



Contents

- [1 Introduction](#)
- [2 Prerequisites](#)
- [3 Installing](#)
- [4 How to use](#)
 - ◆ [4.1 Properties](#)
 - ◆ [4.2 Events on radio buttons](#)
 - ◇ [4.2.1 onChange](#)
 - ◆ [4.3 Creating and setting up a button](#)
 - ◆ [4.4 Formatting the label of the RadioButton](#)
- [5 Downloads](#)
- [6 Conclusion](#)

Introduction

The default radio button component of the Flash CS3/CS4 is very useful to web pages and desktop applications, but it's not true to mobile applications because of the small screen and big resolution, mainly in newer devices (5800, N97). The component shown here is a custom radio button, where the user can provide your own MovieClip as the button.

The radio buttons are combined in groups, the user can only select one radio button in a group at one time. Clicking on another causes the existing one to become unchecked.

Prerequisites

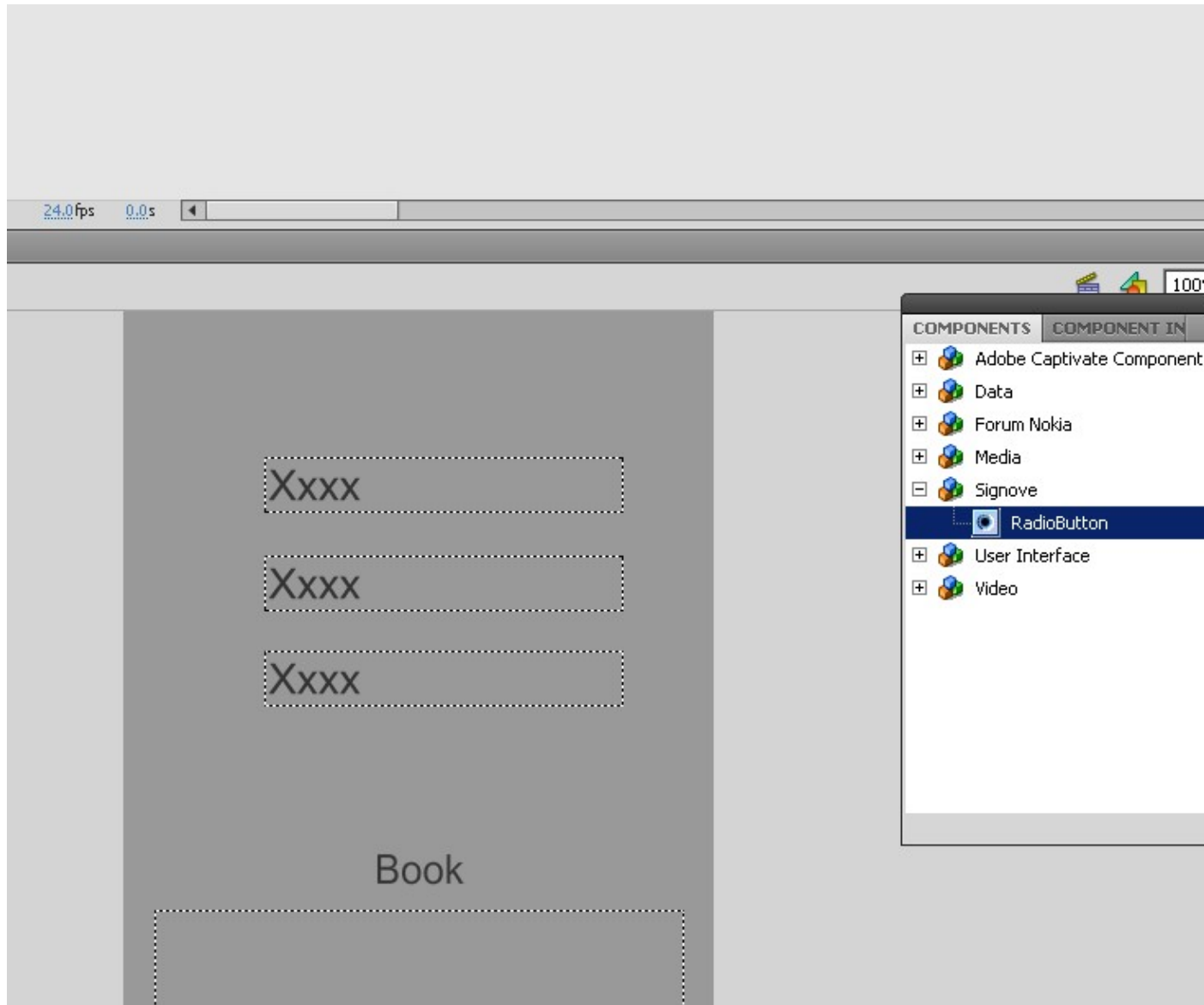
This component was developed in Flash Player 8 and ActionScript 2.0. It was tested on Nokia Xpress Music 5800.

