Visualforce order of execution I: GET requests

Following with a previous post about triggers and order of execution, I want to review in this new post how the order of execution on a Visualforce page is. This time we need to distinguish between two different cases: Visualforce pages that perform a get request and those which perform a post request.

In this post I will explain how GET requests work. I will elaborate a second part of the post, talking about POST requests.

I have based my investigation on this page from Salesforce help, however the empirical results I have got are a bit different from what the help says, so I will try to explain everything that I have found in detail.

Once a visualforce page is requested through a GET request, the server will perform all these steps:

1. Evaluate **constructors on controller and extensions**: this is, if I am using an Apex class as controller or as a controller extension, their constructors will be first evaluated. Remember we can have multiple controller extensions in a single visualforce page. In that case, the order in which the constructors is evaluated, is the order in which they are indicated in the visualforce page definition.

For illustrating that, I have created the next visualforce page, with the next controller and extensions:

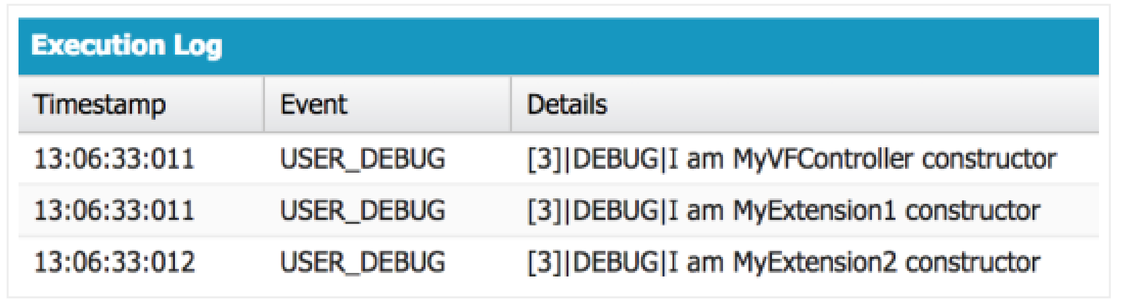
```
<apex:page controller="MyVFController" extensions="MyExtension1,MyExtension2">
   hello!
</apex:page>
```

```
public class MyVFController {
    public MyVFController() {
        System.debug('I am MyVFController constructor');
    }
}
```

```
public class MyExtension1 {
    public MyExtension1(MyVFController controller) {
        System.debug('I am MyExtension1 constructor');
    }
}
```

```
public class MyExtension2 {
    public MyExtension2(MyVFController controller) {
        System.debug('I am MyExtension2 constructor');
    }
}
```

As it was expected, I obtain the next debug log:


Execution Log

Timestamp	Event	Details
13:06:33:011	USER_DEBUG	[3]|DEBUG|I am MyVFController constructor
13:06:33:011	USER_DEBUG	[3]|DEBUG|I am MyExtension1 constructor
13:06:33:012	USER_DEBUG	[3]|DEBUG|I am MyExtension2 constructor

2. According to Salesforce help, the next step is, if there are custom **components** on page, **evaluate constructors on controller and extensions**. However, what I have found is that the component controller and extension constructors are executed before the visualforce page ones. For checking this, I have created the next component:

```
<apex:component controller="ComponentController" extensions="ComponentExtension1,ComponentExtension2">
    <apex:attribute name="param1" description="This is my param1" type="String" assignTo="{!param1Component}"/>
    <p> Param1Component: {!param1Component} </p>
    <p> Param1ComponentCopy: {!param1ComponentCopy} </p>
</apex:component>
```

```
public class ComponentController {
    public String param1Component {get; set;}
    public String param1ComponentCopy {get; private set;}

    public ComponentController() {
        System.debug('I am ComponentController constructor');
        System.debug('The value of param1Component is: ' + param1Component);
    param1ComponentCopy = param1Component;
    }
}
```

```
public class ComponentExtension1 {
    public ComponentExtension1(ComponentController controller) {
        System.debug('I am ComponentExtension1 constructor');
    }
}
```

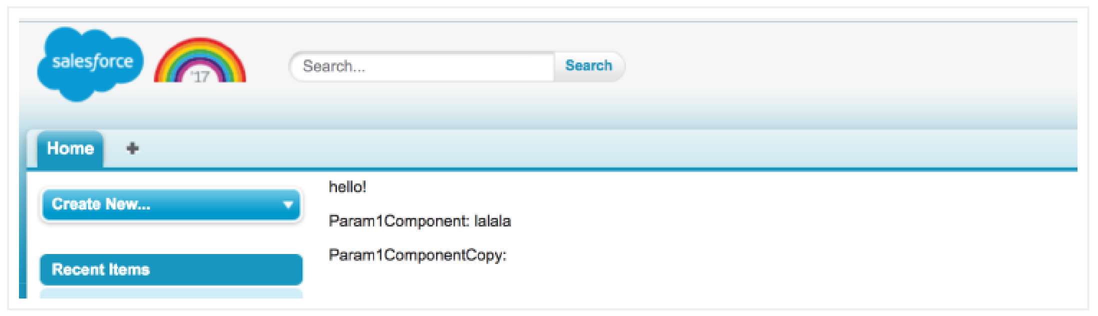
```
public class ComponentExtension2 {
    public ComponentExtension2(ComponentController controller) {
        System.debug('I am ComponentExtension2 constructor');
    }
}
```

