

Java DLL Builder

How does it work

Dynamic link libraries (DLL) provide an efficient mechanism for sharing code and cross-language linkage. Using `JavaDllBuilder` you can create a DLL that exports selected Java methods in a public Java class as Windows native APIs. Now whether you build a C++, VB, Delphi, .NET, or an Office application your code can call those Java methods as though they were Windows functions.

`JavaDllBuilder` uses `Coroutine` for Java library to emit Windows executable and doesn't require C++ compiler or any other tool outside JRE.

`JavaDllBuilder` class provides a single static method that you use to generate a DLL

```
public static void generateDll(String dllname, Class clazz,
                             String [] methodlist)
```

Parameter `dllname` above specifies the name of the DLL to be generated. This parameter may include full pathname. Parameter `clazz` specifies the class which methods will be exported. Methods names are listed in the last parameter `methodlist`.

`JavaDllBuilder` searches `clazz` for all methods listed in `methodlist`. Only methods that have integer parameters or no parameters are selected. If more than one method found for a name, “__N” is appended to exported name where N specifies number of parameters in a method.

`JavaDllBuilder` doesn't compile method's bytecode to machine code. Essentially, each API generated by `JavaDllBuilder` contains a native trampoline stub that forwards API arguments to a JNI call into Java method and passes back the value returned from the method.

Each API uses `__stdcall` calling convention - the standard calling convention of Win32 API functions. Exported names have no decoration.

At run time, a JVM is created within the hosting process on the first call to any of exported Java methods. Consequently a Java object which methods are exported is created. This object is available to all subsequent calls till JVM is destroyed when a hosting process unloads the DLL.

Building Java Methods

Each Java method that you want to expose as a native API

- must be defined as a public instance method of a public class.
- must have as many integer parameters as there are four byte parameters in native API.

For example, if native API has an 8-byte double in its parameter list, Java method must define two integers.

API definition	Java Implementation
double JavaDoubler(double);	<pre>public double JavaDoubler(int p1, int p2) { double j= Coroutine.doubleFrom2Int(p1,p2); return 2*j; }</pre>

The following table lists most typical data types that native API may use, their Java equivalents, and conversion procedures.

Native Type	API parameter	Java method parameter	Conversion
8-bit character	char p1	int p1	char c=(char)p1
16-bit integer	short int p1	int p1	short j=(short)p1
32-bit integer	int p1	int p1	int j=p1;
64-bit integer	__int64 p1	int p1, int p2	long j=Coroutine.longFrom2Int(p1,p2);
Boolean	bool p1	int p1	boolean j=(p1>0);
IEEE 32-bit float	float p1	int p1	float j=Float.intBitsToFloat(p1);
IEEE 64-bit double	double p1	int p1, int p2	double j=Coroutine.doubleFrom2Int(p1,p2);
32-bit pointer	void * p1	int p1	Int j=p1;
function pointer	void * p1	int p1	Coroutine j=Coroutine.BuildCoroutineFromPtr(p1);
C string	char* p1	int p1	String j= Coroutine.StringFromPtr(p1);
Pointer to a byte array. Array size is N bytes.	char* p1	int p1	byte [] j= Coroutine.byteArrayFromPtr(p1,N);
Structure by reference.	struct * p1	int p1	byte [] j= Coroutine.byteArrayFromPtr(p1,N); Use Coroutine.getXXXAtOffset() to extract

Structure size is N bytes			members
8-bit character by reference	char*	int p1	byte j=Coroutine. byteFromPtr(p1); ... Coroutine.byteToPtr(p1,j);
16-bit integer by reference	short int *p1	int p1	short j= Coroutine. shortFromPtr(p1); .. Coroutine.shortToPtr(p1,j);
32-bit integer by reference	int *p1	int p1	int j= Coroutine.intFromPtr(p1); ... Coroutine.intToPtr(p1,j);
64-bit integer by reference	__int64 *p1	int p1	long j=Coroutine.longFromPtr(p1); ... Coroutine.longToPtr(p1,j);
Boolean by reference	bool *p1	int p1	boolean j=Coroutine.booleanFromPtr(p1); ... Coroutine.booleanToPtr(p1,j);
IEEE 32-bit float by reference	float *p1	int p1	float j=Coroutine.floatFromPtr(p1); ... Coroutine.floatToPtr(p1,j);
IEEE 64-bit double by reference	double *p1	int p1	double j=Coroutine.doubleFromPtr(p1); ... Coroutine.doubleToPtr(p1,j);
C string by reference	char**p1	int p1	String j=Coroutine.stringFromPtr(Coroutine.intFromPtr(p1)); ... int ptr=Coroutine.stringAsPtr(j); Coroutine.intToPtr(p1,ptr); Windows application must free the memory by using CoTaskMemFree()
Byte array by reference	char**p1	int p1	byte [] j= Coroutine.byteArrayFromPtr(Coroutine.intFromPtr(p1)),N); ... int ptr=Coroutine.byteArrayAsPtr(j); Coroutine.intToPtr(p1,ptr); Windows application must free the memory by using CoTaskMemFree()

Return value

Native Type	API return	Java type and conversion
8-bit character	char	char j; ... return j; or byte j; ... return j;
16-bit integer	short int	short j; ... return j;
32-bit integer	int	int j; ... return j;
64-bit integer	__int64	long j; ... return j;
Boolean	bool	boolean j; ... return j;
IEEE 32-bit float	float	float j; ... return j;
IEEE 64-bit double	double	double j; return j;
32-bit pointer	void *	Int j; ... return j;
C string	char*	String j; ... return Coroutine.stringAsPtr(j); Windows application must free the memory by

		using CoTaskMemFree()
Pointer to a byte array.	char* p1	byte [] j; ... return Coroutine.byteArrayAsPtr(j); Windows application must free the memory by using CoTaskMemFree()

Set Java VM Runtime Options

To control Java VM runtime options you use a `.jvmoptions` file. The `.jvmoptions` file is an ASCII file that must be placed in the same directory as the dll, with the same name as the dll with `.jvmoptions` appended to the end of the name. Each line in the file specifies a single JVM option. If the first character on the line is `#`, the line is treated as a comment.

Here is an example of a `.jvmoptions` file

```
#disable JIT
-Djava.compiler=NONE
#user classes
-Djava.class.path=X:/WINXP/JAVACLS0
#JVM location
jvm=K:\Program Files\Java\jre1.5.0_09\bin\client\jvm.dll
```

Please note that you can specify the location of the `jvm.dll` file to use as your Java Virtual Machine by using `jvm=<fully qualified path to the Java Virtual Machine dynamic link library>`. For Sun's JRE, this is usually `jvm={JRE_HOME}\bin\client\jvm.dll`.

If you don't specify JVM location, JavaDll Runtime will search the Registry at `HKLM\SOFTWARE\JavaSoft\Java Runtime Environment` and will pickup the latest JVM installed.

Alternatively, to control Java VM runtime options you can call the `SetJVMOptions` API that is always available in a DLL generated by `JavaDllBuilder`. Make sure that you call `SetJVMOptions` before JVM is created, that is, before any of Java methods is called. The `SetJVMOptions` API takes two parameters: an integer that specifies number of option strings in the second parameter and a `char**` array of option strings

```
void __stdcall SetJVMOptions(int argc, char** argv);
```

Using Java DLL: Load-Time Dynamic Linking

With load-time dynamic linking the hosting application can call the library functions as if they were local functions and require no extra programming effort. Typically, C++ applications use load-time dynamic linking. However, in order to use load-time dynamic linking you need to link the hosting C++ application with an import library for the DLL.

An import library contains no executable code. It consists of only names and locations of exported functions in a DLL.

