

# JavaDde

Neva Object Technology, Inc.

JavaDde bean enables Java applets and applications to interact with Windows applications and other applets/applications using Dynamic Data Exchange Protocol. JavaDde bean is written in 100% Java and uses Coroutine to interact with Windows.

The following provides an overview of the JavaDde classes and interfaces.

## Dde Server

DdeServer is an invisible bean that handles basic DDE server functionality. This bean has a single property named `service` that lists service names for the server to support.

The `start` method starts the server. Before you call this method, you should set the `service` property and register both connection and transaction event listeners.

To stop the server, you call the `stop` method.

The `postAdvise` method causes the server to trigger `onAdvReq` event on `DdeServerTransactionEventListener` interface for each client with an active advise loop on the specified topic and item.

`DdeServerConnectionEvent` is a custom event that is generated by `DdeServer`. This object encapsulates the DDE server side connection event data. The object that implements `DdeServerConnectionEventListener` interface gets a `DdeServerConnectionEvent` when the event occurs.

`DdeServerConnectionEventListener` is the listener interface for receiving `DdeServerConnectionEvent`. The class that is interested in processing DDE server connection events implements this interface, and an instance of that class is registered with a `DdeServer` using the `addDdeServerConnectionEventListener` method.

Class `DdeServerConnectionEventAdaptor` provides an empty implementation of all methods defined in `DdeServerConnectionEventListener`.

`DdeServerTransactionEvent` is a custom event that is generated by `DdeServer`. This object encapsulates the DDE server side transaction data. The object that implements `DdeServerTransactionEventListener` interface gets a `DdeServerTransactionEvent` when the event occurs.

`DdeServerTransactionEventListener` is the listener interface for receiving DDE server transaction events. The class that is interested in processing DDE server transaction events implements this interface, and an instance of that class is registered with a `DdeServer` using the `addDdeServerTransactionEventListener` method.

Class `DdeServerTransactionEventAdaptor` provides an empty implementation of all methods defined in `DdeServerTransactionEventListener`.

## Dde Client

DdeClient is an invisible bean that handles basic DDE client functionality.

Class DdeClient offers public methods to connect to a given DDE server, terminate connection, perform various DDE transactions.

To establish a DDE conversation with a server, JavaDde client uses the *connect* method that takes three parameters: a String that specifies the name of the service to connect to or a null (when a null is specified, a conversation is established with any available server); a String that specifies the name of the topic or a null (when a null is specified, a conversation on any topic supported by the server is established); the last parameter specifies the maximum length of time, in milliseconds, that the client will wait for a response from the server.

When a DDE client wants to establish conversations with all server applications that support the specified service name and topic name pair, it uses the *connectList* method. After a successful execution of the *connectList* method, a DDE client can retrieve a list of all connection handlers for the connection by using the *enumConnections* method.

DdeClientTransactionEvent is a custom event that is generated by DdeClient. This object encapsulates the DDE client side transaction data. The object that implements DdeClientTransactionEventListener interface gets a DdeClientTransactionEvent when the event occurs.

DdeClientTransactionEventListener is the listener interface for receiving DDE client side transaction events. The class that is interested in processing DDE client transaction events implements this interface and an instance of that class is registered with a DdeClient using the *addDdeClientTransactionEventListener* method.

Class DdeServerTransactionEventAdaptor provides an empty implementation of all methods defined in DdeServerTransactionEventListener.

A client can send either synchronous or asynchronous transactions. In a synchronous transaction, the client specifies a time-out value that indicates the maximum amount of time it will wait for the server to process the transaction. During a synchronous transaction, DdeClient transaction methods (*request*, *execute*, and *poke*) do not return until the server processes the transaction, the transaction fails, or the time-out value expires.

When the *connectList* method is used to establish original connections, the client must use the *setCurrentConnection* method to specify the destination for the current transaction.

During an asynchronous transaction, DdeClient transaction methods (*requestAsync*, *executeAsync*, *pokeAsync*) return immediately after the transaction has begun, answering a transaction identifier for further reference. When the DDE server finishes processing an asynchronous transaction, the DDE sends a notification back to the client. As a result, the *onAsyncComplete* method in DdeClientTransactionEventListener is invoked providing the client with the

opportunity to verify the results of the asynchronous transaction. Finally, a client application can choose to abandon an asynchronous transaction by calling the *abandonTransaction* method.

To close a connection, JavaDDE client uses the disconnect method. When the *connectList* method is used to establish original connections, the disconnect method closes all connections; to close a single connection, JavaDde client must use the *disconnectSingle* method.

## Dde Remote Client

DdeRemoteClient offers a way of extending Windows DDE protocol to non-Windows clients via Java RMI.

