FLEX - UICOMPONENT CLASS

Introduction

The **UIComponent** class is the base class for all visual components, both interactive and noninteractive.

Class declaration

Following is the declaration for **mx.core.UIComponent** class:

```
public class UIComponent
   extends FlexSprite
      implements IAutomationObject, IChildList, IConstraintClient,
         IDeferredInstantiationUIComponent, IFlexDisplayObject,
            IFlexModule,IInvalidating, ILayoutManagerClient,
               IPropertyChangeNotifier,IRepeaterClient, IStateClient,
                  IAdvancedStyleClient, IToolTipManagerClient,
                     IUIComponent, IValidatorListener, IVisualElement
```

Public Properties

Following are the public properties for **mx.core.UIComponent** class:

S.N.	Name & Description
1	**accessibilityDescription : String** A convenience accessor for the description property in this UIComponent's accessibilityProperties object.
2	**accessibilityEnabled : Boolean** A convenience accessor for the silent property in this UIComponent's accessibilityProperties object.
3	**accessibilityName : String** A convenience accessor for the name property in this UIComponent's accessibilityProperties object.
4	**accessibilityShortcut : String** A convenience accessor for the shortcut property in this UIComponent's accessibilityProperties object.
5	**activeEffects : Array** [read-only] The list of effects that are currently playing on the component, as an Array of EffectInstance instances.
6	**automationDelegate : Object** The delegate object that handles the automation-related functionality.
7	**automationEnabled : Boolean** [read-only] True if this component is enabled for automation, false otherwise.
8	**automationName : String** Name that can be used as an identifier for this object.
9	**automationOwner : DisplayObjectContainer** [read-only] The owner of this component for automation purposes.
10	**automationParent : DisplayObjectContainer** [read-only] The parent of this component for automation purposes.
11	**automationTabularData : Object** [read-only] An implementation of the IAutomationTabularData interface, which can be used to retrieve the data.
12	**automationValue : Array** [read-only] This value generally corresponds to the rendered appearance of the object and should be usable for correlating the identifier with the object as it appears visually within the application.
13	**automationVisible : Boolean** [read-only] True if this component is visible for automation, false otherwise.
14	**baseline : Object** For components, this layout constraint property is a facade on top of the similarly-named style.
15	**baselinePosition : Number** [read-only] The y-coordinate of the baseline of the first line of text of the component.
16	**bottom : Object** For components, this layout constraint property is a facade on top of the similarly-named style.
17	**cacheHeuristic : Boolean** [write-only] Used by Flex to suggest bitmap caching for the object.
18	**cachePolicy : String** Specifies the bitmap caching policy for this object.
19	**className : String**

[read-only] The name of this instance's class, such as "Button".

20
contentMouseX : Number

[read-only] Returns the x position of the mouse, in the content coordinate system.

21
contentMouseY : Number

[read-only] Returns the y position of the mouse, in the content coordinate system.

22
currentState : String

The current view state of the component.

23
cursorManager : ICursorManager

[read-only] Gets the CursorManager that controls the cursor for this component and its peers.

24
depth : Number

Determines the order in which items inside of containers are rendered.

25
descriptor : UIComponentDescriptor

Reference to the UIComponentDescriptor, if any, that was used by the createComponentFromDescriptor method to create this UIComponent instance.

26
designLayer : DesignLayer

Specifies the optional DesignLayer instance associated with this visual element.

27
document : Object

A reference to the document object associated with this UIComponent.

28
doubleClickEnabled : Boolean

[override] Specifies whether the UIComponent object receives doubleClick events.

29
enabled : Boolean

Whether the component can accept user interaction.

30
errorString : String

The text that displayed by a component's error tip when a component is monitored by a Validator and validation fails.

31
explicitHeight : Number

Number that specifies the explicit height of the component, in pixels, in the component's coordinates.

32
explicitMaxHeight : Number

The maximum recommended height of the component to be considered by the parent during layout.

33
explicitMaxWidth : Number

The maximum recommended width of the component to be considered by the parent during layout.

34
explicitMinHeight : Number

The minimum recommended height of the component to be considered by the parent during layout.

35
explicitMinWidth : Number

The minimum recommended width of the component to be considered by the parent during layout.

36
explicitWidth : Number

Number that specifies the explicit width of the component, in pixels, in the component's coordinates.