Here is how you build an import library for a Java DLL. When you execute `generateDll` method you may notice that two additional files are generated: a C++ header file and a module definition `.def` file. You should use C++ compiler and appropriate options to compile C++ header file and generate import library. For Microsoft C++ compiler, the command string will look like the following (replace `<name>` with the name of your DLL):

```
copy <name>.h <name>.cpp
cl /LD /D_GENLIB=1 /Feplib<name>.dll <name>.cpp <name>.def
```

Above will generate `lib<name>.dll` which you need to discard and an import library `lib<name>.lib` that you use when you link C++ hosting application.

Using Java DLL: Run-Time Dynamic Linking

Every programming language in Microsoft Windows environment supports run-time dynamic linking. On a very low level, run-time dynamic linking involves three steps in which the DLL hosting application a) calls `LoadLibrary` API to load a DLL; b) calls `GetProcAddress` to obtain the address of each DLL function it requires; c) calls `FreeLibrary` to unload the DLL from the memory when finished with the DLL.

Most of high level languages offer tools that enable to declare a local function or a method as having its implementation in an external DLL so there is no need for explicit calls to `LoadLibrary/GetProcAddress/FreeLibrary`. For example, C# in .NET offers the `[DllImport()]` attribute that one can use to declare a C# static method as being implemented in a DLL export. Similarly, Microsoft Visual basic offers `Define Function` statement.

Debugging Java DLL

To enable Java remote debugging set the following JVM options in `.jvmoptions` file

```
-Djava.compiler=NONE  
-Xdebug  
-Xnoagent  
-Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8002
```

Of course, you can modify `-Xrunjdwp` sub-option to adjust to your debugging environment.

Examples

Microsoft Excel Add-In Written in Java

The XLL Add-In interface to Excel allows creating global worksheet functions. It was introduced in Excel97 as the C equivalent to the built-in macro language. Even now after years of proliferation of MS Office COM Add-ins XLL interface still remains the easiest and the fastest way to build custom Excel Add-Ins.

An XLL Add-In is just a Windows Dynamic Link Library with several mandatory exports that allow Excel and Excel Add-In Manager to open an Add-In and register the exported functions in such a way that they can be used directly on the worksheet.

Excel calls the `xllAutoOpen` function implemented and exported by an XLL whenever the XLL is added to Excel, either via Add-In Manager, REGISTER macro, or when an XLL file is in the Excel home directory. This function does not take any parameters. Within `xllAutoOpen`, your code must register all the worksheet functions, add any menus and sub-menus to Excel's application menu bar, and perform any other initialization that is required. Note that this is the only function that is required for Microsoft Excel to run an XLL.

Optional functions that an XLL can implement and export include `xllAutoAdd`, `xllAutoRemove`, `xllAutoClose`, `xllAddInManagerInfo`.

`xllAutoAdd` is called when an XLL is added via Add-In manager. This function has no arguments.

`xllAutoRemove` and `xllAutoClose` are called when an XLL is removed via Add-In manager. These functions have no arguments.

`xllAddInManagerInfo` is called by Microsoft Excel Add-In Manager and provides textual information about your XLL

Microsoft Excel offers the `Excel4` function in `XL_CALL32.DLL` that enables your XLL to call back into Excel. You use this function to call internal Excel macro functions and native Excel worksheet functions

The `Excel4` function takes three or more arguments:

```
int _cdecl Excel4(int iFunction, XLOPER *pxRes, int iCount, ...)
```

The first parameter specifies internal Microsoft Excel function you want to call. You should always use one of the predefined constants from `xllcall.h` in Excel SDK. The second argument is a pointer to a `XLOPER` data structure that accepts the result of the call. If you use `NULL` in place of the second parameter, Excel will discard the return value. The third parameter specifies how many arguments to the Excel function will follow (from 0 to 30). All the arguments must be pointers to `XLOPER` data structures. The return value (one of `xllret` constants in `xllcall.h`) indicates whether the call to `Excel4` function was successful.

Note that you can call `Excel4` only when control has been passed to your XLL by Microsoft Excel. For example, you cannot create a thread from your XLL implemented function and call `Excel4` from that thread. In the example below, we will show you how to make asynchronous calls to Excel.

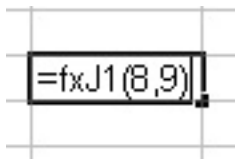
The `XLOPER` data structure mentioned above is essentially a 16-byte variant structure that can be used to pass floating point values, integers, strings, error codes, arrays, references, etc. In the example that follows we created Java wrapper class for this structure.

Finally, functions that will be called from Microsoft Excel should use `__stdcall` calling convention and must not have name decoration.

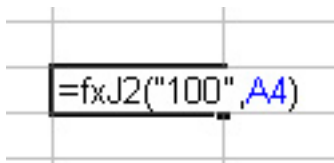
You can find more information on XLL interface on MSDN (currently at <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/office97/html/SF8B8.asp>)

In the example that follows we create an Excel Add-In that contains three worksheet functions `fxJ1`, `fxJ2`, and `fxJ3`.

`fxJ1` takes two 8-byte floating-point numbers and returns the product of its two parameters.



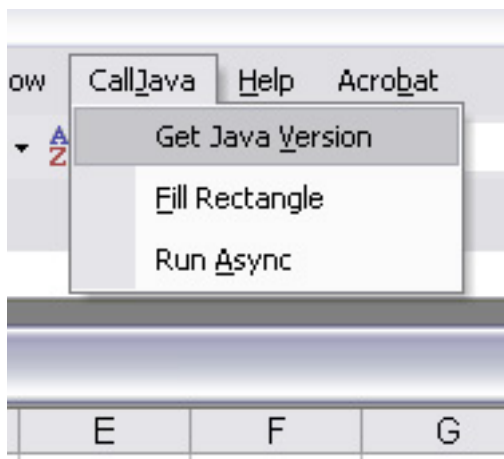
`fxJ2` returns the product of its two parameters; valid parameters are those that contain or can be coerced to a numeric value, or a reference to a cell that contains a numeric value.



Finally, `fxJ3` takes a rectangle array of cells and computes a sum of all elements

	C	D
	67.88	877.98
	98.1	2132.9
	56.6	56.8
	565	522
	33.98	112
	45.7	98.876
	<code>=fxJ3(C1:D6)</code>	

Additionally, Add-in adds `CallJava` drop down menu to Excel menu bar.



`Get Java Version` displays Java Runtime version and the name of the active worksheet; `Fill Rectangle` fills selected rectangle with zeroes; `Run Async` launches asynchronous execution of a Java method that updates active worksheet.