I have modified the visualforce page I created for the first example to include the component, and also I have modified its constructor for having to set a parameter first:

```
<apex:page controller="MyVFController" extensions="MyExtension1,MyExtension2">
    hello!
    <c:OrderOfExecution param1="{!param1}"/>
</apex:page>
```

```
public class MyVFController {
    public String param1 {get; private set;}
    public MyVFController() {
        System.debug('I am MyVFController constructor');
        param1 = 'lalala';
    }
}
```

If I open the visualforce page, I see this:

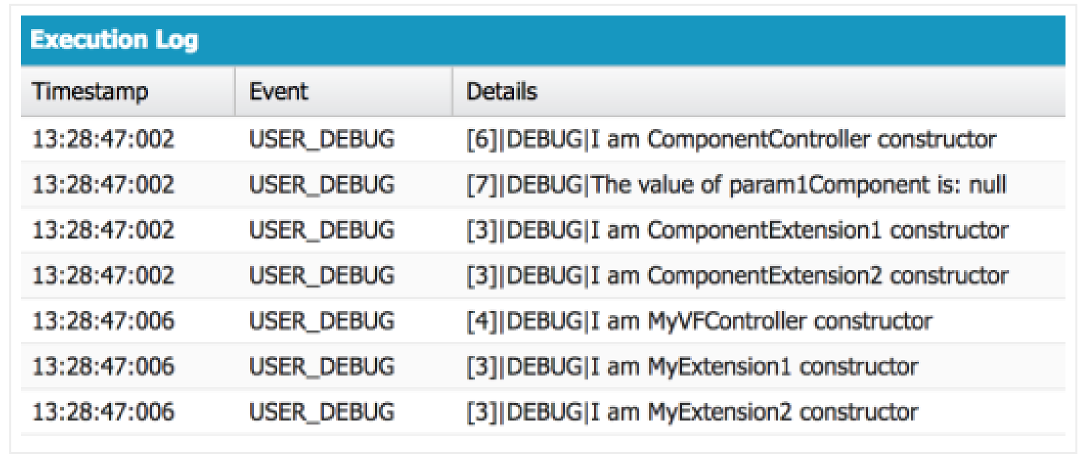


hello!

Param1Component: lalala

Param1ComponentCopy:

And if I take a look at the debug log that has been generated, I see the next:



Execution Log

Timestamp	Event	Details
13:28:47:002	USER_DEBUG	[6]|DEBUG|I am ComponentController constructor
13:28:47:002	USER_DEBUG	[7]|DEBUG|The value of param1Component is: null
13:28:47:002	USER_DEBUG	[3]|DEBUG|I am ComponentExtension1 constructor
13:28:47:002	USER_DEBUG	[3]|DEBUG|I am ComponentExtension2 constructor
13:28:47:006	USER_DEBUG	[4]|DEBUG|I am MyVFController constructor
13:28:47:006	USER_DEBUG	[3]|DEBUG|I am MyExtension1 constructor
13:28:47:006	USER_DEBUG	[3]|DEBUG|I am MyExtension2 constructor

So, what this says to me is that:

- First, **component controller and extensions constructors** are evaluated, no matter if the parameters that are passed in to the component are being initialised in the visualforce page constructor (I read in a stackexchange answer that the component constructor was evaluated first in that case, but that is not what I have found in my tests).
- Second, **visualforce page controller and extensions constructors** are evaluated.
- Third, variables passed in with **assignTo** are passed to the component.

If you see in the example, we have found a common problem, **parameters passed to a component controller through assignTo attribute, won't be ready in the component controller constructor** (or its extensions!).

3. The next step is evaluation of the **action attribute on the <apex:page>** component , expressions, and the required getter and setter methods.

In our example, apart from the {!param1} that we pass to the component and that has already been evaluated, we have the next expressions:

- **{!param1Component}**: this expression calls the param1Component getter in the component constructor. As param1Component was assigned a value with the assignTo attribute, we see in the rendered page that it has been correctly rendered.
- **{!param1ComponentCopy}**: this expression calls the param1ComponentCopy getter in the component constructor. As when the component constructor was executed, param1Component did not have a value yet, we see in the rendered page that the value of param1ComponentCopy is null.

