Flextras Mobile Components Documentation

www.flextras.com

Contents

Welcome

Hi, and thanks for checking out the Flextras mobile Component set. This is documentation for the first update to the Flextras mobile component set. It contains these components:

1) The Flextras AutoCompleteComboBox component
2) Flex DropDownList
3) A RadioButton Item Renderer
4) A Square Button Skin

We are still working on expanding the set over time, so please let us know what you'd be interested in seeing.

This document is intended to get you started using our mobile flex components in your mobile applications.

For full details on the API, you can review the ASDocs included in the free developer edition download, or on the Flextras web site. The download also includes sample code, so be sure to check out those samples.

Thanks for taking a look. We are here to help, so don't hesitate to drop us an e-mail, send us an IM, or give us call with your questions.

Jeffry Houser (The Brains Behind Flextras)

Jeffry@dot-com-it.com | flextras@gmail.com (On GTalk) | 203-379-0773

A Note on Sample Code

The source code for all sample code in this document is included as part of the developer edition download. Where appropriate, we list the relevant files at the beginning of each instruction.

Document History

Date	Description
5/30/2011	• Initial Creation
8/27/2011	• Added more tutorials regarding pop up DropDownList skins and RadioButtonRenderer • Formatted Code

Introduction

This section will give you an introduction to using the Flextras AutoComplete or ComboBox in a mobile project.

Creating a Simple AutoComplete

Introduction

This example will show you a simple Mobile Application that uses the Flextras AutoCompleteComboBox.

Files used

- Sample1.mxml

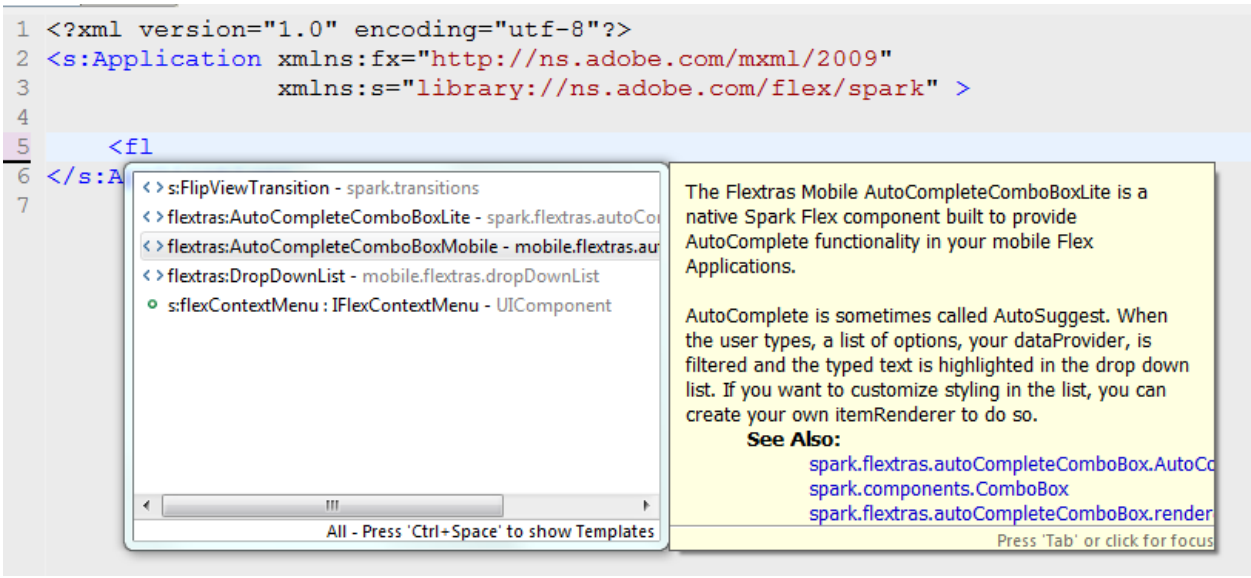
Tutorial

First create a new Flex Mobile Project in your application and add the SWC to your library path. More information how to use SWCs is located in this blog post. http://www.flextras.com/blog/index.cfm/2009/12/18/What-do-you-do-with-a-SWC .

When creating a Mobile Project in Flash Builder, Adobe provides three different templates. For the purposes of this demo, the blank application template will be used. The application will look something like this:

```
<?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"
               minWidth="955" minHeight="600">

</s:Application>
```

Start typing “<fl” in the main application and you should see a list of all Flextras components in the library, like below:



Select the AutoCompleteComboBoxLite and hit enter. The AutoCompleteComboBox code will automatically drop in:

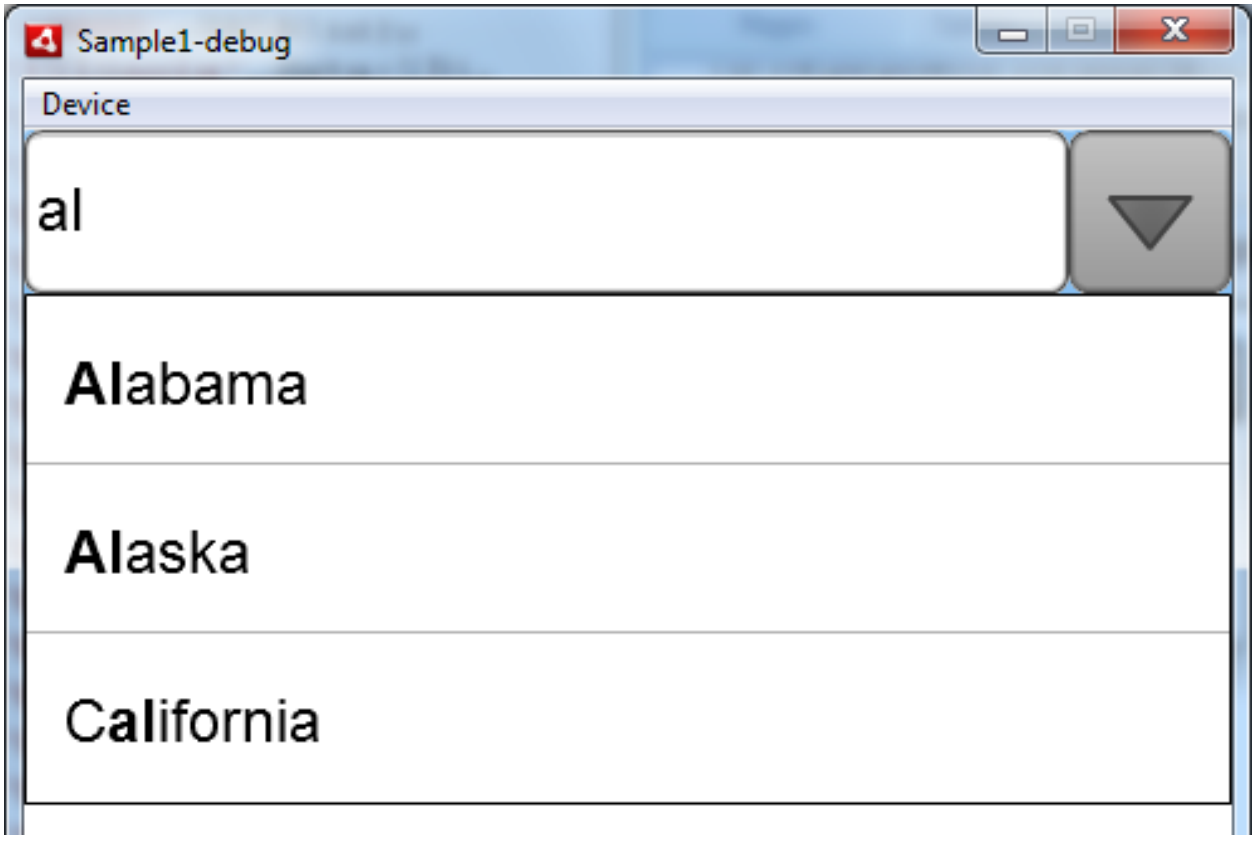
```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
                    xmlns:s="library://ns.adobe.com/flex/spark"
                    xmlns:mx="library://ns.adobe.com/flex/mx"
                   minWidth="955" minHeight="600"
                   xmlns:flextras="http://www.flextras.com/mxml">

      <flextras:AutoCompleteComboBoxMobile >
</s:Application>
```

You'll see that the Flextras namespace was automatically added to the main application tag and the component now resides in the body. The purpose of an AutoComplete component is to be able to filter data as the user types. To make that happen, you'll have to specify a dataProvider. For the purpose of this sample, I'll give you a hard coded dataProvider, but in your own applications, you can source it from any XML or backend, just like you were using other Flex List based components:

```
<?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"
                   minWidth="955" minHeight="600"
                   xmlns:flextras="http://www.flextras.com/mxml">

      <fx:Script>
            <![CDATA[
            import mx.collections.ArrayCollection;

            [Bindable]
            public var dataProvider : ArrayCollection =
```

```
                new ArrayCollection([
                      {label:'Alabama',data:1},
                      {label:'Alaska',data:2},
                      {label:'Arizona',data:3},
                      {label:'Arkansas',data:4},
                      {label:'California',data:5},
                ]);
          ]]>
      </fx:Script>

      <flextras:AutoCompleteComboBoxMobile
                id="accb"
                dataProvider="{this.dataProvider}" />
</s:Application>
```

For the purposes of this document, I kept the dataProvider short, but the sample in the documentation lists all the US States. Execute your sample, and you should see, something like this



Start typing and watch the dataProvider filter. I'm running this sample in the Emulator included in Flash Builder 4.5; however this should work the same on devices.

Final Code

```
<?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"
                minWidth="955" minHeight="600"
                xmlns:flextras="http://www.flextras.com/mxml">

    <fx:Script>
        <![CDATA[
        import mx.collections.ArrayCollection;

        [Bindable]
        public var dataProvider : ArrayCollection =
            new ArrayCollection([
                {label:'United States of America',data:0},
                {label:'Alabama',data:1},
                {label:'Alaska',data:2},
                {label:'Arizona',data:3},
                {label:'Arkansas',data:4},
                {label:'California',data:5},
                {label:'Colorado',data:6},
                {label:'Connecticut',data:7},
                {label:'Delaware',data:8},
                {label:'Florida',data:9},
                {label:'Georgia',data:10},
                {label:'Hawaii',data:11},
                {label:'Idaho',data:12},
                {label:'Illinois',data:13},
                {label:'Indiana',data:14},
                {label:'Iowa',data:15},
                {label:'Kansas',data:16},
                {label:'Kentucky',data:17},
                {label:'Louisiana',data:18},
                {label:'Maine',data:19},
                {label:'Maryland',data:20},
                {label:'Massachusetts',data:21},
                {label:'Michigan',data:22},
                {label:'Minnesota',data:23},
                {label:'Mississippi',data:24},
                {label:'Missouri',data:25},
                {label:'Montana',data:26},
                {label:'Nebraska',data:27},
                {label:'Nevada',data:28},
                {label:'New Hampshire',data:29},
                {label:'New Jersey',data:30},
                {label:'New Mexico',data:31},
                {label:'New York',data:32},
                {label:'North Carolina',data:33},
                {label:'North Dakota',data:3},
                {label:'Ohio',data:35},
                {label:'Oklahoma',data:36},
                {label:'Oregon',data:37},
                {label:'Pennsylvania',data:38},
                {label:'Rhode Island',data:39},
```

```
                        {label:'South Carolina',data:40},
                        {label:'South Dakota',data:41},
                        {label:'Tennessee',data:42},
                        {label:'Texas',data:43},
                        {label:'Utah',data:44},
                        {label:'Vermont',data:45},
                        {label:'Virginia',data:46},
                        {label:'Washington',data:47},
                        {label:'West Virginia',data:48},
                        {label:'Wisconsin',data:49},
                        {label:'Wyoming',data:50},
                    ]);
            ]]>
        </fx:Script>

        <flextras:AutoCompleteComboBoxMobile
            id="accb"
            dataProvider="{this.dataProvider}"
        />
</s:Application>
```

Creating a Simple DropDownList

Introduction

This example will show you a simple Mobile Application that uses the Flextras mobile optimized DropDownList.