Installing

It's very simple to install the component, just download the MXP file and open it in the Adobe Extension Manager.

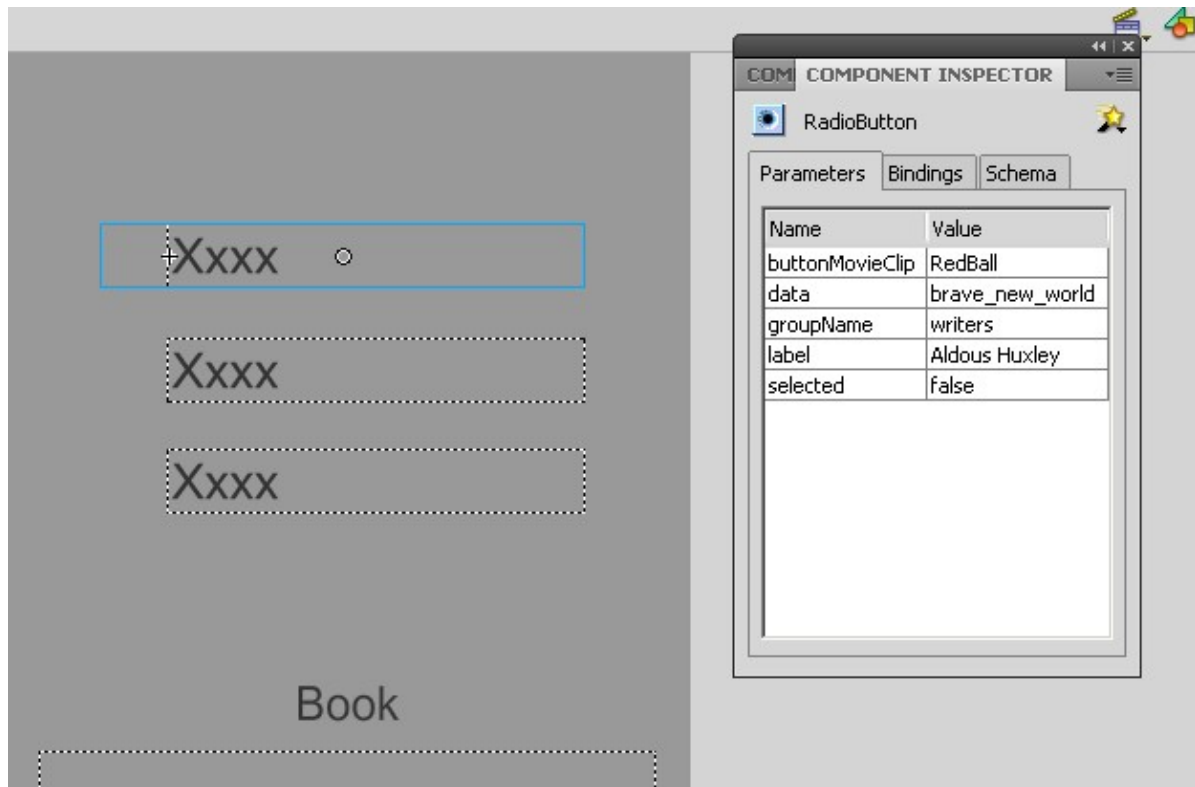
How to use

After the installation, the component appears in the Component panel (Window -> Components -> Signove -> RadioButton) like shown in the below figure. Now, it's just drag and drop the component to the Stage and configure the properties.



