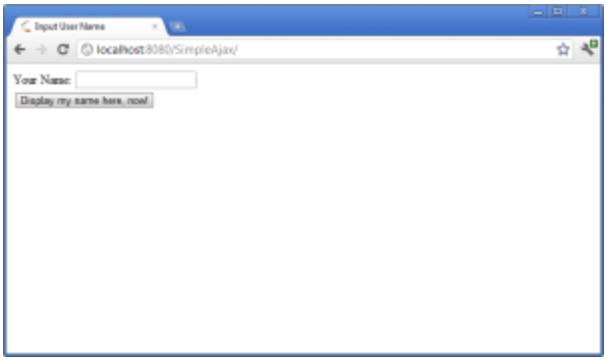
Exercice11 : Ajax et JSF

Cet exercice JSF est sur les f :ajax.

Etape1 :

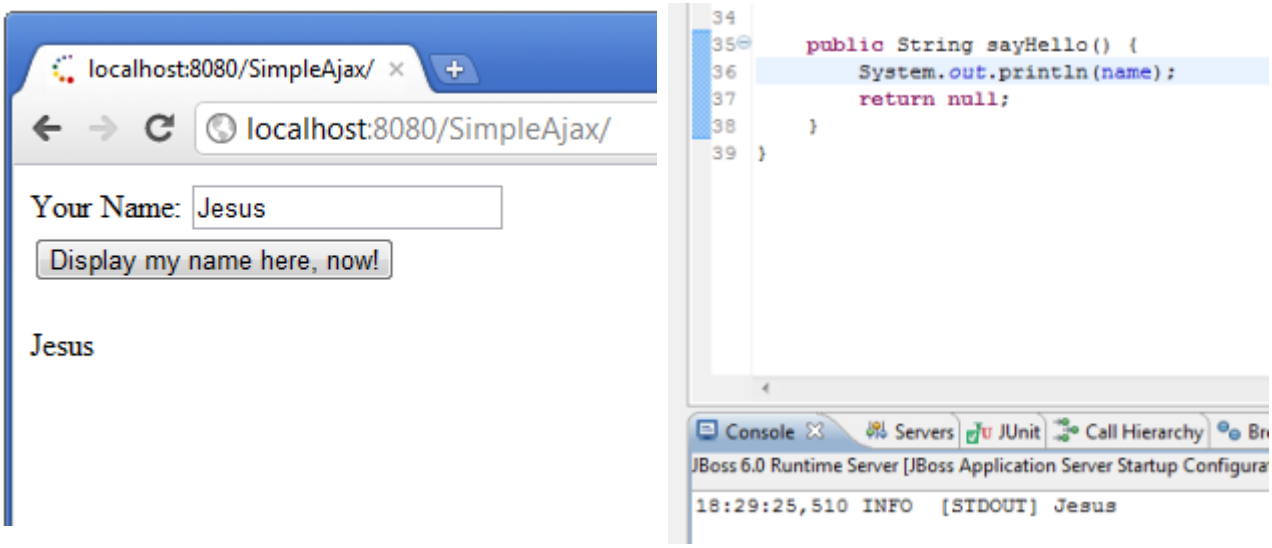
Soit l'écran suivant



Faire un projet Web dynamique qui contient la page d'accueil suivante

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">
<h:head>
</h:head>
<h:body>
   <h:form>
      Your Name: <h:inputText id="inputname" label="${msgs.prompt}"
value="#{user.name}"/>
     <br />
     <h:commandButton action="#{user.sayHello}" value="Display my name here, now!"/>
     <br />
   </h:form>
</h:body>
</html>
```

Comment pourrait-on afficher le nom dactylographié sur le même écran en utilisant Ajax? Il suffit d'ajouter le "f: ajax" dans le composant. Vérifiez la mise à jour du code et le résultat:



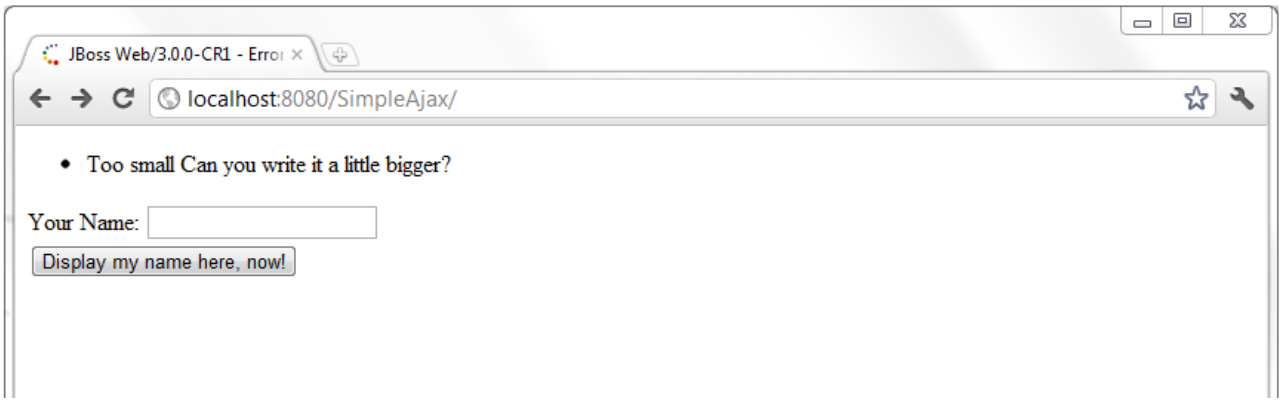

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">
<h:head>
</h:head>
<h:body>
    <h:form>
        Your Name: <h:inputText id="inputname" label="${msgs.prompt}"
value="#{user.name}"/>
        <br />
        <h:commandButton action="#{user.sayHello}" value="Display my name here, now!">
...
        </h:commandButton>
        <br />
        <br />
        <h:outputText id="myName" value="#{user.name}" />
    </h:form>
</h:body>
</html>
```