37
flexContextMenu : IFlexContextMenu

The context menu for this UIComponent.

38
focusEnabled : Boolean

Indicates whether the component can receive focus when tabbed to.

39
focusManager : IFocusManager

Gets the FocusManager that controls focus for this component and its peers.

40
focusPane : Sprite

The focus pane associated with this object.

41
hasFocusableChildren : Boolean

A flag that indicates whether child objects can receive focus.

42
hasLayoutMatrix3D : Boolean

[read-only] Contains true if the element has 3D Matrix.

43
height : Number

[override] Number that specifies the height of the component, in pixels, in the parent's coordinates.

44
horizontalCenter : Object

For components, this layout constraint property is a facade on top of the similarly-named style.

45
id : String

ID of the component.

46
includeInLayout : Boolean

Specifies whether this component is included in the layout of the parent container.

47
inheritingStyles : Object

The beginning of this component's chain of inheriting styles.

48
initialized : Boolean

A flag that determines if an object has been through all three phases of layout: commitment, measurement, and layout *providedthatanywererequired*.

49
instanceIndex : int

[read-only] The index of a repeated component.

50
instanceIndices : Array

An Array containing the indices required to reference this UIComponent object from its parent document.

51
is3D : Boolean

[read-only] Contains true when the element is in 3D.

52
isDocument : Boolean

[read-only] Contains true if this UIComponent instance is a document object.

53
isPopUp : Boolean

Set to true by the PopUpManager to indicate that component has been popped up.

54
layoutMatrix3D : Matrix3D

[write-only] The transform matrix that is used to calculate a component's layout relative to its siblings.

55
left : Object

For components, this layout constraint property is a facade on top of the similarly-named style.

56
maintainProjectionCenter : Boolean

When true, the component keeps its projection matrix centered on the middle of its bounding box.

57
maxHeight : Number

The maximum recommended height of the component to be considered by the parent during layout.

58
maxWidth : Number

The maximum recommended width of the component to be considered by the parent during layout.

59
measuredHeight : Number

The default height of the component, in pixels.

60
measuredMinHeight : Number

The default minimum height of the component, in pixels.

61
measuredMinWidth : Number

The default minimum width of the component, in pixels.

62
measuredWidth : Number

The default width of the component, in pixels.

63
minHeight : Number

The minimum recommended height of the component to be considered by the parent during layout.

64
minWidth : Number

The minimum recommended width of the component to be considered by the parent during layout.

65
moduleFactory : IFlexModuleFactory

A module factory is used as context for using embedded fonts and for finding the style manager that controls the styles for this component.

66
mouseFocusEnabled : Boolean

Whether you can receive focus when clicked on.

67
nestLevel : int

Depth of this object in the containment hierarchy.

68
nonInheritingStyles : Object

The beginning of this component's chain of non-inheriting styles.

69
numAutomationChildren : int

[read-only] The number of automation children this container has.

70
owner : DisplayObjectContainer

The owner of this IVisualElement object.

71
parent : DisplayObjectContainer

[override] [read-only] The parent container or component for this component.

72
parentApplication : Object

[read-only] A reference to the Application object that contains this UIComponent instance.

73
parentDocument : Object

[read-only] A reference to the parent document object for this UIComponent.

74
percentHeight : Number

Specifies the height of a component as a percentage of its parent's size.

75
percentWidth : Number

Specifies the width of a component as a percentage of its parent's size.

76
postLayoutTransformOffsets : mx.geom:TransformOffsets

Defines a set of adjustments that can be applied to the object's transform in a way that is invisible to its parent's layout.

77
processedDescriptors : Boolean

Set to true after immediate or deferred child creation, depending on which one happens.

78
repeater : IRepeater

[read-only] A reference to the Repeater object in the parent document that produced this UIComponent.

79
repeaterIndex : int

[read-only] The index of the item in the data provider of the Repeater that produced this UIComponent.

80
repeaterIndices : Array

An Array containing the indices of the items in the data provider of the Repeaters in the parent document that produced this UIComponent.

81
repeaters : Array

An Array containing references to the Repeater objects in the parent document that produced this UIComponent.

82
right : Object

For components, this layout constraint property is a facade on top of the similarly-named style.

83
rotation : Number

[override] Indicates the rotation of the DisplayObject instance, in degrees, from its original orientation.