Files used

- Sample2.mxml

Tutorial

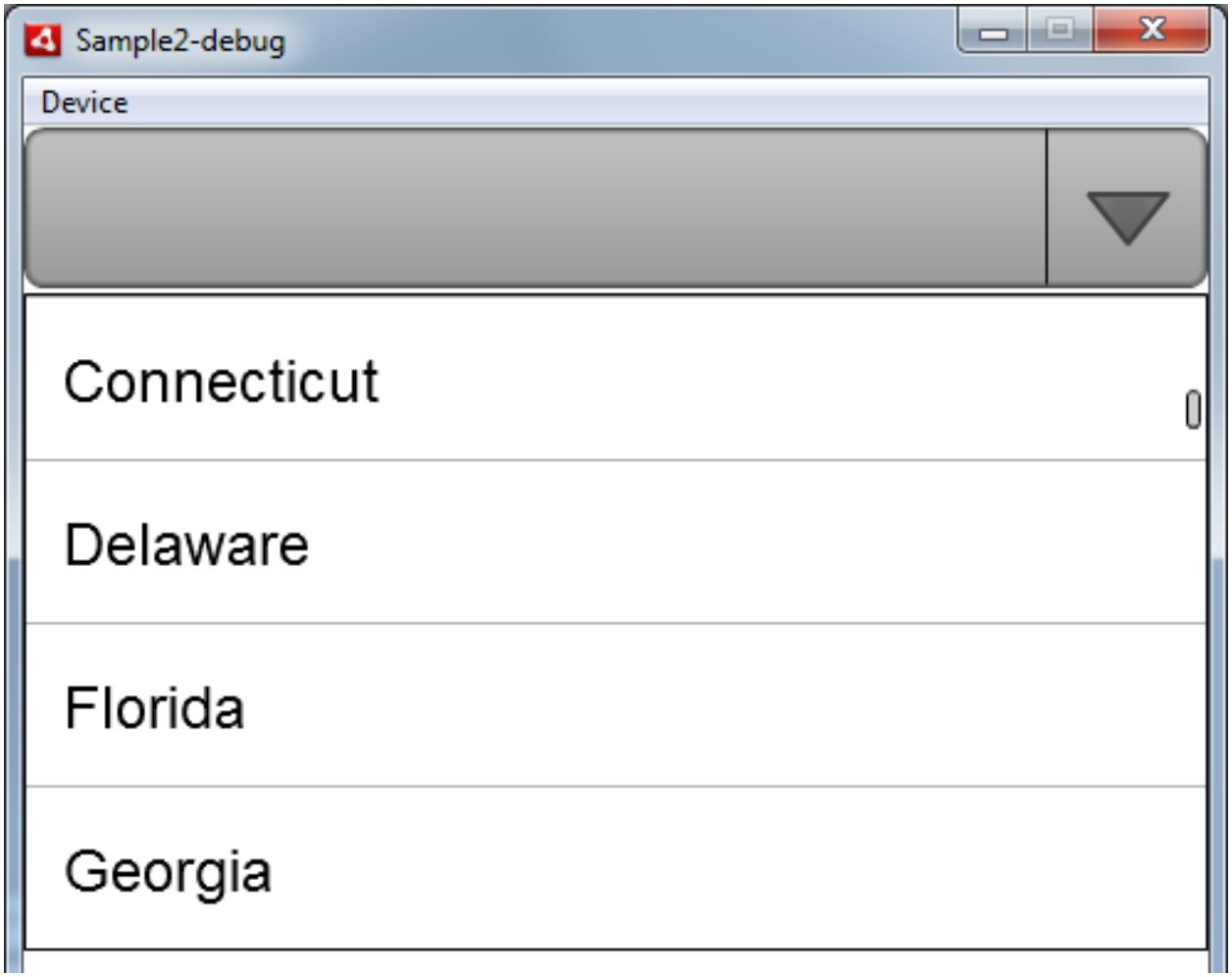
The instructions here are almost identical to the previous sample. You can create the same project, and use the same dataProvider. Just use the DropDownList instead of the AutoCompleteComboBoxMobile. You're first code segment may look like this:

```
<?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"
                   minWidth="955" minHeight="600"
                   xmlns:flextras="http://www.flextras.com/mxml">

      <flextras:DropDownList id="dropDownList" >
</s:Application>
```

Next add an ArrayCollection as a dataProvider:

```
<?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:flextras="http://www.flextras.com/mxml">

     <fx:Script>
          <![CDATA[
          import mx.collections.ArrayCollection;

          [Bindable]
          public var dataProvider : ArrayCollection =
               new ArrayCollection([
                    {label:'Alabama',data:1},
                    {label:'Alaska',data:2},
                    {label:'Arizona',data:3},
                    {label:'Arkansas',data:4},
                    {label:'California',data:5},
               ]);
          ]]>
     </fx:Script>

     <flextras:DropDownList id="dropDownList"
                            dataProvider="{this.dataProvider}"
                            width="100%" />
</s:Application>
```

The code should run and you'll something like this:



Final Code

```
<?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:flextras="http://www.flextras.com/mxml">

     <fx:Script>
          <![CDATA[
               import mx.collections.ArrayCollection;
               [Bindable]
               public var dataProvider : ArrayCollection =
                         new ArrayCollection([
                    {label:'United States of America',data:0},
                    {label:'Alabama',data:1},
                    {label:'Alaska',data:2},
                    {label:'Arizona',data:3},
                    {label:'Arkansas',data:4},
```

```
{label:'California',data:5},
{label:'Colorado',data:6},
{label:'Connecticut',data:7},
{label:'Delaware',data:8},
{label:'Florida',data:9},
{label:'Georgia',data:10},
{label:'Hawaii',data:11},
{label:'Idaho',data:12},
{label:'Illinois',data:13},
{label:'Indiana',data:14},
{label:'Iowa',data:15},
{label:'Kansas',data:16},
{label:'Kentucky',data:17},
{label:'Louisiana',data:18},
{label:'Maine',data:19},
{label:'Maryland',data:20},
{label:'Massachusetts',data:21},
{label:'Michigan',data:22},
{label:'Minnesota',data:23},
{label:'Mississippi',data:24},
{label:'Missouri',data:25},
{label:'Montana',data:26},
{label:'Nebraska',data:27},
{label:'Nevada',data:28},
{label:'New Hampshire',data:29},
{label:'New Jersey',data:30},
{label:'New Mexico',data:31},
{label:'New York',data:32},
{label:'North Carolina',data:33},
{label:'North Dakota',data:3},
{label:'Ohio',data:35},
{label:'Oklahoma',data:36},
{label:'Oregon',data:37},
{label:'Pennsylvania',data:38},
{label:'Rhode Island',data:39},
{label:'South Carolina',data:40},
{label:'South Dakota',data:41},
{label:'Tennessee',data:42},
{label:'Texas',data:43},
{label:'Utah',data:44},
{label:'Vermont',data:45},
{label:'Virginia',data:46},
{label:'Washington',data:47},
{label:'West Virginia',data:48},
{label:'Wisconsin',data:49},
{label:'Wyoming',data:50},
```

```
                ]);


            ]]>
        </fx:Script>

        <flextras:DropDownList id="dropDownList"
                               dataProvider="{this.dataProvider}"
                               width="100%" />
</s:Application>
```

Using Different Mobile Skins

With our Mobile AutoComplete component we provide multiple skins, which you can use in your applications. The primary difference between each skin is the design of the down arrow. The default skins for the DropDown classes make heavy use of MXML Graphics which are considered too performance intensive for mobile devices. Instead our new skins focus on using FXG assets or simple drawings with the graphic API.

This section will tell you about our mobile skins, and how to use them.

Using the Default AutoComplete Skin

The primary difference between all the skins is how the down arrow is implemented. In the default skin, the down arrow extends the Mobile ButtonSkin class and uses an FXG asset for the arrow. The background of the button is drawn using the Flash graphics drawing API, just like the Flex Button.

You don't have to do anything special to use this skin; as it is the default. The class for the default skin is:

- com.flextras.mobile.autoCompleteComboBox. AutoCompleteComboBoxMobileSkin_Default

Using the Default DropDownList skin

The DropDownList skin is very similar to the AutoComplete skin; and even uses the same default down arrow button skin. As with the default AutoComplete skin, you don't have to do anything special to use it; the skin is specified by default.

The class for the default DropDownList skin is:

- com.flextras.mobile.dropDownList.DropDownListMobileSkin_Default

Using the Bonus AutoComplete Skins

Introduction

We have provided additional mobile skins for the Flextras AutoCompleteComboBoxMobile component. In these skins, the down arrow's graphical assets are implemented entirely as FXG assets and the down arrow button extends MobileSkin. This section will show you how to use the extra skin.

Files used

- Sample3.mxml

Tutorial

There are four separate skins:

- com.flextras.mobile.autoCompleteComboBox.AutoCompleteComboBoxMobileSkin_1
- com.flextras.mobile.autoCompleteComboBox.AutoCompleteComboBoxMobileSkin_2
- com.flextras.mobile.autoCompleteComboBox.AutoCompleteComboBoxMobileSkin_3
- com.flextras.mobile.autoCompleteComboBox.AutoCompleteComboBoxMobileSkin_4

You can use these alternate skins in your application by specifying he skinClass style on the component. Start by creating a new application:

```
<?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">

</s:Application>
```

Next, start typing flextras and bring up the Flex AutoCompleteComboBox. I created four instance of the AutoComplete:

```
<?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:flextras="http://www.flextras.com/mxml">

     <flextras:AutoCompleteComboBoxMobile id="accb1" y="10" />
     <flextras:AutoCompleteComboBoxMobile id="accb2" y="100" />
     <flextras:AutoCompleteComboBoxMobile id="accb3" y="200" />
     <flextras:AutoCompleteComboBoxMobile id="accb4" y="300" />
</s:Application>
```

I added some manual spacing with the y attribute. Finally, specify the skinClass style. This is easy to do in MXML. You can ignore the long word wrap on the component path:

```
<?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:flextras="http://www.flextras.com/mxml">

     <flextras:AutoCompleteComboBoxMobile id="accb1" y="10"
     skinClass="com.flextras.mobile.autoCompleteComboBox.skins.
          AutoCompleteComboBoxMobileSkin_1"  />
<flextras:AutoCompleteComboBoxMobile id="accb2" y="100"
     skinClass=" com.flextras.mobile.autoCompleteComboBox.skins.
          AutoCompleteComboBoxMobileSkin_2"  />
<flextras:AutoCompleteComboBoxMobile id="accb3" y="200"
     skinClass=" com.flextras.mobile.autoCompleteComboBox.skins.
          AutoCompleteComboBoxMobileSkin_3"  />
<flextras:AutoCompleteComboBoxMobile id="accb4" y="300"
     skinClass=" com.flextras.mobile.autoCompleteComboBox.skins.
          AutoCompleteComboBoxMobileSkin_4"  />
</s:Application>
```

In normal cases you'd want to specify a dataProvider of some sort to the AutoComplete component, but for the purposes of this sample, I left that out.

This is a screenshot of Skin 1:

This is screenshot of Skin 2, my personal favorite:

This is a screenshot of Skin 3:

This is screenshot of Skin 4:

Using the DropDownList PopUp Skins

Introduction

We have provided additional mobile skins for the Flextras DropDownList mobile component. These skins act more like a native DropDownList than the traditional Flex DropDownList. Instead of displaying the drop down below the component, it is shown in a pop up.