The DdeRemoteClient implementer - DdeRemoteClientImpl class - simultaneously serves as an RMI server and a DDE client.

To obtain an instance of DdeRemoteClient in your client application you need to acquire a reference to DdeRemoteClientFactory via RMI naming service. Then you call the *getDdeRemoteClient* method that answers an instance of DdeRemoteClient.

```
String host="tignes.nevaobject.com: 1099");
com.neva.DdeRemoteClientFactory factory =
    (com.neva.DdeRemoteClientFactory) Naming.lookup("rmi://" + host + "/DdeRemoteClientFactory");
com.neva.DdeRemoteClient cl = factory.getDdeRemoteClient();
```

On Windows machine that runs your DDE server you need to register an instance of DdeRemoteClientFactoryImpl with RMI Registry. JavaDde.jar supplies the DdeRemoteClientFactoryHost class that does the following:

```
package com.neva;
import java.rmi.*;
import java.rmi.server.*;
import java.rmi.registry.LocateRegistry;
public class DdeRemoteClientFactoryHost {
    public static void main(String [] s) {
        try {
            int port=1099;
            if(s.length>0)
                port=Integer.parseInt(s[0]);
            LocateRegistry.createRegistry( port);
            DdeRemoteClientFactoryImpl impl=new DdeRemoteClientFactoryImpl();
            Naming.rebind("//:" + port + "/DdeRemoteClientFactory",impl);
        } catch(Throwable ex) {
            ex.printStackTrace();
        }
    }
}
```

So you can simply run from command prompt

```
java com.neva.DdeRemoteClientFactoryHost
```

To receive DDE client transaction events, your remote application must implement `DdeRemoteClientTransactionEventListener` interface by extending the `DdeRemoteClientTransactionEventAdaptor` class and register it with a `DdeRemoteClient` using the `addDdeClientTransactionEventListener` method.

## Examples

### Basic DDE server written in Java to serve data to Excel

The following example shows a very basic DDE server application that serves data to DDE clients such as Microsoft Excel.

The application first creates a `DdeServer` and initializes it with the service name. Then, it registers connection and transaction event listeners, and starts the server. Finally, the application creates a thread to simulate changing data. The thread updates a counter variable continuously, and advises any DDE clients listening as it changes.

To test your DDE server:

- Start `JavaDdeServer` Java application
- Start Excel, and type the following in a cell:  
=JavaDdeServer|MyTopic!MyItem
- The cell should obtain the value of `g_cnt` from the server application, and increment rapidly as it receives updates.

```
import com.neva.*;
public class JavaDdeServer implements Runnable {
    private int g_cnt=0;
    private Thread th;
    private com.neva.DdeServer ddeServer;
    private String MyTopic="MyTopic";
    private String MyItem="MyItem";
    private int xtablefmt;
    private int CF_TEXT=com.neva.DdeUtil.CF_TEXT;

    public static void main(String [] param) {
        new JavaDdeServer().runServer();
    }
    public void runServer() {
        try {
            xtablefmt=com.neva.DdeServer.ddeRegisterClipboardFormat("XITable");
            ddeServer=new com.neva.DdeServer();
            ddeServer.setService("JavaDdeServer");
            //Register connection event listener
            ddeServer.addDdeServerConnectionEventListener(
```

```

        new com.neva.DdeServerConnectionEventAdaptor() {
    public void onConnect(com.neva.DdeServerConnectionEvent e)
        throws com.neva.DdeServerConnectionRejectedException {
        // Make sure the client is asking for the right topic
        if(e.getTopic().equals(MyTopic)) {
            System.out.println("New Connection established");
            return;
        }
        throw new com.neva.DdeServerConnectionRejectedException();
    }
});
//Register transaction event listener
ddeServer.addDdeServerTransactionEventListener(
    new com.neva.DdeServerTransactionEventAdaptor() {
    public void onAdvStart(com.neva.DdeServerTransactionEvent e)
        throws com.neva.DdeServerTransactionRejectedException {
        // Make sure the client is asking for the right topic and item
        if(e.getTopic().equals(MyTopic) && e.getItem().equals(MyItem)) {
            return;
        }
        throw new com.neva.DdeServerTransactionRejectedException();
    }
    public void onAdvReq(com.neva.DdeServerTransactionEvent e)
        throws com.neva.DdeServerTransactionRejectedException {
        //Data is ready. Check whether the format is correct
        if(e.getFormat() == xtablefmt) {
            byte [] xldata=buildXLTable(""+g_cnt);
            e.setRequestedData(xldata);
            return;
        } else if(e.getFormat() == CF_TEXT) {
            e.setRequestedData(new String(""+g_cnt).getBytes());
            return;
        }
        throw new com.neva.DdeServerTransactionRejectedException();
    }
    public void onRequest(com.neva.DdeServerTransactionEvent e)
        throws com.neva.DdeServerTransactionRejectedException {
        // Make sure the client is asking for the right topic and item
        if(!e.getTopic().equals(MyTopic) || !e.getItem().equals(MyItem)) {
            throw new com.neva.DdeServerTransactionRejectedException();
        }
        //Return requested data. Check whether the requested
        //format is the one that we support
        if(e.getFormat() == xtablefmt) {
            byte [] xldata=buildXLTable(""+g_cnt);
            e.setRequestedData(xldata);
            return;
        } else if(e.getFormat() == CF_TEXT) {
            e.setRequestedData(new String(""+g_cnt).getBytes());
            return;
        }
        throw new com.neva.DdeServerTransactionRejectedException();
    }
});
// Start DDE server
ddeServer.start();
System.out.println("Dde server started.");
System.out.println("Waiting for transaction...");
// Create a thread to simulate changing data
th=new Thread(this);
th.start();
} catch(Exception exc) {
    System.out.println("Unable to start DDE server");
    exc.printStackTrace();
    System.exit(0);
}
}
byte [] buildXLTable(String s) {
    // This method answers data in xtable format
    byte [] arr=new byte[12+1+s.length()];

