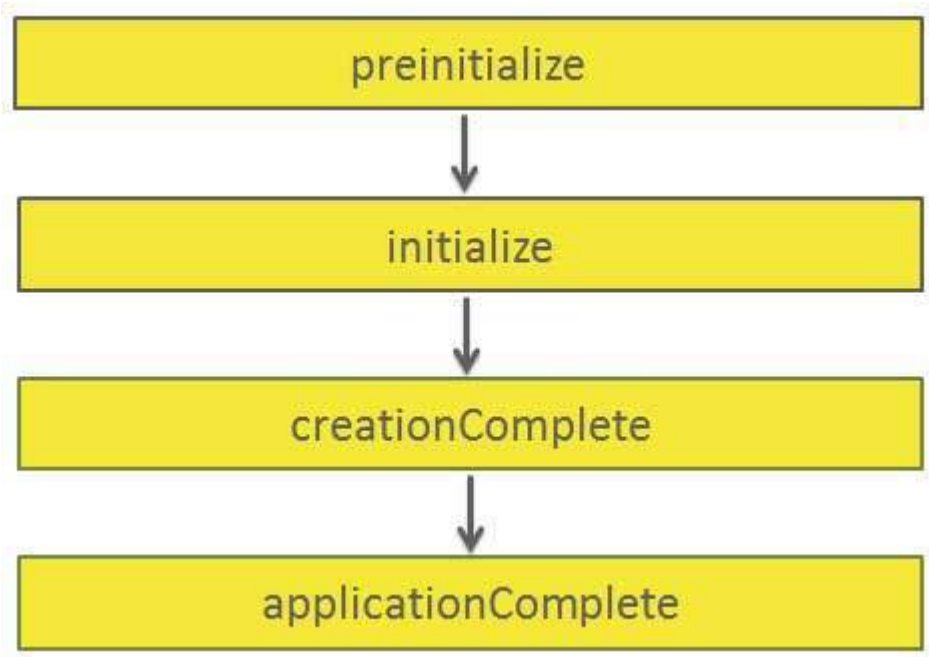
FLEX - LIFE CYCLE PHASES

Life Cycle of flex application:

Although you can build Flex applications without understanding life cycle phases of an application life cycle, but it is good to know the basic mechanism: the order in which things occur.It will help you configure features such as load other Flex applications at runtime, and manage the process of loading and unloading class libraries and assets at runtime.

A good understanding of the Flex application life cycle will enable you to build better applications and optimize them because you will know where to optimally run code. For example, if you need to ensure that some code runs during a preloader, you need to know where to place the code for that event.

When we loads flex application in browser the following events occurs during the lifeCycle of flex application.



Following is the brief detail about different Flex Life Cycle Events.

S.N.	Event & Description
1	**preInitialize:** mx.core.UIComponent.preinitialize **Event Type:** mx.events.FlexEvent.PREINITIALIZE This event is dispatched at the beginning of the component initialization sequence. The component is in a very raw state when this event is dispatched. Many components, such as Button control, creates internal child components to implement functionality. For example, the Button control creates an internal UITextField component to represent its label text. When Flex dispatches the preinitialize event, the children, including all the internal children, of a component have not yet been created.
2	**initialize:** mx.core.UIComponent.initialize **Event Type:** mx.events.FlexEvent.INITIALIZE

This event is dispatched after preinitialize phase. Flex framework initializes the internal structure of this component during this phase. This event automatically fires when the component is added to a parent.

you do not need to call initialize generally.

3

creationComplete: mx.core.UIComponent.creationComplete

Event Type: mx.events.FlexEvent.CREATION_COMPLETE

This event is dispatched when the component has finished its construction, property processing, measuring, layout, and drawing.

At this point, depending on its visible property, the component is not visible even though it has been drawn.

4

applicationComplete: spark.components.Application.applicationComplete

Event Type:mx.events.FlexEvent.APPLICATION_COMPLETE

Dispatched after the Application has been initialized, processed by the LayoutManager, and attached to the display list.

This is the last event of the application creation life cycle and signifies that application has been loaded completely.

Flex Life Cycle Example

Let us follow the following steps to test life cycle of a Flex application by creating a test application:

Step	Description
1	Create a project with a name *HelloWorld* under a package *com.tutorialspoint.client* as explained in the *Flex - Create Application* chapter.
2	Modify *HelloWorld.mxml* as explained below. Keep rest of the files unchanged.
3	Compile and run the application to make sure business logic is working as per the requirements.

Following is the content of the modified mxml file **src/com.tutorialspoint/HelloWorld.mxml**.

```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
   xmlns:s="library://ns.adobe.com/flex/spark"
   xmlns:mx="library://ns.adobe.com/flex/mx"
   width="100%" height="100%" minWidth="500" minHeight="500"
   initialize="reportEvent(event)"
   preinitialize="reportEvent(event)"
   creationComplete="reportEvent(event)"
   applicationComplete="reportEvent(event)">
   <fx:Style source="/com/tutorialspoint/client/Style.css"/>
   <fx:Script>
      <![CDATA[
         import mx.controls.Alert;
         import mx.events.FlexEvent;

         [Bindable]
         private var report:String = "";
```

```
            private function reportEvent(event:FlexEvent):void{
               report += "\n" + (event.type + " event occured at: "
               + getTimer() + " ms" + "\n");
            }
         ]]>
      </fx:Script>
      <s:BorderContainer width="500" height="500"
         styleName="container">
         <s:VGroup width="100%" height="100%" gap="50"
            horizontalAlign="center" verticalAlign="middle">
            <s:Label textAlign="center" width="100%"
            fontSize="40" color="0x777777" styleName="heading"
            text="Life Cycle Events Demonstration"/>
            <s:TextArea
            width="300" height="200">
            </s:TextArea>
         </s:VGroup>
      </s:BorderContainer>
</s:Application>
```

Once you are ready with all the changes done, let us compile and run the application in normal mode as we did in Flex - Create Application chapter. If everything is fine with your application, this will produce following result: [Try it online]