

Flash Lite: Visual Guide

Version 1.0; May 22, 2006

Flash



NOKIA

Copyright © 2006 Nokia Corporation. All rights reserved.

Nokia and Forum Nokia are registered trademarks of Nokia Corporation. Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. Other product and company names mentioned herein may be trademarks or trade names of their respective owners.

Disclaimer

The information in this document is provided “as is,” with no warranties whatsoever, including any warranty of merchantability, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. Furthermore, information provided in this document is preliminary, and may be changed substantially prior to final release. This document is provided for informational purposes only.

Nokia Corporation disclaims all liability, including liability for infringement of any proprietary rights, relating to implementation of information presented in this document. Nokia Corporation does not warrant or represent that such use will not infringe such rights.

Nokia Corporation retains the right to make changes to this specification at any time, without notice.

License

A license is hereby granted to download and print a copy of this specification for personal use only. No other license to any other intellectual property rights is granted herein.

Contents

1	Introduction.....	5
2	Flash Lite for what?.....	6
3	A brief technology review of Flash Lite	7
4	Nokia devices with Flash Lite Player 1.1.....	8
5	What you need	9
6	From Flash to Flash Lite.....	10
7	Usability issues with Flash Lite applications.....	11
7.1	Existing usability guidelines for mobile applications.....	11
7.2	Flash Lite-specific usability guidelines.....	14
8	Innovative Flash Lite user interface design.....	21
9	Performance.....	23
10	Conclusions.....	24
11	References	25
12	Evaluate this resource	27

Change history

May 22, 2006	Version 1.0	Initial document release

1 Introduction

Creating mobile interfaces and applications has never been as easy as it is with Flash Lite [3]. The WYSIWYG editor of Flash enables fast and fun creation of user interfaces and with Action Script 2.0 supporting object oriented programming it is possible to make a fully functional program.

In terms of user interfaces, almost everything is possible with Flash Lite. This is the biggest benefit and at the same time the biggest challenge of using Flash Lite. This document describes how to develop S60 Mobile Flash user interfaces that take the best out of the possibilities that Flash Lite offers while being usable at the same time. Also other factors that affect the total user experience of Flash Lite applications, such as performance, are briefly discussed.

We have interviewed Flash Lite developers and utilized their experiences when writing this document. They have also provided us with some good examples. Special thanks go to Adobe, Aniway, Blueskynorth, Flacell Justin Everett-Church, and Idean Creative.

This document is intended for all Flash developers who want to experiment with Flash Lite in S60 — from beginner to intermediate level. However, this document will definitely give good advice for more advanced developers as well. And if there is something that you would like to see in the next versions of this document, please give us feedback!

2 Flash Lite for what?

Most Flash Lite developers have a successful background as Web Flash developers — beautiful intros, desirable banners, WOW-factorized corporate sites, and breathtaking demos are pretty regular stuff for most of you. But it is a completely different thing to create a mobile Flash Lite application than a typical Web-based Flash application. You need to concentrate much more on the interaction side of development. For example, a typical S60 user has adapted a strong way of navigating from the S60 UI style. It is important to keep your Flash Lite application as consistent as possible, carefully plan the usage of small screen size, and combine it with a proper menu layout. And still, there is lots of room for creativity. Actually Flash Lite is currently one of best imaginable ways to realize your cool ideas to a huge number of potential users!

One might ask about the purpose of bringing yet another technology to the mobile device market — don't we already have enough of them? When talking about Flash Lite, it can be honestly stated that it fulfils needs that the existing technologies, like Java™ or C++, have not been able to satisfy.

The most obvious of these needs is fast prototyping. With Flash Lite it is rather straightforward to test a mobile concept by doing a set of sketches that have limited functionality. In S60 this is extremely useful because of the strong style given by S60 itself. You do not need to think so much about the graphical UI but more about interaction and the concept itself. In addition to prototyping, Flash Lite is widely used for casual games, such as puzzles. Games and prototypes are probably the most common use cases for Flash Lite, but it is also used for creating different kinds of guides, information leaflets, and brochures for mobile devices.