```

```

Coroutine.setWORDAtOffset(arr,16,0);
Coroutine.setWORDAtOffset(arr,4,2);
Coroutine.setWORDAtOffset(arr,1,4);
Coroutine.setWORDAtOffset(arr,1,6);
Coroutine.setWORDAtOffset(arr,2,8);
Coroutine.setWORDAtOffset(arr,1+s.length(),10);
Coroutine.setBYTEAtOffset(arr,s.length(),12);
System.arraycopy(s.getBytes(),0,arr,13,s.length());
return arr;
}
public void run() {
    //This thread updates g_cnt counter variable continuously,
    //and advises any DDE clients listening as it changes
    while(true) {
        try {
            g_cnt++;
            //Notify server; postAdvise() will trigger onAdvReq() event
            ddeServer.postAdvise(MyTopic,MyItem);
            Thread.sleep(1000);
        } catch(Exception e) {
        }
    }
}
}
}

```

## Java server that receives notifications from Netscape browser client

Below is Java application that runs DDE server to receive notifications when Netscape loads new URL

To test your DDE server:

- Start Netscape Navigator
- Start EchoURL application

```

import com.neva.*;
public class EchoURL {
    static String our_service="EchoURL";
    static String our_topic="WWW_URLEcho";
    public static void main(String [] argv) {
        int ret;
        String service="NETSCAPE";
        String topic="WWW_RegisterURLEcho";
        if(argv.length>0)
            service=argv[0];
        service=service.toUpperCase();
        try {
            com.neva.DdeServer svr=new com.neva.DdeServer();
            svr.setService(our_service);
            //Register connection event listener
            svr.addDdeServerConnectionEventListener(new com.neva.DdeServerConnectionEventAdaptor() {
                public void onConnect(com.neva.DdeServerConnectionEvent e)
                    throws com.neva.DdeServerConnectionRejectedException {
                    //Only WWW_URLEcho is allowed
                    if(e.getTopic().equals("WWW_URLEcho"))
                        return;
                    throw new com.neva.DdeServerConnectionRejectedException();
                }
            });
            //Register transaction event listener
            svr.addDdeServerTransactionEventListener(
                new com.neva.DdeServerTransactionEventAdaptor() {

```

```

    public void onPoke(com.neva.DdeServerTransactionEvent e)
        throws com.neva.DdeServerTransactionRejectedException {
        String item=e.getItem();
        System.out.println(item);
    }
});
//Start echo server
svr.start();
//Create Dde client
com.neva.DdeClient cli=new com.neva.DdeClient();
//Connect to Browser
cli.connect(service,topic);
if(service.equals("NETSCAPE"))
    //Tell Netscape the name of the Echo server
    cli.poke(our_service,new byte[1],1,10000);
if(service.equals("IEXPLORE"))
    //Tell IE the name of the Echo server
    cli.request(our_service,1,1000);
//we are done here
cli.disconnect();
} catch(Exception e) {
    e.printStackTrace();
    System.exit(0);
}
}
}

```

## Java client that monitors changes in Excel sheet

The following example implements DDE client that establishes a DDE link with Microsoft Excel, places some values into cells, and establishes a hot link to monitor changes.

