

JAVA ET LE WEB

Fayçal BRAÏKI
etudiantssp@free.fr

Licence Devops
JEE
2018/2019

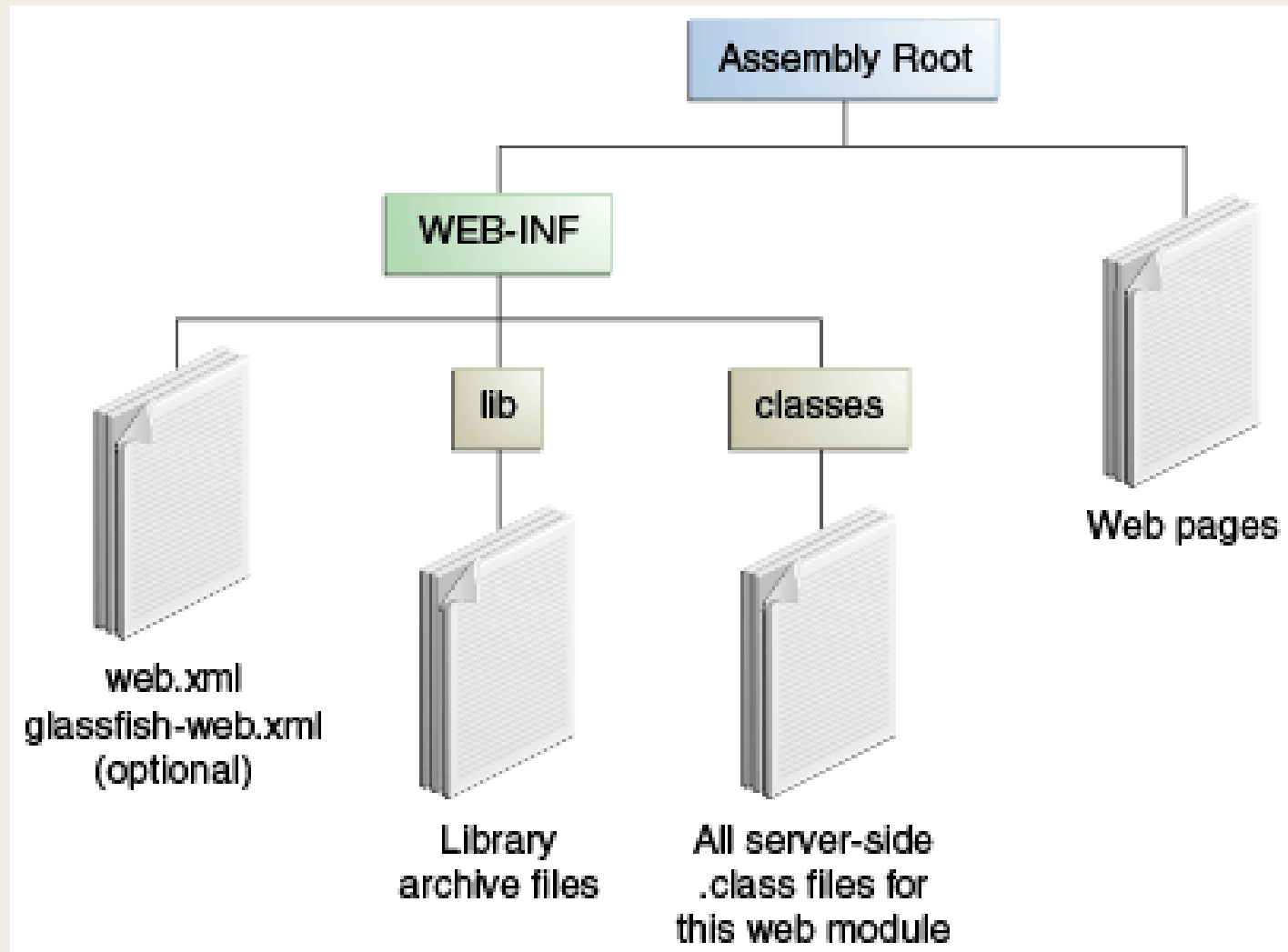
Sommaire

- I. Structure d'une application web
- II. Le fichier de configuration web.xml
- III. Portabilité des applications web