First we create two Java helper classes: `XLOPER.java` that manages access to `XLOPER` data structures and `Excel4.java` that wraps calls to `Excel4` Excel helper function.

```
import com.neva.*;
public class Excel4 {
    //Return codes
    public static int xlretSuccess    = 0;
    public static int xlretAbort      = 1;
    public static int xlretInvXlfn    = 2;
    public static int xlretInvCount   = 4;
    public static int xlretInvXloper  = 8;
    public static int xlretStackOvfl  = 16;
    public static int xlretFailed     = 32;
```

```

public static int xlretUncalced = 64;
// Function number bits
public static int xlCommand=0x8000;
public static int xlSpecial=0x4000;
public static int xlInt1=0x2000;
public static int xlPrompt=0x1000;
//Auxiliary function numbers
public static int xlFree =(0 | xlSpecial);
public static int xlStack=(1 | xlSpecial);
public static int xlCoerce =(2 | xlSpecial);
//...
//some code is removed for clarity
//...
// Wrapper for Excel4 helper function
public static int CallExcel(int func, XLOPER ret, XLOPER [] param) {
    Coroutine co=new Coroutine("XLCALL32","Excel4");
    co.addArg(func);
    if(ret==null)
        co.addArg(0);
    else
        co.addArg(ret.asPointer());
    if(param==null)
        co.addArg(0);
    else {
        co.addArg(param.length);
        for(int j=0;j<param.length;j++)
            co.addArg(param[j].asPointer());
    }
    int r=co.invoke();
    if(r!=0)
        throw new RuntimeException(
            "Invocation error in Excel4: "+co.lastError());
    return co.answerAsInteger();
}
//This function sets an integer value in a range of cells
public static void setValue(int rowFirst, int rowLast,
    int colFirst, int colLast, int value) {
    XLOPER xRef=new XLOPER(rowFirst,rowLast,colFirst,colLast);
    XLOPER val=new XLOPER(value);
    CallExcel(xlSet,null,new XLOPER[] {xRef,val});
}
//This function calls alert to display a message
public static void alert(String text, int type) {
    XLOPER msgtype=new XLOPER(type);
    XLOPER msg=new XLOPER(text);
    XLOPER [] param=new XLOPER[] {msg,msgtype};
    CallExcel(xlcAlert,null,param);
}
//Answers xll name
public static String getXllName() {
    XLOPER xDll=new XLOPER();
    CallExcel(xlGetName,xDll,null);
    String name=xDll.asString();
    xDll.free();
    return name;
}
//Registers a worksheet function
public static void registerFunction(XLOPER [] param) {
    XLOPER ret=new XLOPER();
    int rc=Excel4.CallExcel(xlfRegister,ret,param);
    if(rc!=xlretSuccess)
        throw new RuntimeException("Error "+rc+
            " in Excel4 while in registerFunction");
}

```

```

    }
    //...
    //some code is removed for clarity
    //...
}

```

Here is XLOPER Java class

```

import com.neva.*;
public class XLOPER {
    //...
    //some code is removed for clarity
    //...
    public static short xltypeNum=0x0001;
    public static short xltypeStr=0x0002;
    public static short xltypeBool=0x0004;
    public static short xltypeRef=0x0008;
    public static short xltypeErr=0x0010;
    public static short xltypeFlow=0x0020;
    public static short xltypeMulti=0x0040;
    public static short xltypeMissing=0x0080;
    public static short xltypeNil=0x0100;
    public static short xltypeSRef=0x0400;
    public static short xltypeInt=0x0800;

    private int ptr;
    private ExternalObject eo=null;
    private ExternalObject wrapper=null;
    private int sizeofXLOPER=16;
    public XLOPER() {
        byte [] b=new byte[sizeofXLOPER];
        Coroutine.setWORDAtOffset(b,xltypeNil,8);
        wrapper=new ExternalObject(b,b.length);
        ptr=wrapper.getValue();
    }
    public XLOPER(String s) {
        byte [] by=new String(" "+s+"\0").getBytes();
        by[0]=(byte)(s.length()+1);
        eo=new ExternalObject(by,by.length);
        byte [] b=new byte[sizeofXLOPER];
        Coroutine.setDWORDAtOffset(b,eo.getValue(),0);
        Coroutine.setWORDAtOffset(b,xltypeStr,8);
        wrapper=new ExternalObject(b,b.length);
        ptr=wrapper.getValue();
    }
    public XLOPER(int i) {
        byte [] b=new byte[sizeofXLOPER];
        Coroutine.setWORDAtOffset(b,xltypeInt,8);
        Coroutine.setDWORDAtOffset(b,i,0);
        wrapper=new ExternalObject(b,b.length);
        ptr=wrapper.getValue();
    }
    public XLOPER(double d) {
        byte [] b=new byte[sizeofXLOPER];
        Coroutine.setINT64AtOffset(b,Double.doubleToLongBits(d),0);
        Coroutine.setWORDAtOffset(b,xltypeNum,8);
        wrapper=new ExternalObject(b,b.length);
        ptr=wrapper.getValue();
    }
    protected void finalize() {
        freeWrapper();
    }
}

```

```

public int asPointer() {
    return ptr;
}
//answers an XLOPER from a memory location
public static XLOPER fromPointer(int ptr) {
    XLOPER ret=new XLOPER();
    ret.wrapper=null;
    ret.ptr=ptr;
    return ret;
}
//free memory allocated for Java wrapper
public void freeWrapper() {
    if(wrapper!=null)
        wrapper.free();
    if(eo!=null)
        eo.free();
    wrapper=null;
    eo=null;
    ptr=0;
}
//frees auxiliary memory allocated by Excel4
public void free() {
    Excel4.CallExcel(Excel4.xlFree,null,new XLOPER[] {this});
}
//Answers data type for this XLOPER
public int getType() {
    byte [] bret=Coroutine.byteArrayFromPtr(asPointer(),sizeofXLOPER);
    return Coroutine.getWORDAtOffset(bret,8);
}
public String asString() {
    byte [] bret=Coroutine.byteArrayFromPtr(asPointer(),sizeofXLOPER);
    int pstr=Coroutine.getDWORDAtOffset(bret,0);
    int type=getType();
    if(type!=XLOPER.xltypeStr)
        throw new RuntimeException("This XOPER doesn't contain a "+
            "STRING; type="+type);
    int sz=Coroutine.byteFromPtr(pstr);
    return new String(Coroutine.byteArrayFromPtr(pstr+1,sz));
}
public double asNumber() {
    byte [] bret=Coroutine.byteArrayFromPtr(asPointer(),sizeofXLOPER);
    long lval=Coroutine.getInt64AtOffset(bret,0);
    int type=getType();
    if(type!=XLOPER.xltypeNum)
        throw new RuntimeException("This XOPER doesn't contain a "+
            "NUMBER; type="+type);
    double val=Double.longBitsToDouble(lval);
    return val;
}
public int asInteger() {
    byte [] bret=Coroutine.byteArrayFromPtr(asPointer(),sizeofXLOPER);
    long lval=Coroutine.getInt64AtOffset(bret,0);
    int type=getType();
    if(type!=XLOPER.xltypeInt)
        throw new RuntimeException("This XOPER doesn't contain an "+
            "INTEGER; type="+type);
    return Coroutine.getWORDAtOffset(bret,0);
}
//Answers number of rows in array stored in this XLOPER
public int getArrayRows() {
    byte [] bret=Coroutine.byteArrayFromPtr(asPointer(),sizeofXLOPER);
    int row=Coroutine.getWORDAtOffset(bret,4);
    int type=getType();
}

```

```

        if(type!=XLOPER.xltypeMulti)
            throw new RuntimeException("This XOPER doesn't contain type "+
                " Multi; type="+type);
        return row;
    }
    //Answers number of columns in array stored in this XLOPER
    public int getArrayColumns() {
        byte [] bret=Coroutine.byteArrayFromPtr(asPointer(),sizeofXLOPER);
        int columns=Coroutine.getWORDAtOffset(bret,6);
        int type=getType();
        if(type!=XLOPER.xltypeMulti)
            throw new RuntimeException("This XOPER doesn't contain type "+
                " Multi; type="+type);
        return columns;
    }
    //Answers a two dimensional array of XLOPERs.
    public XLOPER[][] getArray() {
        int rows=getArrayRows();
        int columns=getArrayColumns();
        XLOPER [][] ret=new XLOPER[rows][columns];
        byte [] bret=Coroutine.byteArrayFromPtr(asPointer(),sizeofXLOPER);
        int ptr=Coroutine.getDWORDAtOffset(bret,0);
        for(int j=0;j<rows;j++) {
            for(int i=0;i<columns;i++) {
                ret[j][i]=XLOPER.fromPointer(ptr);
                ptr+=sizeofXLOPER;
            }
        }
        return ret;
    }
    //This function converts one type of XLOPER to another and returns
    //the coerced value
    public XLOPER coerce(int newtype) {
        XLOPER ret=new XLOPER();
        int rc=Excel4.CallExcel(Excel4.xlCoerce,ret,new XLOPER[]
            {this,new XLOPER(newtype)});
        if(rc!=Excel4.xlretSuccess)
            throw new RuntimeException("Coerce failed with error="+rc);
        return ret;
    }
    //...
    //some code is removed for clarity
    //...
}

