
OurUniversity
EFMD CLIP ACCREDITED
Force.com Visualforce Pages
salesforce
global strategic
consulting partner
Lesson 05 : Visualforce Components for Modularity

Lesson Objectives

By the end of this lesson, you will be able to :

- List Key tags and define what their attributes control
- Create Visualforce pages that use these tags to custom components and templates

<apex:component>

The <apex:component> tag can be used to create your own custom, reusable components.

- Create custom components via Your Name | Setup | Develop | Components

Custom components can be accessed using the c namespace (as in <c:componentname>, and are automatically added to the online reference.

Attribute include:

- Controller or extension (String): name of the controller used to control the behavior of this component
- Access (Sting): indicates the ability to use the component outside of the same namespace

<apex:attribute> & <apex:componentBody>

Use the <apex:attribute> tag within a component tag to define the custom attributes for your custom component.

- Attributes can define the name, data type, and other aspects of the custom attribute

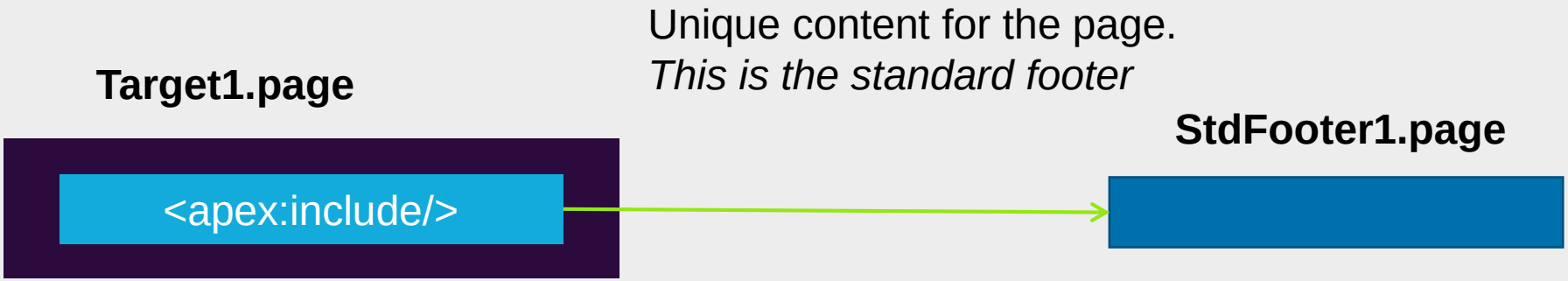
Use the <apex:componentBody> tag within a component tag to pull the body of the components implementation into the component definition.

- This is often used for custom iteration components

Page Inclusions

Tags that allow you to insert other Web content within an iframe.
<apex:iframe> : insert Web content using a URL
<apex:inlcude>: insert another Visualforce page

<apex:include> Example



Target1 Page

```
1   <apex:page showHeader="false">
2     Unique content for this page.
3       <apex:include pageName="StdFooter1.page" />
4   </apex:page>
```

StdFooter1 Page

```
1   <apex:page>
2     <br/>
3     <i>This is the standard footer.</i>
4   </apex:page>
```

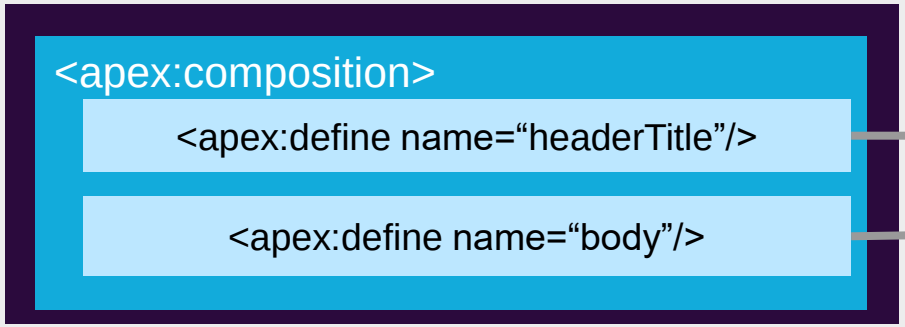
Template Tags

Tags for defining reusable content and styles
<apex:composition> : a container that defines the name of the template Visualforce page to be applied to the target page
<apex:define>: provides dynamic content to be inserted into the template using insert components
<apex:insert>: used in templates to insert a named area defined by a define tag in the target page

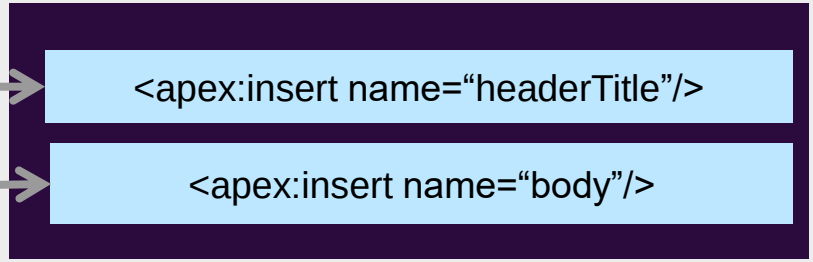
<apex:include> Example

Welcome to the Products page.
Unique content for this page

Target2.page



Template1.page



<apex:include> Example(Cont.)

Target2 Page

```
1    <apex:page showHeader="false">
2      <apex:composition template="Template1" >
3        <apex:define name="headerTitle">Product</apex:define>
4     <apex:define name="body">Unique content for this
     page</apex:define>
5    </apex:composition> </apex:page>
```

Template1 Page

```
1    <apex:page>
2      Welcome to the <b><apex:insert name="headerTitle" /></b> page.<br/>
3      <apex:insert name="body" />
4    </apex:page>
```

Demo

Creating page using Page Templates

<messaging:emailTemplate>

The messaging components are used to create Visualforce email template.

- All email templates must be wrapped inside a single emailTemplate component
- Visualforce email templates have advantages over traditional email templates
- Email templates can be edited under Your Name | Setup | Email | MyTemplates or Your Name | Setup | Communication Templates | Email Templates (for public templates)
- Email templates can be translated

You can use the following tags within email templates, which add the corresponding elements to the email.

- <messaging:emailHeader>
- <messaging:htmlEmailBody>
- <messaging:plainTextEmailBody>
- <messaging:attachment>

Visualforce Performance Considerations

If your users experience delays, unexpected behavior, or other issues specifically around Visualforce.

Determine if Visualforce is the cause

- Ensure that the issue is not confined to a single user's computer
- Check the load time of other Salesforce pages to ensure that slow load times are not the result of a network issue

Follow general Web design best practices:

- Optimize JavaScript and CSS
- Optimize images for the Web
- Avoid iframes whenever possible

Use the Apex Developer Console to step through the request and determine which items in the request used the most system resources

Visualforce Performance

Possible solutions to common Visualforce performance issues:
Reduce View State size by using only one <apex:form> tag on a page
Reduce Page size < 15MB
Cache frequently accessed resources (such as icon graphics, etc.)
Increase the interval attribute of the <apex:actionPoller> component to 15 seconds instead of 5
Remove unnecessary fields to reduce the amount of data returned

Summary

<apex:component> tag can be used to create your own custom, reusable components

Template Tags

Visualforce Performance Considerations