Structure d'une application web

- La structure (organisation des dossiers, nommage des fichiers de configuration,...) obéit à la spécification JEE relative aux servlets :
<http://download.oracle.com/otndocs/jcp/servlet-4-final-eval-spec/index.html>
- Actuellement, nous sommes à la v8 de la spécification JEE disponible ici :
<https://jcp.org/aboutJava/communityprocess/final/jsr366/index.html>

Structure d'une application web



Structure d'une application web

- La racine de l'application (assembly root) contient :
 - *Les pages web (HTML, JSP) de l'application*
 - *Un répertoire nommé WEB-INF*
- Ce dernier contient :
 - *Un fichier de configuration (optionnel depuis JAVA 7) servant au déploiement de l'application nommé web.xml*
 - *Un répertoire contenant servlets et classes utilitaires nommé classes*
 - *Un répertoire contenant les éventuelles archives java (fichiers jar) nommé lib*

Le fichier de configuration web.xml

- Le fichier web.xml permet de définir les paramètres de déploiement de l'application web et des servlets
- Il est optionnel depuis java 7 mais reste néanmoins très utile
- ➔ il n'est donc plus généré automatiquement (en particulier dans netbeans)
- Il contient les principales balises suivantes :
 - *Web-app* : sert à la description de l'application
 - *Servlet* : sert à la description et au paramétrage d'une servlet
 - *Servlet-mapping* : sert à gérer le lien entre la servlet et son (ses) url(s) d'accès
- Syntaxe :
 - `<web-app>....</web-app>`
- Il n'y a pas d'obligation à implémenter toutes les balises

Le fichier de configuration web.xml

- La balise `<servlet>` `</servlet>`
 - Les paramètres disponibles :
 - `<icon>test.gif</icon>`: icône associée (optionnel)
 - `<servlet-name>Test</servlet-name>`: utilisé comme référence pour la définition de la servlet
 - `<display-name>Servlet de test</display-name>`: nom d'affichage de la servlet (optionnel)
 - `<description>Ma Description</description>`: description de la servlet (optionnel)
 - `<servlet-class>servlet.TestServlet</servlet-class>`: classe d'implémentation de la servlet
 - `<jsp-file>JSP/test.jsp</jsp-file>`: éventuel fichier JSP associé
 - `<init-param>`: paramètre(s) d'initialisation
 - `<security-role-ref>`: autorisations sur le servlet

Le fichier de configuration web.xml

- La balise `<servlet-mapping>` `</servlet-mapping>`
 - *Elle permet de faire le lien entre une servlet et son(es) url(s)*
 - *Les paramètres disponibles :*
 - `<servlet-name>xxxxxxx</servlet-name>` : *nom de la servlet*
 - `<url-pattern>/xxxxxxx/*</url-pattern>` : *url d'accès à partir de l'emplacement de la servlet*

Le fichier de configuration web.xml

Exemple :

```
<servlet-mapping>  
    <servlet-name>NewServlet</servlet-name>  
    <url-pattern>/NewServlet/*</url-pattern>  
</servlet-mapping>
```

Toutes les urls positionnées après le pattern exécuteront la servlet

- Exemple de fichier web.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<web-app >
```

