



Ajax with PrimeFaces

Originals of slides and source code for examples: <http://www.coreservlets.com/JSF-Tutorial/primefaces/>

Also see the JSF 2 tutorial – <http://www.coreservlets.com/JSF-Tutorial/jsf2/>

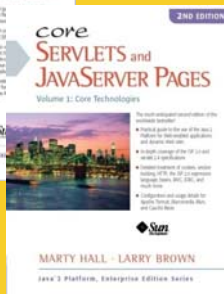
and customized JSF2 and PrimeFaces training courses – <http://courses.coreservlets.com/jsf-training.html>



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



For live training on JSF 2, PrimeFaces, or other Java EE topics, email hall@coreservlets.com
Marty is also available for consulting and development support

Taught by the author of *Core Servlets and JSP*, this tutorial, and JSF 2.2 version of *Core JSF*. Available at public venues, or customized versions can be held on-site at your organization.

- Courses developed and taught by Marty Hall
 - JSF 2, PrimeFaces, Ajax, jQuery, Spring MVC, JSP, Android, general Java, Java 8 lambdas/streams, GWT, custom topic mix
 - Courses available in any location worldwide. Maryland/DC area companies can also choose afternoon/evening courses.
- Courses developed and taught by coreservlets.com experts (edited by Marty)
 - Hadoop, Spring, Hibernate/JPA, RESTful Web Services

Contact hall@coreservlets.com for details



Topics in This Section

- **f:ajax vs. p:ajax**
- **update**
- **process**
- **Ajax-based validation**
- **Designating the event that triggers Ajax**
- **Showing temporary content during slow Ajax requests**

4

© 2015 Marty Hall



Ajax with PrimeFaces vs. Ajax with Standard JSF



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

Assumptions for This Tutorial Section

- **You are familiar with idea of Ajax**
 - Motivation in general
 - Motivation for library integrated with JSF/PrimeFaces
- **You are familiar with f:ajax**
 - render: id(s) of element(s) to update
 - execute: id(s) of element(s) to process on server
 - event: the DOM event that f:ajax responds to
 - Understanding the default (value-change for non-buttons)
 - Changing the default
 - onevent: JavaScript code to run before and/or after
- **Tutorial on Ajax in standard JSF 2**
 - <http://www.coreservlets.com/JSF-Tutorial/jsf2/>
 - Scroll down to “Ajax” section

6

p:ajax vs. f:ajax: Summary

- **update/process instead of render/execute**
 - `<p:ajax update="id-to-update" process="id-to-process"/>`
- **Ajax attributes built into command button**
 - `<p:commandButton ... update="..." process="...">`
- **Component-specific events**
 - Knows how to respond to calendar-selection, spinner clicks, slider drags, clicking on bar chart, dragging panels, etc.
- **p:ajaxStatus**
 - Simpler way to show things before and after Ajax call
- **Uses jQuery**
 - Rather than a custom JavaScript library
- **Documentation**
 - Listed in User's Guide under “AjaxBehavior”, not “Ajax”

7

p:ajax vs. f:ajax: update/process

- **Standard JSF**

`<f:ajax render="id-to-update" execute="id-to-process"/>`

- **PrimeFaces**

`<p:ajax update="id-to-update" process="id-to-process"/>`

- Multiple ids should be separate by spaces, same as in standard JSF 2

- **Notes**

- p:ajax existed before f:ajax
 - So you cannot complain about the naming inconsistencies
- The PrimeFaces names make more sense
 - Making you wonder why f:ajax did not adopt them

8

p:ajax vs. f:ajax: Ajax Attributes in p:commandButton

- **Standard JSF**

`<h:commandButton action="..." value="..." ...>`

`<f:ajax render="..." execute="..." />`

`</h:commandButton>`

- **PrimeFaces**

`<p:commandButton action="..." value="..." ...
update="..." process="..." />`

- **Note**

- If you want to use p:commandButton without Ajax, you must use ajax="false"

`<p:commandButton ... ajax="false"/>`

9



Basics: Specifying Elements to Update



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

p:ajax Basics

- **Goal**
 - Show random number. Make button that updates it.
- **Approach**
 - Use the update attribute of p:commandButton. If you have multiple ids, separate them by spaces.
 - No need for p:ajax at all
- **Basic code**

```
<p:commandButton value="Show Number"
    action="#{code-to-update-num}"
    update="id-of-field-showing-num"/>
```


Example HTML: No Ajax (Full Page Reload)

```
<h:form>
  <p:commandButton value="Show Number"
    action="#{numberGenerator.randomize}"
    ajax="false"/>
  <h2>#{numberGenerator.number}</h2>
</h:form>
```