```

Finally, here is JavaXll class that implements Excel Add-in

```

import com.neva.*;
import java.util.*;
import java.io.*;
public class JavaXll {
    static boolean runTimer=false;
    static String MacroType1="1";
    static String MacroTypeHidden="0";
    //This array describes all worksheet functions implemented in this Java class
    String [][] functions=new String[][]{
        {"fxJ1","BBB","fxJ1","",MacroType1,"Java Built Add-Ins","", "", "", ""},
        {"fxJ2","RRR","fxJ2","",MacroType1,"Java Built Add-Ins","", "", "", ""},
        {"fxJ3","RR","fxJ3","",MacroType1,"Java Built Add-Ins","", "", "", ""},
        {"MenuItem1","I","MenuItem1","",MacroTypeHidden","", "", "", "", ""},
        {"MenuItem2","I","MenuItem2","",MacroTypeHidden","", "", "", "", ""},
        {"MenuItem3","I","MenuItem3","",MacroTypeHidden","", "", "", "", ""},
    }
}

```

```

    {"OnTimerEvent","I","OnTimerEvent","",MacroTypeHidden,"","","",""},
};
//This array describes the drop-down menu.
String [][] menu=new String[][] {
    {"Call&Java" ,"","","","Call Java", ""},
    {"Get Java &Version","MenuItem1","", "...", ""},
    {"&Fill Rectangle","MenuItem2","", "...", ""},
    {"Run &Async ","MenuItem3","", "...", ""}
};
//This function is called by Excel when Excel opens this XLL
public int xlAutoOpen() {
    try {
        // Get XLL file name
        String path=Excel4.getXllName();
        XLOPER [] param;
        for(int j=0;j<functions.length;j++) {
            param=new XLOPER[] {new XLOPER(path), new XLOPER(functions[j][0]),
                new XLOPER(functions[j][1]), new XLOPER(functions[j][2]),
                new XLOPER(functions[j][3]),
                new XLOPER(functions[j][4]),new XLOPER(functions[j][5]),
                new XLOPER(functions[j][6]),
                new XLOPER(functions[j][7]),new XLOPER(functions[j][8]),
                new XLOPER(functions[j][9])};
            Excel4.registerFunction(param);
        }
        //check whether the menu already exist
        XLOPER xTest=new XLOPER();
        Excel4.CallExcel(Excel4.xlfGetBar,xTest,
            new XLOPER[]{new XLOPER(10),new XLOPER("CallJava"),new XLOPER(0)});
        if(XLOPER.xltypeErr==xTest.getType()) {
            //if menu doesn't exist add
            XLOPER xMenu=new XLOPER(menu);
            Excel4.CallExcel(Excel4.xlfAddMenu,null,
                new XLOPER[] {new XLOPER(10),xMenu,new XLOPER("Help")});
        }
        xTest.free();
        return 1;
    } catch(Throwable ex) {
        OutputDebugString(ex);
        return 0;
    }
}
//Excel calls this function when XLL is unloaded
public int xlAutoClose() {
    //Remove menu
    Excel4.CallExcel(Excel4.xlfDeleteMenu,null,
        new XLOPER[]{new XLOPER(10),new XLOPER("CallJava")});
    //remove names from the Wizard
    String path=Excel4.getXllName();
    XLOPER [] param;
    for(int j=0;j<functions.length;j++) {
        param=new XLOPER[] {new XLOPER(path), new XLOPER(functions[j][0]),
            new XLOPER(functions[j][1]), new XLOPER(functions[j][2]),
            new XLOPER(functions[j][3]),
            new XLOPER(MacroTypeHidden),new XLOPER(functions[j][5]),
            new XLOPER(functions[j][6]),
            new XLOPER(functions[j][7]),new XLOPER(functions[j][8]),
            new XLOPER(functions[j][9])};
        Excel4.registerFunction(param);
    }
    return 1;
}

```

```

//This function is called by the Add-in Manager to find the long
//name of the add-in
public int xlAddInManagerInfo(int p) {
    try {
        XLOPER action=XLOPER.fromPointer(p);
        if(action.asNumber()==1.0) {
            return new XLOPER("Java built Add-In.").asPointer();
        }
        //return #VALUE!
        return 0;
    } catch(Throwable ex) {
        OutputDebugString(ex);
        return 0;
    }
}
//This function takes two 8-byte floating-point numbers
//and returns the product of its two parameters
public double fxJ1(int p1, int p2, int p3, int p4) {
    double d1=Coroutine.doubleFrom2Int(p1,p2);
    double d2=Coroutine.doubleFrom2Int(p3,p4);
    return d1*d2;
}
//This function takes two XLOPERs and returns the product
//of its two parameters in an XLOPER. Valid parameters are those
//that contain or can be coerced to a numeric value, or a
//reference to a cell that contains a numeric value
public int fxJ2(int p1, int p2) {
    XLOPER op1=XLOPER.fromPointer(p1);
    XLOPER op2=XLOPER.fromPointer(p2);
    //if one of parameters is missing return #NUM!
    if(op1.getType()==XLOPER.xltypeMissing ||
        op2.getType()==XLOPER.xltypeMissing)
        return 0;
    try {
        double d1=op1.coerce(XLOPER.xltypeNum).asNumber();
        double d2=op2.coerce(XLOPER.xltypeNum).asNumber();
        return new XLOPER(d1*d2).asPointer();
    } catch(Exception ex) {
        OutputDebugString(ex);
        //we are here if one of parameters contains invalid value
        //return #NUM!
        return 0;
    }
}
//This function takes a rectangle array of cells and computes a
//sum of all elements
public int fxJ3(int p1) {
    XLOPER op1=XLOPER.fromPointer(p1);
    double summ=0;
    if(op1.getType()==XLOPER.xltypeSRef || op1.getType()==XLOPER.xltypeRef ||
        op1.getType()==XLOPER.xltypeMulti) {
        try {
            XLOPER op2=op1.coerce(XLOPER.xltypeMulti);
            int rows=op2.getArrayRows();
            int columns=op2.getArrayColumns();
            XLOPER[][] array=op2.getArray();
            for(int j=0;j<array.length;j++) {
                for(int i=0;i<array[j].length;i++) {
                    try {
                        XLOPER op=array[j][i].coerce(XLOPER.xltypeNum);
                        if(op.getType()==XLOPER.xltypeMissing) {
                        } else {
                            double d1=op.asNumber();