Also I have modified a bit the page to include an action param:

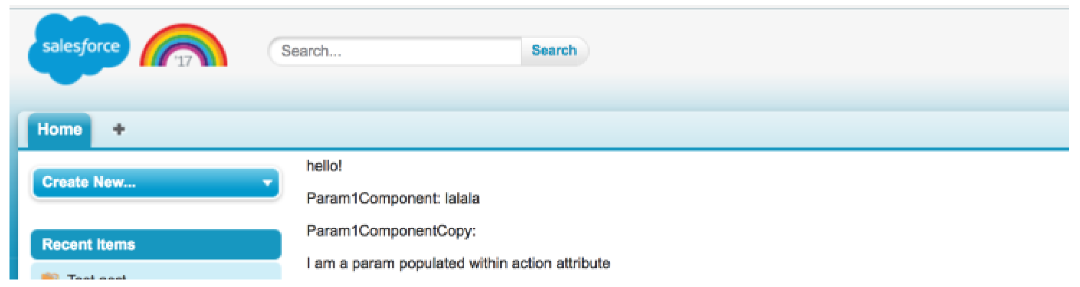
```
<apex:page action="{!populateActionParam}" controller="MyVFController" extensions="MyExtension1,MyExtension2">
    hello!
    <c:OrderOfExecution param1="{!param1}"/>
    {!actionParam}
</apex:page>
```

```
public class MyVFController {
    public String param1 {get; private set;}
    public String actionParam {get; private set;}

    public MyVFController() {
           System.debug('I am MyVFController constructor');
           param1 = 'lalala';
        }

    public void populateActionParam() {
           System.debug('I am populating the actionParam var');
           actionParam = 'I am a param populated within action attribute';
    }
}
```

Salesforce help says that the order of these evaluations (action page attribute and expressions in the page) is **indeterminate**. However in my experience I understood that the action param is always evaluated before any expression on the page. If I open the visualforce page, with the modifications I have done, I see the next:



hello!

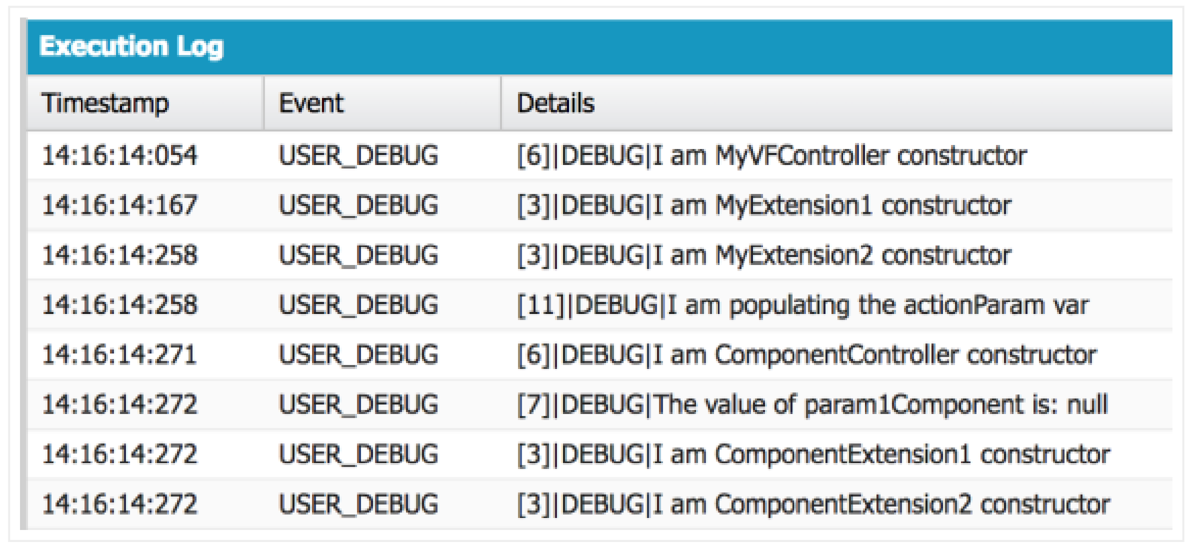
Param1Component: lalala

Param1ComponentCopy:

I am a param populated within action attribute

That means the action param method must have been evaluated before the expressions in the page, because if not, I would not be able of seeing the last sentence.

But here is when I find the surprise. If I take a look at the debug log:



Execution Log

Timestamp	Event	Details
14:16:14:054	USER_DEBUG	[6]|DEBUG|I am MyVFController constructor
14:16:14:167	USER_DEBUG	[3]|DEBUG|I am MyExtension1 constructor
14:16:14:258	USER_DEBUG	[3]|DEBUG|I am MyExtension2 constructor
14:16:14:258	USER_DEBUG	[11]|DEBUG|I am populating the actionParam var
14:16:14:271	USER_DEBUG	[6]|DEBUG|I am ComponentController constructor
14:16:14:272	USER_DEBUG	[7]|DEBUG|The value of param1Component is: null
14:16:14:272	USER_DEBUG	[3]|DEBUG|I am ComponentExtension1 constructor
14:16:14:272	USER_DEBUG	[3]|DEBUG|I am ComponentExtension2 constructor