84
rotationX : Number

[override] Indicates the x-axis rotation of the DisplayObject instance, in degrees, from its original orientation relative to the 3D parent container.

85
rotationY : Number

[override] Indicates the y-axis rotation of the DisplayObject instance, in degrees, from its original orientation relative to the 3D parent container.

86
rotationZ : Number

[override] Indicates the z-axis rotation of the DisplayObject instance, in degrees, from its original orientation relative to the 3D parent container.

87
scaleX : Number

[override] Number that specifies the horizontal scaling factor.

88
scaleY : Number

[override] Number that specifies the vertical scaling factor.

89
scaleZ : Number

[override] Number that specifies the scaling factor along the z axis.

90
screen : Rectangle

[read-only] Returns an object that contains the size and position of the base drawing surface for this object.

91

showInAutomationHierarchy : Boolean

A flag that determines if an automation object shows in the automation hierarchy.

92
states : Array

The view states that are defined for this component.

93
styleDeclaration : CSSStyleDeclaration

Storage for the inline inheriting styles on this object.

94
styleManager : IStyleManager2

[read-only] Returns the StyleManager instance used by this component.

95
styleName : Object

The class style used by this component.

96
styleParent : IAdvancedStyleClient

A component's parent is used to evaluate descendant selectors.

97
systemManager : ISystemManager

Returns the SystemManager object used by this component.

98
tabFocusEnabled : Boolean

A flag that indicates whether this object can receive focus via the TAB key This is similar to the tabEnabled property used by the Flash Player. This is usually true for components that handle keyboard input, but some components in controlbars have them set to false because they should not steal focus from another component like an editor.

99
toolTip : String

Text to display in the ToolTip.

100
top : Object

For components, this layout constraint property is a facade on top of the similarly-named style.

101
transform : flash.geom:Transform

[override] An object with properties pertaining to a display object's matrix, color transform, and pixel bounds.

102
transformX : Number

Sets the x coordinate for the transform center of the component.

103
transformY : Number

Sets the y coordinate for the transform center of the component.

104
transformZ : Number

Sets the z coordinate for the transform center of the component.

105
transitions : Array

An Array of Transition objects, where each Transition object defines a set of effects to play when a view state change occurs.

106
tweeningProperties : Array

Array of properties that are currently being tweened on this object.

107
uid : String

A unique identifier for the object.

108
updateCompletePendingFlag : Boolean

A flag that determines if an object has been through all three phases of layout validation *providedthatanywererequired*.

109
validationSubField : String

Used by a validator to associate a subfield with this component.

110
verticalCenter : Object

For components, this layout constraint property is a facade on top of the similarly-named style.

111
visible : Boolean

[override] Whether or not the display object is visible.

112
width : Number

[override] Number that specifies the width of the component, in pixels, in the parent's coordinates.

113
x : Number

[override] Number that specifies the component's horizontal position, in pixels, within its parent container.

114
y : Number

[override] Number that specifies the component's vertical position, in pixels, within its parent container.

115	**z : Number** [override] Indicates the z coordinate position along the z-axis of the DisplayObject instance relative to the 3D parent container.

Protected Properties

Following are the protected properties for **mx.core.UIComponent** class:

S.N.	Name & Description
1	**currentCSSState : String** [read-only] The state to be used when matching CSS pseudo-selectors.
2	**hasComplexLayoutMatrix : Boolean** [read-only] Returns true if the UIComponent has any non-translation *x, y* transform properties.
3	**resourceManager : IResourceManager** [read-only] A reference to the object which manages all of the application's localized resources.
4	**unscaledHeight : Number** [read-only] A convenience method for determining the unscaled height of the component.
5	**unscaledWidth : Number** [read-only] A convenience method for determining the unscaled width of the component All of a component's drawing and child layout should be done within a bounding rectangle of this width, which is also passed as an argument to updateDisplayList.

S.N.	Event & Description
1	**activate** Dispatched when the Flash Player gains operating system focus and becomes active.
2	**detivate** Dispatched when the Flash Player loses operating system focus and becomes inactive.