12

Example Java: Used in Both Ajax and Non-Ajax Versions

```
@ManagedBean
public class NumberGenerator {
    private double number = Math.random();
    private double range = 1.0;

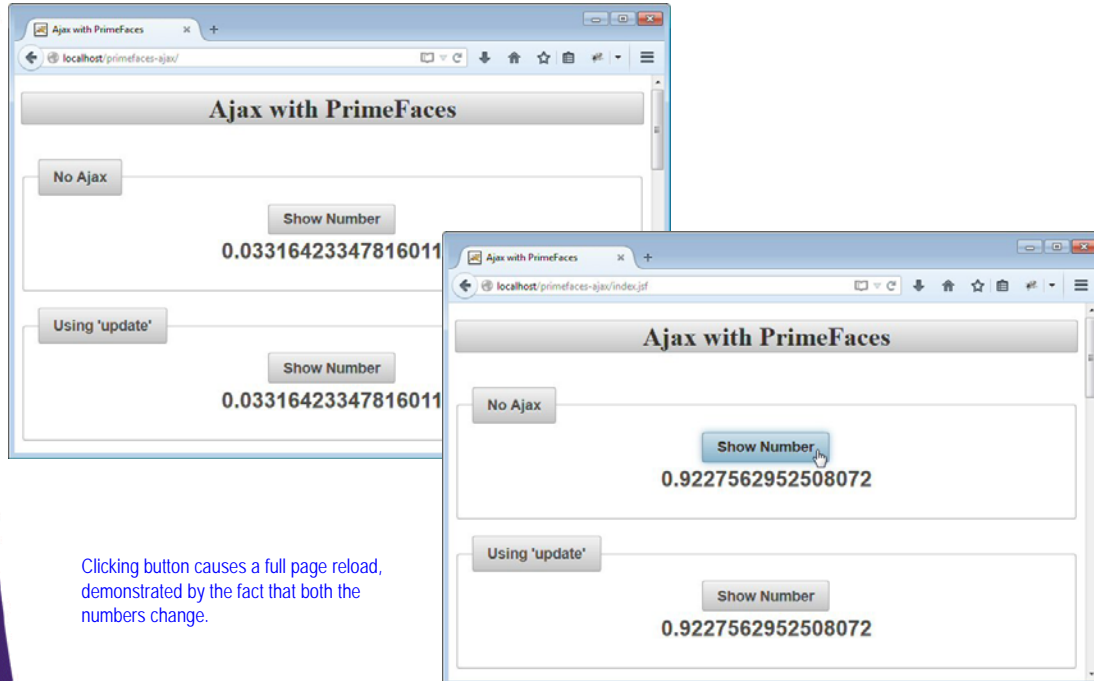
    // Getter and setter for range

    public double getNumber() {
        return(range * number);
    }

    public String randomize() {
        number = Math.random();
        return(null);
    }
}
```

13

No Ajax: Results



14

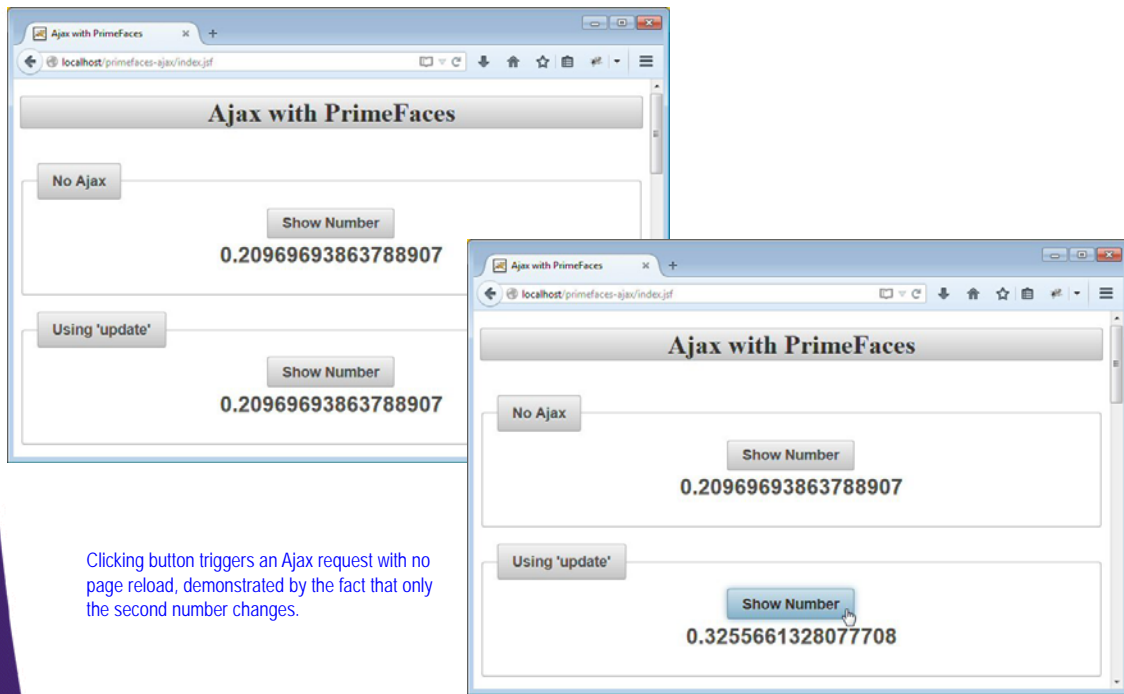
Example HTML: Ajax

```
<h:form>
<p:commandButton value="Show Number"
    action="#{numberGenerator.randomize}"
    update="num-field"/>
<h2><h:outputText value="#{numberGenerator.number}"
    id="num-field"/></h2>
</h:form>
```

