**Custom Visualforce Component example**

Custom visualforce components are very useful. In our projects, many times we develop codes which are required again and again. So instead of repeating same code again and again, we can create visualforce component. Then we can use visualforce component in every place where we need that particular piece of code. In other words, custom visualforce component allows us to create reusable component.

All custom visualforce component definitions must be wrapped inside a single <apex:component > tag.

We can also use <apex:attribute> tag to use customize the component so that custom component can be used in different manners depending on value of different attributes. It helps us in creating reusable generic component and also saves time and number of lines we write in apex and visualforce page.

**Component Code:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | <apex:component >      <apex:attribute name="textValue" description="This is the value for the component" type="String" required="true"/>      <apex:attribute name="textColor" description="This is color for the border." type="String" required="true"/>      <apex:outputText value="{!textValue}" style="color:{!textColor};"/>  </apex:component> |

This component is creating two attributes using <apex:attribute> tag. First attribute is deciding what text text should be displayed and second attribute is deciding color of text. We can use any number of attribute in component. Component can also have controller which helps in more customizable component.

Now we need to use this component. We can use component in visualforce page using <c:componentName>.

**Visualforce Code:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | <apex:page tabStyle="Account">      <apex:pageBlock >          <apex:pageBlockSection title="myComponent Test" collapsible="false">              <c:myComponent textValue="This Text is blue" textColor="blue" />              <c:myComponent textValue="But this is red" textColor="red" />          </apex:pageBlockSection>      </apex:pageBlock>  </apex:page> |