To test your DDE client:

- Start Microsoft Excel
- Start DdeExcelLinkClient application
- Modify Excel worksheet and watch changes printed by Java application

```

import java.util.*;
import com.neva.*;

public class DdeExcelLinkClient {

    public static void main(String args[]) {
        try {
            Class.forName("com.neva.Coroutine");
        } catch(Exception e) {
            System.out.println("Unable to locate com.neva.Coroutine/com.neva.Jddeml");
            return;
        }
        try {
            int timeout=5000;
            com.neva.DdeClient cl=new com.neva.DdeClient();
            int format=com.neva.DdeUtil.CF_TEXT;
            int i,j;

            //Establish spreadsheet link
            cl.connect("Excel","System");

            //Create new worksheet
            cl.execute("[New(1)]\0",timeout);
            //Get worksheet name
            byte [] data=cl.request("Selection\0",format,timeout);

```

```

String worksheet=new String(data);
int ndx=worksheet.indexOf((int) '!');
if(ndx>0) {
    worksheet=worksheet.substring(0,ndx);
}
cl.disconnect();

//Establish link with new worksheet
cl=new com.neva.DdeClient();
cl.addDdeClientTransactionEventListener(
    new com.neva.DdeClientTransactionEventAdaptor() {
        public void onAdviseData(com.neva.DdeClientTransactionEvent e) {
            String data=new String(e.getDdeData());
            System.out.println("onAdviseData: item="+e.getItem()+" data=\n"+data);
        }

        public void onDisconnect(com.neva.DdeClientTransactionEvent e) {
            System.out.println("Server has gone !");
            System.exit(0);
        }
    });
cl.connect("Excel",worksheet);
//Put some values into worksheet
for(j=1;j<=10;j++)
    for(i=1;i<=10;i++)
        cl.poke("R"+j+"C"+i,new String(""+(i*j)).getBytes(),format,timeout);

String hotitem="R1C1:R10C10" ;

//Establish hot link on range of cells
int csvformat=com.neva.DdeServer.ddeRegisterClipboardFormat("CSV");
cl.startAdvise(hotitem,csvformat,timeout);
System.out.println("Advise link established! "+
    "Go ahead and modify Excel worksheet");
} catch(Exception e) {
    e.printStackTrace();
    System.exit(0);
}
}
}

```

The following example implements DDE client that monitors **all** currently open Excel workbooks

```

import java.util.*;
import com.neva.*;
public class DdeExcelLinkClientMulti {
    int [] conv;
    com.neva.DdeClient cl;
    public static void main(String args[]) {
        new DdeExcelLinkClientMulti().doit();
    }
    void doit() {
        try {
            int timeout=5000;
            int format=com.neva.DdeUtil.CF_TEXT;
            int csvformat=com.neva.DdeServer.ddeRegisterClipboardFormat("CSV");
            int m,max;
            cl=new com.neva.DdeClient();
            cl.addDdeClientTransactionEventListener(new com.neva.DdeClientTransactionEventAdaptor() {
                public void onAdviseData(com.neva.DdeClientTransactionEvent e) {
                    System.out.println("onAdviseData: topic="+e.getTopic()+" item="+e.getItem()+" conv=0x"+
                        Integer.toString(e.getConvHandle(),15) + "\n"+new String(e.getDdeData()));
                }
            });
            public void onDisconnect(com.neva.DdeClientTransactionEvent e) {
                System.out.println("Server conv=0x"+Integer.toString(e.getConvHandle(),16)+" has gone !");
                synchronized(conv) {

```



```

public class Dde2Word {
    public static void main(String args[]) {
        try {
            int timeout=5000;
            int j;
            int format=com.neva.DdeUtil.CF_TEXT;
            byte [] repl;
            String doc=null;
            String [] Topics;

            com.neva.DdeClient cli=new com.neva.DdeClient();
            //connect to MsWord
            cli.connect("WINWORD","System");
            //request the list of topics
            repl=cli.request("Topics",format,timeout);

            Topics=tabparser(new String(repl));
            System.out.println("List of topics available:");
            for(j=0;j<Topics.length;j++) {
                System.out.println(Topics[j]);
                if(Topics[j].endsWith(".doc") || Topics[j].endsWith(".DOC"))
                    doc=Topics[j];
            }

            //disconnect from MsWord
            cli.disconnect();
            if(doc==null) {
                System.out.println("Document is not available...");
                System.exit(0);
            }

            //connect to the document
            cli=new com.neva.DdeClient();
            cli.connect("WINWORD",doc);

            //retrieve the contents of the entire document
            repl=cli.request("\\Doc",format,timeout);
            String text=new String(repl);
            System.out.println(text);

            //Disconnect from the document
            cli.disconnect();
        } catch(Exception e) {
            e.printStackTrace();
            System.exit(0);
        }
    }

    static String [] tabparser(String in) {
        //parses the tab-delimited list of information
        int pos=0;
        int cnt=1;
        while(true) {
            pos=in.indexOf('\t',pos+1);
            if(pos==-1)
                break;
            cnt++;
        }
        String [] out=new String[cnt];
        cnt=0;
        pos=0;
        while(true) {
            int pos2=in.indexOf('\t',pos);
            if(pos2==-1) {
                out[cnt++]=in.substring(pos);
                break;
            } else
                out[cnt++]=in.substring(pos,pos2);
            pos=pos2+1;
        }
    }
}