Files used

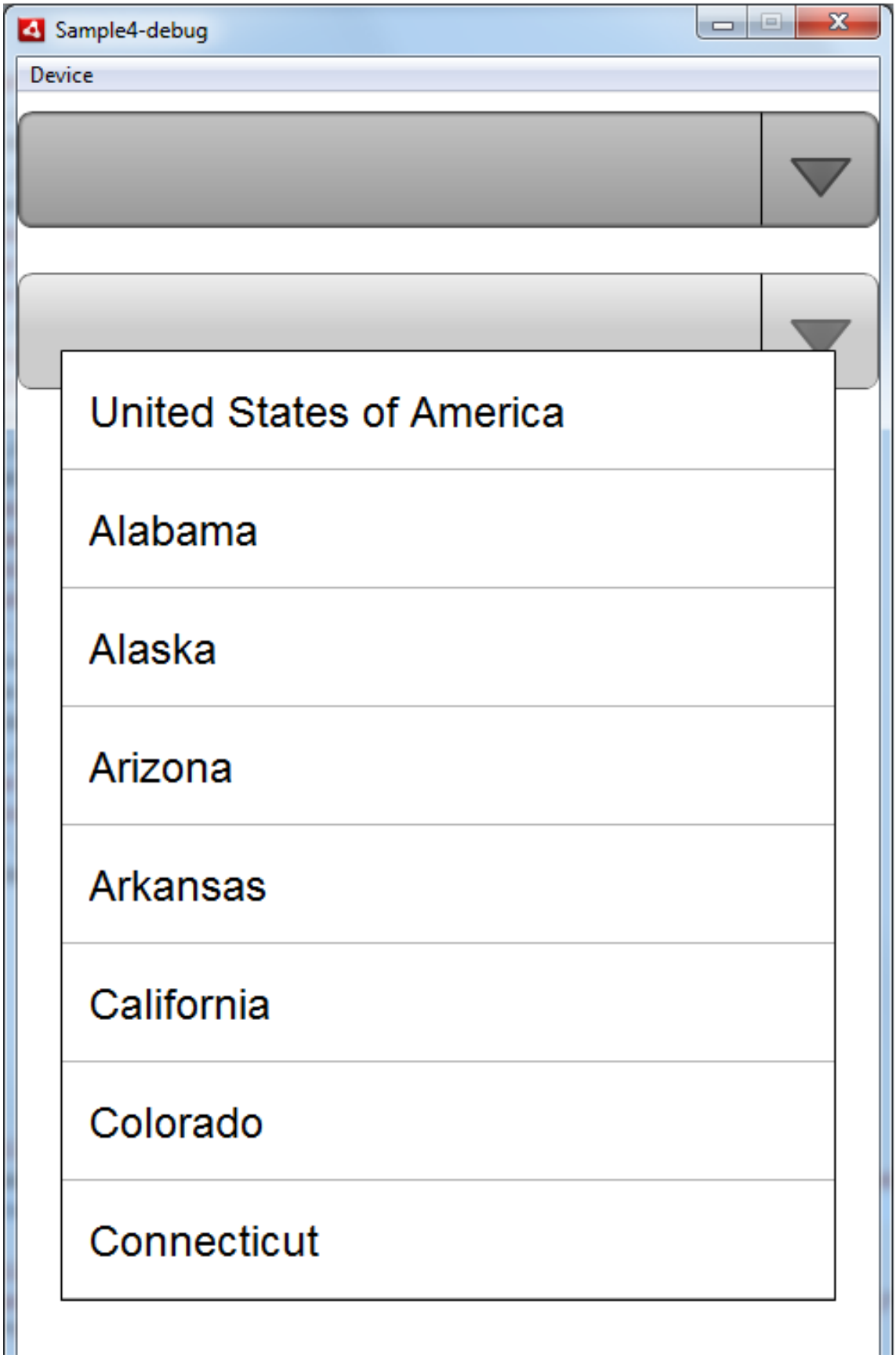
- Sample4.mxml

Tutorial

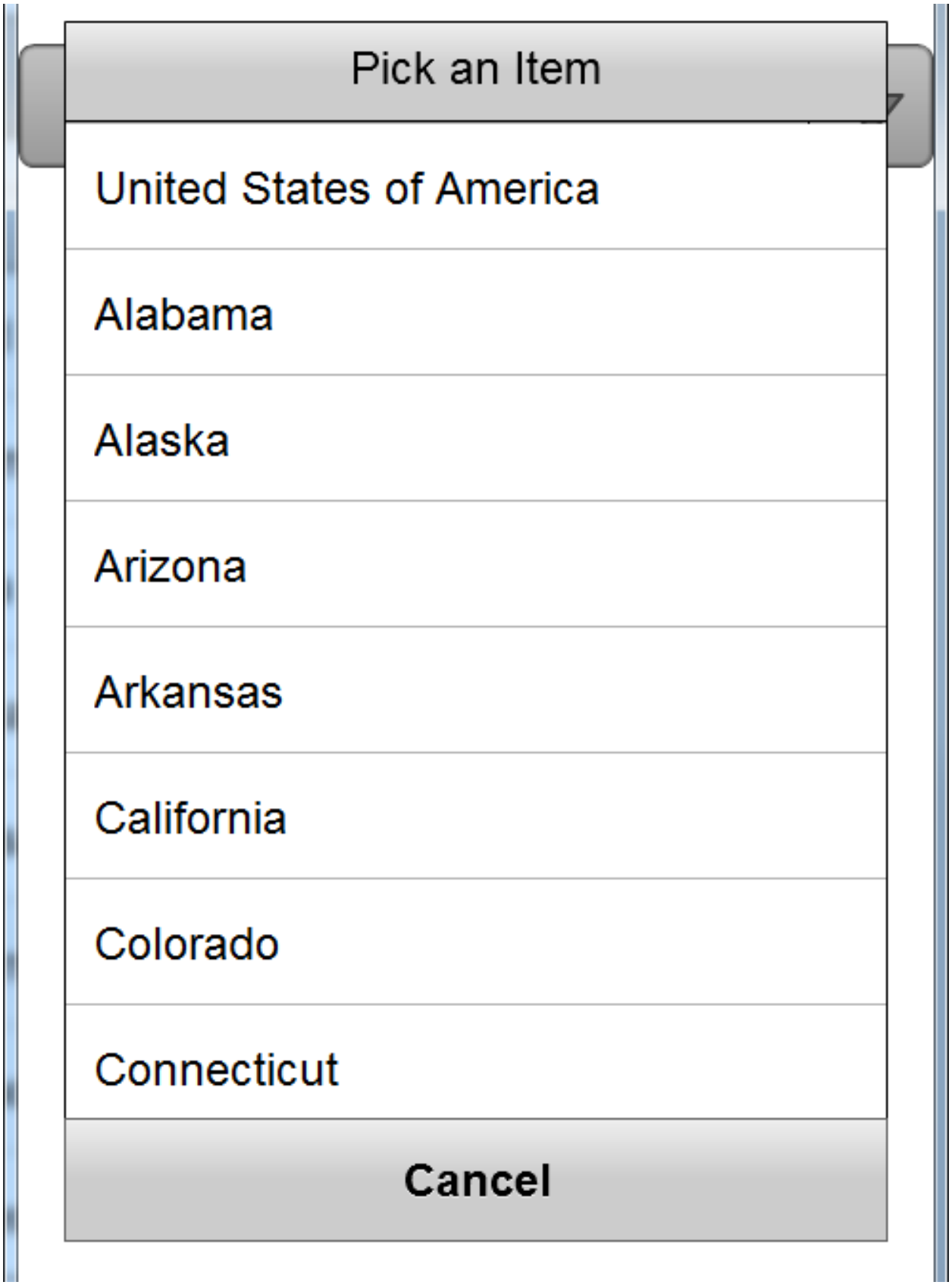
There are separate skins:

- com.flextras.mobile.dropDownList.skins DropDownListMobileSkin_PopUp
- com.flextras.mobile.dropDownList.skins DropDownListMobileSkin_PopUp_2

The first skin shows the drop down in a popup window over all controls.


Sample4-debug
Device
United States of America
Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut

The second skin also shows the drop down in a popup, but adds a header and cancel button.


Pick an Item
United States of America
Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut
Cancel

You can use these alternate skins in your application by specifying the skinClass style on the component. Start by creating a new application:

```
<?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">

</s:Application>
```

Next, start typing flextras and bring up the Flex DropDownList. I created two instance of the DropDownList:

```
<?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:flextras="http://www.flextras.com/mxml">

          <flextras:DropDownList id="ddl1" y="10" />
          <flextras:DropDownList id="ddl2" y="100" />
</s:Application>
```

The next step is to specify the custom skins on the two components:

```
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:flextras="http://www.flextras.com/mxml">

 <flextras:DropDownList id="ddl1" y="10"
     skinClass="com.flextras.mobile.dropDownList.skins.
               DropDownListMobileSkin_PopUp"/>
 <flextras:DropDownList id="ddl2" y="100"
     skinClass="com.flextras.mobile.dropDownList.skins.
               DropDownListMobileSkin_PopUp_2" />
</s:Application>
```

Excuse the word wrap in the skin class name. Both skins support a style named requestedRowCount. The default is four, however you can change that based on the amount of space you want on the device. For the purpose of these samples, you can specify 8—the same number shown in the above screenshots.

```
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:flextras="http://www.flextras.com/mxml">
```

```
 <flextras:DropDownList id="ddl1" y="10" requestedRowCount="8"
          skinClass="com.flextras.mobile.dropDownList.skins.
               DropDownListMobileSkin_PopUp" />
 <flextras:DropDownList id="ddl2" y="100" requestedRowCount="8"
          skinClass="com.flextras.mobile.dropDownList.skins.
               DropDownListMobileSkin_PopUp_2" />
</s:Application>
```

The second pop up skin contains both a header and a button. You can control the text on those properties using two properties on the DropDownList:

- **label**: The label property is used to specify the text that appears in the header of the pop up. It has no default value.
- **closeButtonLabel**: The closeButtonLabel property is used to specify the text that appears in the close button. The default value is 'Cancel'.

In the final code for the sample, you can keep the closeButtonLabel as the default value, but specify "Pick an Item" for the header label:

```
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:flextras="http://www.flextras.com/mxml">

 <flextras:DropDownList id="ddl1" y="10" requestedRowCount="8"
     skinClass="com.flextras.mobile.dropDownList.skins.
               DropDownListMobileSkin_PopUp" />
 <flextras:DropDownList id="ddl2" y="100" requestedRowCount="8"
     skinClass="com.flextras.mobile.dropDownList.skins.
               DropDownListMobileSkin_PopUp_2"
     label="Pick an Item" />
</s:Application>
```