The biggest limitation of Flash Lite at the moment is probably its lack of handling secure information securely enough. Thus it is not recommended to use Flash Lite to develop a service that deals with Digital Rights Management (DRM) — yet. However, this does not mean that Flash Lite could not be used for serious purposes. With the help of different extensions of the Flash Lite technology, Flash Lite has been used for making operating systems for mobile devices, vehicle on-board computers, MP3-players, and digital cameras.

Most Flash Lite users are currently in Asia, especially in Japan. Japanese operators, like DoCoMo [5], have been integrating Flash to their services already for a while. Devices subscribing to DoCoMo's services use a Flash Lite technology called Flash Cast that pushes content to the device, similar to Nokia's Kanavat application. In Europe and the United States the Flash Lite market is still growing and much of the future growth depends on the co-operation between operators, Adobe, and mobile device manufacturers. As we can see, much is expected from Flash Lite!

For more information, see the document [Flash Lite for S60 - an emerging global ecosystem](#) [1].

3 A brief technology review of Flash Lite

Currently there are altogether three versions of Flash Lite: 1.0, 1.1, and 2.0. As Flash Lite 1.1 supports ActionScript introduced in Flash 4, Flash Lite 2.0 is compatible with the latest version of ActionScript, namely ActionScript 2.0. In addition to the standard Action Script classes there are also special Flash Lite classes that, for example, enable accessing device data and the network via the device.

There are also different Flash player versions available that are compatible with different Flash Lite versions. Most of the newest phones have Flash Lite Player 1.1 preinstalled on them. There is also a newer Player, version 2, which has almost equal function with Flash Player 7. This player can be purchased from the Adobe/Macromedia Web site [2].

If you are developing Flash Lite content for masses, you better stick to Flash Lite 1.1 since it is the technology that is easily available for most consumers. There has been quite a big technological leap between Flash Lite 1.1 and Flash Lite 2.0. However, strictly from the UI point of view almost the same things can be done with Flash Lite 1.1 as with the newer version. So even though this document refers to Flash Lite 2.0, most of the things are also valid for Flash Lite 1.1.

The good news about developing with Flash Lite for S60 is that S60 is the reference platform for the Flash Lite technology. Nokia and Adobe have made a licensing agreement on integrating Macromedia Flash technology into the S60 platform. Nokia has also agreed on integrating Flash into its other platforms, such as Series 40. More about this can be read from the press release published on Adobe's Web site [3].

As you noticed there is a lot of technical stuff you need to know to be able to create Flash Lite applications for S60. However, from the "ultimate-creative-designer" point of view it is a good idea to move your thinking to a particularly challenging topic: developing your interaction design skills.

4 Nokia devices with Flash Lite Player 1.1

There are currently 16 Nokia devices that have Flash Lite 1.1 support preinstalled in them. Naturally it is possible to purchase the Flash Lite Player 2.0 and upgrade your device for more advanced Flash Lite applications. Out of the 16 Flash Lite enabled devices 12 are S60 devices and the rest are Series 40 devices. The S60 devices can be found at: http://www.adobe.com/devnet/devices/nokia_s60.html. The list is updated regularly so it is worthwhile to check it every now and then.

5 What you need

Besides a lot of enthusiasm, visual talent, and interaction skills you will need a couple of commercially available programs. Flash Lite 1.1 applications can be developed with the Adobe Flash IDE. If you want to develop applications using Flash Lite 2.0, you have to download an update from Adobe's Web site [7]. This update is only applicable to Flash 8 Professional.

Before transferring your Flash Lite masterpiece to your mobile device, you can test it on an emulator included in Flash 8 Professional. For Flash Lite 2.0 there are initially no phones (except for the generic phone) available in the emulator, but you can install a set of phones by applying a device profile update [11].

To make the beginning as a Flash Lite developer easier for you, Adobe has created a Flash Lite 2 Content Development Kit (CDK) that includes documentation, samples, and code [12]. It is also recommended that you check out the Flash Lite 2 Developer Forum [8].

