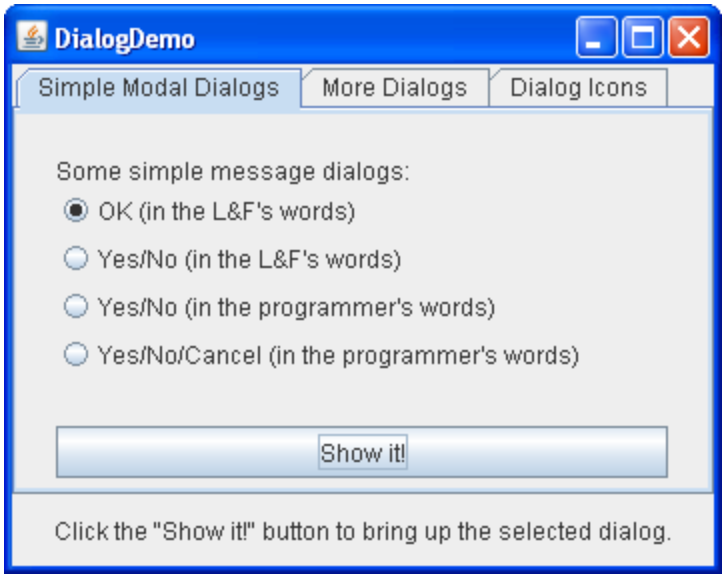
```
set UGII_CLASSPATH_PRELOAD = CLASS_JAR_LOADING_dll
```

GUI Development Using Java

- UIStyler not for Java
- Java GUI is much more powerful and flexible than UIStyler
  - Controller
  - Layout control
  - Other

- APIs available
  - AWT (Abstract Window Toolkit)
  - Swing
  - Other: SWT (Standard Widget Toolkit), etc