```
  <servlet-mapping>
```

```
    <servlet-name>NewServlet</servlet-name>
```

```
    <url-pattern>/NewServlet/*</url-pattern>
```

```
  </servlet-mapping>
```

```
<display-name>Ma servlet FB</display-name>
```

```
  <servlet>
```

```
    <description>Ma Servlet FB</description>
```

```
    <servlet-name>NewServlet</servlet-name>
```

```
    <servlet-class>NewServlet</servlet-class>
```

```
    <init-param>
```

```
      <param-name>valeur</param-name>
```

```
      <param-value>12</param-value>
```

```
    </init-param>
```

```
  </servlet>
```

```
</web-app>
```

Le fichier de configuration web.xml

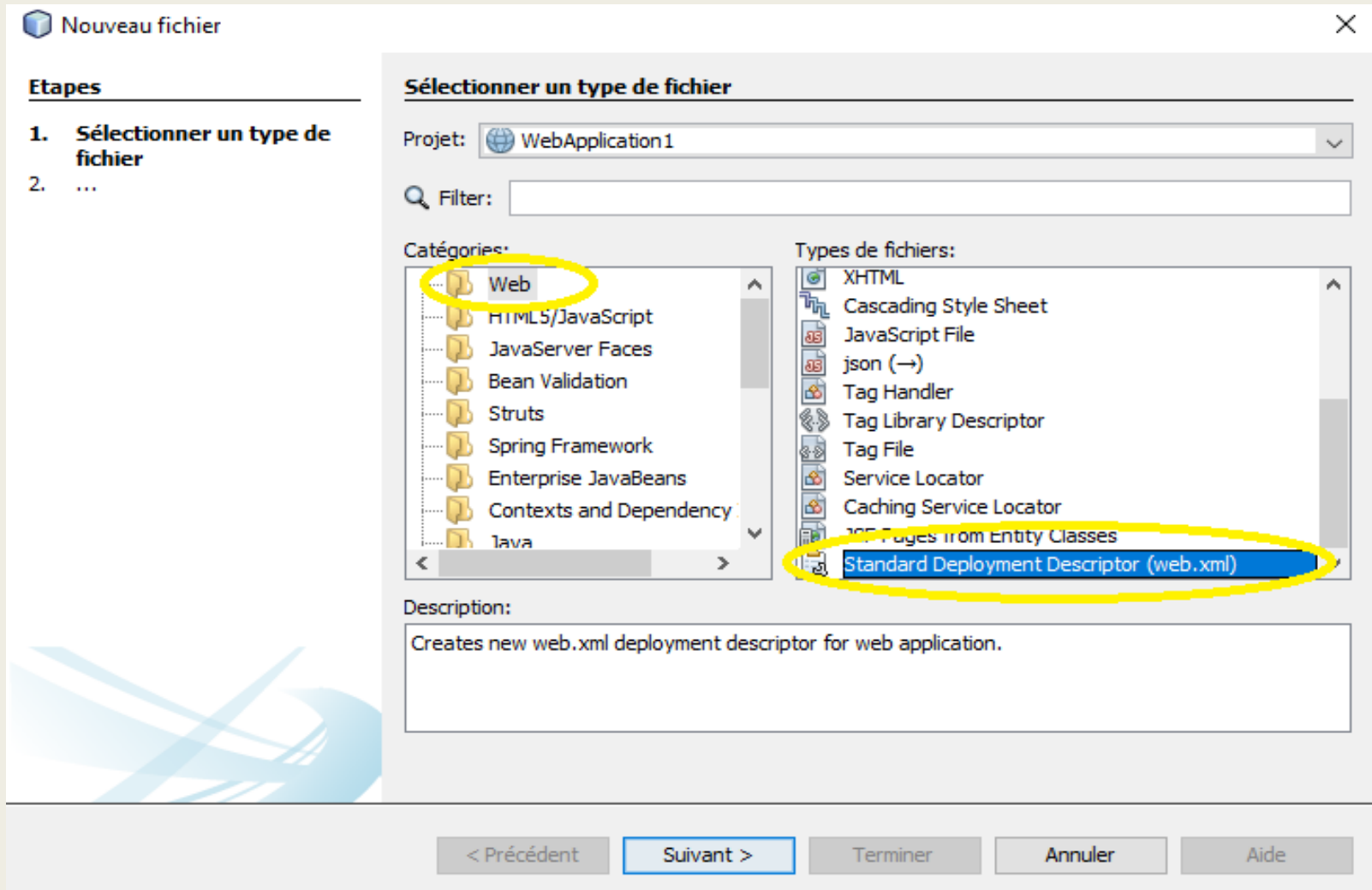
Remarques :

