AJAX in JSF

Introduction

- Ajax is an acronym for Asynchronous JavaScript and XML, a group of web technologies that enable creation of dynamic and highly responsive web applications.
- Using Ajax, web applications can retrieve content from the server without interfering with the display on the client. In the Java EE 7 platform, JavaServer Faces technology provides built-in support for Ajax.

Introduction

- Early web applications were created mostly as static web pages. When a static web page is updated by a client, the entire page has to reload to reflect the update. In effect, every update needs a page reload to reflect the change. Repetitive page reloads can result in excessive network access and can impact application performance. Technologies such as Ajax were created to overcome these deficiencies.

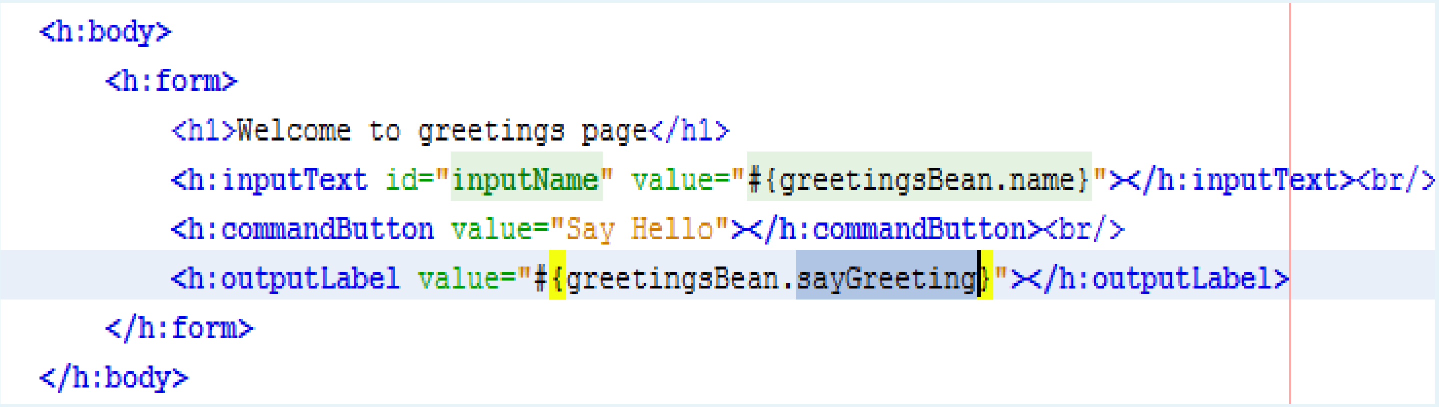
JSF With Ajax

- When you create a JSF application you should consider what happens when user press button in web browser to make some action in your app.
- Let's see an example to get this thing more clear to you.
- Open the Welcome project from examples folder.

JSF With Ajax

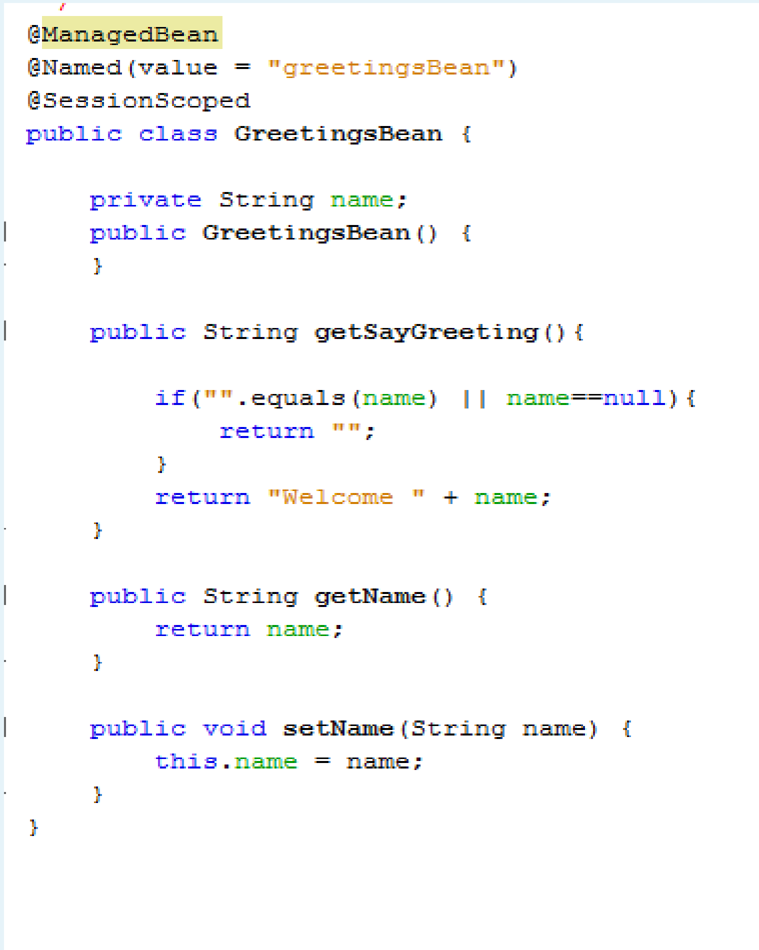
- If you run the Welcome example, enter some name on the text field and press the button, you can see from the title, that the whole page is refreshed (rendered again because of the greeting label).
- This is not what we want, we just wanted to update the label text!
- And here where the AJAX comes into the big picture.