```

```

        summ+=d1;
    }
    } catch(Exception ex1) {
        //we are here if one of elements contains invalid value
        //return #NUM!
        OutputDebugString(ex1);
        return 0;
    }
}
}
}
return new XLOPER(summ).asPointer();
} catch(Exception ex) {
    OutputDebugString(ex);
    //coerce may fail due to an uncalced cell
    return 0;
}
}
//return #NUM!
return 0;
}
//Displays Java version and active sheet name
public int MenuItem1() {
    String sheet="Error: Unable to find active sheet.";
    XLOPER xName=new XLOPER();
    XLOPER xId=new XLOPER();
    if(Excel4.xlretSuccess==Excel4.CallExcel(Excel4.xlSheetId,xId,
                                             new XLOPER[]{})) {
        if(Excel4.xlretSuccess==Excel4.CallExcel(Excel4.xlSheetNm,xName,
                                             new XLOPER[]{})) {
            sheet=xName.asString();
        }
    }
    String message=System.getProperty("java.vendor")+" "+
        System.getProperty("java.version")+"\n"+sheet;
    xName.free();
    xId.free();
    Excel4.alert(message,3);
    return 0;
}
//Fills current selection with 0
public int MenuItem2() {
    //Check whether a workbook is opened
    XLOPER xref=new XLOPER();
    Excel4.CallExcel(Excel4.xlfGetWorkbook,xref,new XLOPER[]{new XLOPER(1)});
    if(XLOPER.xltypeMulti!=xref.getType()) {
        xref.free();
        Excel4.alert("Unable to find a workbook!",3);
        return 0;
    }
    xref.free();
    XLOPER xRef=new XLOPER();
    Excel4.CallExcel(Excel4.xlfSelection,xRef,new XLOPER[]{});
    int [] rect=xRef.getRectRef();
    Excel4.setValue(rect[0],rect[1],rect[2],rect[3],0);
    return 0;
}
//Starts and stops five second timer
public int MenuItem3() {
    if(runTimer) {
        runTimer=false;
        return 0;
    }
    set5secTimer();
}

```

```

    return 0;
}
//Launches five second timer to execute OnTimerEvent function
public void set5secTimer() {
    int rc;
    XLOPER xTime=new XLOPER("NOW()+TimeValue(\"00:00:05\")");
    XLOPER xFunc=new XLOPER("OnTimerEvent");
    XLOPER xTimeEvaluated=new XLOPER();
    XLOPER xFuncEvaluated=new XLOPER();
    XLOPER xErr=new XLOPER();
    try {
        rc=Excel4.CallExcel(Excel4.xlIntl|Excel4.xlfEvaluate,xTimeEvaluated,
            new XLOPER[]{xTime});
        if(rc!=Excel4.xlretSuccess ||
            xTimeEvaluated.getType()==XLOPER.xltypeErr) {
            Excel4.alert("Error: Evaluate time failed!",3);
            throw new RuntimeException();
        }
        rc=Excel4.CallExcel(Excel4.xlIntl|Excel4.xlfEvaluate,xFuncEvaluated,
            new XLOPER[]{xFunc});
        if(rc!=Excel4.xlretSuccess ||
            xFuncEvaluated.getType()==XLOPER.xltypeErr) {
            Excel4.alert("Error: Evaluate function failed!",3);
            throw new RuntimeException();
        }
        rc=Excel4.CallExcel(Excel4.xlIntl|Excel4.xlcOnTime,xErr,
            new XLOPER[]{xTimeEvaluated,xFuncEvaluated});
        if(rc!=Excel4.xlretSuccess || xErr.getType()==XLOPER.xltypeErr) {
            Excel4.alert("Error: xlcOnTime function failed",3);
            throw new RuntimeException();
        }
        runTimer=true;
    } catch(Throwable th) {
        OutputDebugString(th);
    }
    xTimeEvaluated.free();
    xFuncEvaluated.free();
    xErr.free();
    xTime.free();
    xFunc.free();
}
//This code is executed asynchronously
public int OnTimerEvent() {
    if(!runTimer) {
        Excel4.setValue(0,0,0,0,"");
        return 0;
    }
    //display current time in cell A1
    Date now=new Date();
    Excel4.setValue(0,0,0,0,
        ""+now.getHours()+":"+now.getMinutes()+":"+now.getSeconds());
    //reset timer
    set5secTimer();
    return 0;
}
//Some code is removed for clarity
//...
//Builds JavaXll.xll
public static void main(String [] s) {
    try {
        JavaDllBuilder.generateDll("javaxll.xll",JavaXll.class,

```

```

        new String []{"xlAutoOpen", "xlAutoClose", "xlAddInManagerInfo",
        "fxJ1", "fxJ2", "fxJ3", "MenuItem1", "MenuItem2", "MenuItem3",
        "OnTimerEvent"});
    } catch(Exception ex) {
        ex.printStackTrace();
    }
}
}
}

```

Compile above classes, build JavaXll.jar archive

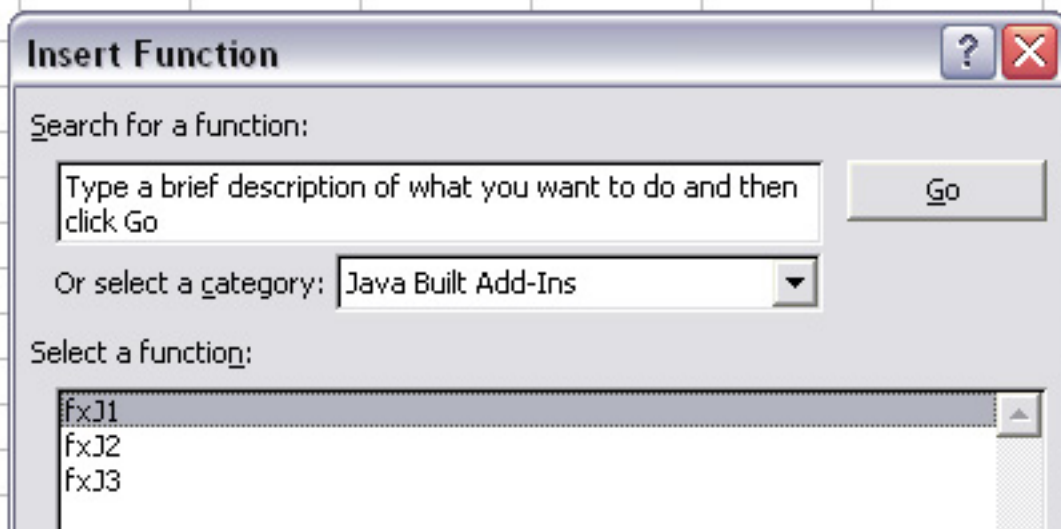
```

>javac JavaXll.java XLOPER.java Excel4.java
>jar c0vf JavaXll.jar *.class

```

and run JavaXll main() to produce javaxll.xll

To use JavaXll Add-in with Microsoft Excel, copy JavaXll.jar to the current JRE lib/ext directory, start Excel, create a new workbook, Click Add-ins on the Tools menu. Browse to javaxll.xll and click OK. Notice when you click OK Call Java menu is added to Excel menubar. You will also discover Java Built Add-ins listed in Insert Function dialog



The example source code is available in Examples/JavaXll folder.

.NET Remoting Server that does Java RMI

In the following example we will add Java RMI support to an existing .NET server. As a result, .NET application will serve both .NET and Java clients.

Here is simple .NET server that distributes California highway traffic conditions found on California Department of Transportation Web site to remote clients.

The implementation uses the shared interface approach when two interfaces, `IRemoteServerFactory` and `IRemoteServer`, are in an assembly that is shared between the client and the server while the implementation resides on the server:

```
namespace com.neva {
    public interface IRemoteServerFactory {
        IRemoteServer GetServer();
    }
    public interface IRemoteServer {
        String GetCAHighwayStatus(String highway);
    }
}
```

`IRemoteServerFactory` has a single `GetServer` method, which returns an object that implements the `IRemoteServer` interface. `IRemoteServer` interface, in turn, has a single `GetCAHighwayStatus` method that takes a valid California highway number ("10", "405", etc.) and returns a string that describes traffic conditions on the highway.