Properties

To view the properties of the component go to (Window -> Component Inspector) and select the component in the Stage. There are just five properties in the component, as shown in the below figure.



- **buttonMovieClip** : This movie clip is the button of the radio button. We'll see more details about it in "Creating and setting up a button" section.
- **data** : The value of the radio button. Must be unique in a group.
- **groupName** : The name of the group. Radio buttons in the same group must have the same groupName property.
- **label** : The text shown in the radio button.
- **selected** : If the radio button must appears selected or not when the application starts.

Events on radio buttons

onChange

There is just one piece of ActionScript code needed in this event. It's shown below.

```
//Import the RadioButtonGroup class.
import com.signove.RadioButton.RadioButtonGroupClass

//Set a listener to the onChange event. When the user click on a radio button, the onChange method
var onChangeListener:Object = new Object();
onChangeListener.onChange = function(evt):Void{
    option_value.text = evt.target.selectedData;
}
```

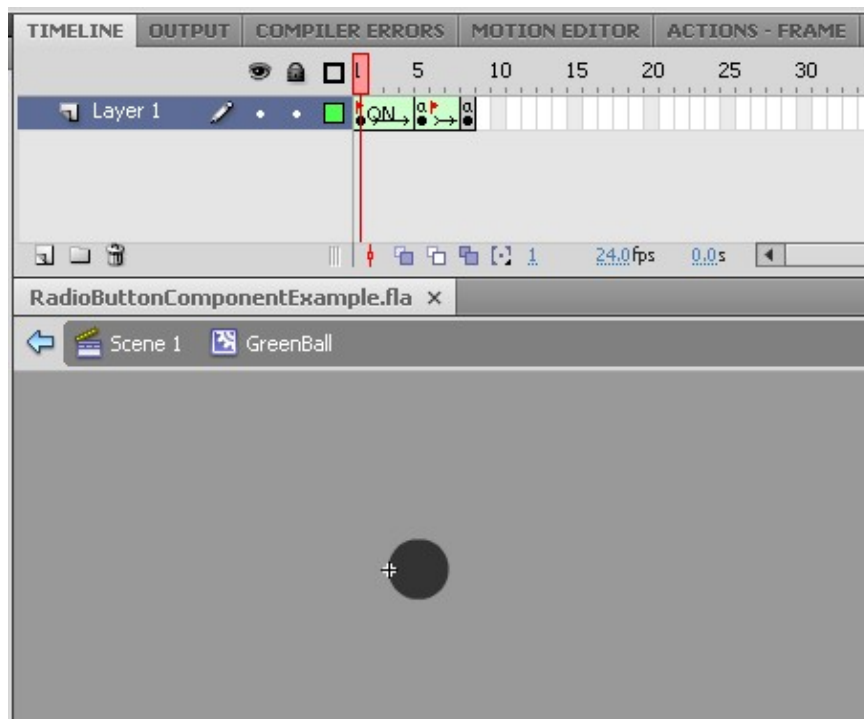
RadioButton_Component_for_Flash_Lite

```
//Retrieve the group and set the listener.  
var group = RadioButtonGroupClass.getGroup('writers');  
group.addEventListener('onChange', onChangeListener);
```

The code shown above is about events on radio buttons. In the first version of the component, there is just one event to be caught (onChange).