Especially if your Flash Lite application uses external files, such as .txt or .xml files, it is recommended that you create a .sis file out of everything that is included in your Flash Lite program. Creating a .sis file is also recommended if you are using just a single .swf file. Here are some benefits of the .sis compression:

1. You can set an icon to your file.
2. You can install multiple files at a time.
3. The user can easily uninstall all of the content.
4. You can assign a version number to your content.
5. You can select a non-default location for your content.
6. You can also install .txt files to be used with loadVariables.

Source: [10] and Aniway

To be able to create a .sis file of your Flash content you need a Software Development Kit (SDK) for Symbian that is downloadable from Forum Nokia. You also need to order a set of UIDs (Unique Identification Number) from Symbian. More information on this can be found at the Symbian Web site [9].

You can also find valuable information on Adobe's Web site [Getting started developing for Flash Lite](#) [13].

6 From Flash to Flash Lite

If you are familiar with Flash, it is easy to start developing a Flash Lite application. The only differences in the Flash development kit are that the document dimensions are fixed (to, for example, 176 x 208 pixels) and that some classes and functions in the ActionScript reference are different. Even though these sound like small issues, in practice the effects to development are huge.

This is the first trap you can fall in to: It is fun and easy — even too easy — to create a Flash Lite application if you are familiar with Flash, as long as you do it on your computer. The real challenges begin when you start testing your application on different devices with different displays, memory capacities, and processors. Thus, rule number one is: Think about optimization at all times and your application will more probably work fantastically on more devices. If you have a background from the good-ol'-days of the 1980's with C-64, Sinclair, and Atari or other computers with less than 32 kB of memory, you will feel like coming back to reconstructed home: It really makes a difference to develop tight and optimal code while keeping your application cool and respectable. For Flash Lite Player 1.1 the preferred file size is about 100 kB, but it can be also higher. For Flash Lite 2.0 the preferred file size is about 4 times bigger, approximately 400 kB. These values can and will vary from device to device, these are just rough guidelines. We'll go a bit deeper in the area of optimization in Chapter 9, "Performance."

7 Usability issues with Flash Lite applications

The user experience of a mobile application is formed by many factors, an important one of them being usability. It is sometimes hard to separate usability from other matters that affect the user's attitudes and feelings towards a service: For example, the performance of an application surely affects the user experience, but it is not strictly taken a usability issue.

Usability of Flash Lite applications is a rather unstudied field and thus no known specific conventions and guidelines exist yet. However, the most general design principles for mobile user experience also apply to Flash Lite user interfaces. This chapter briefly explains some existing design conventions for mobile applications that we consider relevant for Flash Lite applications and Flash Lite-specific guidelines that have been created based on interviews of Flash Lite designers. These guidelines help you to ensure the basic usability of your application. If and when you decide to try out something totally new, it is always best to test your design with real end users and see how they cope with your design.

7.1 Existing usability guidelines for mobile applications

There are loads of existing guidelines for mobile applications, including guidelines for Java games available on the Forum Nokia Web site, some of them being specific and some more general. The guidelines presented here are mainly based on our experiences as user interface designers and researchers. We have also included some of the points listed in the recently published [Mobile Web Best Practices 1.0](#) by W3C [15].




7.1.1 Clear focus

Good usability is much about showing the user where he or she is and what the options are. Thus clear focus indication is a must. In Flash Lite (1.1 and 2.0) you should absolutely not trust the automatic highlight of the Flash Player, but add `onPress` event handlers (see the following ActionScript Example) to your buttons and design proper hover states for them. Also remember to set the `_focusrect` property to `false` in order to avoid the automatic yellow focus. Items that are not focused still have to be visible and clear so that the user knows what the options are.

ActionScript
<pre>my_button.onPress = function () { selectedItem = this; gotoAndStop("frame_01"); };</pre>
Applying onPress event handlers to a button.

