Netbeans

Contents

1 Testing

1.1 New project

```
File > New Project > Java > Java Application
```

```
// TODO code application logic here
System.out.println("Hello World!");
```

```
shift-F11
```

1.2 Web Start

1.2.1 Enable Java Web Start

With Java Web Start, the user can launch a Java application by clicking an HTML link to a JNLP file for this application inside a web browser. The JNLP file, which is a special configuration file, instructs Java Web Start to download, cache, and run the Java application. To run applications with Java Web Start, it is enough to have a compatible version of the Java Runtime Environment (JRE) installed on the client machine. The installation of the Java Development Kit (JDK) is not required.

To enable your Java application to run with Java Web Start, you need to configure the properties of how the IDE should build the project. When Java Web Start is enabled in project properties, the IDE automatically creates a JNLP file and an HTML page with the link to the JNLP file, together with the JAR file. Configuring the Project to Enable Java Web Start

At first, we configure the project to make it Java Web Start enabled and test its execution locally.

1. Right-click the project node and choose Properties.

2. Under Categories, choose Web Start and select the Enable Web Start checkbox.

3. Leave the Local Execution option selected as the Codebase as we will first run the application locally.
4. The Codebase Preview field shows the path to local application files.
5. Ensure that the Self-signed checkbox is selected.
6. The application JAR file will be signed by a certificate that is generated automatically when the project is built. With the self-signed certificate, the application will be able to access the same resources from the computer as a regular application that is running locally. For example, the self-signed certificate allows an application to access local files and the network.Modifying the JNLP File
7. (Optional). In the Project Properties dialog box, select the Applications panel and change the application title and vendor name.
8. Click OK to close the Project Properties dialog box.

1.2.2 Modifying the JNLP File

To launch the application from the web, you need to provide a link to the applications source file on the web in the JNLP file.

1. Right-click the project node, choose Properties and select Web Start under Categories.
2. Choose User Defined as the Codebase.
3. In the Codebase Preview field, enter the URL where you will be uploading the source files. For example, `http://lpnp90.in2p3.fr/~roche/nectari/dist`.
4. Click OK in the Project Properties window.
5. Right-click the project node and choose Clean and Build.

1.2.3 Uploading the Source Files

Just copy the `/dist` directory to the web served directories.

```
$ rm -fr /htdocs/nectari/dist
$ cp -fr cvs/nectar/dist /htdocs/nectari/.
```

Modify the `codebase` in the *launch.jnlp* file and eventualy remove the *.jar* library that cannot be signed :

```
<?xml version="1.0" encoding="UTF-8"?>
<jnlp spec="1.0+" codebase="http://lpnp90.in2p3.fr/~roche/nectari/dist/" href="launch.jnlp">
    <information>
        <title>nectar</title>
        <vendor>Herman</vendor>
        <description>nectar</description>
        <description kind="short">nectar</description>
        <homepage href=""/>
        <icon href="shot0000.jpg" kind="default"/>
        <offline-allowed/>
    </information>
    <security>
        <all-permissions/>
    </security>
    <resources>
        <j2se version="1.5+" java-vm-args="-Djava.security.policy=applet.policy"/>
        <jar href="nectar.jar" main="true" download="eager"/>
```

```
        <jar href="lib/aida-3.3.jar" download="eager"/>
...
        <jar href="lib/xercesImpl-2.6.2.jar" download="eager"/>
    </resources>
    <application-desc main-class="nectar.Main">
    </application-desc>
</jnlp>
```

2 Configuration

2.1 Adding the sun JDK

- `Tools >>> Java Platforms >>> Add Platform...`
- Open to the */usr/java/jdk1.6.0_19/default* directory.
- Click the `Next >` button that was just enabled.
- Make the `Platform Sources` point to */usr/lib/jvm/java-6-sun-1.6.0.12/bin/*
- Click the `Finish` button.

2.2 Adding JDK's Javadoc

- test : clic on a word of the code and then do `ctrl-shift espace`. You should read : `Javadoc not found`
- Download JDK documentation from here
- In NetBeans IDE :
  1. Choose Tools > Java Platform Manager from the main window.
  2. Select the platform to which you want to add Javadoc in the left panel of the dialog box.
  3. In the Javadoc tab, click Add ZIP/Folder and specify the location of the Javadoc files.
  4. Click Close.

note: the javadoc of JAIDA is available as a `.jar` file in the */cvs/nectar/lib* directory.

2.3 Getting the sources and libraries from CVS

- Check out: In NetBeans IDE, choose `Team >>> CVS >>> Checkout` from the main menu. The Checkout wizard opens.
- enter this CVS Root: *:pserver:roche@lpnp90.in2p3.fr:2401/home/cvsroot*
- ask for the *nectar* module.
- uncompress the */NetBeansProjects/nectar/lib/jdom-1.1.1.tar.gz* lib files for instance.

2.4 Make the sun JDK the default one

- Right-click the project's root node in the Projects window and choose **Properties**.
- In the Project Properties dialog box, select the **Libraries** node in the left pane.
- Choose the desired Java platform in the Java Platform combo box.

2.5 Configuring libraries

1. From the ant subdirectory of the just-unpacked library, run `$ ant javadoc` to generate the Javadocs locally. You can skip this step, but then you'll be missing the API documentation.
2. In NetBeans, select the Library Manager item from the Tools menu
3. Click on the *New Library...* button and enter the library name NETBEANS prompt for.
4. With the Classpath tab selected, click on the *Add JAR/Folder...* button and select the *.jar* files from the *build* and the *lib* directories.
5. With the Sources tab selected, click on the *Add JAR/Folder...* button and select the source directory for this library.
6. With the Javadoc tab selected, click on the *Add ZIP/Folder...* button and select the javadoc directory for this library.

At this point, you have complete the configuration of the libraries and you should be able to compile.

- compile: Maj+F11
- run: F6

2.6 Configuring the running jvm with more

- Right-click the project's root node in the Projects window and choose **Properties**.
- In the Project Properties dialog box, select the **Run** node in the left pane.
- Insert the following options for initial and maximum memory size of the JVM into the **VM Option** box :

  ```
  -Xms512m -Xmx1024m
  ```

3 Best practices

3.1 Unitary tests

Each *.java* file contain a *static void main* function and is executable alone. You have to click on the file and then press `Maj+F6`.

Without NETBEANS you should do:

```
$ ls src/nectar/*.java
$ java -cp dist/nectar.jar nectar.ConfXml
```

3.2 Packaging

```
$ cd nectar
$ ant dist                        # compile software
$ java -jar dist/nectar.jar       # test software
$ tar -zcf nectar.tgz dist/       # archive software
```

Comment: so as to add the XML configuration file you will have to add the folowing target to the *build.xml* file:

```
<target name="-post-jar">
    <copy file="nectar.xml" flatten="true" todir="${dist.dir}"/>
</target>
```