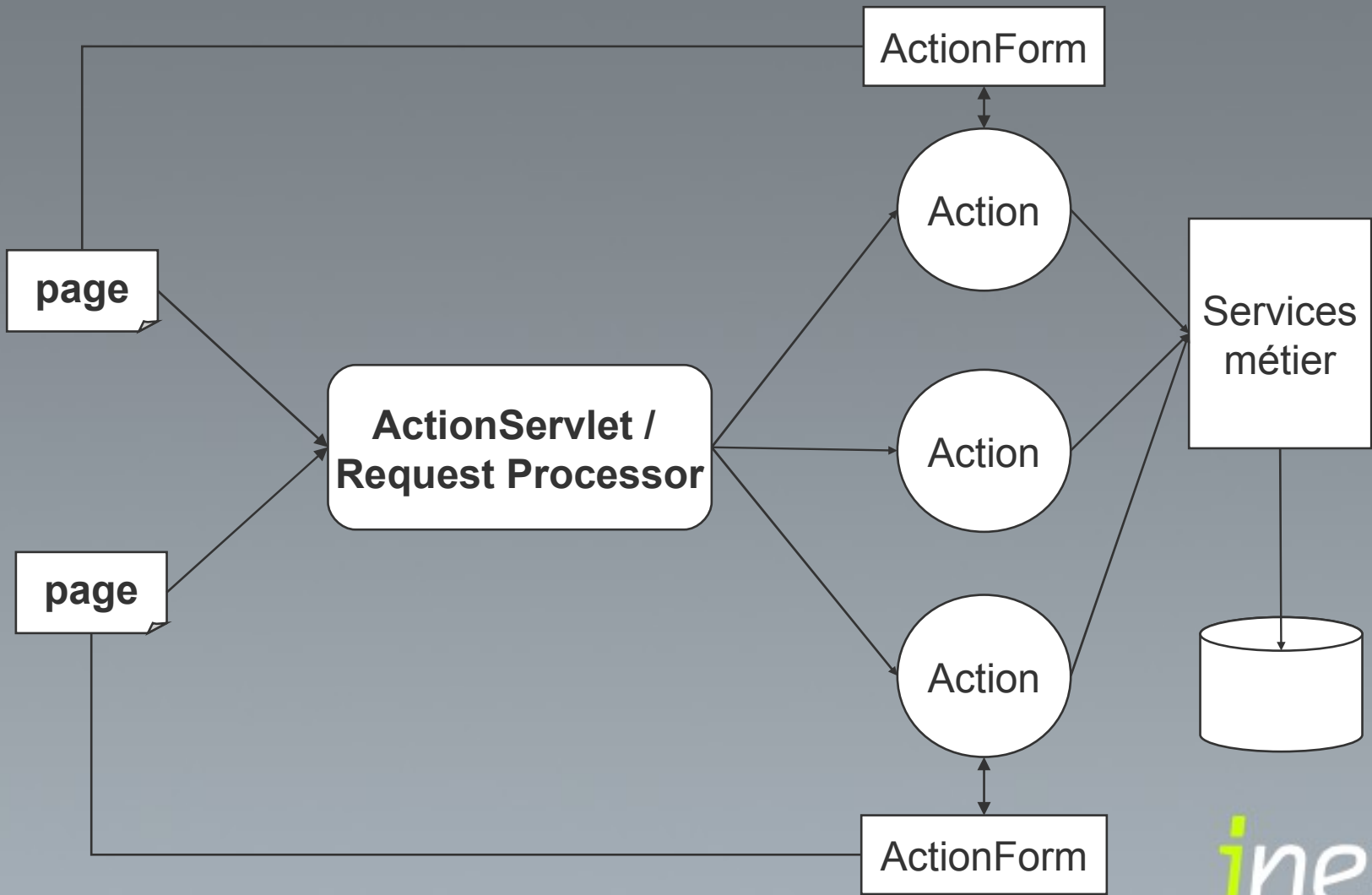


# ActionForm

- **Struts dispose d'une fonction d'encapsulation des formulaires permettant**
  - Le remplissage automatique de Bean à partir des paramètres de la requête.
    - Transformation chaîne -> int, float, etc.
  - Le remplissage automatique de <Form> d'une JSP à partir de Bean (taglib spécifique)
- **L'objectif est bien évidemment de pouvoir manipuler cette ActionForm au sein d'une Action**

