

Display List & DOM Events

Ted Patrick

Flex Evangelist

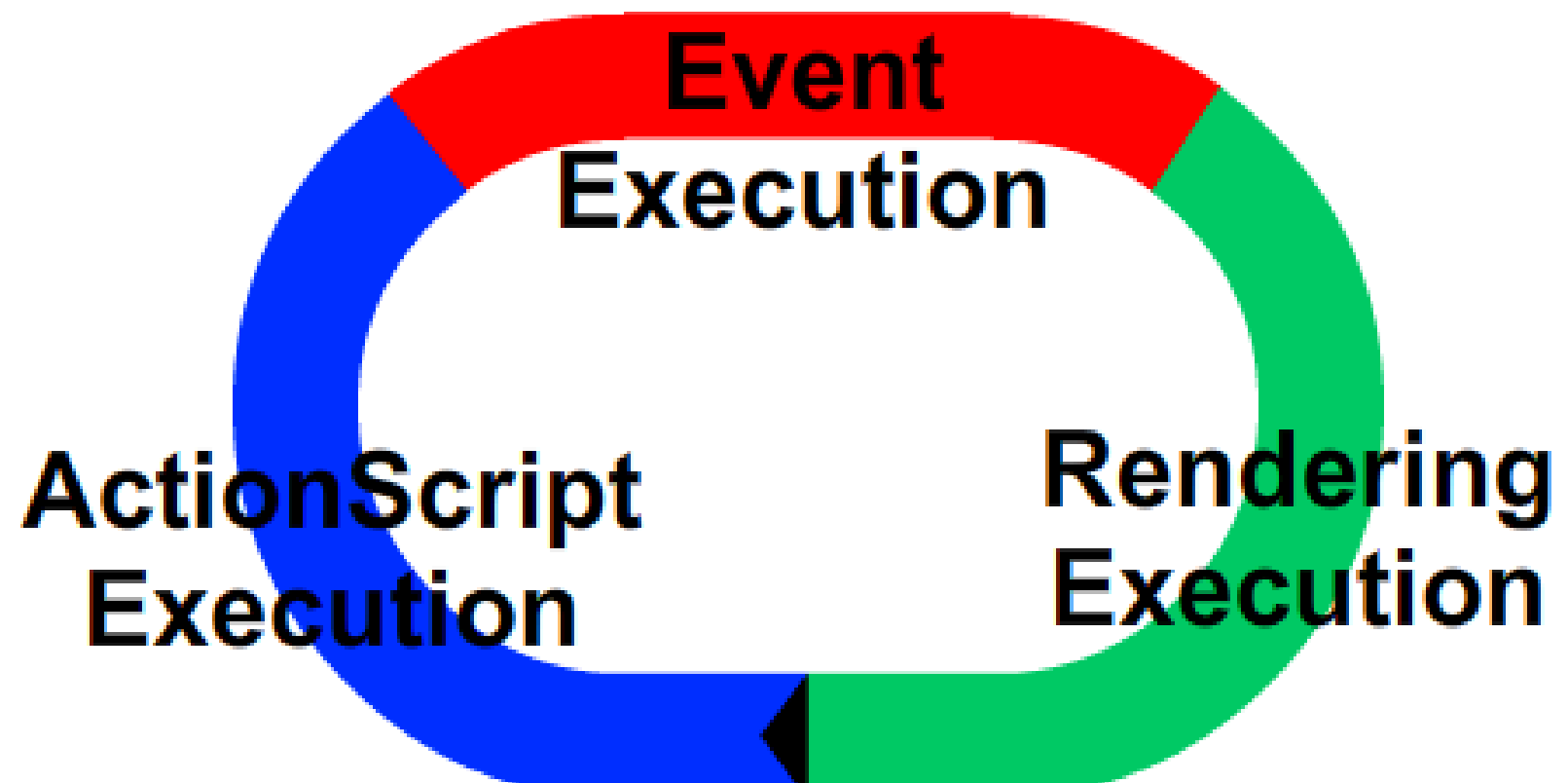
June 23, 2006



What is the Display List?