Mobile devices often do not have good color contrast and they are used in different lighting conditions. Hence information highlighted in color may not be visible to users. If color is used to indicate a feature, that feature should generally also be indicated in a way that is not color-dependent. [15]

Prefer	Avoid	Avoid
--------	-------	-------

		
<p>The focus indication is clear and also the other options are visible.</p>	<p>The focus indication is clear but on the expense of other selectable elements.</p>	<p>The focus indication is not clear.</p>

<p style="text-align: center; color: red;">Avoid</p>

<p>The standard Flash Player tab selection has not been overridden.</p>


7.1.2 Stepwise navigation and information

Mobile devices have small displays and limited input systems. Besides this, they are often used when being on the move. Thus the information that is being shown to the user needs to be simple, to-the-point, and offered in small pieces. Also pay attention to the language: Prefer brevity and directness instead of a discursive writing style. [15]

Prefer

Making Coffee

Making coffee is easy. Just follow the instructions and succeed!




Start →

Options Exit

Step 1

Decide how many cups you want to make.




← 1 2 3 4 5 →

Options Exit

Step 4

Switch the coffee machine on and wait until there the coffee is ready.



← 1 2 3 4 5 →

Options Exit

The text is split into 5 separate steps and the language is compact and structured.

Avoid

Making Coffee

Making coffee is rather simple. What you need is a coffee machine, some coffee, a measure spoon, water and a filter paper. First you decide how many cups you want to cook. Let's say you want to have 5 cups. You measure 5 cups water and pour it to the machine. Then you open up the paper filter and place it to its place. In the filter you pour 5 full spoons of coffee. After that you switch on the coffee machine and wait until you don't hear

Options ⬆ Back

The text is long, no paragraphs are used, and the text is presented in one long piece.

7.1.3 Consistent style

Consistency is vital for usability but it also helps you to keep the whole project sorted out. Being consistent is much about planning and deciding at least the major guidelines before starting to do the programming and visual design. It is about being creative and clever while choosing the style and then sticking to it no matter what. Consistency strictly from the usability perspective means consistency of visual design and consistency of interaction.

7.1.4 User input

As mentioned before, mobile devices have limited input and viewing capabilities. The less the user needs to insert text, the better. Instead of free text input you should use selection lists, radio buttons, and other controls that do not require typing. Also try to offer default values that are relevant to the

user (for example, in the address field of a mobile browser having the `www.` prewritten can be considered helpful). Previous entries can also be sometimes helpful when used as default values. [15]

7.1.5 Color

Most of the mobile devices in the market emphasize blue tones. Thus a blue gradient will look better than, for example, a red one. This does not mean that you should not use any other colors than blue — this is for information only.

Ensure that critical information must be also understandable when viewed without color, because mobile devices often do not have good color contrast and are often used in less-than-ideal lighting conditions. [15]



7.2 Flash Lite-specific usability guidelines

This section presents issues that have come up while discussing Flash Lite and usability with Flash Lite developers.

7.2.1 Guideline 1: Respect the UI conventions of the platform you are designing for

People are used to a certain interaction style when using their devices. Nokia S60 devices have a consistent UI style that the users feel safe with. The UI conventions of the platform that your Flash Lite application is supposed to be running on are a good starting point for your design. With Nokia S60 devices the most important design guidelines are:

1. Placement of softkey labels
 - Options for exiting and moving backwards are behind the right softkey.
 - Menus and other kinds of options are behind the left softkey.

Prefer	Avoid
	
Courtesy of Blueskynorth	

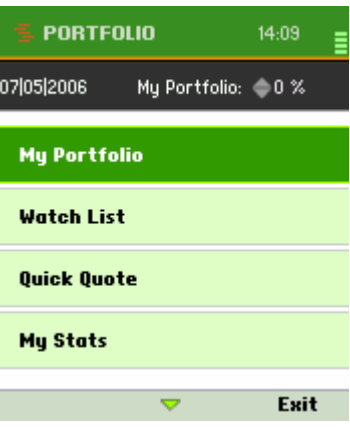
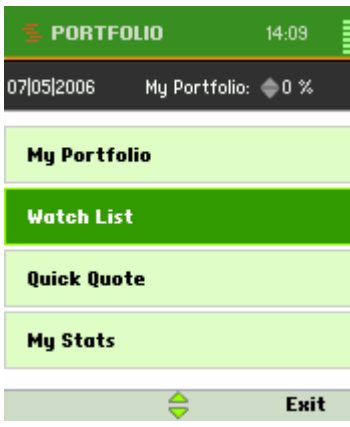
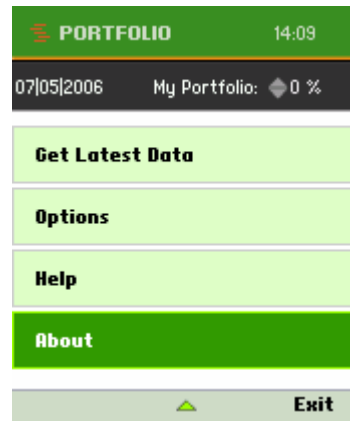
2. Visibility of softkey labels
 - If there are actions assigned to the softkeys, the labels are always visible.
3. Size of softkey labels
 - Softkey labels should be located so that they are clearly on the right and on the left.

