

TP n°1

Objectifs

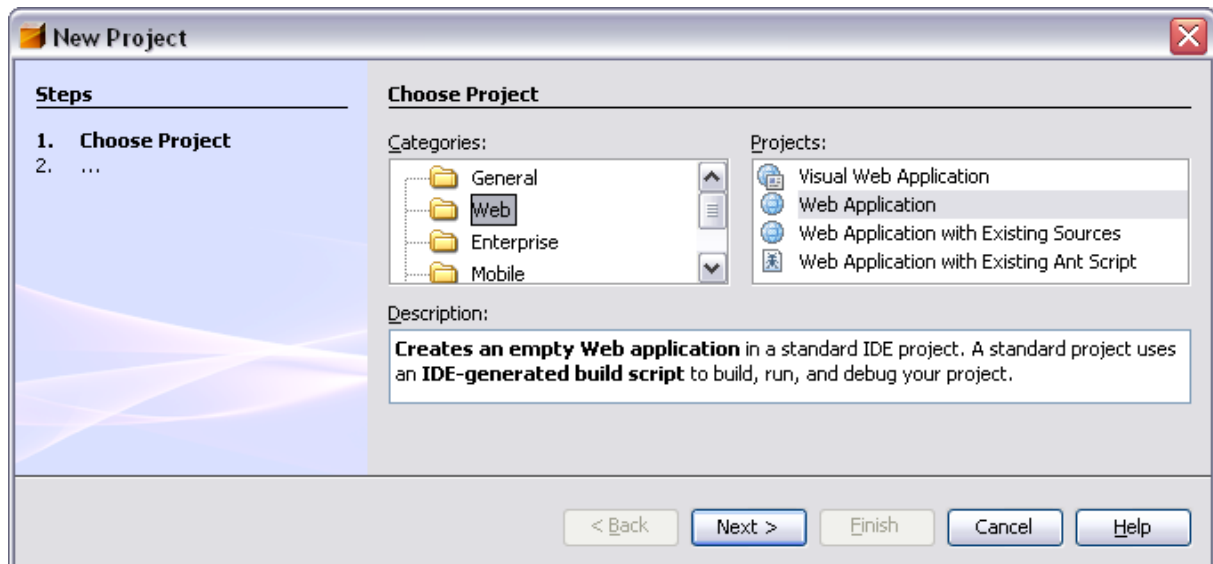
- Introduction à l'IDE netbeans
- Création d'un formulaire HTML
- Création d'une servlet
- Déploiement d'une application web