```

```

    return out;
  }
}

```

## JSP page that generates an Excel spreadsheet via DDE to Excel

The following example demonstrates how to convert an HTML table to an Excel spreadsheet.

For information on deploying and running this JSP onto your application server, see the appropriate server documentation.

```

<%@ page language="java" import="com.neva.* java.io.*" %>
<html>
<!-- Copyright (c) 1996-2003 Neva Object Technology, Inc. . All rights reserved. -->
<head>
<title>Using JavaDde from within JSP</title>
</head>
<body bgcolor="white">
<div align="center">
<form method='GET' action='ddexl.jsp'>
<table border="0" bgcolor="#F1F1F1">
<tr>
<td ><input type="text" name="R1C1" size="10"></td>
<td ><input type="text" name="R1C2" size="10"></td>
<td ><input type="text" name="R1C3" size="10"></td>
<td ><input type="text" name="R1C4" size="10"></td>
</tr>
<tr>
<td ><input type="text" name="R2C1" size="10"></td>
<td ><input type="text" name="R2C2" size="10"></td>
<td ><input type="text" name="R2C3" size="10"></td>
<td ><input type="text" name="R2C4" size="10"></td>
</tr>
<tr>
<td ><input type="text" name="R3C1" size="10"></td>
<td ><input type="text" name="R3C2" size="10"></td>
<td ><input type="text" name="R3C3" size="10"></td>
<td ><input type="text" name="R3C4" size="10"></td>
</tr>
<tr>
<td ><input type="text" name="R4C1" size="10"></td>
<td ><input type="text" name="R4C2" size="10"></td>
<td ><input type="text" name="R4C3" size="10"></td>
<td ><input type="text" name="R4C4" size="10"></td>
</tr>
<tr>
<td ><input type="text" name="R5C1" size="10"></td>
<td ><input type="text" name="R5C2" size="10"></td>
<td ><input type="text" name="R5C3" size="10"></td>
<td ><input type="text" name="R5C4" size="10"></td>
</tr>
</table>
<form method="GET" action="java:void(0)">
<p><input type="radio" value="V1" checked name="download">Download
<input type="radio" name="download" value="V2">Display</p>
<input type='submit' value='Make Excel Workbook' name='GO'>
</form>
</form>
</div>
<%

```

```

try {
    String path=getFilePath()+".xls";
    String r1c1=request.getParameter("R1C1");
    boolean download=request.getParameter("download")!=null &&
        request.getParameter("download").equals("V1");
    if(r1c1!=null && r1c1.length()>0) {
        //Launch Excel
        String excelpath="E:\\Program Files\\Microsoft Office \\Office \\Excel.exe";
        if(0!=CreateProcess(excelpath,false)) {
            return;
        }
        Thread.sleep(5000);
        int timeout=5000;
        DdeClient sys=new DdeClient();
        int format=com.neva.DdeUtil.CF_TEXT;
        int i,j;
        //Connect to Excel
        sys.connect("Excel","System");
        //Create new worksheet
        sys.execute("[New(1)]\0",timeout);
        //Get worksheet name
        byte [] data=sys.request("Selection\0",format,timeout);
        String worksheet=new String(data);
        int ndx=worksheet.indexOf('(');
        if(ndx>0) {
            worksheet=worksheet.substring(0,ndx);
        }
        //Establish link with new worksheet
        DdeClient cl=new DdeClient();
        cl.connect("Excel",worksheet);
        //Put values into worksheet
        for(j=1;j<=4;j++)
            for(i=1;i<=5;i++) {
                String cell="R"+i+"C"+j;
                String value=request.getParameter(cell);
                if(value!=null && value.length()>0)
                    cl.poke(cell,value.getBytes(),format,timeout);
            }
        //save worksheet
        cl.execute("[SAVE.AS(\""+path+"\")\0",timeout);
        cl.disconnect();
        //close worksheet
        sys.execute("[CLOSE.ALL]\0",timeout);
        try {
            //quit Excel
            sys.execute("[QUIT]\0",timeout);
            sys.disconnect();
        } catch(Exception exx) {}
        out.clearBuffer();
        File file=new File(path);
        response.setContentType("application/vnd.ms-excel");
        response.setContentLength((int)file.length());
        if(download)
            response.setHeader("Content-Disposition","attachment; filename=result.xls");
        response.setHeader("Cache-Control","no-cache");
        response.setHeader("Pragma","no-cache");
        response.setDateHeader("Expires", 0);
        int c=0;
        FileInputStream fin=null;
        try {
            fin=new FileInputStream(file);
            while((c=fin.read()) != -1)
                out.write(c);
            out.flush();
        } finally {
            if(fin!=null)
                fin.close();
        }
        file.delete();
        new File(path).delete();
    }
}

