

Extending Components

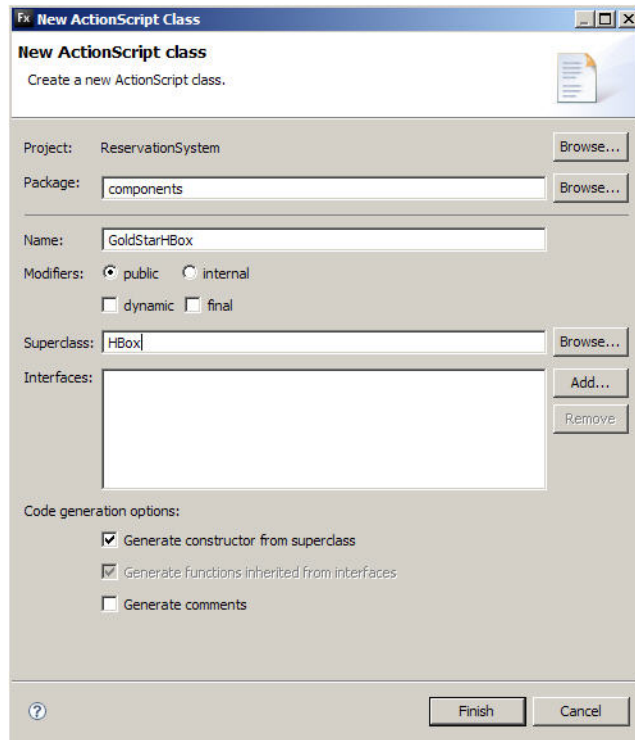
Import the project

1. In Flex Builder, select **File > Import > Flex Project**.
2. In the dialog window, select **Archive File** and browse to where **Ex19_Starter.zip** is located in your local file system.
3. Click **Finish**.

Note: Alternatively you can start with the code you completed from exercise 18.

Extend an HBox component

1. In the Flex Builder **Flex Navigation** view, right-click on the **components** folder and select **New > ActionScript Class**.
2. In the **New ActionScript Class** dialog window enter these values:
 - Package : `components`
 - Name: `GoldStarHBox`
 - Modifiers : `public`
 - Superclass : `HBox`
 - Generate constructor from superclass : `checked`



3. Click **Finish**.
4. After the class definition and before the constructor, create a `public` variable named `numStars` datatyped as `int` with a value of 0 (zero).

```
package components
{
    import mx.containers.HBox;

    public class GoldStarHBox extends HBox
    {
        public var numStars:int = 0;

        public function GoldStarHBox()
        {
            super();
        }
    }
}
```

5. Open the **GoldStar.mxml** file from the **components** folder.

6. Copy the `drawStar()` function and paste it into **GoldStarHBox.as** after the constructor.

```
public class GoldStarHBox extends HBox
{
    public var numStars:int = 0;

    public function GoldStarHBox()
    {
        super();
    }

    private function drawStar():void
    {
        . . .
    }
}
```

7. Within the function as the first line of code, create a local variable named `pos` datatyped as `int` having a value of 0 (zero). This variable is used to position each star.

```
private function drawStar():void
{
    var pos:int = 0;
    this.graphics.clear();
    . . .
}
```

4. After the `lineStyle()` method, create a for loop using the following statements:

- `var x:int = 0`
- `x < numStars`
- `x++`

```
for (var x:int = 0; x < numStars; x++)
{
}
}
```

5. Move the remainder lines of code to the loop.

```
for (var x:int = 0; x < numStars; x++){
    this.graphics.moveTo(9, 2);
    this.graphics.lineTo(5, 14);
}
```

```
        this.graphics.lineTo(16, 7);
        this.graphics.lineTo(3, 7);
        this.graphics.lineTo(14, 14);
        this.graphics.lineTo(9, 2);
    }
```

6. Change the first parameter of the `moveTo()` method to `pos + 9`.

```
        this.graphics.moveTo(pos + 9, 2);
```

8. Change the first parameter of all the remaining `lineTo()` methods to add `pos +`.

```
        this.graphics.lineTo(pos + 5, 14);
        this.graphics.lineTo(pos + 16, 7);
        this.graphics.lineTo(pos + 3, 7);
        this.graphics.lineTo(pos + 14, 14);
        this.graphics.lineTo(pos + 9, 2);
```

7. After the last `lineTo()` method, increment the `pos` variable by 20. You want the stars 20 pixels apart.

```
        pos += 20;
```

Your code should appear as follows:

```
private function drawStar():void
{
    var pos:int = 0;
    this.graphics.clear();
    this.graphics.beginFill(0xFFD700,1);
    this.graphics.lineStyle(2, 0xFFD700, 1, false,
"normal", "square", "miter", 8);

    for (var x:int = 0; x < numStars; x++){
        this.graphics.moveTo(pos + 9, 2);
        this.graphics.lineTo(pos + 5, 14);
        this.graphics.lineTo(pos + 16, 7);
        this.graphics.lineTo(pos + 3, 7);
        this.graphics.lineTo(pos + 14, 14);
        this.graphics.lineTo(pos + 9, 2);

        pos += 20;
    }
}
```

```
    }  
}
```

9. After the `drawStar()` method, override the protected function named `updateDisplayList()`. Pass the `unscaledWidth` and the `unscaledHeight` variables datatyped as `Number`. The function returns `void`. The `updateDisplayList()` is called when the class is initialized and subsequently whenever the component is manipulated in any way.

```
protected override function updateDisplayList  
(unscaledWidth:Number, unscaledHeight:Number):void  
{  
  
}
```

10. Within the function call the `super.updateDisplayList()` method and pass `unscaledWidth` and `unscaledHeight` as parameters.

```
    super.updateDisplayList(unscaledWidth, unscaledHeight);
```

11. Call the `drawStar()` method.

Your code should appear as follows:

```
protected override function updateDisplayList  
(unscaledWidth:Number, unscaledHeight:Number):void {  
    super.updateDisplayList(unscaledWidth,  
        unscaledHeight);  
  
    drawStar();  
}
```

12. Save the file.

Use the component

13. Open **RoomRenderer.mxml** from the **components** folder.
14. Using the **Outline** panel, locate the last `Text` control within the second `Canvas` container.
15. Add the `GoldStarHBox` component from the `comp` namespace.
16. Add an `x` property having a value of 0, a `y` property having a value of 271 and a `numStars` property bound to `data.stars`. You are passing the number of

stars the room should have which is contained in the data property of the item renderer.

```
<comp:GoldStarHBox x="0" y="271"  
    numStars="{data.stars}"/>
```

17. Save the file and run the application.
18. Log in and scroll the list of rooms.
Note that there are stars displayed with each room.

