Flex Viewer Widget Communication Explained

In Flex Viewer there is a built in widget communication/messaging capability that is very useful for passing data from one widget to another, but it is a aspect of the Flex Viewer that in un documented.

This document will attempt to explain in detail the methods used in widget messaging and what is going on behind the scenes. I will also cover advanced topics such as opening one widget from another so you can message data to that widget.

I'll begin by explaining that widget messaging is accomplished by dispatching and listening for AppEvents. Widget messaging is managed using the DataManager.as. The DataManager listens for AppEvent.DATA_FETCH_ALL, AppEvent.DATA_PUBLISH and AppEvent.DATA_FETCH.

The below model is the flow of data when Widget-A and Widget-B are both open and Widget-B is listening for AppEvent.DATA_PUBLISH

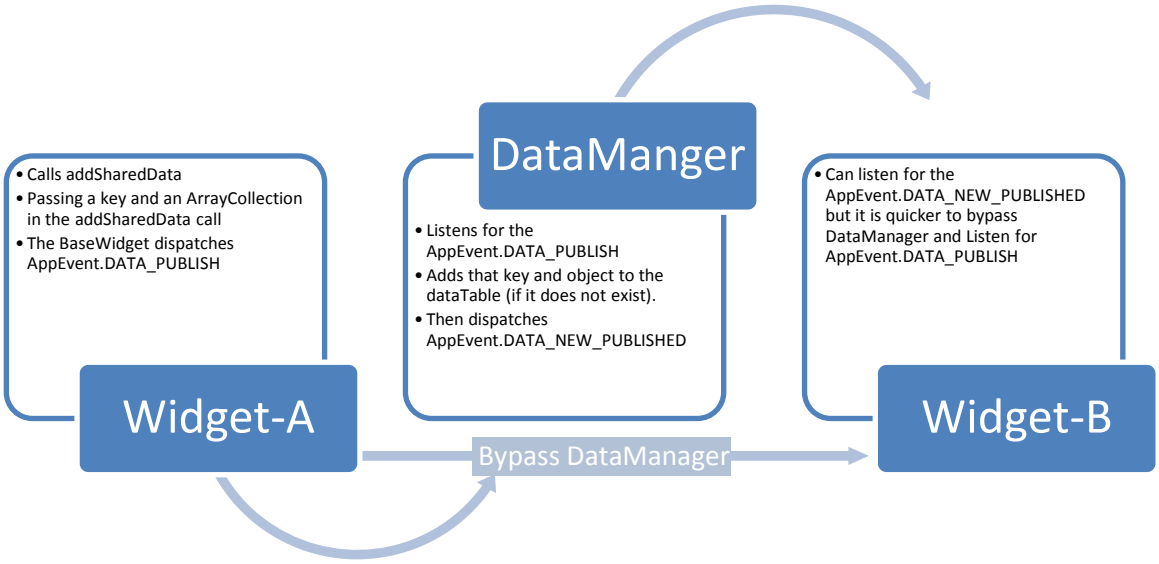

Figure 1

If Widget A calls addSharedData and Widget-B is not opened yet than listening for AppEvent.DATA_PUBLISH will not catch the event because it has already been dispatched. In this case you will have to call fetchSharedData in Widget-B and listen for AppEvent.DATA_SENT. There is a big difference between what is returned by AppEvent.DATA_PUBLISH and AppEvent.DATA_SENT. AppEvent.DATA_PUBLISH will have an ArrayCollection as the event.data and AppEvent.DATA_SENT will have a HashTable containing multiple ArrayCollections that have unique keys to identify them.

The below diagram is the flow of events when you need to message data from Widget-A to Widget-B when Widget-B is not yet initialized (opened)