Nous avons juste besoin de passer la valeur qui sera porté par le ManagedBean et exécuter avec le paramètre render" paramètre nous dire de quel composant JSF sera «rafraîchie».

Etape2 :

Notez également que le nom dactylographié apparaît dans la console.

Avec ce code, on peut "rafraîchir" tous les types de composants. Laissez un message d'erreur si l'utilisateur tape un nom de moins de 4 caractères.




```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">
<h:head>
</h:head>
<h:body>
    <h:form>
...
        Your Name: <h:inputText id="inputname" label="${msgs.prompt}"
value="#{user.name}"/>
        <br />
        <h:commandButton action="#{user.sayHello}" value="Display my name here, now!">
...
        </h:commandButton>
        <br />
        <br />
        <h:outputText id="myName" value="#{user.name}" />
    </h:form>
</h:body>
</html>
```

Bien entendu, il faut définir le managed bean associé

```
package exercice5.etape2;

...

...
public class User {

    private String name;

    public String sayHello() {
...
        System.out.println(name);
        return null;
    }
```

```
    private boolean isNameIncorrect() {
        return "".equals(name.trim()) || name.length() < 3;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```

Notez que nous avons la composante "h: Messages" et son ID est utilisé dans le "f: ajax" composant. Ce code fonctionne également lorsque vous utilisez le "h: message pour =" YYY "" composant.

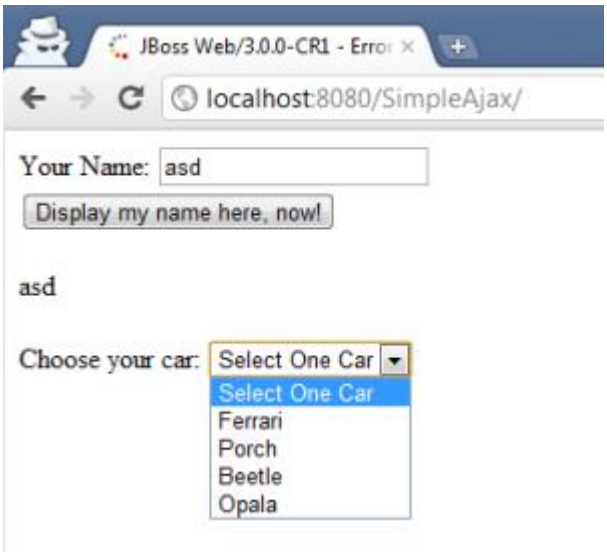
Etape3 :

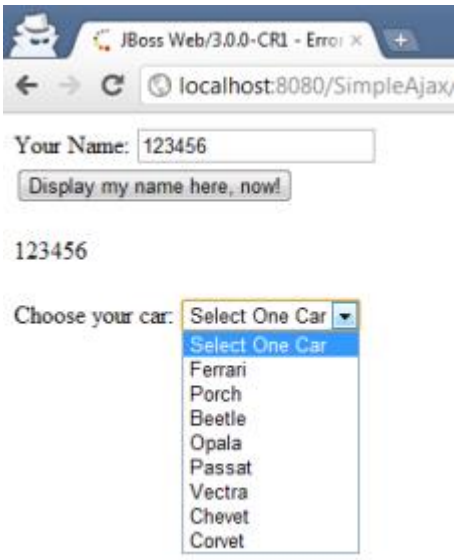
Travaillons maintenant avec des comboboxes? Afficher une liste déroulante qui contient 4 éléments lorsque nous avons un nom de moins de 6 caractères, et une liste de plus de 4 points si le nom dactylographié a plus de 6 caractères.

```
package demo;

import java.util.ArrayList;
import java.util.List;

...

...
public class User {

    private String name;

    private List<String> cars;

    private String selectedCar;
    private HtmlSelectOneMenu htmlSelectCars;

    private static final String SELECT_A_CAR = "Select One Car";

    public User() {
        cars = new ArrayList<String>();
    }

    public String sayHello() {
...