Public methods

S.N.	Method & Description
1	**UIComponent** Constructor.
2	**addStyleClient***styleClient: IAdvancedStyleClient***:void** Adds a non-visual style client to this component instance.
3	**callLater***method: Function, args: Array = null***:void** Queues a function to be called later.
4	**clearStyle***styleProp: String***:void** Deletes a style property from this component instance.
5	**contentToGlobal***point: Point***:Point** Converts a Point object from content coordinates to global coordinates.
6	**contentToLocal***point: Point***:Point** Converts a Point object from content to local coordinates.
7	**createAutomationIDPart***child: IAutomationObject***:Object** Returns a set of properties that identify the child within this container.
8	**createAutomationIDPartWithRequiredProperties***child: IAutomationObject, properties: Array***:Object** Returns a set of properties that identify the child within this container.
9	**createReferenceOnParentDocument***parentDocument: IFlexDisplayObject***:void** Creates an id reference to this IUIComponent object on its parent document object.
10	**deleteReferenceOnParentDocument***parentDocument: IFlexDisplayObject***:void** Deletes the id reference to this IUIComponent object on its parent document object.
11	**determineTextFormatFromStyles:mx.core:UITextFormat** Returns a UITextFormat object corresponding to the text styles for this UIComponent.
12	**dispatchEvent***event: Event***:Boolean** [override] Dispatches an event into the event flow.
13	**drawFocus***isFocused: Boolean***:void** Shows or hides the focus indicator around this component.
14	**drawRoundRect***x: Number, y: Number, w: Number, h: Number, r: Object = null, c: Object = null, alpha: Object = null, rot: Object = null, gradient: String = null, ratios: Array = null, hole: Object = null***:void**

Programmatically draws a rectangle into this skin's Graphics object.

15

effectFinished*effectInst: IEffectInstance***:void**

Called by the effect instance when it stops playing on the component.

16

effectStarted*effectInst: IEffectInstance***:void**

Called by the effect instance when it starts playing on the component.

17

endEffectsStarted:void

Ends all currently playing effects on the component.

18

executeBindings*recurse: Boolean = false***:void**

Executes all the bindings for which the UIComponent object is the destination.

19

finishPrint*obj: Object, target: IFlexDisplayObject***:void**

Called after printing is complete.

20

getAutomationChildAt*index: int***:IAutomationObject**

Provides the automation object at the specified index.

21

getAutomationChildren:Array

Provides the automation object list .

22

getBoundsXAtSize*width: Number, height: Number, postLayoutTransform: Boolean = true***:Number**

Returns the x coordinate of the element's bounds at the specified element size.

23

getBoundsYAtSize*width: Number, height: Number, postLayoutTransform: Boolean = true***:Number**

Returns the y coordinate of the element's bounds at the specified element size.

24

getClassStyleDeclarations:Array

Finds the type selectors for this UIComponent instance.

25

getConstraintValue*constraintName: String***:***

Returns a layout constraint value, which is the same as getting the constraint style for this component.

26

getExplicitOrMeasuredHeight:Number

A convenience method for determining whether to use the explicit or measured height

27

getExplicitOrMeasuredWidth:Number

A convenience method for determining whether to use the explicit or measured width

28

getFocus:InteractiveObject

Gets the object that currently has focus.

29

getLayoutBoundsHeight*postLayoutTransform: Boolean = true***:Number**

Returns the element's layout height.

30

getLayoutBoundsWidth*postLayoutTransform: Boolean = true***:Number**

Returns the element's layout width.

31

getLayoutBoundsX*postLayoutTransform: Boolean = true***:Number**

Returns the x coordinate that the element uses to draw on screen.

32

getLayoutBoundsY*postLayoutTransform: Boolean = true***:Number**

Returns the y coordinate that the element uses to draw on screen.

33

getLayoutMatrix:Matrix

Returns the transform matrix that is used to calculate the component's layout relative to its siblings.

34

getLayoutMatrix3D:Matrix3D

Returns the layout transform Matrix3D for this element.

35

getMaxBoundsHeight*postLayoutTransform: Boolean = true***:Number**

Returns the element's maximum height.

36

getMaxBoundsWidth*postLayoutTransform: Boolean = true***:Number**

Returns the element's maximum width.

37

getMinBoundsHeight*postLayoutTransform: Boolean = true***:Number**

Returns the element's minimum height.

38

getMinBoundsWidth*postLayoutTransform: Boolean = true***:Number**

Returns the element's minimum width.

39

getPreferredBoundsHeight*postLayoutTransform: Boolean = true***:Number**

Returns the element's preferred height.

40

getPreferredBoundsWidth*postLayoutTransform: Boolean = true***:Number**

Returns the element's preferred width.