Prepared By: Robert J. Scheitlin

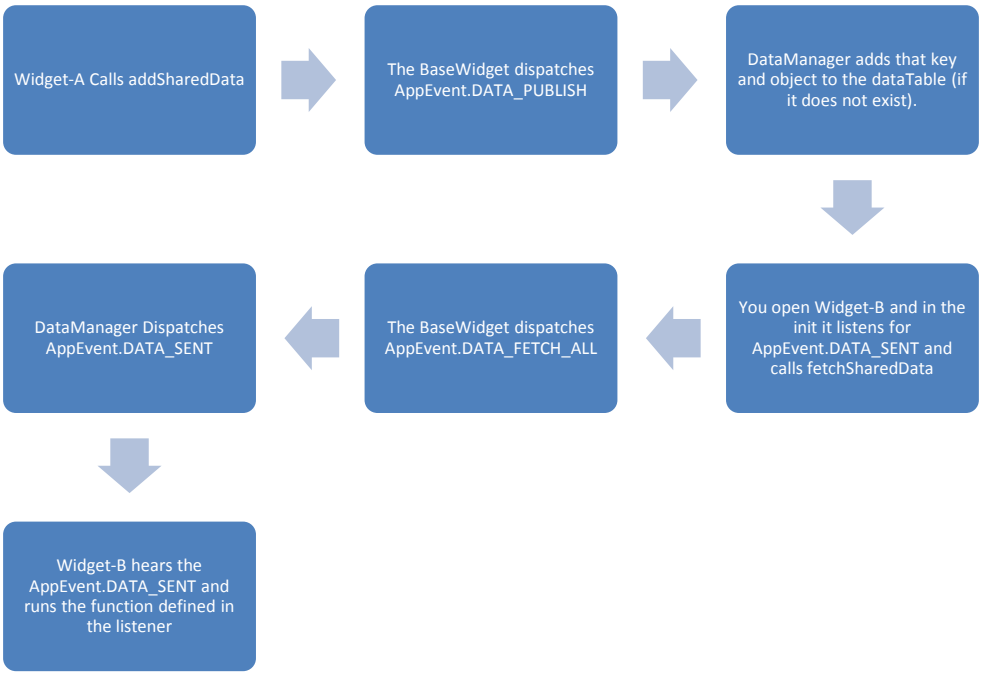

Figure 2

So to have a widget that listens for data from another widget whether that widget is opened yet or not you have to listen for two separate AppEvents (AppEvent.DATA_PUBLISH & AppEvent.DATA_SENT) and call fetchSharedData in its init function.

In order to have Widget-A open Widget-B so that it can share data with it you have to use the label of Widget-B and some new standard Flex Viewer functionality in 2.4. Here is a code block that demonstrates that:

```
var id:Number = ViewerContainer.getInstance().widgetManager.getWidgetId("Enhanced Search");

var bWidget:IBaseWidget = ViewerContainer.getInstance().widgetManager.getWidget(id, true) as IBaseWidget;

if (bWidget){
        var vSW:* = bWidget;
        vSW.queryFromURL("aaa",2,0);
}
```

In the code block above I use the Singleton ViewerContainer to get access to the widgetManager and a function called getWidgetId and pass that function the label of the widget I am searching for, that returns a number (id) of that widget. Using that id I can get the widget using the getWidget function in the widgetManager. As the returned IBaseWidget could be any widget I set a variable to an unknown type using :* and that allows me to call a specific function on a unknown widget type.

To put all this together this is the code you will need to add to each widget:

Widget-A (the widget that will shared some specific data with Widget-B)

Required Imports:

```
import mx.collections.ArrayCollection;
import com.esri.viewer.AppEvent;
import com.esri.viewer.BaseWidget;
import com.esri.viewer.IBaseWidget;
import com.esri.viewer.ViewerContainer;
```

Some Function to call the addSharedData and launch the Widget-B:

```
private function someFunction(evt:Event):void
{
	var msArr:ArrayCollection = new ArrayCollection();
	//Add an object or for example a string to the ArrayCollection
	msArr.addItem("Some text");
	addSharedData("MiniSearch_Search", msArr);
	//These next lines are what is needed if Widget-B is not already opened
	//and you want to open it and send your shared data
	//This code will only work in Flex Viewer 2.4
	var id:Number = ViewerContainer.getInstance().widgetManager.getWidgetId("Enhanced Search");
	var bWidget:IBaseWidget = ViewerContainer.getInstance().widgetManager.getWidget(id, true) as IBaseWidget;
	if (bWidget){
		var vSW:* = bWidget;
		vSW.queryFromURL("aaa",2,0);
	}
}
```

Widget-B (the widget that will receive the shared data from Widget-A)

Required Imports:

```
import com.esri.viewer.AppEvent;
import com.esri.viewer.utils.Hashtable;
```

Add these lines to the end of the Init Function in Widget-B:

```
AppEvent.addListener(AppEvent.DATA_PUBLISH, sharedDataUpdated);
AppEvent.addListener(AppEvent.DATA_SENT, sharedDataUpdated2);
fetchSharedData();
```


Prepared By: Robert J. Scheitlin

Most all Flex Viewer 2.4 widgets already have a sharedDataUpdated function. So you just need to add to it. This is the function that will get called if Widget-B is already open when Widget-A calls addSharedData()

```
private function sharedDataUpdated(event:AppEvent):void
{
   var data:Object = event.data;

   if (data.key == "Deactivate_DrawTool")
   {
      setMapAction(null, null, null, null);
      if (selectedDrawingIcon)
      {
         selectedDrawingIcon.filters = [];
         selectedDrawingIcon = null;
      }
   }
   if (data.key == "MiniSearch_Search")
   {
            if (data.collection[0]){
                  //The function you want to call in this widget
                  //data.collection[0] is the first item in the arrayCollection
                  //example:
                  queryFromURL(data.collection[0], 2, 0);
            }
   }
}
```

This is the function that will get called when the widget is initialized and fetchSharedData is called, which in turn dispatches an AppEvent.DATA_SENT event.

```
private function sharedDataUpdated2(event:AppEvent):void
{
      var dataTable:Hashtable = event.data as Hashtable;
      if (dataTable.containsKey("MiniSearch_Search"))
      {
            var recAC:ArrayCollection = dataTable.find("MiniSearch_Search") as ArrayCollection;
            if (recAC[0]){
                  queryFromURL(recAC[0], 2, 0);
            }
      }
}
```


Prepared By: Robert J. Scheitlin