The corresponding Java code is unchanged from what was shown earlier.

15

Ajax: Results



Clicking button triggers an Ajax request with no page reload, demonstrated by the fact that only the second number changes.

16

© 2015 Marty Hall



Designating Form Elements to Process on Server



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

The process Attribute

- **Goal**

- Show random number. Use number spinner to change range. Make button that generates and displays random number from 0 up to that range.

- **Approach**

- Use the process attribute to designate the spinner. If you have multiple ids, separate them by spaces. Alternatively, omit ids for input elements and use process="@form".

- **Basic code**

```
<p:commandButton value="Show Number"
                 action="#{code-to-update-num}"
                 update="id-of-field-showing-num"
                 process="id-of-input-element"/>
```

18

Example HTML

```
<h:form>
Range:
<p:spinner value="#{numberGenerator.range}"
           id="range-field"/>
<p:commandButton value="Show Number"
                 process="range-field"
                 action="#{numberGenerator.randomize}"
                 update="num-field"/>
<h2><h:outputText value="#{numberGenerator.number}"
                  id="num-field"/></h2>
</h:form>
```

The corresponding Java code is unchanged from what was shown earlier.

19

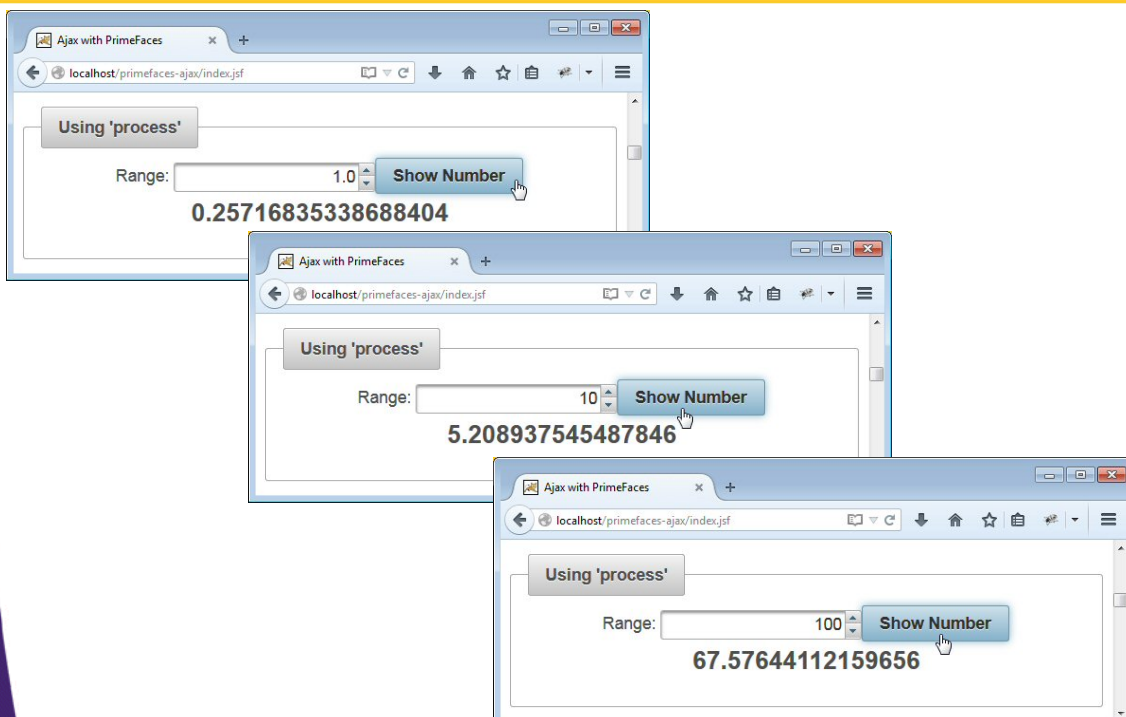
Example HTML (Alternative Version)

```
<h:form>
Range:
<p:spinner value="#{numberGenerator.range}"/>
<p:commandButton value="Show Number"
    process="@form"
    action="#{numberGenerator.randomize}"
    update="num-field"/>
<h2><h:outputText value="#{numberGenerator.number}"
    id="num-field"/></h2>
</h:form>
```