The server implementations of above interfaces, `RemoteServerFactory` and `RemoteServer`, are contained in their own assembly.

```
public class RemoteServerFactory: MarshalByRefObject, IRemoteServerFactory {
    public IRemoteServer GetServer() {
        return new RemoteServer();
    }
}

public class RemoteServer: MarshalByRefObject, IRemoteServer {
    public String GetCAHighwayStatus(String highway) {
        try {
            //Connect to California DOT Web site and POST a request
            //Read the response and remove all the HTML and formatting
            //Return the raw status string back to the client
            int hiwy=Int32.Parse(highway);
            Uri uri = new Uri("http://www.dot.ca.gov/cgi-bin/roads.cgi");
            HttpWebRequest request = (HttpWebRequest) WebRequest.Create(uri);
            request.Referer = "http://www.dot.ca.gov";
            String postsourcedata;
            postsourcedata = "roadnumber="+hiwy;
            request.Method = "POST";
            request.ContentType = "application/x-www-form-urlencoded";
            request.ContentLength = postsourcedata.Length;
            request.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)";
            Stream writeStream = request.GetRequestStream();
            UTF8Encoding encoding = new UTF8Encoding();
            byte[] bytes = encoding.GetBytes(postsourcedata);
            writeStream.Write(bytes, 0, bytes.Length);
            writeStream.Close();
            HttpWebResponse response=(HttpWebResponse) request.GetResponse();
            Stream responseStream = response.GetResponseStream();
        }
    }
}
```



```
[STAThread]
static void Main(string[] args)    {
    String highway="5";
    String host="localhost";
    String port="8099";
    if(args.Length>0)
        highway=args[0];
    if(args.Length>1)
        host=args[1];
    if(args.Length>2)
        port=args[2];
    IRemoteServerFactory factory=(IRemoteServerFactory)
    Activator.GetObject(typeof(IRemoteServerFactory),
    "http://" + host + ":" + port + "/" + "RemoteServerFactory.soap");
    IRemoteServer server=factory.GetServer();
    String s=server.GetCAHighwayStatus(highway);
    Console.WriteLine(s);
}
}
```

When we run above client from the command prompt

```
>NetClient 5 TIGNES.NEVAOBJECT.COM
```

We are getting

```

I 5 [SAN DIEGO & IMPERIAL CO'S] THE NORTHBOUND & SOUTHBOUND CONNECTORS
TO NORTHBOUND SR 163 /IN SAN DIEGO/(SAN DIEGO CO) ARE CLOSED FROM 2200 HRS EACH
NIGHT TO 0500 HRS EACH MORNING MONDAY THRU FRIDAY THRU 7/9/04 - DUE TO
CONSTRUCTION - DETOURS ARE AVAILABLE [ORANGE CO] NO TRAFFIC RESTRICTIONS
ARE REPORTED FOR THIS AREA. [LOS ANGELES & VENTURA CO'S] IS CLOSED TO
SOUTHBOUND TRAFFIC AT LORENA ST /IN LOS ANGELES/ (LOS ANGELES CO) - DUE TO AN
ACCIDENT - A DETOUR IS AVAILABLE [SOUTH CENTRAL CALIFORNIA] A HIGH WIND
ADVISORY IS IN EFFECT FROM THE LOS ANGELES/KERN CO LINE TO 2 MI SOUTH OF
WHEELER RIDGE (KERN CO) /GRAPEVINE/ - TRAVEL IS NOT RECOMMENDED FOR CAMPERS,
TRAILERS OR PERMIT LOADS [CENTRAL CALIFORNIA] NO TRAFFIC RESTRICTIONS
ARE REPORTED FOR THIS AREA. [NORTH CENTRAL CALIFORNIA] NO TRAFFIC
RESTRICTIONS ARE REPORTED FOR THIS AREA. [NORTHERN CALIFORNIA] TRAFFIC IS
REDUCED TO 1 LANE IN EACH DIRECTION FROM 21 MI NORTH TO 24 MI NORTH OF REDDING
/OVER THE SACRAMENTO RIVER BRIDGE/ (SHASTA CO) 24 HRS A DAY 7 DAYS A WEEK THRU
9/30/05 - DUE TO CONSTRUCTION - MOTORISTS ARE SUBJECT TO DELAYS

```

Now back to Java. Our goal is to run Java RMI server in above .NET server. When a Java client calls `GetCAHighwayStatus` via RMI, Java server should delegate the task to .NET implementation then return the result back to the client.

Here is Java definition of `IRemoteServerFactory` and `IRemoteServer` interfaces

```

public interface IRemoteServerFactory extends Remote {
    public IRemoteServer GetServer() throws RemoteException;
}
public interface IRemoteServer extends Remote {
    public String GetCAHighwayStatus(String s) throws RemoteException;
}

```

The following depicts the implementation of above interfaces. Please note the `start` method in `IRemoteServerFactoryImpl` below. This method is used by .NET server to launch Java RMI server and to pass an address of .NET implemented `GetCAHighwayStatus`.

```

public class IRemoteServerFactoryImpl extends UnicastRemoteObject
    implements IRemoteServerFactory {
    static int pcallback;
    public IRemoteServerFactoryImpl() throws RemoteException {
    }
    public IRemoteServer GetServer() throws RemoteException {
        return new IRemoteServerImpl(pcallback);
    }
    public int start(int port, int ptr) {
        try {
            LocateRegistry.createRegistry(port);
            IRemoteServerFactoryImpl impl=new IRemoteServerFactoryImpl();
            Naming.rebind("//:"+port+"/IRemoteServerFactory",impl);
            pcallback=ptr;
            return 0;
        } catch(Throwable ex) {
            return -1;
        }
    }
}
public class IRemoteServerImpl extends UnicastRemoteObject
    implements IRemoteServer {
    int pCallback=0;
}

```

```

public IRemoteServerImpl(int ptr) throws RemoteException {
    pCallback=ptr;
}
public String GetCAHighwayStatus(String s) throws RemoteException {
    try {
        //here we call GetCAHighwayStatus implemented in C#
        Coroutine co=Coroutine.BuildCoroutineFromPtr(pCallback);
        co.addArg(s);
        if(co.invoke()!=0) {
            throw new RuntimeException("Error in invoke ["+co.lastError()+"]");
        }
        //Get a string returned by C#
        String res=co.answerAsString();
        //Since .NET interop marshaler uses CoTaskMemAlloc when a string is
        //passed to unmanaged code, we will use CoTaskMemFree to free the memory
        int ptr=co.answerAsInteger();
        CoTaskMemFree(ptr);
        return res;
    } catch(Throwable th) {
        RemoteException ex=new RemoteException(th.toString());
        ex.initCause(th);
        throw ex;
    }
}
//Java wrapper for CoTaskMemFree API
void CoTaskMemFree(int ptr) {
    Coroutine co=new Coroutine("ole32","CoTaskMemFree");
    co.addArg(ptr);
    if(0!=co.invoke())
        throw new RuntimeException("Error in invoke ["+co.lastError()+"]");
}
}
}

```

To expose the start method above as Windows API we run the following Java code that generates JavaRemoteServer.dll

```

JavaDllBuilder.generateDll("JavaRemoteServer.dll",
    IRemoteServerFactoryImpl.class,
    new String []{"start"});

```

Now we need to modify .NET server to include our Java stuff.