```

```

    }
  } catch(Exception ex) {
    out.println("<B>Error: " + ex.getMessage() + "</B>");
  }
}
%>
<%!
public int CreateProcess(String commandLine,boolean showWindow) {
  //some code is removed for clarity
}
String getFilePath() {
  //some code is removed for clarity
}
}
%>
<hr>
</body>
</html>

```

## Java application that uses asynchronous DDE transactions to control Word

In the following example, Java application launches Microsoft Word and uses several asynchronous DDE transactions to create and format new Word document, run spell checker, and to save document on hard disk.

```

import java.awt.*;
import java.awt.event.*;
import java.util.*;
import com.neva.*;
class ControlPanel extends java.awt.Panel {
  public Dimension getPreferredSize() { return new Dimension(400,40);}
  public Dimension getMinimumSize() { return new Dimension(400,40);}
}
public class Java2Word extends Frame {
  public Button build;
  public TextArea text;
  public ControlPanel panel;
  int spellingId=0;
  int saveId=0;
  com.neva.DdeClient cli=null;
  public static void main(String [] param) {
    new Java2Word();
  }
  public Java2Word() {
    super("Build Word Document");
    GridBagLayout gbl=new GridBagLayout();
    GridBagConstraints gbc=new GridBagConstraints();
    setLayout(gbl);
    gbc.gridwidth=GridBagConstraints.REMAINDER;
    gbc.anchor=GridBagConstraints.NORTH;
    gbc.weightx=1.0;
    gbc.weighty=1.0;
    gbc.fill=GridBagConstraints.BOTH;
    text=new TextArea();
    gbl.setConstraints(text,gbc);
    add(text);
    panel=new ControlPanel();
    panel.setLayout(null);
    gbc=new GridBagConstraints();
    gbc.gridwidth=GridBagConstraints.REMAINDER;
    gbc.anchor=GridBagConstraints.SOUTH;
    gbl.setConstraints(panel,gbc);
  }
}

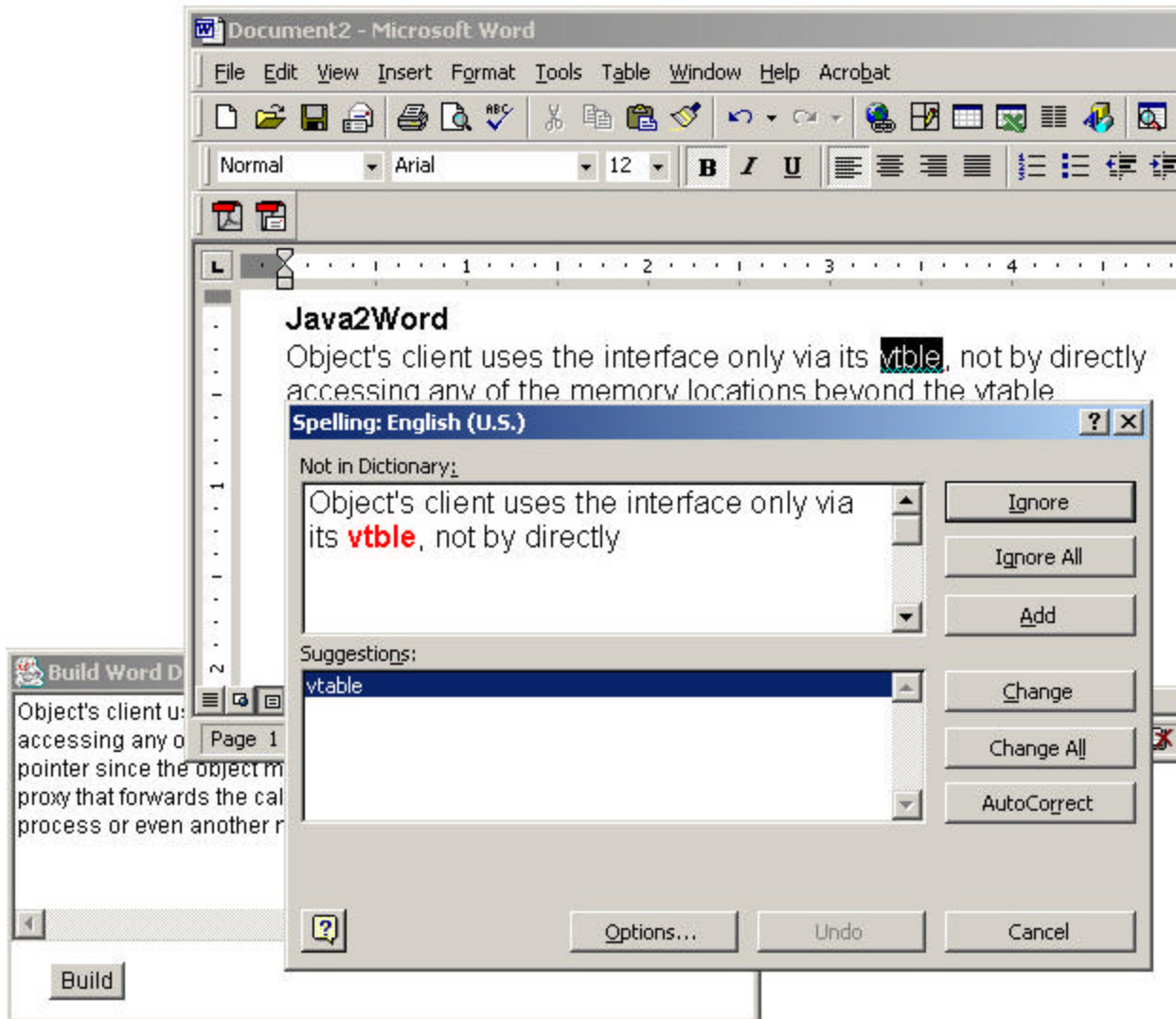
```

```

add(panel);
build=new Button("Build");
panel.add(build);
build.setBounds(20,10,40,20);
build.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        buildWordDocument();
    }
});
enableEvents(java.awt.event.WindowEvent.WINDOW_CLOSING);
setSize(400,200);
setLocation(10,10);
show();
}
String getText() {
    return text.getText();
}
boolean buildWordDocument() {