41
getRepeaterItem*whichRepeater: int = -1***:Object**

Returns the item in the dataProvider that was used by the specified Repeater to produce this Repeater, or null if this Repeater isn't repeated.

42
getStyle*styleProp: String***:***

Gets a style property that has been set anywhere in this component's style lookup chain.

43
globalToContent*point: Point***:Point**

Converts a Point object from global to content coordinates.

45
hasCSSState:Boolean

Returns true if currentCSSState is not null.

46
hasState*stateName: String***:Boolean**

Determines whether the specified state has been defined on this UIComponent.

47
horizontalGradientMatrix*x: Number, y: Number, width: Number, height: Number***:Matrix**

Returns a box Matrix which can be passed to the drawRoundRect method as the rot parameter when drawing a horizontal gradient.

48
initialize:void

Initializes the internal structure of this component.

49
initializeRepeaterArrays*parent: IRepeaterClient***:void**

Initializes various properties which keep track of repeated instances of this component.

50
invalidateDisplayList:void

Marks a component so that its updateDisplayList method gets called during a later screen update.

51
invalidateLayering:void

Called by a component's items to indicate that their depth property has changed.

52
invalidateLayoutDirection:void

An element must call this method when its layoutDirection changes or when its parent's layoutDirection changes.

53
invalidateProperties:void

Marks a component so that its commitProperties method gets called during a later screen update.

54
invalidateSize:void

Marks a component so that its measure method gets called during a later screen update.

55
localToContent*point: Point***:Point**

Converts a Point object from local to content coordinates.

56
matchesCSSState*cssState: String***:Boolean**

Returns true if cssState matches currentCSSState.

57
matchesCSSType*cssType: String***:Boolean**

Determines whether this instance is the same as, or is a subclass of, the given type.

58
measureHTMLText*htmlText: String***:flash.text:TextLineMetrics**

Measures the specified HTML text, which can contain HTML tags such as &lt;font&> and &<b&>, assuming that it is displayed in a single-line UITextField using a UITextFormat determined by the styles of this UIComponent.

59
measureText*text: String***:flash.text:TextLineMetrics**

Measures the specified text, assuming that it is displayed in a single-line UITextField *orUIFTETextField* using a UITextFormat determined by the styles of this UIComponent.

60
move*x: Number, y: Number***:void**

Moves the component to a specified position within its parent.

61
notifyStyleChangeInChildren*styleProp: String, recursive: Boolean***:void**

Propagates style changes to the children.

62
owns*child: DisplayObject***:Boolean**

Returns true if the chain of owner properties points from child to this UIComponent.

63
parentChanged*p: DisplayObjectContainer***:void**

Called by Flex when a UIComponent object is added to or removed from a parent.

64
prepareToPrint*target: IFlexDisplayObject***:Object**

Prepares an IFlexDisplayObject for printing.

65
regenerateStyleCache*recursive: Boolean***:void**

Builds or rebuilds the CSS style cache for this component and, if the recursive parameter is true, for all descendants of this component as well.

66
registerEffects*effects: Array***:void**

For each effect event, registers the EffectManager as one of the event listeners.

67
removeStyleClient*styleClient*: *IAdvancedStyleClient***:void**

Removes a non-visual style client from this component instance.

68
replayAutomatableEvent*event*: *Event***:Boolean**

Replays the specified event.

69
resolveAutomationIDPart*criteria*: *Object***:Array**

Resolves a child by using the id provided.

70
resumeBackgroundProcessing:void

[static] Resumes the background processing of methods queued by callLater, after a call to suspendBackgroundProcessing.

71
setActualSize*w*: *Number*, *h*: *Number***:void**

Sizes the object.

72
setConstraintValue*constraintName*: *String*, *value*: * **:void**

Sets a layout constraint value, which is the same as setting the constraint style for this component.

73
setCurrentState*stateName*: *String*, *playTransition*: *Boolean = true***:void**

Set the current state.

74
setFocus:void

Sets the focus to this component.

75
setLayoutBoundsPosition*x*: *Number*, *y*: *Number*, *postLayoutTransform*: *Boolean = true***:void**

Sets the coordinates that the element uses to draw on screen.

76
set Layout Bounds Size*width*: *Number*, *height*: *Number*, *postLayoutTransform*: *Boolean = true***:void**

Sets the layout size of the element.