No need for id

20

Example: Results



21



Ajax-Based Validation



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

Ajax Validation with Explicit Update

- **Goal**
 - Update with Ajax as before, but perform field validation.
 - See validation sections of tutorial on standard JSF 2
- **Approach**
 - Give id to p:message or p:messages or p:growl. Include this id in the update values.
 - Validation error: error message shown, range of 1 used
 - Validation passes: no error message, specified range used
 - Also see overlays lecture for special case of validation with dialogs

- **Basic code**

```
<p:commandButton value="Show Number"  
    action="#{code-to-update-num}"  
    update="output-id message-id"  
    process="id-of-input-element"/>
```

PrimeFaces Error Message Output

- **p:messages**
 - Like h:messages, but takes on the look and feel of the current PrimeFaces theme
- **p:message**
 - Like h:message, but takes on the look and feel of the current PrimeFaces theme
- **p:growl**
 - Like p:messages, but drops down from the top right corner of browser, rather than appearing where the tag is located in the HTML page
- **Details and examples**
 - Are given in the lecture on overlays and dialogs

24

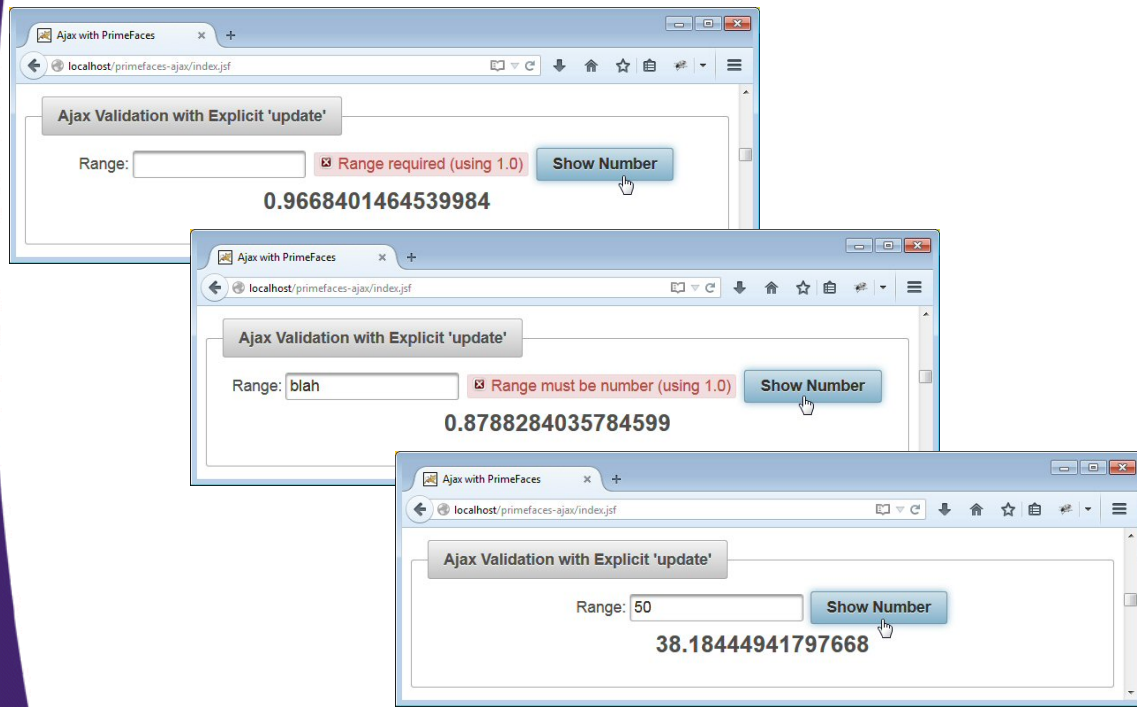
Example HTML

```
<h:form>
<h:panelGrid columns="4">
Range:
<p:inputText value="#{numberGenerator.range}"
             id="range-field"
             required="true"
             requiredMessage="Range required (using 1.0)"
             converterMessage="Range must be number (using 1.0)">
</p:inputText>
<p:message for="range-field" id="error-field"/>
<p:commandButton value="Show Number"
                 action="#{numberGenerator.randomize}"
                 process="range-field"
                 update="num-field error-field"/>
</h:panelGrid>
<h2><h:outputText value="#{numberGenerator.number}"
                  id="num-field"/></h2>
</h:form>
```

25

The corresponding Java code is unchanged from what was shown earlier.