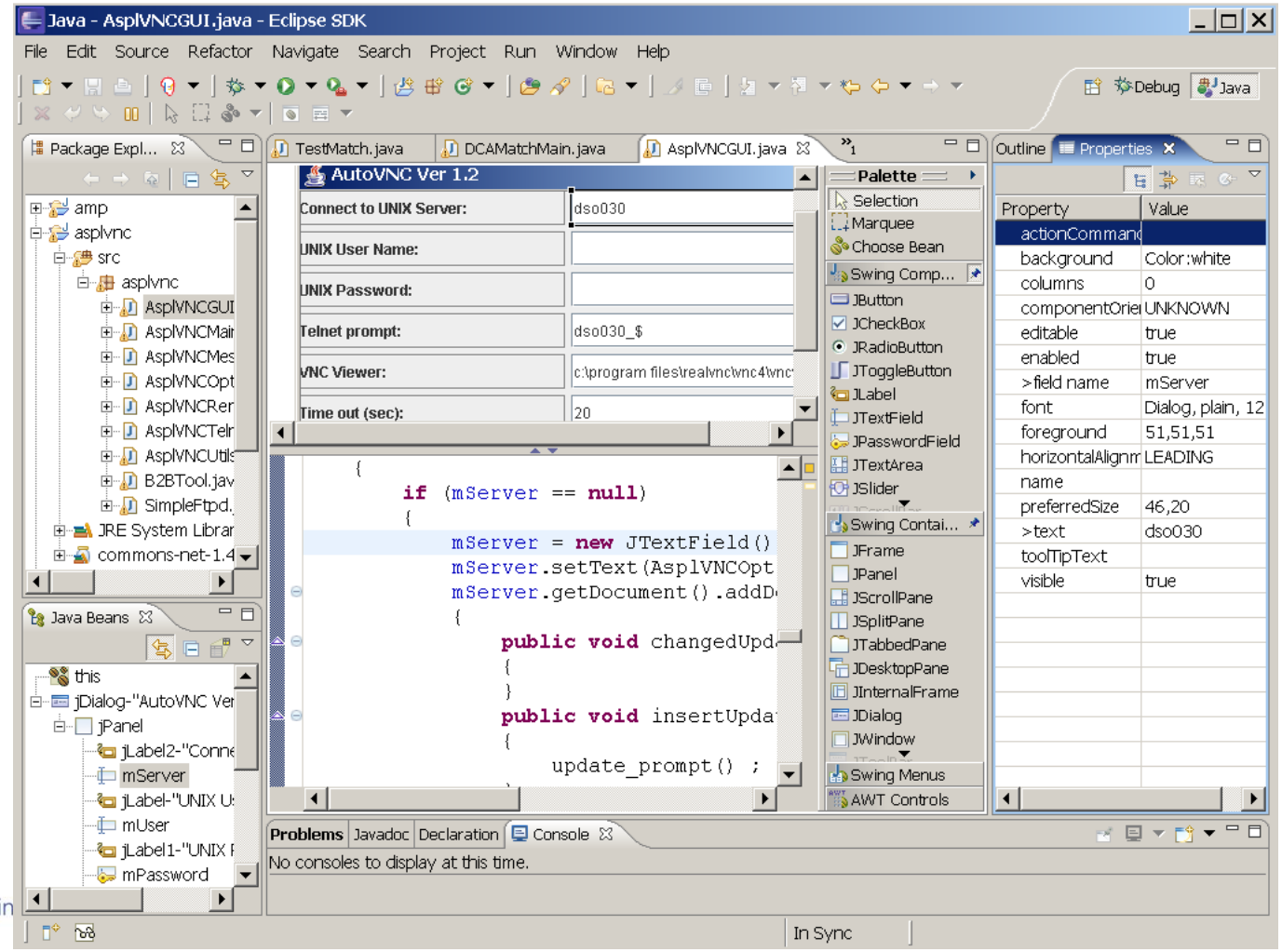
Example




```
public DialogDemo(JFrame frame) {
    super(new BorderLayout());
    this.frame = frame;
    customDialog = new CustomDialog(frame, "geisel", this);
    customDialog.pack();

    //Create the components.
    JPanel frequentPanel = createSimpleDialogBox();
    JPanel featurePanel = createFeatureDialogBox();
    JPanel iconPanel = createIconDialogBox();
    label = new JLabel("Click the \"Show it!\" button to bring up the selected
dialog.",JLabel.CENTER);

    //Lay them out.
    Border padding = BorderFactory.createEmptyBorder(20,20,5,20);
    frequentPanel.setBorder(padding);
    featurePanel.setBorder(padding);
    iconPanel.setBorder(padding);

    JTabbedPane tabbedPane = new JTabbedPane();
    tabbedPane.addTab("Simple Modal Dialogs", null,
            frequentPanel,
            simpleDialogDesc); //tooltip text
    tabbedPane.addTab("More Dialogs", null,
            featurePanel,
            moreDialogDesc); //tooltip text
    tabbedPane.addTab("Dialog Icons", null,
            iconPanel,
            iconDesc); //tooltip text
```