77
setLayoutMatrix*value*: *Matrix*, *invalidateLayout*: *Boolean***:void**

Sets the transform Matrix that is used to calculate the component's layout size and position relative to its siblings.

78
setLayoutMatrix3D*value*: *Matrix3D*, *invalidateLayout*: *Boolean***:void**

Sets the transform Matrix3D that is used to calculate the component's layout size and position relative to its siblings.

79
setStyle*styleProp*: *String*, *newValue*: * **:void**

Sets a style property on this component instance.

80
setVisible*value*: *Boolean*, *noEvent*: *Boolean = false***:void**

Called when the visible property changes.

81
styleChanged*styleProp*: *String***:void**

Detects changes to style properties.

82
stylesInitialized:void

Flex calls the stylesInitialized method when the styles for a component are first initialized.

83
suspendBackgroundProcessing:void

[static] Blocks the background processing of methods queued by callLater, until resumeBackgroundProcessing is called.

84
transformAround
transformCenter: *Vector3D*, *scale*: *Vector3D = null*, *rotation*: *Vector3D = null*, *translation*: *Vector3D = null*, *postLayoutScale*: *Vector3D = null*, *postLayoutRotation*: *Vector3D = null*, *postLayoutTranslation*: *Vector3D = null*, *invalidateLayout*: *Boolean = true*
:void

A utility method to update the rotation, scale, and translation of the transform while keeping a particular point, specified in the component's own coordinate space, fixed in the parent's coordinate space.

85
transform Point To Parent *localPosition*: *Vector3D*, *position*: *Vector3D*, *postLayoutPosition*: *Vector3D***:void**

A utility method to transform a point specified in the local coordinates of this object to its location in the object's parent's coordinates.

86
validateDisplayList:void

Validates the position and size of children and draws other visuals.

87
validateNow:void

Validate and update the properties and layout of this object and redraw it, if necessary.

88
validateProperties:void

Used by layout logic to validate the properties of a component by calling the commitProperties method.

89
validateSize*recursive*: *Boolean = false***:void**

Validates the measured size of the component If the LayoutManager.invalidateSize method is called with this ILayoutManagerClient, then the validateSize method is called when it's time to do measurements.

90
validationResultHandler*event*: *ValidationResultEvent***:void**

Handles both the valid and invalid events from a validator assigned to this component.

91
verti cal Gradient Matrix*x*: *Number*, *y*: *Number*, *width*: *Number*, *height*: *Number***:Matrix**

Returns a box Matrix which can be passed to drawRoundRect as the rot parameter when drawing a vertical gradient.

Protected methods

S.N.	Method & Description
1	**adjustFocusRect***obj: DisplayObject = null***:void** Adjust the focus rectangle.
2	**applyComputedMatrix:void** Commits the computed matrix built from the combination of the layout matrix and the transform offsets to the flash displayObject's transform.
3	**attachOverlay:void** This is an internal method used by the Flex framework to support the Dissolve effect.
4	**canSkipMeasurement:Boolean** Determines if the call to the measure method can be skipped.
5	**childrenCreated:void** Performs any final processing after child objects are created.
6	**commitProperties:void** Processes the properties set on the component.
7	**createChildren:void** Create child objects of the component.
8	**createInFontContext***classObj: Class***:Object** Creates a new object using a context based on the embedded font being used.
9	**createInModuleContext***moduleFactory: IFlexModuleFactory, className: String***:Object** Creates the object using a given moduleFactory.
10	**dispatchPropertyChangeEvent***prop: String, oldValue: *, value: ****:void** Helper method for dispatching a PropertyChangeEvent when a property is updated.
11	**focusInHandler***event: FocusEvent***:void** The event handler called when a UIComponent object gets focus.
12	**focusOutHandler***event: FocusEvent***:void** The event handler called when a UIComponent object loses focus.
13	**initAdvancedLayoutFeatures:void** Initializes the implementation and storage of some of the less frequently used advanced layout features of a component.
14	**initializationComplete:void** Finalizes the initialization of this component.
15	**initializeAccessibility:void** Initializes this component's accessibility code.
16	**invalidateParentSizeAndDisplayList:void** Helper method to invalidate parent size and display list if this object affects its layout *includeInLayoutistrue*.
17	**isOurFocus***target: DisplayObject***:Boolean** Typically overridden by components containing UITextField objects, where the UITextField object gets focus.
18	**keyDownHandler***event: KeyboardEvent***:void** The event handler called for a keyDown event.
19	**keyUpHandler***event: KeyboardEvent***:void** The event handler called for a keyUp event.
20	**measure:void** Calculates the default size, and optionally the default minimum size, of the component.
21	**resourcesChanged:void** This method is called when a UIComponent is constructed, and again whenever the ResourceManager dispatches a "change" Event to indicate that the localized resources have changed in some way.
22	**setStretchXY***stretchX: Number, stretchY: Number***:void** Specifies a transform stretch factor in the horizontal and vertical direction.
23	**stateChanged***oldState: String, newState: String, recursive: Boolean***:void** This method is called when a state changes to check whether state-specific styles apply to this component
24	**updateDisplayList***unscaledWidth: Number, unscaledHeight: Number***:void** Draws the object and/or sizes and positions its children.