- The maximum length for a softkey label is about 35 % from the left/right margin.

S60 UI	Prefer	Avoid
		

4. Scrolling

- Scrolling is always indicated with arrows or with a scrollbar.
- Limit scrolling to one direction unless secondary scrolling cannot be avoided:
The page layout should be so simple that repeated scrolling in the same direction (axis) allows the user to see all the content. However, some content (such as maps and other images) cannot be displayed without secondary scrolling. [15]

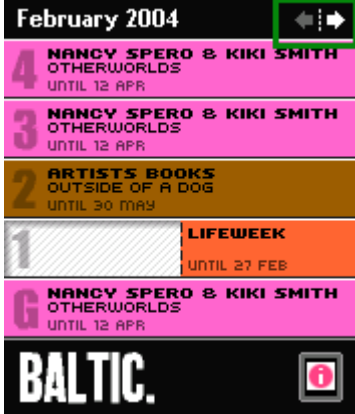

Prefer	Prefer	Prefer
		
In the beginning	In the middle	At the end

Three stages of scrolling through the list (courtesy of Flash Cell). This way of indicating the text or list length is preferred only if the text or list is rather short. With longer content using a scroll bar is recommended.

5. Navigation key and actions on the screen share a logical mapping.

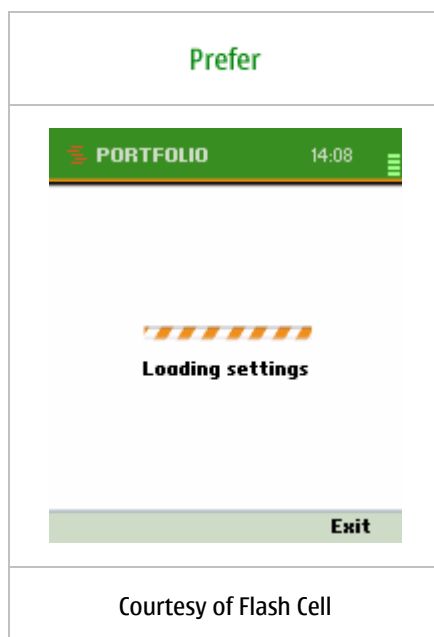
When the user presses the left/right/up/down navigation key, he or she also moves left/right/up/down on the screen. For example, scrolling text should not happen by pressing the left and right softkey.

This still does not mean that you could not be creative with your navigation. Take a look at the examples below.

Prefer	Prefer
	
<p>Up + down move up + down on the list, left and right browse the calendar back and forth. Small arrows, marked here with green, indicate the horizontal movement. (Courtesy of Blueskynorth)</p>	<p>Left + right move from page to page. The pages are listed horizontally at the bottom of the content area and marked here with green. (Courtesy of Blueskynorth)</p>

6. Always allow the user to go back and exit the application with control.

Remember to check that there are no one-way streets in your application. You should always allow the user to browse back to the previous stage. If that does not seem reasonable (for example, in a game), you should allow the user to exit the application in a controlled way providing a visible option for that.





