



# **Cognitier SimpleIPC**

## **Client Code Sample - Java**

Version 1.0.0.1

April 21, 2009

Copyright © 2009 Cognitier, Inc. All rights reserved.

Cognitier and the Cognitier logo are trademarks of Cognitier, Inc. All other company and product names referred to may be trademarks of their respective owners. This documentation is copyrighted material and is intended exclusively for use by licensed users of Cognitier software. The information in this document is subject to change without notice.

## Client Code Sample - Java

The following sample client code is provided to demonstrate the use of the most common SimpleIPC client API calls in a Java programming environment. In this example, your application is the client, and it is written in Java. You want to invoke a custom server program you recently wrote to be hosted by SimpleIPC.

You might use the Java API from a standalone Java application or a Java web server. Your Java application would need to be running on Windows, because SimpleIPC only runs on the Windows operating system. In the following sample code, a SimpleIPC Java client object is instantiated. Thereafter, arguments are passed via the client object to a remote object running in an IPC server, and the remote object is instructed to run a custom piece of code that you, the software developer, would have previously written. You may have any of several reasons for not wanting the code in question to run in your client's process space. When the call returns, there is an array of output values that was prepared by the remote procedure call. The sample code invokes the remote routine several times in a loop for demonstration purposes. It also illustrates checking the return value from each call to see if there was a failure in the remote process. If so, the code obtains a new session (presumably with a new remote process) and resumes processing.

When compiling and running Java code for use with SimpleIPC, you must include the CTJObjects.jar file in your classpath, and you must include the SimpleIPC \bin directory in your library path. The SimpleIPC Java API is for use with Java 1.5 or higher.

```
import java.io.*;
import java.lang.*;
import com.cognitier.*;

class TestClient
{

TestClient()
{

}

    public static void main(String[] args)
    {
        try
        {
            //Instantiate the CTJClient object, which provides access to
make routine calls
```

```

        CTJClient oCTJClient = new CTJClient();

        //Instantiate the CTJSessionReturnStruct object, which bundles
the return values from the call to obtainSession
        CTJSessionReturnStruct oSessionRetStruct = new
CTJSessionReturnStruct();

        //Instantiate the CTJBasicCall1RetStruct object, which bundles
the return values from the call to basicRoutineCall1
        CTJBasicCall1RetStruct oCall1RetStruct = new
CTJBasicCall1RetStruct();

        //Declare variables
        boolean bRet = false;
        String sSessionID = "";
        String sInstanceName = "Inst1";
        String sAppName = "App1";
        String sRoutineName = "HelloWorld";
        int iNumCallsToMake = 3;
        int iCallCount = 0;
        String SESSIONINVALIDORTIMEOUT = "12001";
        String SERVERCOMMUNICATIONERR = "12004";
        String ERRRESUMINGSESSION = "12005";
        boolean bIncrementCallCount = true;
        int iElemCount = 0;

        //Initialize input arguments array
        String[] inputArray = null;
        inputArray = new String[3];
        inputArray[0] = "Sample Input 1";
        inputArray[1] = "Sample Input 2";
        inputArray[2] = "Sample Input 3";

        //Make the call to obtainSession
        bRet = oCTJClient.obtainSession(sInstanceName, sAppName,
oSessionRetStruct);
        if (bRet == false)
        {
            System.out.println("obtainSession return = " + bRet + " ;
rej code = " + oSessionRetStruct.RejectCode);
        }
        else

```

```

        {
            System.out.println("Session obtained; session id = " +
oSessionRetStruct.SessionID);
            //For demonstration purposes, make several routine calls
in a loop
            while (iCallCount < iNumCallsToMake)
            {
                bIncrementCallCount = true;
                //Make call to BasicRoutineCall1
                bRet =
oCTJClient.basicRoutineCall1(oSessionRetStruct.SessionID, sInstanceName,
sAppName, sRoutineName, inputArray, oCall1RetStruct);
                if (bRet == false)
                {
                    if (oCall1RetStruct.errorsVector.size() > 0)
                    {
                        if
(((String)oCall1RetStruct.errorsVector.elementAt(0)).equals(SESSIONINVALIDORTIM
EOUT) ||
((String)oCall1RetStruct.errorsVector.elementAt(0)).equals(SERVERCOMMUNICATIOE
RR) ||
((String)oCall1RetStruct.errorsVector.elementAt(0)).equals(ERRRESUMINGSESSION))
                        {
                            //The call to basicRoutineCall1
failed, but the error code indicates there might be a problem with the session,
so try to get a new session
                                oSessionRetStruct.RejectCode = 0;
                                oSessionRetStruct.SessionID = "";
                                bRet =
oCTJClient.obtainSession(sInstanceName, sAppName, oSessionRetStruct);
                                    if (bRet == false)
                                    {
                                        break;
                                    }
                                    else
                                    {
                                        bIncrementCallCount = false;
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

```

```

                break;
            }
        }
        else
        {
            iElemCount = 0;
            while (iElemCount <
oCalllRetStruct.outVector.size())
            {
                System.out.println("Call " + iCallCount +
"; Output Array Elem " + iElemCount + " = " +
(String)oCalllRetStruct.outVector.elementAt(iElemCount));
                iElemCount++;
            }

            iElemCount = 0;
            while (iElemCount <
oCalllRetStruct.errorsVector.size())
            {
                System.out.println("Call " + iCallCount +
"; Error Array Elem " + iElemCount + " = " +
(String)oCalllRetStruct.errorsVector.elementAt(iElemCount));
                iElemCount++;
            }

            iElemCount = 0;
            while (iElemCount <
oCalllRetStruct.warningsVector.size())
            {
                System.out.println("Call " + iCallCount +
"; Warning Array Elem " + iElemCount + " = " +
(String)oCalllRetStruct.warningsVector.elementAt(iElemCount));
                iElemCount++;
            }
        }

        //Increment the call count unless the previous call
failed and we had to get a new session
        if (bIncrementCallCount == true)
        {
            iCallCount++;
        }
    }
}

```

```
        }
        if (iCallCount < (iNumCallsToMake - 1))
        {
            System.out.println("Unable to complete all routine
calls");
        }

        //Release the session
        oCTJClient.releaseSession(oSessionRetStruct.SessionID);
    }

}
catch (Exception e)
{
    System.out.println("Exception: " + e.getMessage());
}

System.out.println("Execution complete");
}
}
```