Events

Following are the events for **mx.core.UIComponent** class:

S.N.	Event & Description
1	**add** when the component is added to a container as a content child by using the addChild, addChildAt, addElement, or addElementAt method.
2	**creationComplete** when the component has finished its construction, property processing, measuring, layout, and drawing.
3	**currentStateChange** after the view state has changed.
4	**currentStateChanging** after the currentState property changes, but before the view state changes.
5	**dragComplete** by the drag initiator *thecomponentthatisthesourceofthedatabeingdragged* when the drag operation completes, either when you drop the dragged data onto a drop target or when you end the drag-and-drop operation without performing a drop.
6	**dragDrop** by the drop target when the user releases the mouse over it.
7	**dragEnter** by a component when the user moves the mouse over the component during a drag operation.
8	**dragExit** by the component when the user drags outside the component, but does not drop the data onto the target.
9	**dragOver** by a component when the user moves the mouse while over the component during a drag operation.
10	**dragStart** by the drag initiator when starting a drag operation.
11	**effectEnd** after an effect ends.
12	**effectStart** just before an effect starts.
13	**effectStop** after an effect is stopped, which happens only by a call to stop on the effect.
14	**enterState** after the component has entered a view state.
15	**exitState** just before the component exits a view state.
16	**hide** when an object's state changes from visible to invisible.
17	**initialize** when the component has finished its construction and has all initialization properties set.
18	**invalid** when a component is monitored by a Validator and the validation failed.
19	**mouseDownOutside** from a component opened using the PopUpManager when the user clicks outside it.
20	**mouseWheelOutside** from a component opened using the PopUpManager when the user scrolls the mouse wheel outside it.
21	**move** when the object has moved.
22	**preinitialize** at the beginning of the component initialization sequence.
23	**remove** when the component is removed from a container as a content child by using the removeChild, removeChildAt, removeElement, or removeElementAt method.

24
resize

when the component is resized.

25
show

when an object's state changes from invisible to visible.

26
stateChangeComplete

after the component has entered a new state and any state transition animation to that state has finished playing.

27
stateChangeInterrupted

when a component interrupts a transition to its current state in order to switch to a new state.

28
toolTipCreate

by the component when it is time to create a ToolTip.

29
toolTipEnd

by the component when its ToolTip has been hidden and is to be discarded soon.

30
toolTipHide

by the component when its ToolTip is about to be hidden.

31
toolTipShow

by the component when its ToolTip is about to be shown.

32
toolTipShown

by the component when its ToolTip has been shown.

33
toolTipStart

by a component whose toolTip property is set, as soon as the user moves the mouse over it.

34
touchInteractionEnd

A non-cancellable event, by a component when it is done responding to a touch interaction user gesture.

35
touchInteractionStart

A non-cancellable event, by a component when it starts responding to a touch interaction user gesture.

36
touchInteractionStarting

A cancellable event, by a component in an attempt to respond to a touch interaction user gesture.

37
updateComplete

when an object has had its commitProperties, measure, and updateDisplayList methods called *ifneeded*.

38
valid

when a component is monitored by a Validator and the validation succeeded.

39
valueCommit

when values are changed programmatically or by user interaction.

Methods inherited

This class inherits methods from the following classes:

- mx.core.FlexSprite
- flash.display.Sprite
- flash.display.DisplayObjectContainer
- flash.display.InteractiveObject
- flash.display.DisplayObject
- flash.events.EventDispatcher
- Object