7.2.2 Guideline 2: Test your application on all possible devices it can or will be used on

The most radical difference between developing Flash Lite content and regular Flash is probably the testing phase. Regular Flash will run on most of the computers and will look almost the same, but this is not the case with Flash Lite. There are hundreds of different mobile devices available and they all have different displays, memory processing capacities, memory space, and input systems. Thus it is highly recommended that you test your application on as many devices as possible. You will probably notice differences in at least these things:

- Colors
- Performance
- Display dimensions (vector graphics are scalable, but it wont help if the dimensions do not grow linearly)
- Input systems



7.2.3 Guideline 3: Showing device information

Respect the platform conventions also here. If the battery status is visible in the upper right corner, you should keep it there also in your application. If you break the design conventions of the platform, indicate it clearly, for example, by showing a symbol for battery and signal.

Prefer	Avoid
	
<p>Battery and signal status are shown in the upper part of the display. They are both indicated with symbols that are easy to understand (Courtesy of Blueskynorth)</p>	<p>Battery and signal status and the date and time information are in an unexpected place. There are no symbols either to indicate which one is battery and which one signal.</p>

7.2.4 Guideline 4: Use Flash to guide the user in making the right navigation steps

As Flash allows you to create elements that have a unique shape, use it. Use arrows and buttons to show the user where to go next. Emphasize the actions the user might take.

Prefer	Avoid
	
<p>The Play button is animated and big and the most probable navigation choice for the user. (Courtesy of Blueskynorth)</p>	<p>In this example the most probable interaction step (“Enter”) is tiny in comparison to the not so probable ones (“Menu” and “Exit”).</p>

7.2.5 Guideline 5: Prefer pixel/bitmap fonts

When the screen resolution and the display are small, making text readable and easy to understand is not the easiest thing to do. You can achieve better readability by simplifying the text and its layout, but also using pixel fonts. Pixel fonts are fonts that consist only of black/white pixels and are not antialiased. You have two ways to use pixel fonts: You can either purchase/download them or you can apply the bitmap setting on Flash 8 Professional © (see the following picture). You can also use device fonts (also visible in the picture) to ensure readability.

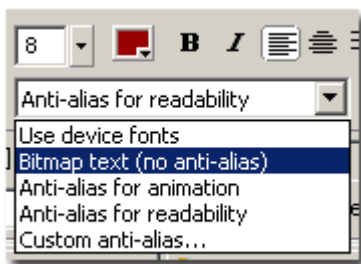


Figure 1: Selecting the bitmap setting from the text properties in Flash 8 Professional ©

When using pixel fonts you must take a few things into account to ensure a successful outcome:

1. **Pixel fonts have only one size.**
The size or sizes are being told when you purchase or download the font. You cannot use the font in any other size without visual fragmentation.
2. **Embed the font.**
Embed only the characters you need. If you embed everything, it will increase the file size.
3. **Place the text on full pixels.**

Use the **Snap to Pixel** feature under **View** to keep fonts clear. If the fonts are not of whole `_x` and `_y` values, they will appear blurry.

4. Round values to integers when using fonts in motion.

When you use pixel fonts in motion, you need to round the `_x` and `_y` values to integers to keep the fonts aliased. Use the following ActionScript to round the x and y coordinate numbers down to the closest integer:

ActionScript

```
_x = Math.floor(_x);
_y = Math.floor(_y);
```

Partly © 2005 Craig Kroeger. [14]

Prefer	Prefer	Avoid
<p>Hello Guideline readers! This is Arial, 11 points, so now you need to scroll a bit. On this text I have also applied the bitmap setting + the text is rendered as HTML. Remember that you need to embed the fonts to your file, otherwise device fonts will be used. Embed only characters you need so that the file size stays reasonable. This is</p>	<p>Hello Guideline readers! This is Arial, 9 points. On this text I have applied the bitmap setting + the text is rendered as HTML. Remember that you need to embed the fonts to your file, otherwise device fonts will be used. Embed only characters you need so that the file size stays reasonable.</p> <p><i>This is now italic. Can you read it? I think it's quite easy to read this, but only if the text is short enough!</i></p>	<p>Hello Guideline readers! This is Arial, 9 points. On this text I have applied the antialias for animation setting + the text is rendered as HTML. Remember that you need to embed the fonts to your file, otherwise device fonts will be used. Embed only characters you need so that the file size stays reasonable.</p> <p><i>it's better not to antialias your text, because it gets so blurry!</i></p>

7.2.6 Rule 6: Be considerate about sounds

If you decide to have sounds in your application, there are few things that you need to take into account:

- With Flash Lite 1.1 you should enable the user to access the **Flash** menu at least on one page because it is the only place where he or she can adjust the volume.
- With Flash Lite 2.0 it is possible to control the volume via ActionScript, so you should override the Flash Player menu and make a neat volume adjustment to your application.
- Also remember that sounds have proved to be quite performance-consuming (at least with Flash 1.1), so leaving them out might make using your application smoother.