- The Display List is a Tree
- The Display List root is the Base Class
- Changes to DisplayObjects are rendered to screen every keyframe.

The RaceTrack Mental Model



AVM2 Changes

DisplayObjects are class instances

```
myCircle = new Circle();
```

API is simple and easy to use

```
this.addChild( myCircle );
```

One class instance, one location on the Display List

Events propagate through the Display List hierarchy

Reparenting

```
a = new Sprite( );  
b = new Sprite( );  
c = new Sprite( );  
addChild( a )  
a.addChild( b )  
b.addChild( c )
```

```
root  
|  
a  
|  
b  
|  
c
```

Reparenting

```
addChild( c )  
c.addChild( b )  
b.addChild( a )
```

```
root  
-  
c  
-  
b  
-  
a
```

Reparenting with UIObject

UIObject does not clean-up 'children' array
unless removeChild is called!!!

Call removeChild before addChild with UIObjects

Handy to know

//self removal

parent.removeChild(this);

//add to the front

addChild(childObj);

//add to the back

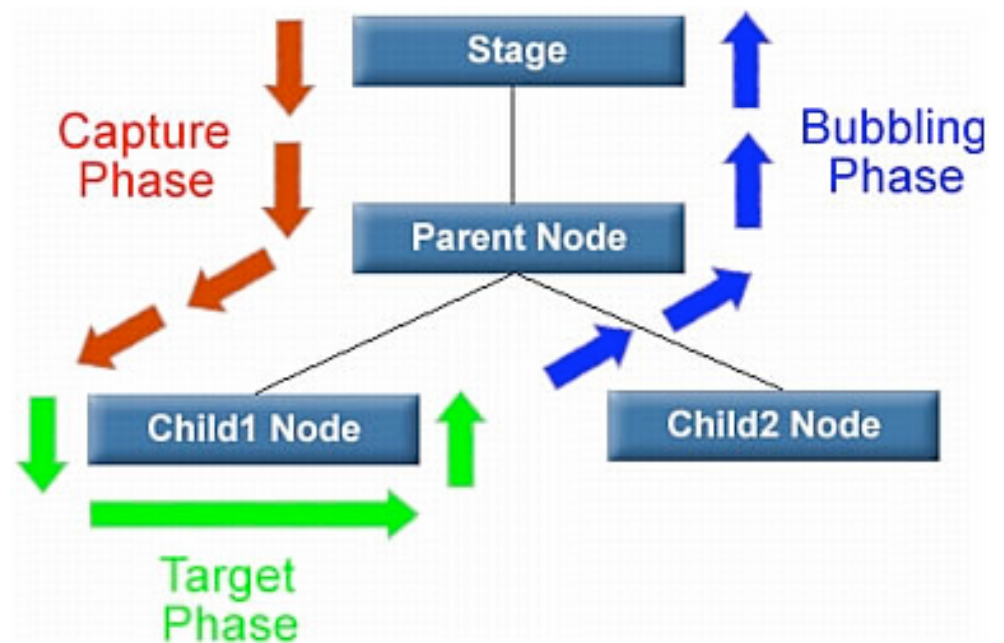
addChildAt(childObj , 0);

3 Phases of DOM Events

Capture – From root to target

Target – At target

Bubble – From target to root



Event API

Listening for an event

```
myDO.addEventListener( MouseEvent.CLICK, clickHandler );
```

Dispatching an event

```
var me = new MouseEvent( MouseEvent.CLICK, true, false)
```

```
myDO.dispatchEvent( me );
```

Capture Logic

- On MouseDown:
 - Subscribe to mouse capture events on System Manager
 - Cancel all mouse events (None shall Pass!!!)
- On Mouse Up:
 - Unsubscribe to System Manager

States in Components

- DisplayObject API enabled States
- States = Denote a series of UI changes w
 - **this.currentState = "closed"**
 - **this.currentState = "docked"**
- States are usable within Components and Applications