Final Code

```
<?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:flextras="http://www.flextras.com/mxml">

     <fx:Script>
          <![CDATA[
               import mx.collections.ArrayCollection;
               [Bindable]
               public var dataProvider : ArrayCollection =
```

```
new ArrayCollection([
      {label:'United States of America',data:0},
      {label:'Alabama',data:1},
      {label:'Alaska',data:2},
      {label:'Arizona',data:3},
      {label:'Arkansas',data:4},
      {label:'California',data:5},
      {label:'Colorado',data:6},
      {label:'Connecticut',data:7},
      {label:'Delaware',data:8},
      {label:'Florida',data:9},
      {label:'Georgia',data:10},
      {label:'Hawaii',data:11},
      {label:'Idaho',data:12},
      {label:'Illinois',data:13},
      {label:'Indiana',data:14},
      {label:'Iowa',data:15},
      {label:'Kansas',data:16},
      {label:'Kentucky',data:17},
      {label:'Louisiana',data:18},
      {label:'Maine',data:19},
      {label:'Maryland',data:20},
      {label:'Massachusetts',data:21},
      {label:'Michigan',data:22},
      {label:'Minnesota',data:23},
      {label:'Mississippi',data:24},
      {label:'Missouri',data:25},
      {label:'Montana',data:26},
      {label:'Nebraska',data:27},
      {label:'Nevada',data:28},
      {label:'New Hampshire',data:29},
      {label:'New Jersey',data:30},
      {label:'New Mexico',data:31},
      {label:'New York',data:32},
      {label:'North Carolina',data:33},
      {label:'North Dakota',data:3},
      {label:'Ohio',data:35},
      {label:'Oklahoma',data:36},
      {label:'Oregon',data:37},
      {label:'Pennsylvania',data:38},
      {label:'Rhode Island',data:39},
      {label:'South Carolina',data:40},
      {label:'South Dakota',data:41},
      {label:'Tennessee',data:42},
      {label:'Texas',data:43},
      {label:'Utah',data:44},
      {label:'Vermont',data:45},
```

```
                    {label:'Virginia',data:46},
                    {label:'Washington',data:47},
                    {label:'West Virginia',data:48},
                    {label:'Wisconsin',data:49},
                    {label:'Wyoming',data:50},

                ]);


        ]]>
    </fx:Script>

    <flextras:DropDownList id="ddl1" y="10" width="100%"
        dataProvider="{this.dataProvider}"
        skinClass="com.flextras.mobile.dropDownList.skins.
                    DropDownListMobileSkin_PopUp"
        requestedRowCount="8"  />

    <flextras:DropDownList id="ddl2" y="100" width="100%"
        dataProvider="{this.dataProvider}"
        skinClass="com.flextras.mobile.dropDownList.skins.
                    DropDownListMobileSkin_PopUp_2"
        requestedRowCount="8" label="Pick an Item"/>
</s:Application>
```

Using the RadioButton ItemRenderer

Introduction

In addition to providing some custom skins, we also provided a mobile optimized itemRenderer that includes a RadioButton inside it. This can make your pop up drop down lists act similar to native drop down lists on mobile devices. This sample will show you how to use that renderer.

Files used

- Sample5.mxml

Tutorial

Start by creating a new application:

```
<?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">

</s:Application>
```

Next, start typing flextras and bring up the Flex DropDownList. I created two instance of the DropDownList:

```
<?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:flextras="http://www.flextras.com/mxml">

          <flextras:DropDownList />
</s:Application>
```

Start by setting some default properties for the DropDownList, including the ID, the width, the skinClass, dataProvider, and the label.

```
<?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:flextras="http://www.flextras.com/mxml">

 <flextras:DropDownList id="ddl" y="10" width="100%"
     dataProvider="{this.dataProvider}"
     skinClass="com.flextras.mobile.dropDownList.skins.
               DropDownListMobileSkin_PopUp_2"
     label="Pick an Item"
     />
</s:Application>
```

The full code will contain the dataProvider, but for the sake of brevity I'll leave it out of this code.

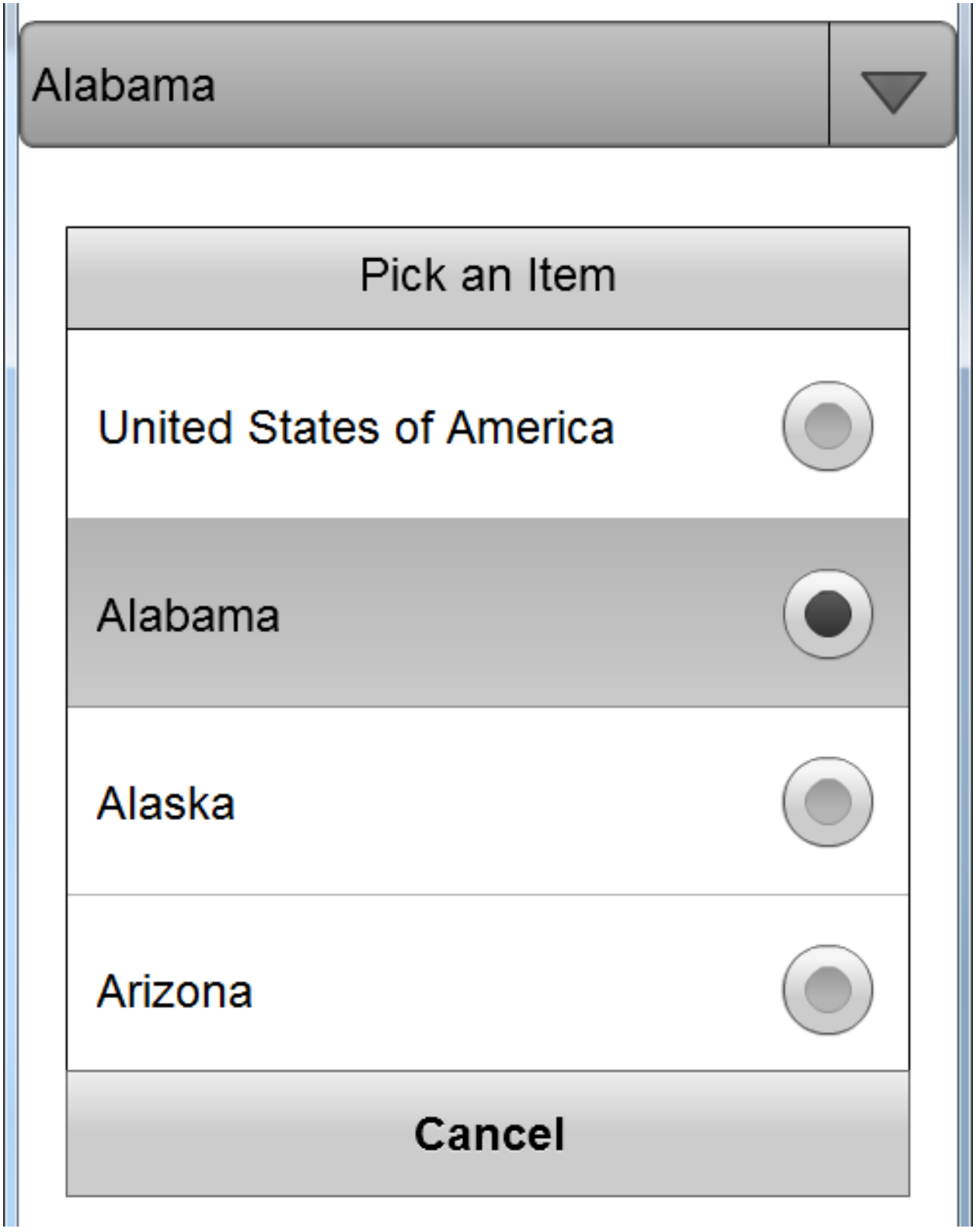
In the previous samples you specified a requestedRowCount of 8. It is important to remember that the radio button renderer will have a larger width than the renderer with just a label; so in this sample you can keep the requestedRowCount at the default value of 4.