Tools Enable Visual Design of GUI

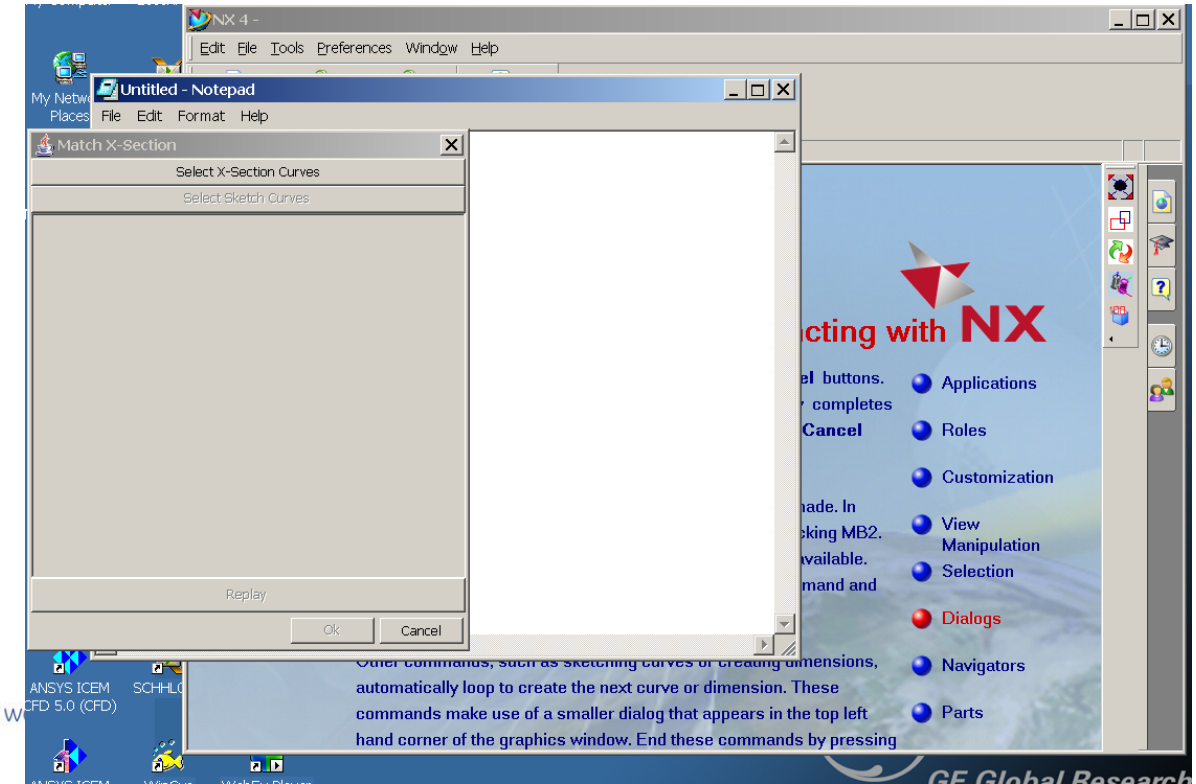
- Available in many Java IDEs (Eclipse, NetBeans, JBuilder etc)
- Example: Visual Editor for Eclipse



Known Issues

- Not native to either OS or UG
- Not integrated with UG GUI. Has different behaviors.
  - Dialog is not owned by UG window
  - Dialog does not stay on top of UG window

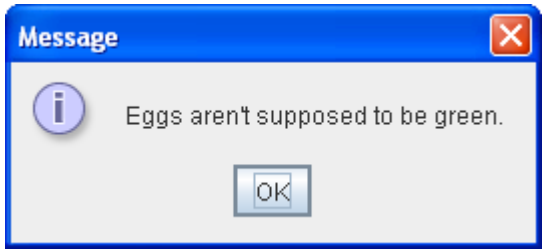
    Solution: Method *SetAlwaysOnTop* since JRE 5.0
  - Dialog does not maximize/minimize when UG window maximizes/minimizes

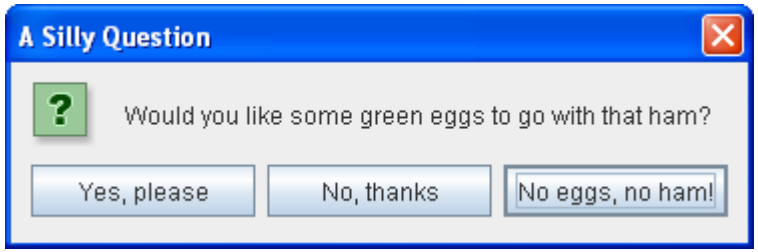


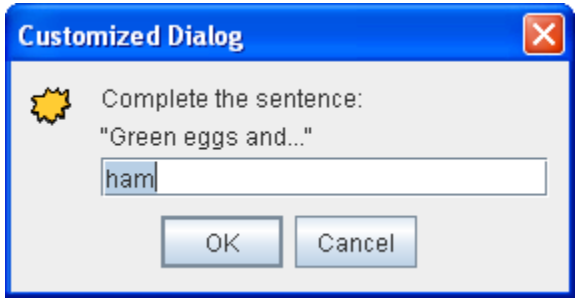
Using Standard SWING Dialogs

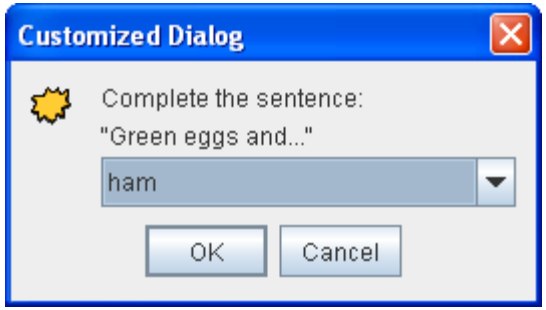
- Some simple UG dialog functions are not wrapped for Java
- Use Class JOptionPane.

```
JOptionPane.showMessageDialog(null, "Eggs are't supposed to be green", "Message",
JOptionPane.INFORMATION_MESSAGE);
```