Example: Results



26

Ajax Validation with autoUpdate

- **Goal**
 - Same as before, but do not explicitly list the id of the message element.
- **Approach**
 - Use `autoUpdate="true"` for `p:messages`
 - Can also use `p:growl`. But not `p:message`.
- **Warning**
 - Only works when you have only one message element in entire page. Otherwise an Ajax update of one form can result in incorrect message shown on different form.
- **Basic code**

```
<p:messages autoUpdate="true" />
```

27

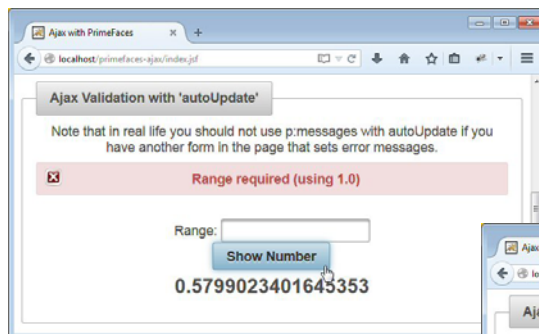
Example HTML

```
<h:form>
<p:messages autoUpdate="true"/><br/>
Range:
<p:inputText value="#{numberGenerator.range}"
             required="true"
             requiredMessage="Range required (using 1.0)"
             converterMessage="Range must be number (using 1.0)">
</p:inputText><br/>
<p:commandButton value="Show Number"
                 action="#{numberGenerator.randomize}"
                 process="@form"
                 update="num-field"/>
<h2><h:outputText value="#{numberGenerator.number}"
                  id="num-field"/></h2>
</h:form>
```

28

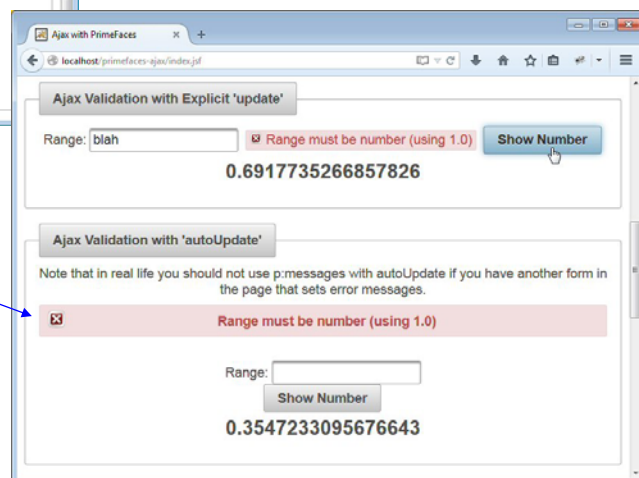
The corresponding Java code is unchanged from what was shown earlier.

Example: Results



autoUpdate works fine in usual case where there is only one form in the page, and thus only a single message element.

autoUpdate does not work when there are two forms in the page that both have message elements. A validation error in one form causes an error message to be shown in other form.



29

Special Case: Dialogs and Validation

- **Problem**

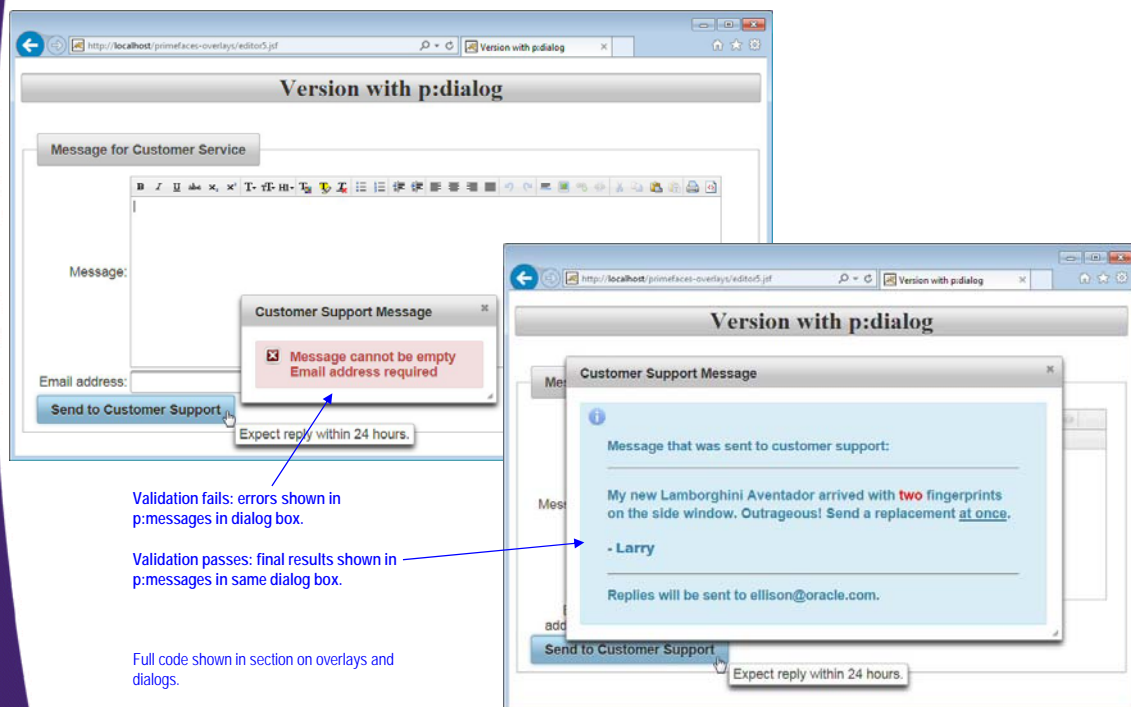
- The oncomplete event determines when Ajax has finished. But, no JavaScript event determines if validation passed or failed. So, dialog box pops up either way.
 - Validation fails: p:messages updated, action method not called.
 - Validation passes: no content in p:messages, action method called. Either way, onsuccess and oncomplete triggered.

