# PC\*Miler|Connect Standalone JNI wrapper

## Introduction

Starting in PC\*Miler v28 we are now offering all of the source code to our existing Connect Java wrapper. Previously we had only provided the jar code, which is only half the code needed to truly manage and expand the PC\*Miler|Connect Java wrapper. We now also supply a standalone version of our C++ JNI code, the second piece to the puzzle.

The JNI layer was, and currently still is, directly integrated with our PC\*Miler|Connect dll, pcmsrv32.dll. Old versions of the jar file or even new versions built with existing java code will still interface directly with pcmsrv32.dll (or pcmsrv64.dll). It is not mandatory to use this new stand alone JNI layer. If you are satisfied with the current behavior then you can bypass the rest of this document.

There is now the option to build a separate JNI layer dll and interface with that instead. This is beneficial as it will allow users to expand the Java wrapper to gain access to any PC\*Miler|Connect API’s we do not currently expose through the Java wrapper. This document will cover the steps necessary to configure, build, and deploy everything necessary to do that.

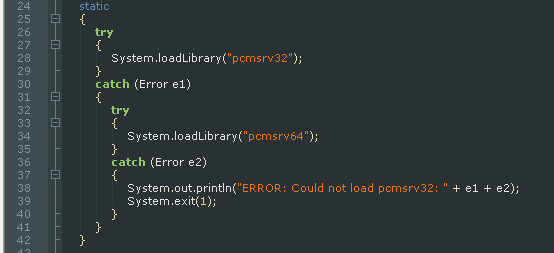
## The Code

There are two sets of code provided now. The native Java code used to build the traditional Java jar file and the C++ JNI layer used to create the JNI wrapper dll for our C-style API’s provided in pcmsrv32.dll. All of this code is architected using standard JNI practices.

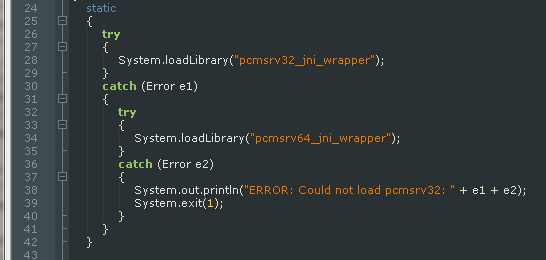
### Java Layer

To use the standalone JNI wrapper there is only one change needed in the Java code. Open up java/alk/connect/PCMSJava.java in an editor of your choice. Find the two System.loadLibrary() calls made at the top of the PCMSJava class. By default they reference the standard PC\*Miler|Connect dll’s. We simply need to redirect them to point to the new standalone JNI wrapper dll’s. The new names will be pcmsrv32\_jni\_wrapper and pcmsrv64\_jni\_wrapper. The screen shots below show the changes needed.

Old:



New:



Once these changes are made you can save and close PCMSJava.java. Navigate back to the Java.new root folder and run 01\_build\_jar.bat. The Java code will be recompiled and a new alk.jar will be put into the Java.new folder.

### C++ JNI Wrapper

The jni\_wrapper folder contains a MS Visual Studio 2005 project, and all the necessary source code used to build a standalone JNI wrapper dll for our PC\*Miler|Connect product. There are no code edits required to get it to work out of the box. You only need to edit the code if you want to expose new PC\*Miler|Connect API’s through the Java wrapper. If you are not doing that you can just run 02\_build\_jni\_wrapper.bat in the Java.new folder. Two dll’s will be produced into the Java.new folder, pcmsrv32\_jni\_wrapper.dll and pcmsrv64\_jni\_wrapper.dll.

## Deployment

Once the two components are built deployment is very simple. As long as the PC\*Miler|Connect dll is accessible via a load library call the jar and jni wrapper dll can be put anywhere on the system. To verify this you can build and run our included PC\*Miler|Connect Java sample application. First you need to run 03\_build\_test.bat and then 04\_run\_test.bat to launch the test application. The output of the test will be dumped to PCMSJavaTest\_output.txt.

If the test succeeded the output will contain a wide variety of API calls to calculate mileages, perform lookups, and run reports.