Outils utilisés

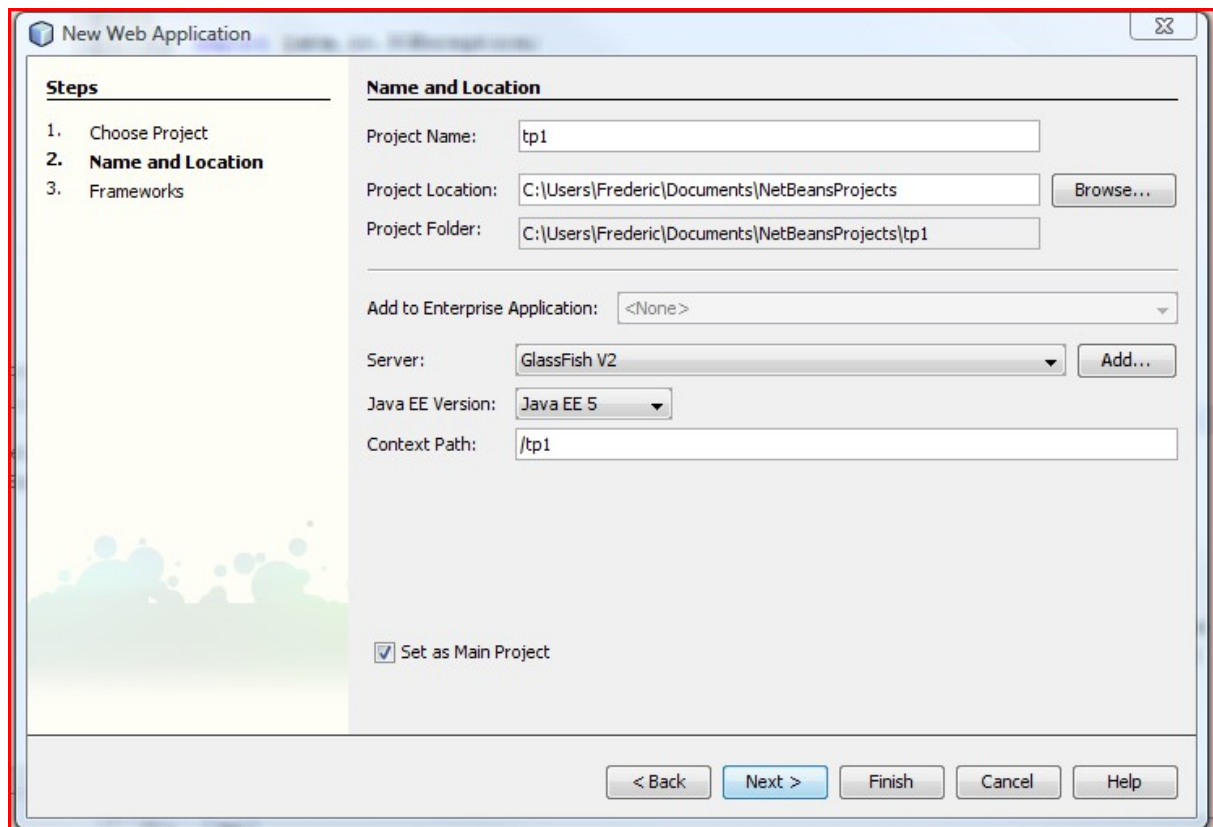
- La plate-forme netbeans intègre un serveur web
- Le serveur web est automatiquement lancé à l'exécution de l'application web ainsi que le navigateur

Construire une application web

- choisir `New Project` dans le menu `File`.
- puis



- donner un nom au projet (ici `tp1`)
- choisir un répertoire d'accueil pour les projets



Toute application web (ici tp1) est construite selon l'arborescence suivante :

exemple

```

Web Pages
  META-INF
    context.html
  WEB-INF
    web.xml
    autres ressources (images, html, jsp,...)
Configuration files
Source packages

```

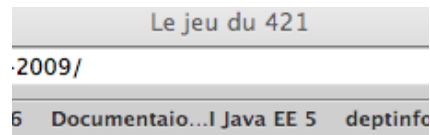
Pour structurer notre application, on créera un package dans lequel on placera la servlet :

- clic droit sur le dossier `Source packages` du projet, puis (`New` → `package`). On donne un nom à ce package (par exemple `servlets`).

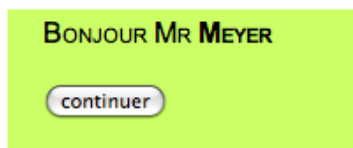
Spécification de l'application web

On se propose de créer une servlet `Authentication` dont le rôle est :

- d'afficher un formulaire (le style du modèle est laissé à votre initiative)
- de récupérer la valeur saisie par l'utilisateur
- de traiter cette information en affichant une nouvelle page web résultat



le formulaire



le résultat

Question 1

Ecrire une page HTML (`authentication.html`) qui affiche le formulaire ci-dessus.

Rappels :

Balise HTML `form` :

- permet de créer un formulaire
- elle comporte 3 attributs :
 - **action** spécifie la destination des infos à envoyer
`<form action="mailto:manager@cnam.fr"></form>`
 - **action** spécifie la destination des infos à envoyer
`<form action="http://deptinfo.cnam.fr/page.jsp"></form>`
 - **action** spécifie la destination des infos à envoyer
`<form action=""></form>` les infos sont reçues sur la page elle-même

- **method** : définit la méthode d'envoi (*post* ou *get*)
- **enctype** : précise la codification souhaitée pour l'envoi de l'information
enctype="text/plain" pour un envoi par courrier électronique

Balise HTML `input` :

- **value** : correspond au texte à afficher sur le bouton
- **type** : *submit* pour envoyer, *reset* pour rétablir la valeur saisie, *text* pour définir une zone de texte et *size* sa taille
- **name** : nom du paramètre auquel est associé la valeur qui sera réceptionnée.

