Ant plugin

1. Installing plugin for Ant

QJ-Pro integrates with Ant version 1.4 or higher. To use the Ant plug-in download `qjpro_Vx.x_.zip` and unzip the file into a directory of your choice.

To use the plugin for Ant in your build scripts you have to add the following taskdef to your `build.xml` file:

```
<property name="qjpro_home" value="c:\qjpro_V2.1\"/>
  <taskdef name="qjpro" classname="net.sourceforge.qjpro.integration.ant.QJProTask">
  <classpath>
    <pathelement location="${qjpro_home}/classes/qjpro.jar"/>
  </classpath>
</taskdef>
```

Note:

You only need to change the `qjpro_home` property.

2. Running QJ-Pro

Invoking QJ-Pro from within Ant is simple, only a few parameters are required. The following table displays the attributes which can be used for the Ant task:

Attribute	Required	Description
cfgFile	No	Location of the configuration file.
classpathref	No	Classpath reference, probably the same as the classpath which the Java compiler uses
sourcepathref	No	Sourcepath reference.
classpath	No	Classpath, probably the same as the classpath which the Java compiler uses

sourcepath	No	Sourcepath.
silent	No	Indicator for the QJ-Pro parser to use the silent mode. Possible values: "false", "true".
javaversion	No	Specify the source code level, possible values: 1.2, 1.3 and 1.4. Standard QJ-Pro uses the same source code level as specified by the default java compiler.
noclean	No	If this switch is set to true, the temporary files will not be removed from the filesystem.
traceon	No	Puts QJ-Pro in debug mode.
fileset	Yes	Specify which sources should be analyzed by QJ-Pro.
reporter	Yes	Settings for the report generator tool. Supported attribute: `outputdir`(required)

Table 1: Attributes which can be passed to QJ-Pro

The reporter property only supports the attribute `outputdir` at the moment, however when more different styles emerge this property can be extended in future releases. The `outputdir` is the place where QJ-Pro stores the results of an analysis run, this is also the place where the `.err` file will be stored. This file will tell you which errors where encountered during the analysis run.

In order to specify the source files or directories to be analyzed, you can use the nested filesets. The classpath and sourcepath references are the same as the ones used to invoke the java compiler.

3. Report

During a run a report will be generated which can be used to view the results. This report is currently available in HTML format, however the reports are generated with the aid of Velocity so you can easily tweak the layout/information of the generated report. As an example of a generated report click on the following link.

4. Example build file

This section describes a standard ant build file used to invoke QJ-Pro

```
<project name="qjpro" basedir="." default="test_qjpro">

<property name="qjpro_home" value="/home/nanneb/qjpro"/>
  <taskdef name="qjpro" classname="net.sourceforge.qjpro.integration.ant.QJProTask">
    <classpath>
      <pathelement location="${qjpro_home}/classes/qjpro.jar"/>
    </classpath>
  </taskdef>

  <path id="source.path">
    <pathelement
      path=" /home/nanneb/workspace/qjpro_HEAD/implementation/components/client/" />
  </path>

  <target name="test_qjpro">
    <qjpro cfgFile="${qjpro_home}/include/default.cfg"
      sourcepathref="source.path">
      <fileset dir="/home/nanneb/workspace/qjpro_HEAD/implementation/components/client/
        <include name="**/*.java" />
      </fileset>
      <reporter outputdir="output"/>
    </qjpro>
  </target>
</project>
```