    try {
        String wordpath="E:\\Program Files\\Microsoft Office\\Office\\winword.exe";
        if(0!=CreateProcess(wordpath,true)) {
            return false;
        }
        Thread.sleep(5000);
        cli=new com.neva.DdeClient();
        cli.addDdeClientTransactionEventListener(new
            com.neva.DdeClientTransactionEventAdaptor() {
            public void onAsyncComplete(DdeClientTransactionEvent event){
                try {
                    if(spellingId==event.getAsyncTransId()) {
                        //Spelling has finished
                        if(event.getAsyncTransSuccess())
                            saveDoc();
                    }
                    if(saveId==event.getAsyncTransId()) {
                        //Doc was saved
                        if(event.getAsyncTransSuccess())
                            quitMSWord();
                    }
                } catch(Exception ex) {
                    ex.printStackTrace();
                }
            }
            public void onDisconnect(com.neva.DdeClientTransactionEvent e) {
                try {
                    cli.disconnect();
                } catch(Exception ex) {
                    ex.printStackTrace();
                }
            }
        });
        cli.connect("WINWORD","System");//connect to MsWord
        cli.executeAsync("[AppRestore]");
        cli.executeAsync("[DocWindowPosTop 100][DocWindowPosLeft 100]");
        cli.executeAsync("[DocWindowWidth 500][DocWindowHeight 300] ");
        cli.executeAsync("[FileNew]");
        cli.executeAsync("[Font \"Arial\"][FontSize 12]");
        cli.executeAsync("[Bold 1][Insert \"Java2Word\"][EndOfLine]");
        cli.executeAsync("[Bold 0][InsertPara][Insert \"\"+getText().replace(\"\\n\", \"\\r\")+\"\\n\"]");
        cli.executeAsync("[EditSelectAll]");
        spellingId=cli.executeAsync("[ToolsSpelling]");
    } catch(Exception ex) {
        ex.printStackTrace();
    }
    return true;
}
void saveDoc() throws com.neva.DdeException, java.lang.InterruptedExcepion {
    saveId=cli.executeAsync("[FileSave]");
}
void quitMSWord() throws com.neva.DdeException, java.lang.InterruptedExcepion {

