

JDT Tutorial: Class Loading in a running plugin

From Wissensbasis

Class loading may be problematic when a running Eclipse plugin wants to load a class from the workspace. The class will not be found unless the class loader of the Java project (where the class to load lives) is used.

We want here to define a class loader for our Eclipse plugin that uses the project class loader to load any given class.

Contents

- 1 Find the projects declared in the workspace
- 2 Which type of class loaders to use?
- 3 Retrieve the class loader of a Java project
- 4 All together
 - 4.1 Related Articles

Find the projects declared in the workspace

First, we retrieve all the projects that live in the workspace where our plugin runs:

```
List<IJavaProject> javaProjects = new ArrayList<IJavaProject>();
IProject[] projects = ResourcesPlugin.getWorkspace().getRoot().getProjects();
for(IProject project: projects){
    project.open(null /* IProgressMonitor */);
    IJavaProject javaProject = JavaCore.create(project);
    javaProjects.add(javaProject);
}
```

Which type of class loaders to use?

Since Eclipse plugins are referring to resources as URLs, it is convenient to use `URLClassLoaders` to load classes from a search path of URLs.

Retrieve the class loader of a Java project

We read the class path entries of the Java project. We create the an `URLClassLoader` whose parent is the class loader of the project and with the class entries of the project.

```
String[] classPathEntries = JavaRuntime.computeDefaultRuntimeClassPath(project);
```

```
List<URL> urlList = new ArrayList<URL>();
for (int i = 0; i < classPathEntries.length; i++) {
    String entry = classPathEntries[i];
    IPath path = new Path(entry);
    URL url = url = path.toFile().toURI().toURL();
    urlList.add(url);
}
```

```
ClassLoader parentClassLoader = project.getClass().getClassLoader();
URL[] urls = (URL[]) urlList.toArray(new URL[urlList.size()]);
URLClassLoader classLoader = new URLClassLoader(urls, parentClassLoader);
```

All together

We create a list containing the class loaders of all the projects in the workspace.

```
loaders = new ArrayList<URLClassLoader>();
for(IJavaProject javaProject: javaProjects){
    loaders.add(getProjectClassLoader(javaProject));
}
```

and we finally define the loadClass method:

```
for(URLClassLoader loader: loaders){
    try {
        Class<?> clazz = loader.loadClass(fullyQualifiedName);
        return clazz;
    } catch (ClassNotFoundException e) {
    }
}
```

Related Articles

- [Class Loading Problem with log4j and Eclipse](#)
- [Class Loading and Eclipse Plugins](#)

Keywords: *Eclipse, JDT, JDTJDT, Java, Class loading, Plugin, Tutorial*

Retrieved from "http://sdqweb.ipd.kit.edu/wiki/JDT_Tutorial:_Class_Loading_in_a_running_plugin"

Category: Tutorials and HowTos