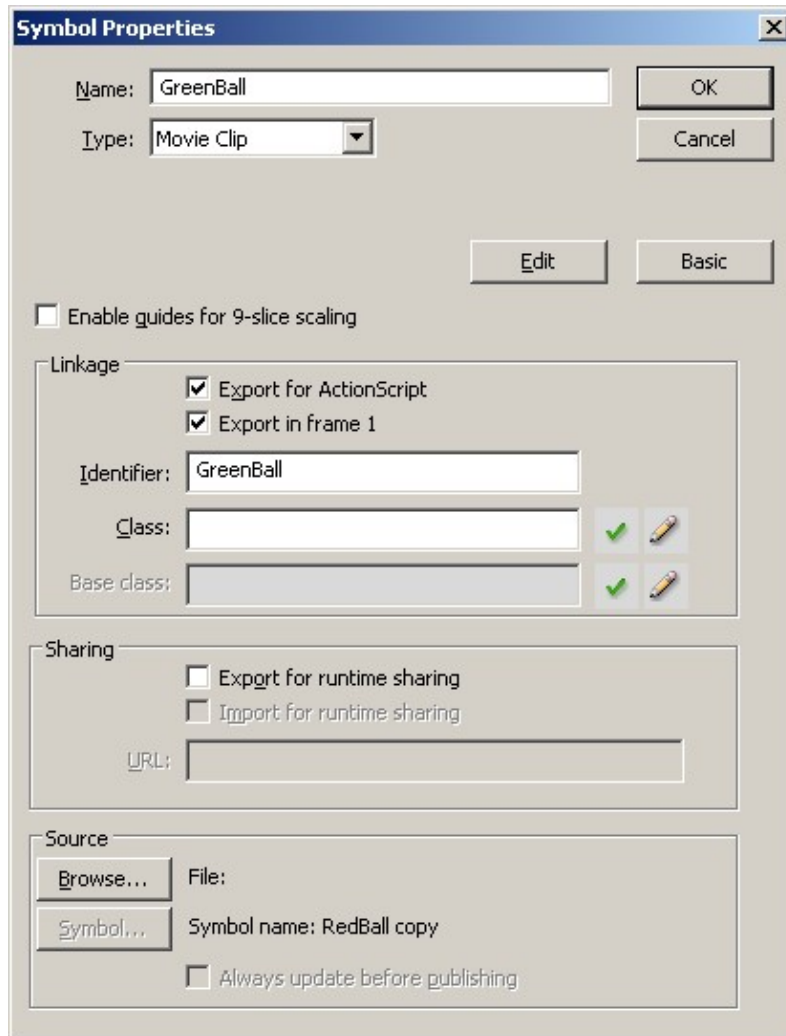
Creating and setting up a button

The component comes with a default button, but the developer can create your own button. The button is just a MovieClip with few restrictions. It has to have two frames named 'ON' and 'OFF', one to start the animation when the user check a radio button and another to start the animation when a radio button become unchecked. An example is shown in the figure below.



After the creation of the MovieClip (don't forget the 'ON' and 'OFF' frames) you have to set a 'Identifier' to the MovieClip. This identifier is used in the buttonMovieClip property of the radio button.

To create an identifier for the MovieClip, go to (Window -> Library), select your new MovieClip, right-click on it and select 'Properties...' option. You will see a screen like the one shown below. Check both options 'Export for ActionScript' and 'Export in frame 1' and type an 'Identifier'



Formatting the label of the RadioButton

The developer can format the label of the radio button with an instance of the `TextFormat` class. The code below shows how to do that.

```
var my_fmt:TextFormat = new TextFormat();
my_fmt.bold = true;
my_fmt.size = 24;
my_fmt.color = 0xFF0000;

//The instance name of the RadioButton component
radio_1.setTextFormat(my_fmt);
```

Downloads

- The source code of the component, download [here](#)
- The MXP file, ready to install, download [here](#)
- A simple example using this component, download [here](#)

Conclusion

It's really very simple to create a component. I've just started to learn two days ago and i have the first version of my first component now. I didn't use Assets in this version because i don't guess it's an easy work and it's not necessary at the moment.

If you want to see more, go to [here](#)