- Les paramètres définis par la balise sont accessibles par les méthodes `ServletConfig.getInitParameterNames()` et `ServletConfig.getInitParameter(String nom_paramètre)` de l'interface `ServletConfig` implémentée par la classe `HttpServlet`
- Autres balises intéressantes du fichier web.xml :
 - `<welcome-file-list></welcome-file-list>` et son paramètre `<welcome-file>nom_fichier</welcome-file>` : permet de spécifier le fichier de démarrage de l'application
 - À défaut de fichier de démarrage spécifié dans le fichier web.xml, le serveur cherche, dans l'ordre, un fichier nommé :
 - 1- index.html 2-index.htm 3-index.jsp
 - `<session-config> </session-config>` et le paramètre `<session-timeout> durée_en_minutes_de_la_session</session-timeout>`
 - pour d'autres paramètres :

https://docs.oracle.com/cd/E13222_01/wls/docs81/webapp/web_xml.html#1015060

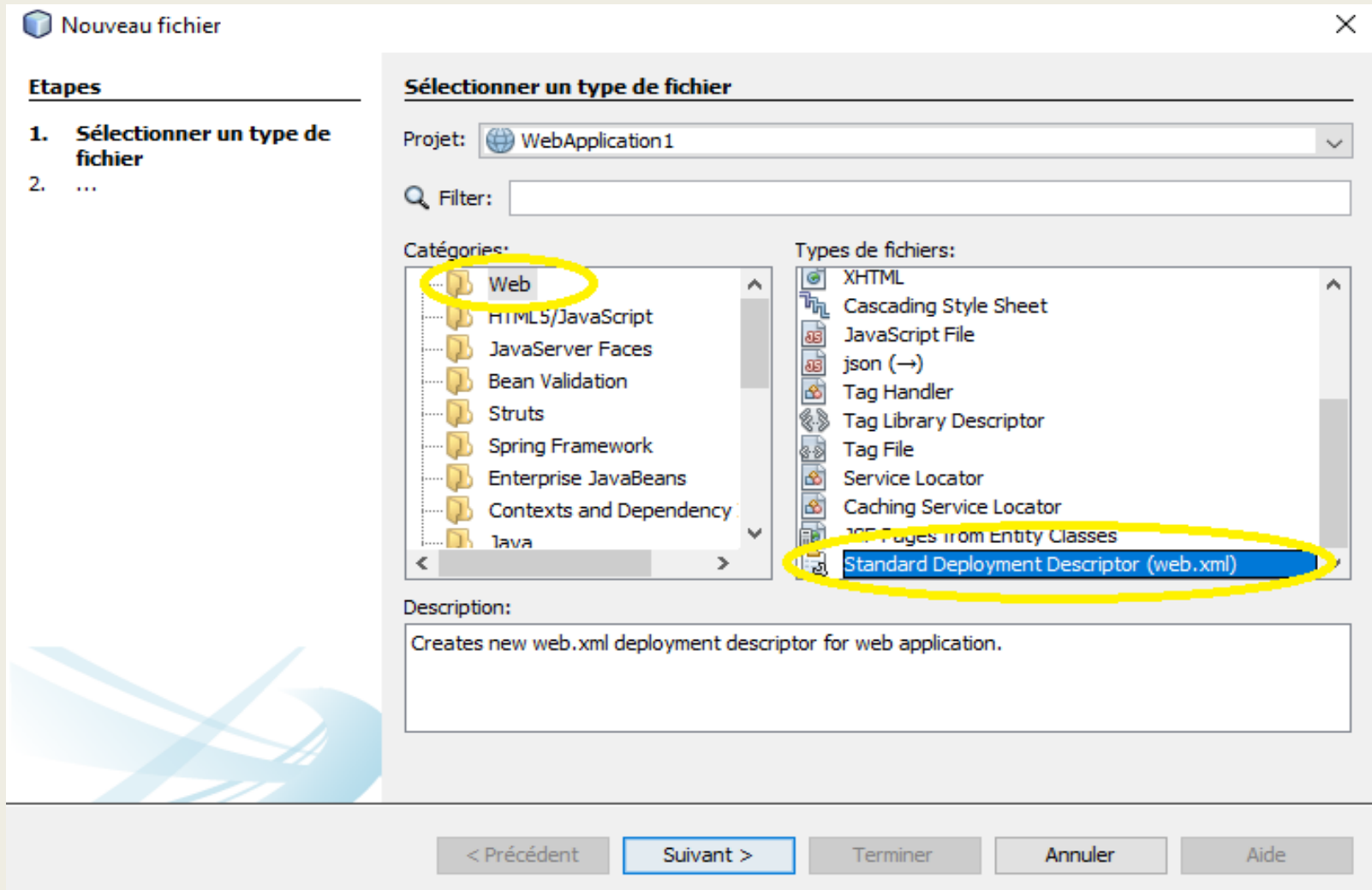
Le fichier de configuration web.xml

- Génération d'un fichier web.xml avec netbeans 8.1



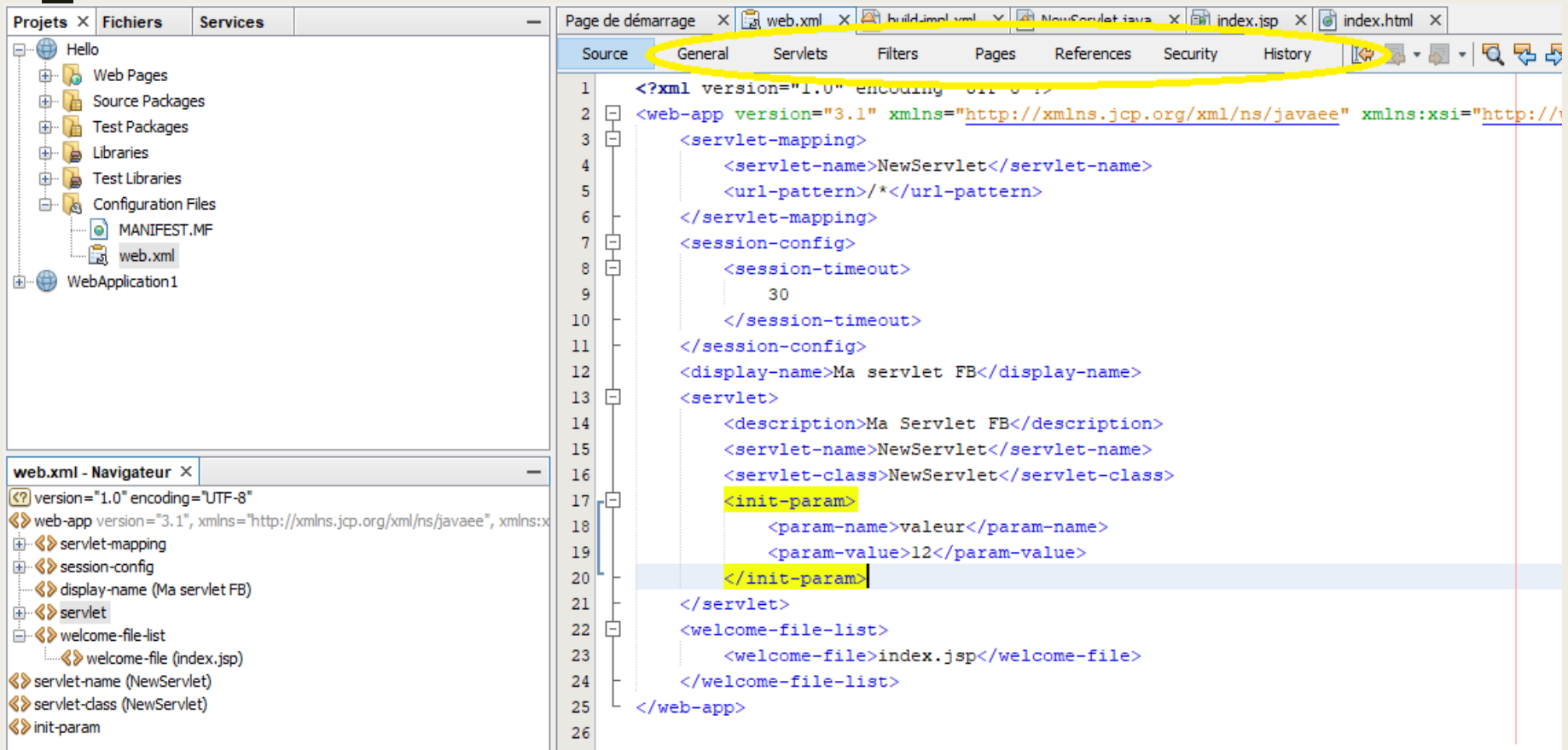
Le fichier de configuration web.xml

- Génération d'un fichier web.xml avec netbeans 8.1



Le fichier de configuration web.xml

- Génération d'un fichier web.xml avec netbeans 8.1
 - *Le fichier est alors éditable dans un traitement de texte, via l'EDI ou encore au travers des différents menus*