First, create a C# wrapper for the start Java method

```

public delegate String FPTR(String highway);
public class JavaCallWrap {
    [DllImport("JavaRemoteServer.dll")]
    public static extern int start(int rmiPort,FPTR cb);
}
class ServerStartup {
    private GCHandle gch;
    static void Main(string[] args) {
        int netPort=8099;
        int rmiPort=1099;
        new ServerStartup().start(netPort,rmiPort);
        // the server will keep running until keypress.
        Console.ReadLine();
    }
    public void start(int netPort, int rmiPort) {
        HttpChannel chnl = new HttpChannel(netPort);
    }
}

```

```

ChannelServices.RegisterChannel(chnl);
RemotingConfiguration.RegisterWellKnownServiceType(
    typeof(RemoteServerFactory),
    "RemoteServerFactory.soap",
    WellKnownObjectMode.Singleton);
Console.WriteLine ("NET Server started");
//start RMI
FPTR fptr=new FPTR(CallCaltrans);
//protect delegate from GC
gch = GCHandle.Alloc(fptra);
//call into Java passing delegate
if(JavaCallWrap.start(rmiPort,fptra)==0)
    Console.WriteLine ("RMI Server started ");
else
    Console.WriteLine ("RMI Server failed");
}
public String CallCaltrans(String hiwy) {
    return new RemoteServer().GetCAHighwayStatus(hiwy);
}
}

```

Finally, here is Java RMI client

```

public class RMIclient {
    public static void main(String [] args) {
        try {
            String highway="5";
            String host="localhost";
            String port="1099";
            if(args.length>0)
                highway=args[0];
            if(args.length>1)
                host=args[1];
            if(args.length>2)
                port=args[2];
            IRemoteServerFactory factory=(IRemoteServerFactory)Naming.lookup(
                "rmi://" + host + ":" + port + "/IRemoteServerFactory");
            IRemoteServer server=factory.GetServer();
            String s=server.GetCAHighwayStatus(highway);
            System.out.println(s);
        } catch(Throwable ex) {
            ex.printStackTrace();
        }
    }
}

```

When we run above client from the command prompt

```
>java RMIclient 405 TIGNES.NEVAOBJECT.COM
```

We are getting

```
I 405 [ORANGE CO.] NO TRAFFIC RESTRICTIONS ARE REPORTED FOR THIS AREA.
[LOS ANGELES & VENTURA CO.'S] THE SOUTHBOUND CONNECTOR TO NORTHBOUND I 710
/IN LONG BEACH/(LOS ANGELES CO) IS CLOSED FROM 1900 HRS EACH NIGHT TO 0600 HRS
EACH MORNINGMONDAY THRU SATURDAY THRU 7/10/04 - DUE TO CONSTRUCTION - A DETOUR
IS AVAILABLE THE SOUTHBOUND CONNECTOR TO EASTBOUND & WESTBOUND I 105 /IN
HAWTHORNE/(LOS ANGELES CO) IS CLOSED FROM 2100 HRS EACH NIGHT TO 0600 HRS EACH
MORNINGMONDAY THRU SATURDAY THRU 7/10/04 - DUE TO CONSTRUCTION - A DETOUR IS
AVAILABLE THE NORTHBOUND CONNECTOR TO WESTBOUND SR 90 /IN CULVER CITY/(LOS
ANGELES CO) IS CLOSED FROM 2100 HRS EACH NIGHT TO 0600 HRS EACH MORNINGMONDAY
THRU SATURDAY THRU 7/10/04 - DUE TO CONSTRUCTION - A DETOUR IS AVAILABLE
THE SOUTHBOUND CONNECTOR TO EASTBOUND & WESTBOUND SR 90 /IN CULVER CITY/ (LOS
ANGELES CO) IS CLOSED FROM 2300 HRS EACH NIGHT TO 0600 HRS EACH MORNING MONDAY
THRU SATURDAY THRU 7/10/04 - DUE TO CONSTRUCTION - A DETOUR IS AVAILABLE
```

The example source code is available in Examples/NetRMI folder.

Windows Control Panel Applet Written in Java

Control Panel applets are used in Windows to enable end users to configure system-wide parameters or to manage the settings of specific software components. For example, the Java Plug-in Control Panel applet enables you to change Java runtime parameters used by Sun Java Plug-in.

Every Control Panel applet is a dynamic-link library built with a special extension, .cpl. The sole function an applet must export from its dynamic-link library is `CPlApplet`. This function receives messages from Control Panel and performs the requested task. When called by Control Panel, `CPlApplet` takes a window handle, a message and a couple of message specific parameters

```
long CPlApplet(HWND hWnd, unsigned int uMsg, long lParam1, long lParam2);
```

The applet's dynamic-link library must reside in the Windows System directory. Alternatively, for Windows 2000 and later systems, Control Panel items can be installed anywhere on the system and the path to the .cpl file must be recorder in the Registry.

See MSDN at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/shellcc/platform/shell/programmersguide/shell_adv/conpanel.asp for more details.

In the following example, we create small Control Panel applet that displays the default location of Coroutine, Java2COM, and JavaDde libraries. Additionally each library archive size and last modification timestamp is displayed.

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.border.*;
import com.neva.*;
import java.io.*;
import java.net.*;
import java.util.*;

public class JavaCPLApplet {
    static final int CPL_INIT=1;
    static final int CPL_GETCOUNT=2;
    static final int CPL_INQUIRE=3;
    static final int CPL_SELECT=4;
    static final int CPL_DBLCLK=5;
    static final int CPL_STOP=6;
    static final int CPL_EXIT=7;
    static final int CPL_NEWINQUIRE=8;
    static final int CPL_STARTWPARAMS=9;
    JFrame frame=null;
    final JTextArea [] tx=new JTextArea[3];
    final JTabbedPane tabbedPane = new JTabbedPane();
    int icon=0;
    int event;
    public JavaCPLApplet() {
        event=CreateEvent(null,false);
    }
    public int CPLApplet(int hwnd,int msg, int lparam1, int lparam2) {
        try {
            switch(msg) {
                case CPL_INIT:
                    //the control panel should proceed
                    return 1;
                case CPL_DBLCLK:
                    //display applet UI
                    showUI("JavaCPL");
                    //this emulates Windows dialog box
                    //CPLApplet will block here till event
                    //is set from WindowsListener's windowClosing method
                    Wait(event,-1);
                    return 0;
                case CPL_STOP:
                    //do cleanup here
                    CloseHandle(event);
                    return 0;
                case CPL_NEWINQUIRE:
                    processNEWINQUIRE(lparam2);
                    return 0;
            }
            return 1;
        } catch(Throwable th) {
            OutputDebugString(th);
            return 1;
        }
    }
    void processNEWINQUIRE(int lparam2) {
        /*
        typedef struct tagNEWCPLINFO {
            DWORD dwSize;
            DWORD dwFlags;
            DWORD dwHelpContext;

```