Finally use the itemRenderer property to point to the Flextras mobile optimized radio button renderer:

```
<?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:flextras="http://www.flextras.com/mxml">

<flextras:DropDownList id="ddl" y="10" width="100%"
     dataProvider="{this.dataProvider}"
     skinClass="com.flextras.mobile.dropDownList.skins.
          DropDownListMobileSkin_PopUp_2"
     label="Pick an Item"
     itemRenderer="com.flextras.mobile.shared.renderers.
```

```
                RadioButtonRenderer" />

</s:Application>
```

Once again, excuse the word wrap in the class names. The final results will look like this:



Final Code

```
<?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:flextras="http://www.flextras.com/mxml">

    <fx:Script>
        <![CDATA[
            import mx.collections.ArrayCollection;
            [Bindable]
            public var dataProvider : ArrayCollection =
                new ArrayCollection([
                {label:'United States of America',data:0},
                {label:'Alabama',data:1},
                {label:'Alaska',data:2},
                {label:'Arizona',data:3},
                {label:'Arkansas',data:4},
                {label:'California',data:5},
                {label:'Colorado',data:6},
                {label:'Connecticut',data:7},
                {label:'Delaware',data:8},
                {label:'Florida',data:9},
                {label:'Georgia',data:10},
                {label:'Hawaii',data:11},
                {label:'Idaho',data:12},
                {label:'Illinois',data:13},
                {label:'Indiana',data:14},
                {label:'Iowa',data:15},
                {label:'Kansas',data:16},
                {label:'Kentucky',data:17},
                {label:'Louisiana',data:18},
                {label:'Maine',data:19},
                {label:'Maryland',data:20},
                {label:'Massachusetts',data:21},
                {label:'Michigan',data:22},
                {label:'Minnesota',data:23},
                {label:'Mississippi',data:24},
                {label:'Missouri',data:25},
                {label:'Montana',data:26},
                {label:'Nebraska',data:27},
                {label:'Nevada',data:28},
                {label:'New Hampshire',data:29},
                {label:'New Jersey',data:30},
                {label:'New Mexico',data:31},
                {label:'New York',data:32},
                {label:'North Carolina',data:33},
                {label:'North Dakota',data:3},
                {label:'Ohio',data:35},
                {label:'Oklahoma',data:36},
                {label:'Oregon',data:37},
```

```
                    {label:'Pennsylvania',data:38},
                    {label:'Rhode Island',data:39},
                    {label:'South Carolina',data:40},
                    {label:'South Dakota',data:41},
                    {label:'Tennessee',data:42},
                    {label:'Texas',data:43},
                    {label:'Utah',data:44},
                    {label:'Vermont',data:45},
                    {label:'Virginia',data:46},
                    {label:'Washington',data:47},
                    {label:'West Virginia',data:48},
                    {label:'Wisconsin',data:49},
                    {label:'Wyoming',data:50},

                ]);


            ]]>
        </fx:Script>

        <flextras:DropDownList id="ddl" y="10" width="100%"
         dataProvider="{this.dataProvider}"
         label="Pick an Item"
         skinClass="com.flextras.mobile.dropDownList.skins.
                    DropDownListMobileSkin_PopUp_2"
         itemRenderer="com.flextras.mobile.shared.renderers.
                       RadioButtonRenderer" />
</s:Application>
```

Frequently Asked Questions

This section deals with some commonly asked questions.

How do I resize the down arrow?

If you're using one of our default skins, the easiest way is to specify a height on either the DropDownList or AutoCompleteComboxBox. The down arrow is sized as a square and will size its self to fill the height; also making the width bigger.

You can also create your own custom skin.

How can I set a default item in the AutoCompleteComboBox or DropDownList?

Use the selectedItem or selectedIndex properties, just as you would with the non mobile counterparts.

Is there a way to easily extend the height of the drop down in the DropDownList or AutoCompleteComboBox?

The best way to do this is to set the requestedRowCount style on either the DropDownList or the AutoCompleteComboBoxMobile. The requestedRowCount is used in the skin to determine the number of items that should be displayed in the drop down; and that number is use to calculate the size of the drop down.

An alternate approach is to create an alternate skin that sizes the drop down using an alternate approach.

Support

We are more than happy to help out with issues surrounding our components. Many of our samples and support documentation came about because people asked us for help.

Here are some ways you can contact us:

- Stop by a Flextras Friday Lunch live session. More info at www.meetup.com/flextras
- Send us an e-mail at info@dot-com-it.com
- Send us an IM on Jabber/GTalk at flextras@gmail.com
- Find us on Twitter; the account name is Flextras.
- Give us a phone call at 203-379-0773.

For best results, please provide as much details as you can, including a sample application and steps to reproduce your problem.

We can also provide consulting services to develop custom extensions of our components, if our component doesn't explicitly fit your use case, but comes very close.

Thank You

We wanted to thank you for checking out our components. We can't wait to see what type of applications you're going to build with this. We're here to help so be sure to contact us if you have questions, enhancement requests, or if you just want to chat.

Jeffry Houser (The Brains Behind Flextras)

Jeffry@dot-com-it.com | reboog711@gmail.com | 203-379-0773