- **Solution**

- Have content of dialog be p:messages only. Have “good” content be a FacesMessage, but with severity set to SEVERITY_INFO. Dialog pops up either way: sometimes showing validation errors, sometimes showing final results
 - Details in section on overlays and dialogs

30

Example



31



Designating Events that Trigger Ajax



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

The event Attribute

- **Goal**
 - Trigger Ajax on keystrokes or another non-default event.
 - In this example, make temperature converter that shows updates for every key that is typed
- **Approach**
 - Understand defaults
 - Command components: click. Input elements: change
 - Change defaults with event="event-name"
 - Can be DOM event such as "keyup" or custom PrimeFaces event such as "dateSelect"
- **Basic code**

```
<p:ajax update="..." process="..."  
    event="event-name"/>
```

Example 1: HTML

```
<h:form>
Temperature in Fahrenheit:
<p:inputText value="#{tempConverter.fahrenheit}">
  <p:ajax event="keyup" update="cField kField"/>
</p:inputText><br/>
<h2>Temperature in Celsius:
<h:outputText value="#{tempConverter.celsius}"
              id="cField"/><br/>
Temperature in Kelvin:
<h:outputText value="#{tempConverter.kelvin}"
              id="kField"/><br/>
</h2>
</h:form>
```

Default event is change, not keyup. So, you must specify event explicitly.

34

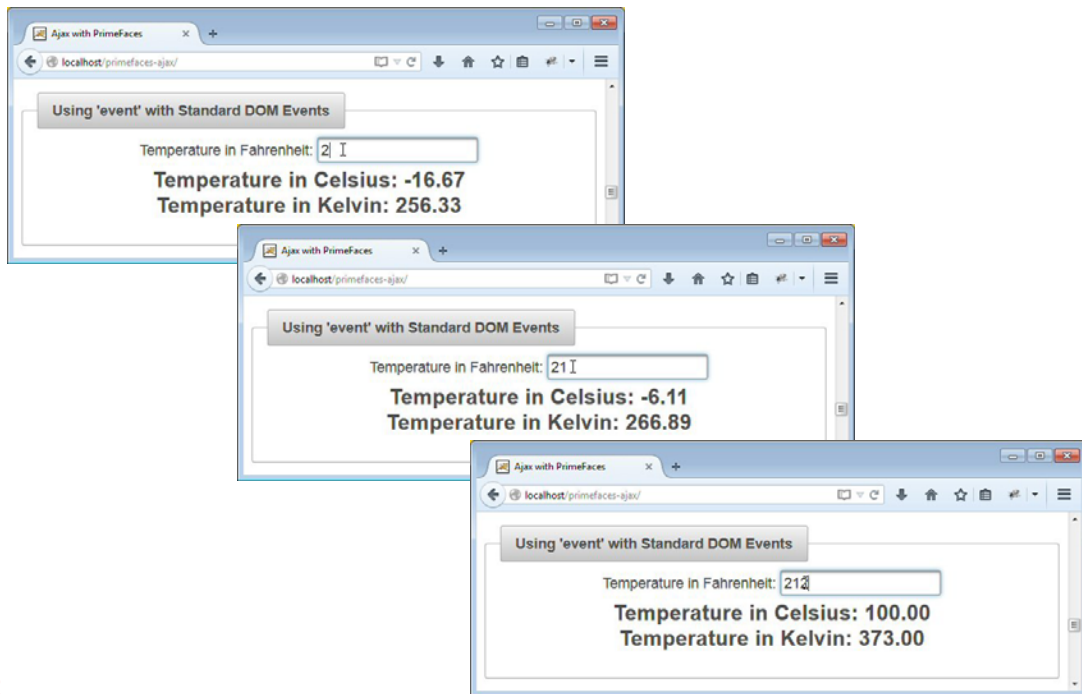
Example 1: Java

```
@ManagedBean
public class TempConverter {
    private String celsius, kelvin;
    // getCelsius, getKelvin, getFahrenheit

    public void setFahrenheit(String fTemp) {
        double f = 0;
        try {
            f = Double.parseDouble(fTemp);
            f = Math.max(f, -459.4); // Absolute zero
            double c = (f - 32)*(5.0/9.0);
            double k = c + 273;
            celsius = String.format("%.2f", c);
            kelvin = String.format("%.2f", k);
        } catch (NumberFormatException nfe) {
            celsius = "Invalid";
            kelvin = "Invalid";
        }
    }
}
```