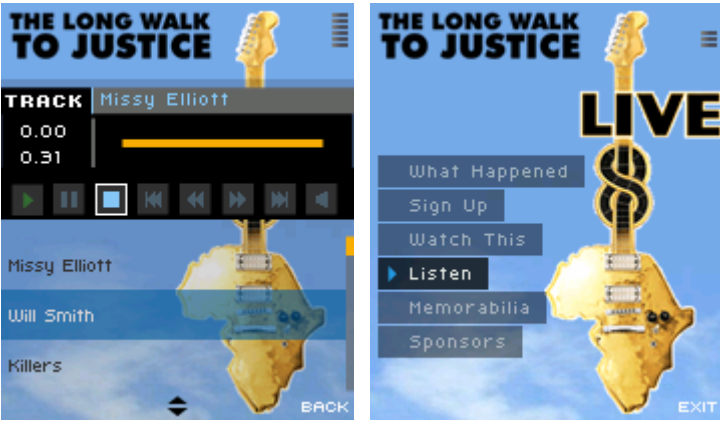
7.2.7 Rule 8: Have an item preselected on the page

It is easier for the user to know where to start from when an item is already selected.

ActionScript

```
if (selectedItem == null) {  
    Selection.setFocus (what_happened);  
}
```

When the application first starts, focus is set to the top menu item 'What Happened'.

Prefer	Prefer
 <p>The screenshot shows a menu titled 'THE LONG WALK TO JUSTICE LIVE' with a guitar graphic. The menu items are: What Happened (selected with a blue highlight), Sign Up, Watch This, Listen, Memorabilia, and Sponsors. An 'EXIT' button is at the bottom right.</p>	 <p>The left screenshot shows a music player interface for 'Missy Elliott' with a progress bar and playback controls. The right screenshot shows the same menu as the first screenshot, but with 'Listen' selected and highlighted in blue. A 'BACK' button is visible at the bottom right.</p>
<p>The first menu item is selected when the page is entered.</p>	<p>When the user returns to the menu, the focus is where he or she left the menu.</p>

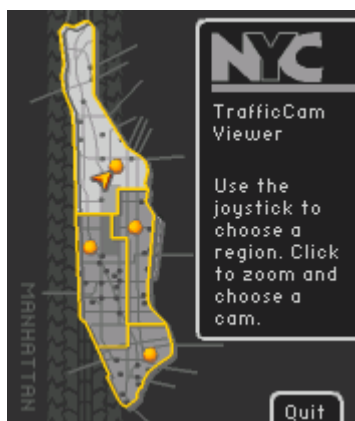
7.2.8 Rule 9: Allow the user to control animations and volume

Especially with long animations the user should be able to pause, stop, and exit them. In Flash Lite 2.0 the volume can be adjusted via ActionScript, so if you are using sounds it is recommendable to let the user set the volume to a level he or she feels ok with. A rule of thumb with sounds in user interface design is that they are OFF by default.

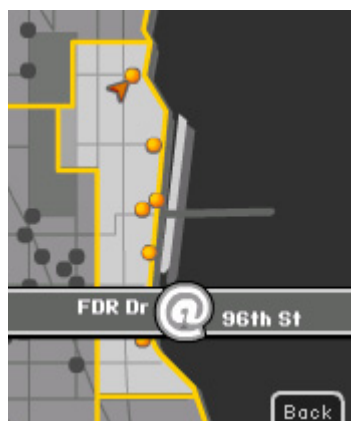
8 Innovative Flash Lite user interface design

Even though we suggest that you respect the conventions of the platform you are designing for, it does not mean that you should or could not be creative with your Flash Lite UI. Respecting conventions is just a starting point, but there is much more to the Flash Lite UI than to just resemble the conventional mobile user interface. What really makes Flash Lite so interesting is that it gives possibilities to experiment with interaction and design. If used properly, Flash Lite gives you an excellent way to highlight indicators and actions with sophisticated animations. Here are some good examples that are both usable and unique.