# ActionForm



# ActionForm

- La classe abstraite ActionForm est fournie par le framework.
  - *org.apache.struts.action.ActionForm*
- Elle encapsule les informations (champs de saisie) d'un formulaire HTML.
- Elle respecte la convention d'écriture des JavaBeans.
- Elle est généralement couplée avec l'usage de taglibs permettant leur manipulation dans les JSP (Taglib HTML / Bean)
- Permettra on le verra la validation des données (gestion des erreurs)

# ActionForm

## ■ Deux types

- Défini par le programmeur

- hérite de *org.apache.struts.action.ActionForm*
- Elle doit respecter la convention d'écriture des JavaBeans

```
<form-bean name="loginform"
           type="com.exemple.LoginForm">
</form-bean>
```

```
package com.exemple;
public class LoginForm extends ActionForm {
    private String login;
    .....
}
```

- Dynamique

- Hérite de *org.apache.struts.action.DynaActionForm*
- Elle est configurée dans le `struts-config.xml`

```
<form-bean name="loginform" dynamic="true" type="org.apache.struts.action.DynaActionForm">
  <form-property name="login" type="java.lang.String"/>
  <form-property name="password" type="java.lang.String"/>
</form-bean>
```

# ActionForm

## ■ Les getters et setters des propriétés

- servent à accéder aux informations à afficher / saisir.
- des traitements de format peuvent y être effectués

## ■ reset()

- Appelé avant que les attributs de l'ActionForm soient définis en fonction des données d'une requête.
- Permet de réinitialiser les variables d'instance.

## ■ validate()

- Permet de valider les données du formulaire, nous verrons cela plus loin dans le cours

# ActionForm (Exemple)

```
<form name="user" method="post" action="login.do">
  <label for="Login">Login :</label>
  <input type="text" name="login" value=""><br>
  <label for="Password">Password :</label>
  <input type="text" name="password" value=""><br><br>
  <label for="role">Role :</label>
  <select name="role">
    <option value="1">Utilisateur</option>
    <option value="2">Administrateur</option>
  </select><br>
  <input type="submit" value="Valider">
</form>
```

## Connexion

Donnez vos Login / mot de passe et rôle

Connexion

Login :

Password :

Role :

```
public class LoginForm
    extends ActionForm {

  private String login;
  private String password;
  private int role;

  public String getLogin() {
    return login;
  }
  .....
}
```

# ActionForm (Example)

```
public class LoginAction  
extends Action {
```

```
public ActionForward  
execute(  
ActionMapping mapping,  
ActionForm form,  
HttpServletRequest  
request,  
HttpServletResponse  
response) {
```

```
    LoginForm f =  
    (LoginForm) form;  
    if (check(f))  
    return  
    mapping.findForward("suc  
    cess");  
    else  
    return  
    mapping.findForward("fail  
    ure");  
}
```

```
<struts-config>  
  
  <form-beans>  
    <form-bean name="user"  
               type="com.example.LoginForm">  
    </form-bean>  
  </form-beans>  
  
  <action-mappings>  
    <action path="/login"  
            type="com.example.LoginAction"  
            scope="request"  
            name="user">  
      <forward name="success"  
                path="/welcome.jsp" />  
      <forward name="failure"  
                path="/error.jsp"/>  
    </action>  
  </action-mappings>  
  
</struts-config>
```

```
public class LoginForm  
extends ActionForm {
```

```
private String login;  
private String password;  
private int role;
```

```
public String getLogin() {  
return login;  
}
```

```
.....
```

```
<form name="user"  
method="post"  
action="/login.do">  
...  
</form>
```

# ActionForm

- On remarque dans l'exemple précédent que l'utilisation d'un Taglib pour la saisie de formulaire n'est pour l'instant pas indispensable.
- Assez étrangement, pour indiquer l'identifiant du formulaire dans la configuration de l'action, on utilise un attribut « name ». Contrairement à ce que l'on pourrait penser, « name » ne représente donc pas un nom que l'on pourrait donner à l'action mais le nom du formulaire associé à cette action.