The Welcome Example JSF code



```
<h:body>
    <h:form>
        <h1>Welcome to greetings page</h1>
        <h:inputText id="inputName" value="#{greetingsBean.name}"></h:inputText><br/>
        <h:commandButton value="Say Hello"></h:commandButton><br/>
        <h:outputLabel value="#{greetingsBean.sayGreeting}"></h:outputLabel>
    </h:form>
</h:body>
```

Managed Bean Code



```java
@ManagedBean
@Named(value = "greetingsBean")
@SessionScoped
public class GreetingsBean {

    private String name;
    public GreetingsBean() {
    }

    public String getSayGreeting(){

        if("".equals(name) || name==null){
            return "";
        }
        return "Welcome " + name;
    }

    public String getName() {
        return name;
    }

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

The Ajax Tag

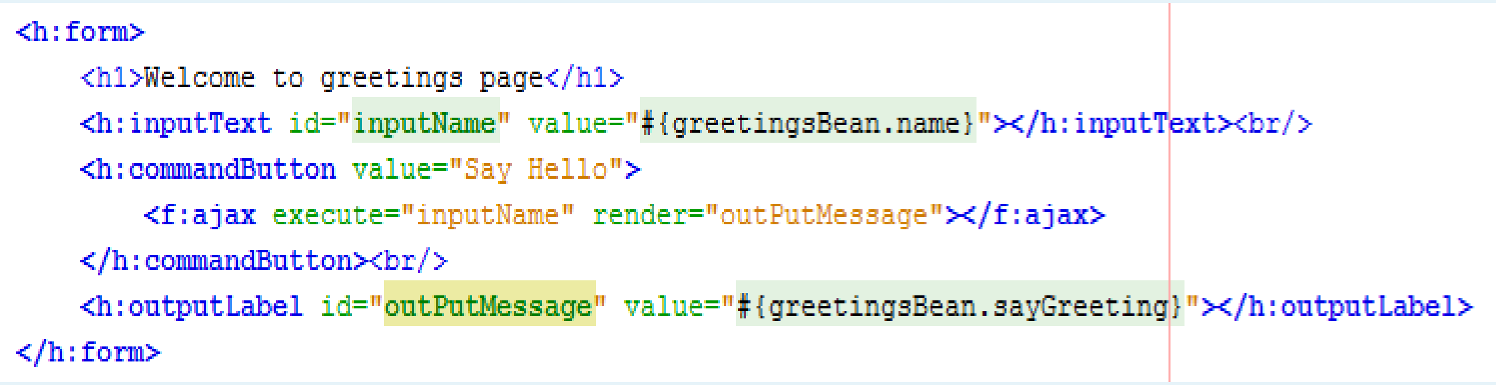
- Before seeing how we fix the problem, we need to understand the <f:ajax> tag and it's attributes.
- JavaServer Faces technology supports Ajax by using a built-in JavaScript resource library that is provided as part of the JavaServer Faces core libraries.

The Ajax Tag

- Ajax resource can be used in JavaServer Faces web applications in one of the following ways:
  - By using the f:ajax tag along with another standard component in a Facelets application. This method adds Ajax functionality to any UI component without additional coding and configuration.
  - By using the JavaScript API method jsf.ajax.request() directly within the Facelets application.This method provides direct access to Ajax methods, and allows customized control of component behavior.

The Ajax Tag

- Example:



```
<h:form>
    <h1>Welcome to greetings page</h1>
    <h:inputText id="inputName" value="#{greetingsBean.name}"></h:inputText><br/>
    <h:commandButton value="Say Hello">
        <f:ajax execute="inputName" render="outPutMessage"></f:ajax>
    </h:commandButton><br/>
    <h:outputLabel id="outPutMessage" value="#{greetingsBean.sayGreeting}"></h:outputLabel>
</h:form>
```

<f:ajax> attributes

- Disabled
  - If true, the Ajax behavior will be applied to any parent or child components. If false, the Ajax behavior will be disabled.
- event
  - The event that will invoke Ajax requests, for example "click", "change", "blur", "keypress", etc.

<f:ajax> attributes

- execute
  - A space-separated List of IDs for components that should be included in the Ajax request.
- immediate
  - If "true" behavior events generated from this behavior are broadcast during Apply Request Values phase. Otherwise, the events will be broadcast during Invoke Applications phase

<f:ajax> attributes

- listener
  - An EL expression for a method in a backing bean to be called during the Ajax request.
- onerror
  - The name of a JavaScript callback function that will be invoked if there is an error during the Ajax request
- onevent
  - The name of a JavaScript callback function that will be invoked to handle UI events.

<f:ajax> attributes

- render
  - A space-separated list of IDs for components that will be updated after an Ajax request.

Exercise

- Copy the Welcome example project in your own working directory.
- Fix the page refresh problem by using the <f:ajax> tag with command button.