```

```
cli.executeAsync("[AppClose]");
}
public void processEvent(AWTEvent e) {
    super.processEvent(e);
    if(e.getID()==java.awt.event.WindowEvent.WINDOW_CLOSING) {
        System.exit(0);
    }
}
public int CreateProcess(String commandLine,boolean showWindow) {
    int [] id=new int[2];
    Coroutine coro=new Coroutine("KERNEL32","CreateProcessA");
    coro.addArg(0);
    coro.addArg(commandLine);
    coro.addArg(0);
    coro.addArg(0);
    coro.addArg(true);
    coro.addArg(0);
    coro.addArg(0);
    coro.addArg(0);
    byte [] si=new byte[68];
    coro.setDWORDAtOffset(si,68,0);
    if(!showWindow) {
        coro.setWORDAtOffset(si,7,48);
        coro.setWORDAtOffset(si,1,44);
    }
    coro.addArg(si);
    coro.addArg(new int[2]);
    int rc=coro.invoke();
    if(rc != 0) {
        coro.freeArg();
        return rc;
    }
    int [] ids=coro.intArrayFromParameterAt(9,2);
    id[0]=ids[0];
    id[1]=ids[1];
    return 0;
}
}
```



## Non-Windows Java client that monitors changes in remote Excel sheet

The following example implements DDE client that establishes a DDE link with Microsoft Excel while running on a non-Windows OS.

To test your client application:

- Start Microsoft Excel
- Start `com.neva.DdeRemoteClientFactoryHost` on the machine that runs Excel
- Start `DdeRemoteExcelLinkClient` application on remote machine. The single parameter specifies the host that runs Excel
- Modify Excel worksheet and watch changes printed by Java application

```

import java.util.*;
import com.neva.*;
import java.rmi.*;
import java.rmi.server.*;
public class DdeRemoteExcelLinkClient {
    public static void main(String args[]) {
        try {
            String host="localhost";
            if(args.length>0)
                host=args[0];
            //Locate DdeRemoteClientFactory
            com.neva.DdeRemoteClientFactory factory =
                (com.neva.DdeRemoteClientFactory)Naming.lookup("rmi://" + host + "/DdeRemoteClientFactory");
            //Get DdeRemoteClient
            com.neva.DdeRemoteClient cl = factory.getDdeRemoteClient();
            int timeout=5000;
            int format=com.neva.DdeUtil.CF_TEXT;
            int i,j;
            //Establish spreadsheet link
            cl.connect("Excel","System");
            //Create new worksheet
            cl.execute("[New(1)]\0",timeout);
            //Get worksheet name
            byte [] data=cl.request("Selection\0",format,timeout);
            String worksheet=new String(data);
            int ndx=worksheet.indexOf((int) '!');
            if(ndx>0) {
                worksheet=worksheet.substring(0,ndx);
            }
            cl.disconnect();
            //Establish link with new worksheet
            EventHandler eh=new EventHandler();
            cl.addDdeRemoteClientTransactionEventListener(eh);
            cl.connect("Excel",worksheet);
            //Put some values into worksheet
            for(j=1;j<=10;j++)
                for(i=1;i<=10;i++)
                    cl.poke("R"+j+"C"+i,new String(""+(i*j)).getBytes(),format,timeout);
            String hotitem="R1C1:R10C10";
            //Establish hot link on range of cells
            cl.startAdvise(hotitem,format,timeout);
            System.out.println("Advise link established! Go ahead and modify Excel worksheet");
            //Unfortunately, this application will terminate immediately hereafter.
            //We somehow need the thread to stay alive and wait for events.
            //Since we do not have a GUI, we can use the following hack for that purpose.
            Object keepAlive = new Object();
            synchronized(keepAlive) {
                try {
                    keepAlive.wait();
                } catch(InterruptedException e) {
                    // do nothing it will not happen
                }
            }
        } catch(java.rmi.RemoteException re) {
            if(re.getCause()!=null && re.getCause().getCause()!=null)
                re.getCause().getCause().printStackTrace();
            else
                re.printStackTrace();
            System.exit(0);
        } catch(Throwable e) {
            e.printStackTrace();
            System.exit(0);
        }
    }
}

class EventHandler extends com.neva.DdeRemoteClientTransactionEventAdaptor {
    public EventHandler() throws RemoteException {}
    public void onAdviseData(com.neva.DdeClientTransactionEvent e) throws RemoteException {
        String data=new String(e.getDdeData());
        System.out.println("onAdviseData: item="+e.getItem()+" data=\n"+data);
    }
}

```

```
}  
public void onDisconnect(com.neva.DdeClientTransactionEvent e) throws RemoteException {  
    System.out.println("Server has gone !");  
    System.exit(0);  
}  
}
```