35

Example 1: Results



36

Example 2: HTML

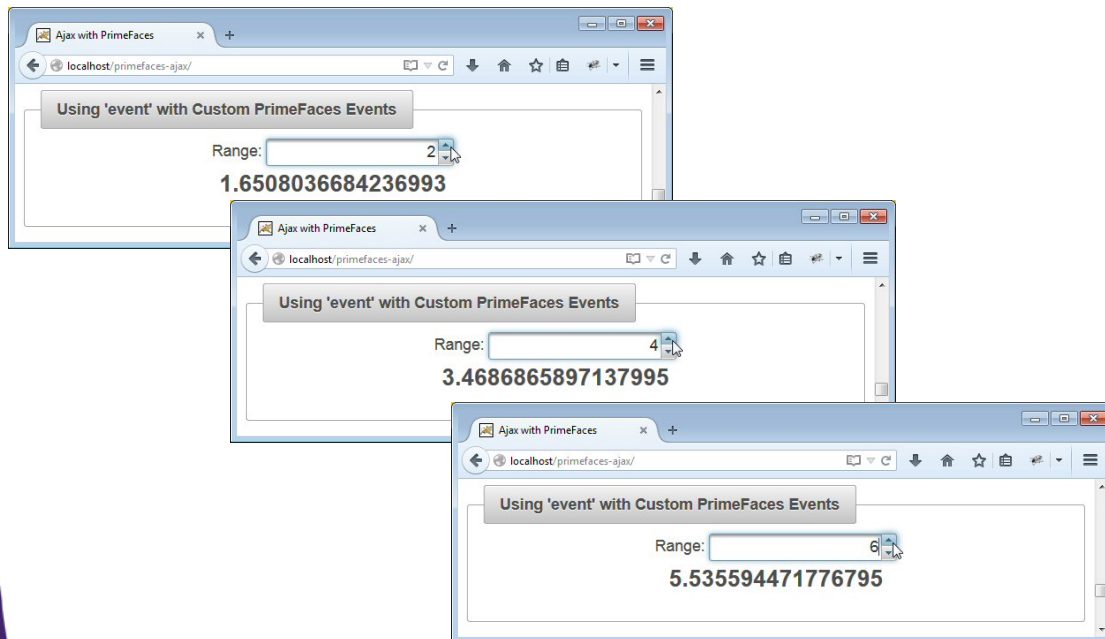
```
<h:form>  
Range:  
<p:spinner value="#{numberGenerator.range}">  
  <p:ajax update="num-field"/>  
</p:spinner>  
<h2><h:outputText value="#{numberGenerator.number}"  
  id="num-field"/></h2>  
</h:form>
```

Default event is change. So, no need to specify event explicitly.

Same Java code as earlier examples that displayed random numbers.

37

Example 2: Results



38

© 2015 [Marty Hall](#)



Showing Temporary Content During Slow Ajax Requests



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

p:ajaxStatus

- **Goal**

- Show “working...” message and animated GIF indicator while waiting for Ajax response

- **Approach**

- Use p:ajaxStatus
 - Mark during-request content with <f:facet name="start">
 - Mark ending content with <f:facet name="complete">

- **Basic code**

```
<p:ajaxStatus>
  <f:facet name="start">...</f:facet>
  <f:facet name="complete">...</f:facet>
</p:ajaxStatus>
```

40

Warning: AjaxStatus is Global

- **Problem**

- AjaxStatus applies to all Ajax requests in entire page
- So, if there is a fast Ajax request and a slow Ajax request, the AjaxStatus example on previous page will show temporary results both times

- **Solution**

- For handling things specific to a particular Ajax call, use the onstart and oncomplete attributes of p:ajax and p:commandButton

```
<p:commandButton ... onstart="showRegion1()"
                  oncomplete="hideRegion1AndShowResult()"/>
```

- But, you will have to write custom JavaScript code to show and hide the regions

41

Aside: Building Animated GIFs

- **ajaxload.info**

- <http://ajaxload.info/> lets you build your own indicator GIFs
- Free for any use and totally without restrictions
 - Cites the <http://www.wtfpl.net/> "license"



