Jar

Contents

1 Introduction

The JAR files (Java ARchive) allow to archive files in a JAVA specific format which is indeed portable. The JAR files are in fact just some ZIP files renamed (we can `unzip` them). The advantage of JAR archives is that tey can be run like `.exe` files under WINDOWS.

2 Build from scratch

- File *MANIFEST.MF*:

```
Main-Class: ClassePrincipale
Class-Path: \Archives\liquidlnf.jar // Le fichier JAR contenant le Look \& Feel Liquid
```

- Command:

```
$ jar cvmf MANIFEST.MF Programme.jar *.class
$ unzip Programme.jar
```

3 Netbeans

3.1 Including libraries

If we enable the WEBSTART option, the `.jar` libraries files will load even if they are not included into the project JAR file.
To do it, you must right click on the project node and choose "proporties" and then "web start".

3.2 Including ressources

In order to include the `etc/` directory (and the `nectar.xml` configaration file) into the sources. To do it, we have to right click on the project node and choose "proporties" and then "sources" an add the `etc/` directory as a "source package folder". **Note** that the `nectar.xml` is not prefixed by the project name nor by `etc/`.

```
try {
    // use the user configuration file (local copy)
    doc = builder.build(new FileInputStream("etc/" + Main.confFile));
    UI_Log.logger.info("LOCAL configuration file loaded");

} catch (Exception f) {
    UI_Log.logger.info("Cannot load local configuration file... try to look into the JAR");
```

```
    // use the default configuration file (into the jar)
    ClassLoader cl = Thread.currentThread().getContextClassLoader();
    InputStream jarFile = cl.getResourceAsStream("nectar.xml");
    if (jarFile == null) {
        UI_Log.logger.severe("Cannot load default configuration from JAR file");
        return;
    }
    else {
        UI_Log.logger.info("DEFAULT configuration file loaded");
        doc = builder.build(jarFile);
    }
}
```