```

    LONG_PTR lpData;
    HICON hIcon;
    TCHAR szName[32];
    TCHAR szInfo[64];
    TCHAR szHelpFile[128];
} NEWCPLINFO, *LPNEWCPLINFO;
*/
//we need to fill up NEWCPLINFO structure
byte [] cplinfo=new byte[244];
Coroutine.setDWORDAtOffset(cplinfo,cplinfo.length,0);
Coroutine.setDWORDAtOffset(cplinfo,cplinfo.length,0);
Coroutine.setDWORDAtOffset(cplinfo,
    loadWin32IconFromArchive("JavaCup.ico"),16);
String name="Coroutine4Java\0";
System.arraycopy(new String(name).getBytes(),0,cplinfo,20,name.length());
String info="Java built CP applet\0";
System.arraycopy(new String(info).getBytes(),0,cplinfo,52,info.length());
new Coroutine().copyMemory(lparam2,
    new ExternalObject(cplinfo,cplinfo.length).getValue(),cplinfo.length);
}
public int showUI(String title) {
    frame = new JFrame(title);
    frame.addWindowListener( new WindowAdapter() {
        public void windowClosing( WindowEvent e ) {
            //This will cause CPlApplet to return
            SetEvent(event);
        }
    });
    frame.getContentPane().setLayout (new CardLayout());
    JPanel p1=new JPanel();
    JPanel p2=new JPanel();
    JPanel p3=new JPanel();
    p1.setLayout(new BorderLayout());
    p2.setLayout(new BorderLayout());
    p3.setLayout(new BorderLayout());
    JTextArea t1=new JTextArea(5, 40);
    p1.add(t1);
    JTextArea t2=new JTextArea(5, 40);
    p2.add(t2);
    JTextArea t3=new JTextArea( 5, 40);
    p3.add(t3);
    tabbedPane.addTab("Coroutine", p1);
    tabbedPane.addTab("Java2COM", p2);
    tabbedPane.addTab("JavaDde", p3);
    frame.getContentPane().add(tabbedPane,"T");
    t1.insert(getClassInformation("com.neva.Coroutine"),0);
    t2.insert(getClassInformation("com.neva.COMIUnknown"),0);
    t3.insert(getClassInformation("com.neva.DdeClient"),0);
    tabbedPane.setSelectedIndex(0);
    frame.setSize( 400, 200 );
    frame.setVisible(true);
    return 0;
}
String getClassInformation(String sclazz) {
    try {
        Class clazz=Class.forName(sclazz);
        URL loc=clazz.getProtectionDomain().getCodeSource().getLocation();
        File fl=new File(URLConnection.decode(loc.getFile(),"UTF-8"));
        String s="Loaded from:\n"+fl.toString();
        if(!fl.isDirectory())
            s+="\nmodified:"+new Date(fl.lastModified())+"\nsize: "+fl.length();
        return s;
    } catch(java.lang.ClassNotFoundException er) {

```

```

        OutputDebugString(er);
        return er.toString();
    } catch(Throwable th) {
        OutputDebugString(th);
        return th.toString();
    }
}
//creates a Win32 icon from resource in jar
int loadWin32IconFromArchive(String name) {
    //extract icon's file data from jar file
    //save to current directory
    int read=0,szb=0,off=0;;
    InputStream is=this.getClass().getResourceAsStream("/"+name);
    if(is==null)
        return 0;
    byte [] icondata;
    try {
        szb=is.available();
        off=0;
        icondata=new byte[szb];
        while(true) {
            read=is.read(icondata,off,is.available());
            if(read<0)
                break;
            off+=read;
        }
        FileOutputStream fos=new FileOutputStream(name);
        fos.write(icondata,0,icondata.length);
        fos.close();
    } catch(java.io.IOException ex) {
        return 0;
    }
    //load icon from file
    return loadIcon(name);
}
//sends a string to the current Windows debugger for display
public static void OutputDebugString(String string) {
    Coroutine co=new Coroutine("kernel32","OutputDebugStringW");
    co.addArgUnicodeString(string+"\n");
    co.invoke();
}
//Sends the stack trace to the current Windows debugger for display
public static void OutputDebugString(Throwable th) {
    ByteArrayOutputStream stream=new ByteArrayOutputStream();
    th.printStackTrace(new PrintStream(stream));
    OutputDebugString(stream.toString());
}
//wrapper for WaitForSingleObject API
int Wait(int handle,int howLong) throws SecurityException {
    Coroutine coro=new Coroutine("KERNEL32","WaitForSingleObject");
    coro.addArg(handle);
    coro.addArg(howLong);
    int rc=coro.invoke();
    if(rc!=0)
        return rc;
    return coro.answerAsInteger();
}
//wrapper for CreateEvent API
int CreateEvent(String name, boolean state) {
    Coroutine coro=new Coroutine("KERNEL32","CreateEventA");
    coro.addArgNull();
    coro.addArg(false);
    coro.addArg(state);
}

```

```

    if(name == null)
        coro.addArgNull();
    else
        coro.addArg(name);
    coro.invoke();
    return coro.answerAsInteger();
}
//wrapper for SetEvent API
boolean SetEvent(int handle) {
    Coroutine coro=new Coroutine("KERNEL32","SetEvent");
    coro.addArg(handle);
    coro.invoke();
    return coro.answerAsBoolean();
}
//wrapper for Closehandle API
public static boolean CloseHandle(int handle) {
    Coroutine coro=new Coroutine("KERNEL32","CloseHandle");
    coro.addArg(handle);
    coro.invoke();
    return coro.answerAsBoolean();
}
//wrapper for LoadImage API
int loadIcon(String path) {
    int IMAGE_ICON=1;
    int LR_DEFAULTSIZE=0x0040;
    int LR_LOADFROMFILE=0x0010;
    Coroutine co=new Coroutine("user32","LoadImageA");
    co.addArg(0);
    co.addArg(path);
    co.addArg(IMAGE_ICON);
    co.addArg(0);
    co.addArg(0);
    co.addArg(LR_LOADFROMFILE|LR_DEFAULTSIZE);
    if(co.invoke()!=0)
        return 0;
    return co.answerAsInteger();
}
public static void main(String [] s) {
    try {
        //this code builds JavaCPlApplet.cpl
        JavaDllBuilder.generatedDll("JavaCPlApplet.cpl",
            JavaCPlApplet.class,new String []{"CPlApplet"});
    } catch(Exception ex) {
        ex.printStackTrace();
    }
}
}

```

Compile above code

```
> javac JavaCPlApplet.java
```

Execute main() method to generate JavaCPlApplet.cpl file

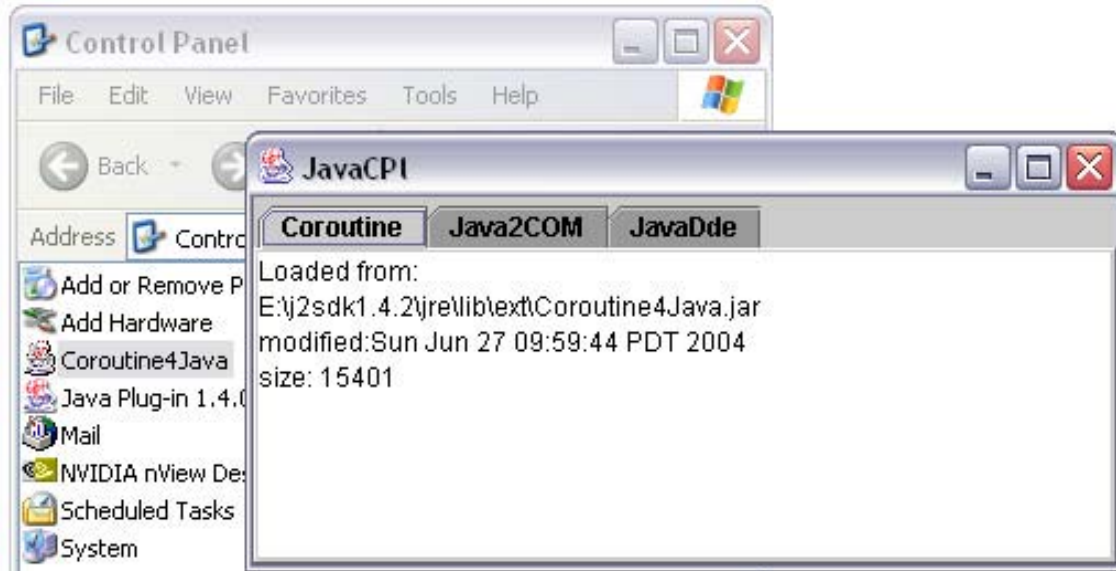
```
> java JavaCPlApplet
```

Build Java archive. Include applet's class files and JavaCup.ico

```
> jar c0vf JavaCPIApplet.jar *.class JavaCup.ico
```

Install JavaCPIApplet.jar as Java extension. Finally, copy JavaCPIApplet.cpl to Windows System directory.

Now click the *Start* button, highlight *Settings*, select *Control Panel*, and click on *Coroutine4Java* icon



The example source code is available in Examples/JavaCPI folder.