From http://java.sun.com/docs/books/tutorial/uiswing/components/dialog.html

Using Standard SWING Dialogs

- By default, the dialog created by JOptionPane is not always on top
- Invoke JOptionPane dialogs in a different way.

```
public static void showMessageDialog(Component parentComponent, Object message,
String title, int messageType)
{
    JOptionPane pane = new JOptionPane(message, messageType);
    JDialog dialog = pane.createDialog(parentComponent, title);
    dialog.setAlwaysOnTop(true) ;
    dialog.setModal(true) ;
    dialog.setVisible(true);
}
```

Java Thread and UG Thread

- JVM created by UG shares the same thread of UG
- Java code should be in a separate thread, if
  - It takes long time
  - It has its own message loop

```
public static void main(String[] args)
{
          //code .....
          SwingUtilities.invokeLater(new Runnable()
          {
                    public void run()
                    {
                              try
                              {
                                //your code here
                              }
                              catch (Exception e)
                              { //exception processing
                              }
                              finally
                              { //clearnup
                              }
                    }
          });
}
```

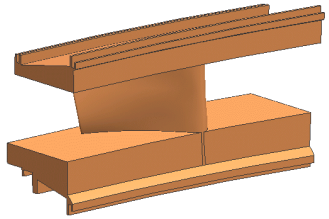
Case Study: X-Section Matching Tool

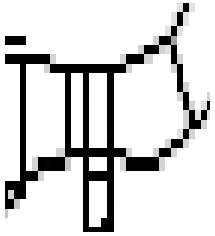
X-Section Matching Tool

Par1 = 1.0

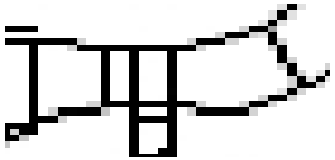
Par2 = 230

...





Parameters 3D models Cross sections



Par1 = ?

Par2 = ?

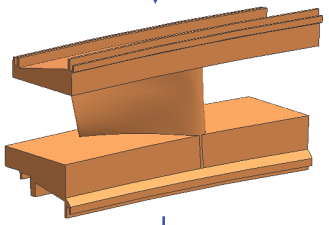
...

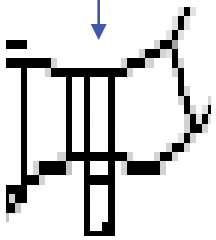
Find the parameter values to match giving x-section

Par1 = 1.0

Par2 = 230

...





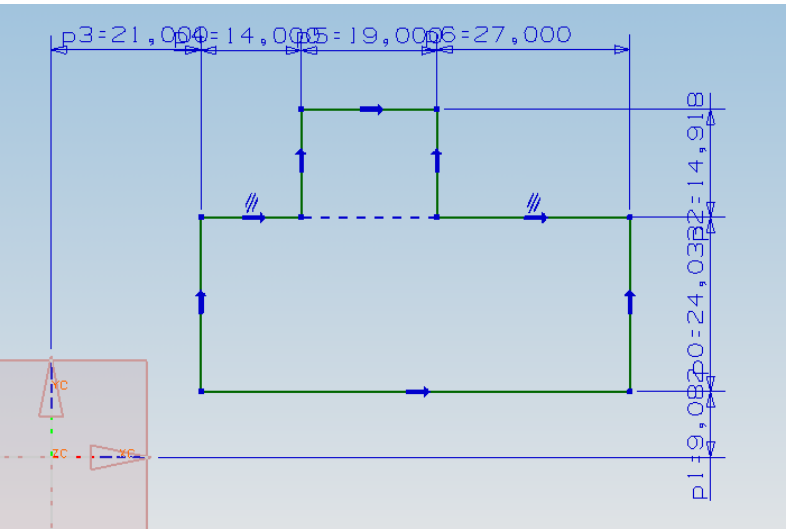
Match given x-section?

