# Mini-report on redesigning the JNI layer

By Chronio (Patrick Kramer)

The current problem with the JNI layer is, that the way of collecting and storing Java elements (such as classes, methods and field) is often quite messy and there are quite a few issues regarding portability and maintainability. A solution can be implemented that has the form described in this mini-report.

The main issue of the current design of the JNI layer is, that it is still undefined as to how the different Java elements are cached. However, there are C++ classes to represent Java classes, methods and objects. The solution is simple yet effective: Since the JNI related classes eventually extend from QObject, the tree structure inherited through QObject allows for JNIMethod objects to be nested inside JNIClasses which, in turn, can be nested in the JavaEnv. This way, the JavaEnv can access the classes it needs (through the findChild and children functions) and the classes and methods can access the Java Environment through the parent function.

Another problem is, that the class and method names have been obfuscated, and as such it is difficult recognising and / or mapping these to the actual jclass and jmethodID values. The solution for this makes use of the abovementioned solution. Since we store the JNIClass objects as children of the JavaEnv object, it becomes possible to address these classes by name, due to QObjects having an instance-specific name. As such, each JNIClass can be assigned a semantically correct name, and each JNIMethod can, as well. Then it becomes possible to first access the correct class through the JavaEnv object, and then get the correct method through the resulting JNIClass object.

As can be seen, the mentioned issues can be solved using the functionality provided by Qt’s QObject class. As such, a change will have to be made in the design of the JNI module. This also means that milestone 0.2 will be a bit further off than expected.

Update

It has been decided that the updater required for identifying the classes of Runescape will be provided through a server. By doing this, it is possible to have the updater work once, and store the results for everyone to collect. Someone (who wishes to remain anonymus) has been found who is willing to set this up for PowerGrid. This allows entire Class structure to be built at startup (while the Runescape client is starting) and as such causes relatively faster startup times compared to other bot clients that need to perform injection before the client can be started. The structure of the JNIClass class itself does not change.