42

Example: HTML

```
<h:form>
<p:commandButton value="Show Number"
                  action="#{numberGenerator.randomize}"
                  update="num-field"/>

<p:ajaxStatus>
<f:facet name="start">
  <h2><p:graphicImage name="images/ajax-loader.gif"/>
    Getting data from server...</h2>
</f:facet>
<f:facet name="complete">
  <h2><h:outputText value="#{numberGenerator.numberSlow}"
                    id="num-field"/></h2>
</f:facet>
</p:ajaxStatus>
</h:form>
```

43

Example: Java

```
@ManagedBean
public class NumberGenerator {

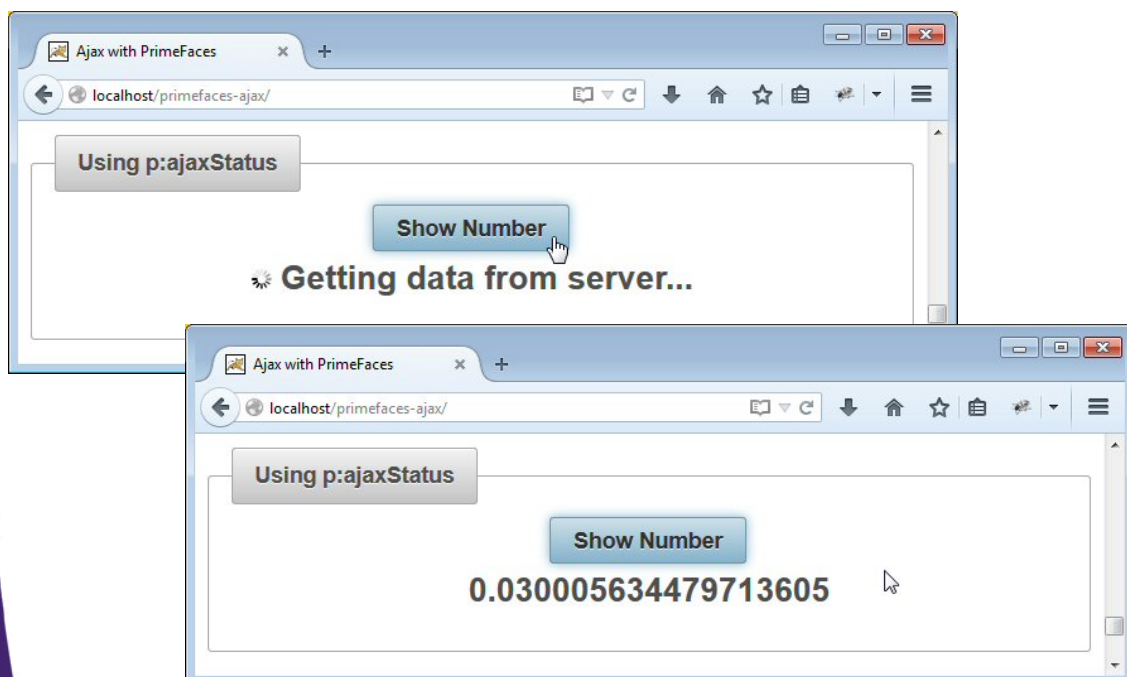
    ...

    public double getNumbersSlow() {
        try {
            Thread.sleep(2500);
        } catch (InterruptedException ie) {
        }
        return(range * number);
    }
}
```

The rest of NumberGenerator shown earlier.

44

Example: Results



45



Wrap-Up



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

Summary

- **Use process and update**
 - Instead of render and execute
 - Can supply directly in p:commandButton
- **Validation**
 - Include the message element (p:message, p:messages, p:growl) in the update list
 - Can use p:messages or p:growl with autoUpdate, but only when there is a single message element in page
- **Use event to change what triggers Ajax**
 - Many custom events (like dateSelect) in docs
- **Show temporary content with p:ajaxStatus**
 - Mark during-request content with <f:facet name="start">
 - Mark ending content with <f:facet name="complete">



Questions?

More info:

<http://www.coreservlets.com/jsf-Tutorial/jsf2/> – JSF 2.2 tutorial

<http://www.coreservlets.com/jsf-Tutorial/primefaces/> – PrimeFaces tutorial

<http://courses.coreservlets.com/jsf-training.html> – Customized JSF and PrimeFaces training courses

<http://coreservlets.com/> – JSF 2, PrimeFaces, Java 7 or 8, Ajax, jQuery, Hadoop, RESTful Web Services, Android, HTML5, Spring, Hibernate, Servlets, JSP, GWT, and other Java EE training



Customized Java EE Training: <http://courses.coreservlets.com/>

Java 7, Java 8, JSF 2, PrimeFaces, Android, JSP, Ajax, jQuery, Spring MVC, RESTful Web Services, GWT, Hadoop

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.