N

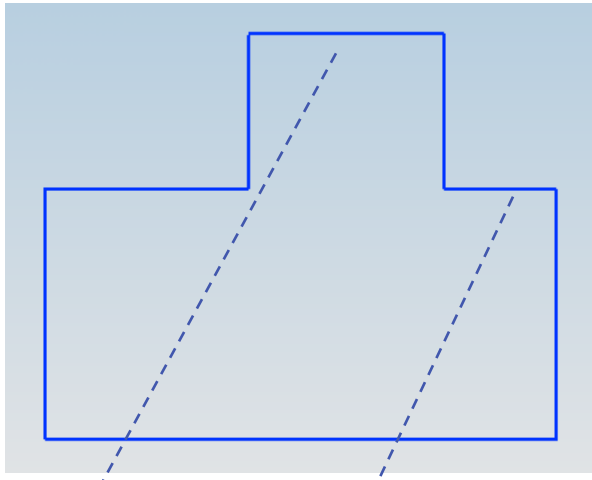
Y

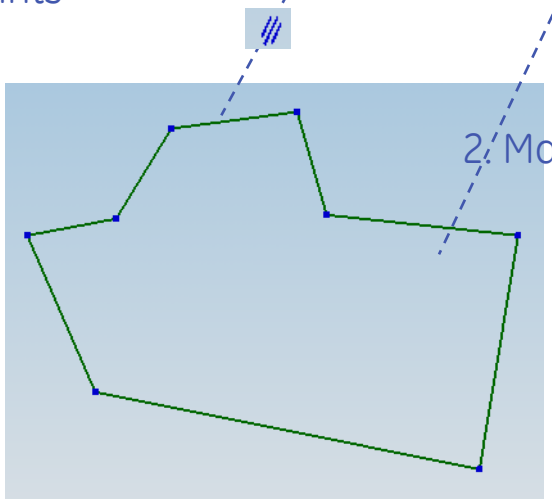
Optimization approach: inefficient

Methodology

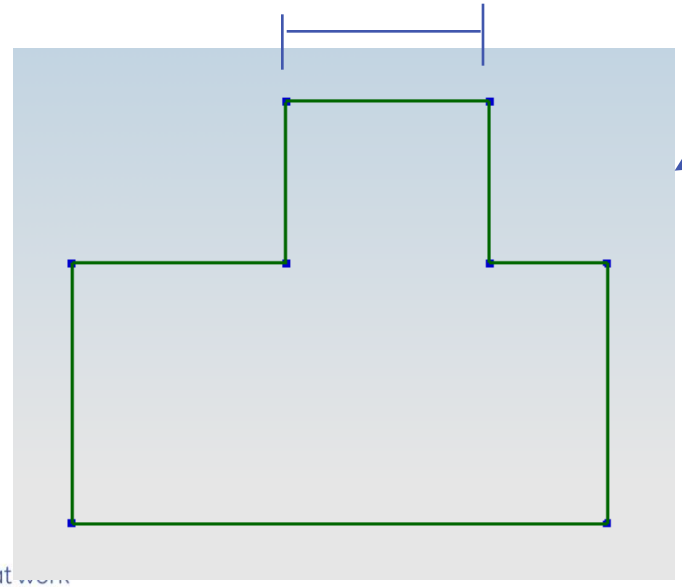


1. Release constraints





2. Match



3. Measure

P0=1.0
P1=34
P4=6
...

Demo

- Parameterize a simple flange cross section

Summary

- Java technology provides a powerful, and reliable platform for UG based CAD tool development.
- Support of Java in UG is getting mature.

- Half of the development in Design Automation group of GRC is done with Java now.

Acknowledgement

- Jeff Erno (GE GRC, Niskayuna, NY)
- Ashok K. Dhakar (GE BEC, Bangalore, India)