



## Contents

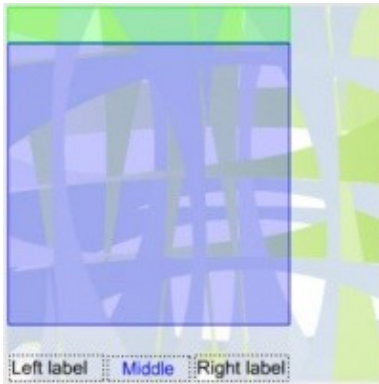
- [1 Purpose of this article](#)
- [2 Square stage](#)
- [3 UI components](#)
- [4 Freeze the stage](#)
- [5 Detecting the orientation switch](#)
  - ◆ [5.1 NOTE:](#)
- [6 Without detection](#)
- [7 Other resolutions](#)
- [8 How to use the template](#)
- [9 The template .fla](#)

## Purpose of this article

The problem of not knowing if the phone is in portrait or landscape orientation is quite annoying, especially when the landscape mode scales the fonts down. This article explains how I created a very simple template for some of my own projects. This is just one way of making sure your application looks OK in both screen orientations (portrait and landscape) in 240-x-320 pixel (QVGA) devices. I'm working on an even more flexible solution which should work with any screen orientation.

See this article for more information about dynamic layouts:  
[http://www.adobe.com/devnet/devices/articles/dynamic\\_layout.html](http://www.adobe.com/devnet/devices/articles/dynamic_layout.html)

## Square stage



I started creating the application with a 320-x-320 pixel stage. To make sure it looks the same in both orientations, I made the background abstract. This type of image (see the picture on the left) is suitable for both portrait and landscape since it will not rotate. You can, of course, have a solid colour, which is easier, or an image that you stretch and/or rotate to suit the current layout, but it requires a bit more work.

The image on the left is an actual screenshot from [Adobe Flash professional](#). I have attached the example FLA file at the end of this article.

However, the key to this sort of dynamic layout design is not the background but creating replacable elements.

## UI components



Next, I created MovieClips for the title, softkey area, and the actual real estate of the application. The objects

## Dynamic\_Layout\_control\_for\_Flash\_Lite

are colored and have outlines for easier handling. If you do not like the colors, feel free to change them ;o). I keep the colors in the FLA, but change or remove them at runtime with ActionScript. The title should pop up briefly in landscape mode but be always visible in portrait mode. The real estate area is 240 x 240 so it will not be resized, giving me full control. The softkey area is designed so that the texts will not be resized either.

## Freeze the stage

Next I made sure the stage stays put and is in full-screen mode.

```
Stage.scaleMode = "noScale"; // Prevent movie from resizing if browser window resizes.
Stage.align = "LT"; // Fix the stage to the top left corner.
fscommand2("Fullscreen", true); //Set to full screen.
```

This means I can anchor my real estate to wherever I wish. In this case I wanted to fix the left top. See Adobe's help for **Stage.align** for more options.

## Detecting the orientation switch

Next, I start sniffing for the softkey change with `setInterval`:

```
setInterval(TurnMe, 100);
```

This might be a bit crude, but I have not really found a way to detect the orientation switch, at least not in most devices. The following piece of code helps me skip the function if there is no change.

```
if(softKeyLocation != PreviousOrientation){ // do only if different from previous
```

It may, of course, be better to set the interval a bit longer than 100 milliseconds, but I did not have any major problems when testing this.

For a more reliable version an extension to the Flash Lite framework is needed, so that the orientation switch can be detected as an event.

### NOTE:

The following method described in Adobe's help does not seem to work - at least not in all devices:

```
var stageListener:Object = new Object();
stageListener.onResize = function() {
    //your code goes here
```

## Without detection



If you don't want to use the above method for detecting the orientation switch, you can still benefit from this

Without detection

## Dynamic\_Layout\_control\_for\_Flash\_Lite

dynamic approach and detect the softkey location/screen size only in the beginning and position your layout elements accordingly. The way to find out the screen size is through the **stage** object

```
Stage.width; //The width of the display  
Stage.height; //The height of the display
```

## Other resolutions

For an overall *real* dynamic application, you must consider the other resolutions as well. Even though a vast majority of Nokia devices come in QVGA (240x320) resolution, there are more options to play with, provided that you wish to offer a specific layout model for a single device. For example, the inside display of the Nokia E90 Communicator has a resolution of 800x352. This means that you have more space to play with and if your application can benefit from it, you can display more data than in the QVGA resolution:



However, the cover display of the Nokia E90 Communicator is still a regular QVGA display, so this device is covered also by your regular QVGA layout version.

The new Nokia 5800 XpressMusic has a resolution of 360x640 and it switches automatically between landscape and portrait mode, depending on the way the user is holding it. The physical size of the display is not that much bigger, so you need to increase the size of your components anyway. However, you also need to consider that it is a touch-only device and it does not have a keyboard. Designing for touch means that you need to enable mouse support in the application since there are no keyboard events. This is a topic for another article and I will add a link to it here once the article is ready ;o)

## How to use the template

Now I just place the actual content inside the real estate element and the title inside the title element, and change the softkey labels to reflect the actual functionality and publish. This can be done with ActionScript or by modifying the UI components by clicking on them and editing them.

## The template .fla

[File:DynamicLayout.zip](#)

Click on the link above and download from the page that opens.