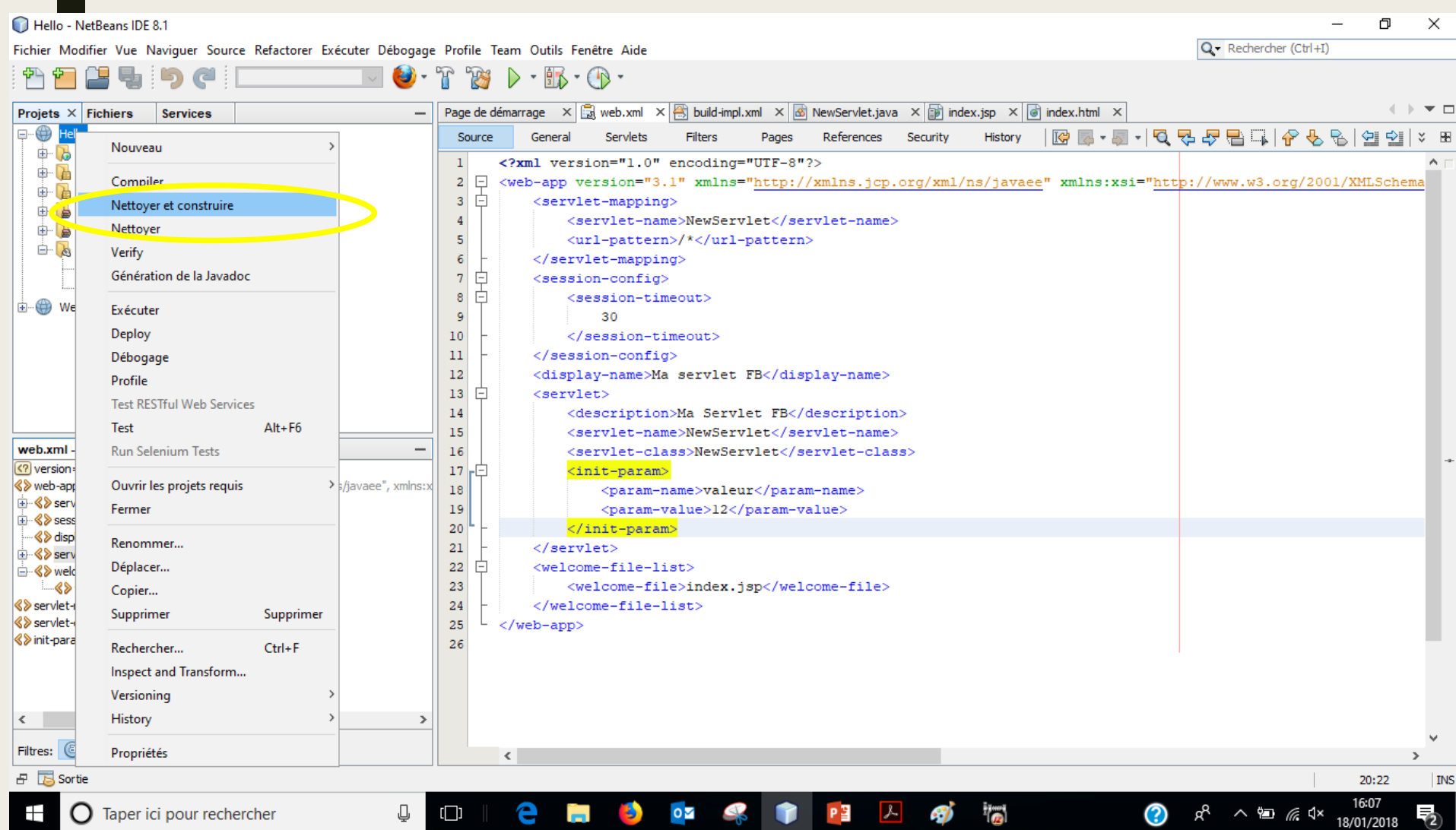


Portabilité des applications web

- Nous avons précédemment vu la structure d'une application web
- Cette structure se veut indépendante du serveur d'application l'implémentant
- Le format de portabilité des applications est .war (Web Application Archive)
- Une fois archivée, l'application est compressée sous la forme d'un fichier *nom_application.war* dans lequel on retrouve la structure de l'application à laquelle s'ajoute un répertoire nommé *meta-inf*
- Ce fichier permet d'assurer la portabilité de l'application web d'un serveur d'application à un autre
- Une fois placé dans le répertoire *web-app* d'un serveur d'application, le fichier est décompressé et prêt à l'emploi