Question 2

Ecrire la servlet `Authentification` qui affiche le même formulaire en vous aidant du code HTML précédent et du squelette à compléter qui suit.

Note : Pour créer la servlet sous netbeans, faites un clic droite sur " Sources Packages -> package `servlets` ", donnez le nom `Authentification` au fichier et enfin cliquez sur `finish`.

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class Authentification extends HttpServlet{

    public void doGet
        (HttpServletRequest request, HttpServletResponse response)
            throws IOException, ServletException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        // on affiche le formulaire
        .....
    }

    public void doPost
        (HttpServletRequest request, HttpServletResponse response)
            throws IOException, ServletException {
        // on passe la main au GET
        doGet(request, response);
    }
}
```

Question 3

Avant d'exécuter cette servlet, il faut la déployer sur le serveur. Pour cela, il faut modifier le fichier descripteur de déploiement `web.xml`.

Note : ouvrir le fichier `web.xml`, puis cliquer sur l'onglet `xml` pour accéder à son code source

Le fichier `web.xml` est, comme son nom l'indique un fichier codé en XML.

2 balises permettent de décrire le déploiement de la servlet :

```

<!--cette balise associe un nom interne à la classe qui lui correspond-->
    <servlet>
        <servlet-name>Authentification</servlet-name>
        <servlet-class>servlets.Authentification</servlet-class>
    </servlet>
<!--cette balise associe au nom interne une URL -->
    <servlet-mapping>
        <servlet-name>Authentification</servlet-name>
        <url-pattern>/Authentification</url-pattern>
    </servlet-mapping>

```

Question 4

On souhaite compléter la page du formulaire en affichant un message de bienvenue sur la page :
"Bienvenue sur cet espace".

Pour que le texte de ce message soit facilement accessible et modifiable, on le placera en paramètre d'initialisation de la servlet `Authentification`.

Pour réaliser cela, il suffit :

- de modifier le fichier de déploiement `web.xml` en ajoutant ce paramètre d'initialisation à la servlet


```

<init-param>
    <param-name>message</param-name>
    <param-value>Bienvenue sur cet espace</param-value>
</init-param>

```
- d'ajouter à la servlet la méthode `void init()` qui le récupérera
- de modifier la méthode `doGet` pour son affichage.

Que se passe-t-il ?

Lorsqu'une servlet est appelée pour la 1ère fois, sa méthode `init` est appelée. C'est le seul cas où elle est appelée.

Si la servlet a été appelée par la méthode HTTP GET, la méthode `doGet` est appelée pour traiter la requête du client.

Si la servlet a été appelée par la méthode HTTP POST, la méthode `doPost` est appelée pour traiter la requête du client.

La méthode `init` sert ici à récupérer des valeurs par défaut. Ce sont des valeurs de configuration de la servlet qui ne changent pas au fil des cycles requête-réponse.

Question 5

On veut maintenant récupérer le nom saisi dans le formulaire et l'afficher dans une nouvelle page (cf page resultat ci-dessus).

Pour cela, on modifiera la méthode `doPost`.

Rappel :

La méthode `doGet` reçoit deux paramètres `request` et `response`.

`request` est un objet de type `HttpServletRequest` représentant la requête du client.

`response` est un objet de type `HttpServletResponse` représentant la réponse à envoyer au client.

`request.getParameter("nom")` est le message permettant de récupérer la valeur du paramètre `nom` dans la requête du client.

`response.getWriter()` est le message qui permet d'obtenir un flot (stream) d'écriture vers le client.

`response.setContentType(String)` établit la valeur de l'entête HTTP Content-type indiquant la nature du document qu'il va recevoir. Le type `text/html` indique un document HTML.

Question 6

Modifier le code précédent de manière à réaliser l'affichage du nom saisi en gras par une méthode ajoutée à la servlet.

Question 7

Dans le fichier `web.xml` :

- modifier le mapping url de la servlet en "accueil". Tester le bon fonctionnement en entrant l'URL : <http://localhost:PORT/PROJET/accueil>

avec PORT: le port de votre serveur web
et PROJET: le nom de votre projet Netbeans

- On souhaite lancer l'application par l'URL suivante:

<http://localhost:PORT/PROJET>

Comment faire ?