        System.out.println(name);
        return null;
    }
```

```
    private boolean isNameInCorrect() {
        return name == null || "".equals(name.trim()) || name.length() < 3;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void editMyCarsList(AjaxBehaviorEvent event) {
        if (htmlSelectCars == null) {
            htmlSelectCars = new HtmlSelectOneMenu();
        }

        htmlSelectCars.getChildren().clear();

        UISelectItems items = new UISelectItems();
        items.setValue(getCars());
        htmlSelectCars.getChildren().add(items);
    }

    public List<String> getCars() {
...

        return cars;
    }

    public void setCars(List<String> cars) {
        this.cars = cars;
    }
...
}
```

Jetez un coup d'oeil maintenant à notre page:






```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">
<h:head>
</h:head>
<h:body>
    <h:form>
...
        Your Name: <h:inputText id="inputname" label="${msgs.prompt}"
value="#{user.name}" />
        <br />
        <h:commandButton action="#{user.sayHello}" value="Display my name here, now!">
...
listener="#{user.editMyCarsList}" />
        </h:commandButton>
        <br />
        <br />
        <h:outputText id="myName" value="#{user.name}" />
        <br />
        <br />
...
        <br />
        <br />
    </h:form>
</h:body>
</html>
```

Notez que notre combobox est mis à jour en fonction du nom dactylographié.

Comme dernier exemple, créer une liste déroulante qui apparaissent et disparaissent en fonction de la valeur sélectionnée dans la liste déroulante de voiture.

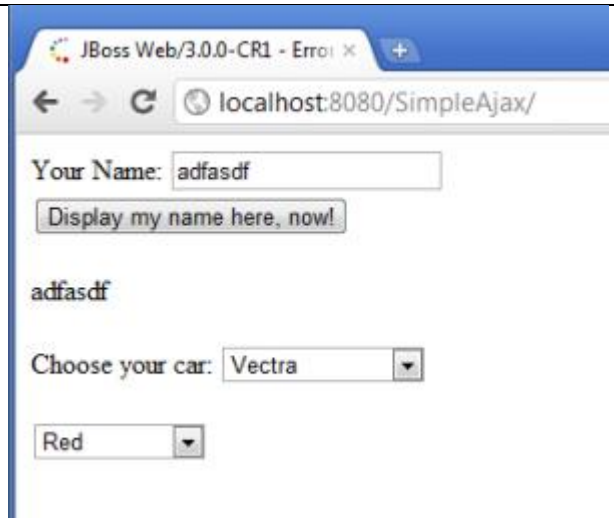
Le ManagedBean:

```
package exercice5.etape2;

...

...
public class User {
    private String name;
    private List<String> cars;
    private List<String> colors;

    private String selectedCar;
    private String selectedColor;
    private HtmlSelectOneMenu htmlSelectCars;

    private static final String SELECT_A_CAR = "Select One Car";

    public User() {
        cars = new ArrayList<String>();
        colors = new ArrayList<String>();

...
    }

    public String sayHello() {
...
        System.out.println(name);
        return null;
    }

    private boolean isNameInCorrect() {
        return name == null || "".equals(name.trim()) || name.length() < 3;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void editMyCarsList(AjaxBehaviorEvent event) {
        if (htmlSelectCars == null) {
            htmlSelectCars = new HtmlSelectOneMenu();
        }

        htmlSelectCars.getChildren().clear();

        UISelectItems items = new UISelectItems();
        items.setValue(getCars());
        htmlSelectCars.getChildren().add(items);
    }

...
```

```
        return cars;
    }

...
}
```

Notre ManagedBean a été légèrement mis à jour, nous venons d'ajouter une liste avec une méthode qui retourne une liste de couleurs qui peuplent notre combobox, nous avons ajouté aussi une méthode qui retourne un Boolean - true si la liste déroulante est autorisé à être affiché.

Visitez la nouvelle page:




```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">
<h:head>
</h:head>
<h:body>
   <h:form>
...
     Your Name: <h:inputText id="inputname" label="${msgs.prompt}" value="#{user.name}"
/>
     <br />
     <h:commandButton action="#{user.sayHello}" value="Display my name here, now!">
...
     </h:commandButton>
     <br />
     <br />
     <h:outputText id="myName" value="#{user.name}" />
     <br />
     <br />
```

```
     Choose your car:
     <h:selectOneMenu id="myCars" binding="#{user.htmlSelectCars}"
value="#{user.selectedCar}">
...
     </h:selectOneMenu>
     <br />
     <br />
     <h:panelGroup id="myColors">
      <h:selectOneMenu value="#{user.selectedColor}"
rendered="#{user.colorsAlloweToDisplay}">
        <f:selectItems value="#{user.colors}" />
      </h:selectOneMenu>
     </h:panelGroup>
    </h:form>
    </h:body>
</html>
```