Portabilité des applications web

- Génération d'un fichier war avec netbeans 8.1

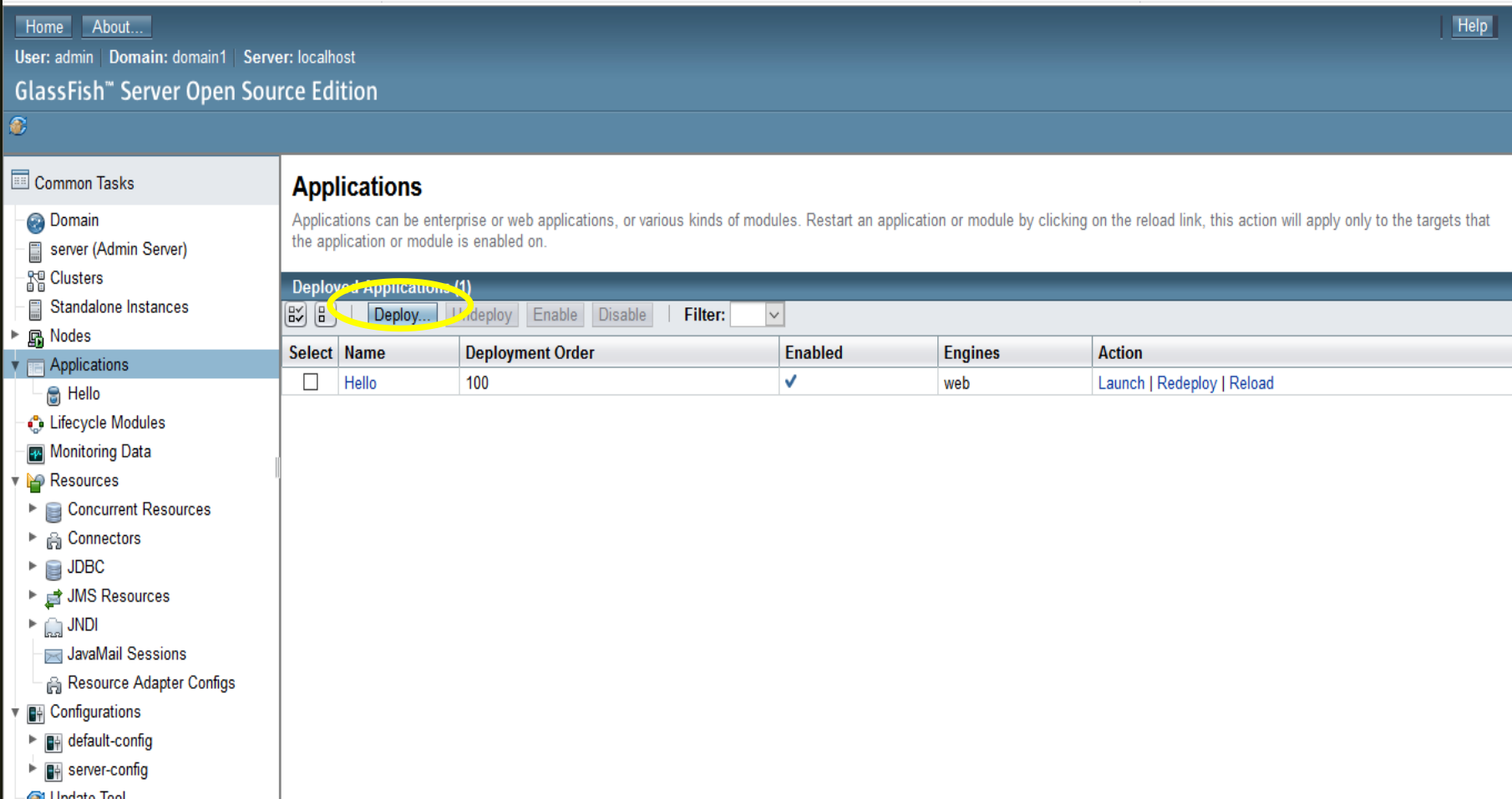


Portabilité des applications web

- Génération d'un fichier war avec netbeans 8.1
 - *Le fichier war est ensuite généré dans le répertoire dist de l'application*
 - *Il est prêt à être importé dans un serveur d'application*

Portabilité des applications web

- Exemple d'importation d'un fichier .war sous glassfish 4.1
 - *La console d'administration de glassfish est accessible directement à partir de netbeans ou à l'url <http://localhost:4848>*



Home About... Help

User: admin Domain: domain1 Server: localhost

GlassFish™ Server Open Source Edition

Common Tasks

- Domain
 - server (Admin Server)
 - Clusters
 - Standalone Instances
 - Nodes
 - Applications**
 - Hello
 - Lifecycle Modules
 - Monitoring Data
 - Resources
 - Concurrent Resources
 - Connectors
 - JDBC
 - JMS Resources
 - JNDI
 - JavaMail Sessions
 - Resource Adapter Configs
 - Configurations
 - default-config
 - server-config

Update Tool

Applications

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

Deployed Applications (1)

☒ ☐ **Deploy...** Undeploy Enable Disable Filter:

Select	Name	Deployment Order	Enabled	Engines	Action
<input type="checkbox"/>	Hello	100	✓	web	Launch Redeploy Reload