The award-winning Manhattan application (downloadable at [2] shows that the device display can also be used the other way round. (Courtesy of Justin Everett-Church)



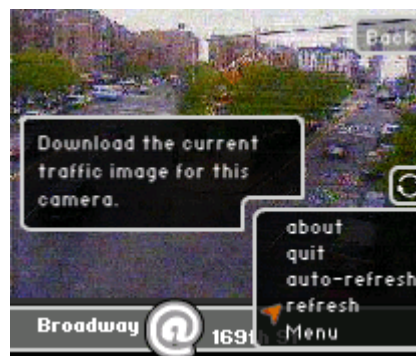
1. Choosing a region



2. Taking a closer look

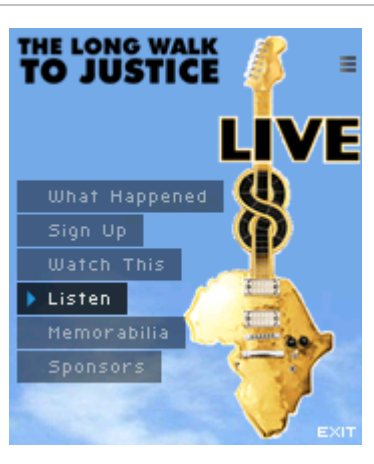
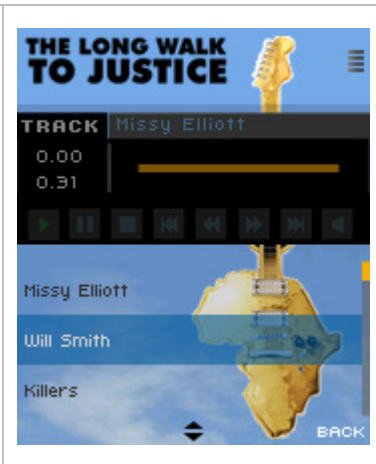



3. The UI transition
The application's UI is transitioned from portrait to landscape. The animation guides the user really intuitively to physically turning the phone the other way round. This is a really good and innovative UI design that can be achieved with Flash Lite. (Download the application and try it out yourself to get a better idea of how it works).

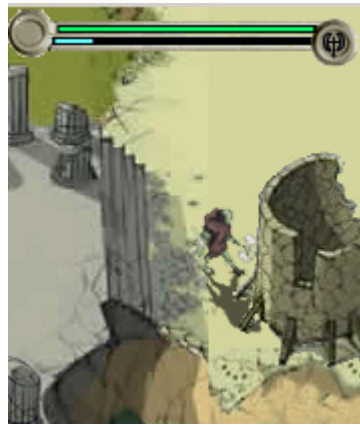


4. The end result
The camera picture is viewed horizontally like in any other camera display.

This “Live 8” example from Flash Cell shows that the UI does not have to be boring or even traditional to be usable. (Courtesy of Flash Cell)

		
<p>The menu works as any menu — it just looks different.</p>	<p>The artists are selected with the vertical (up+down) navigation keys...</p>	<p>...and the music player is activated and used with the horizontal navigation keys (left+right).</p>

A role-playing game “Half Breed” by Aniway has unique hand-drawn graphics. (Courtesy of Aniway)



9 Performance

The performance of your application is one of the key things that will affect the user's experience. Thus you need to think about it all the time when you develop your application. Here are some tips on how to make your application run smoother:

1. Shortening variable names can help if you are using hundreds of name/value pairs. It will not make much difference with 50 or so variables.
2. Changing vectors to bitmaps can help in some situations but not always. You need to test both formats on the actual device to see if the performance improves. It has been noticed that bitmaps work well for static content. If you are using bitmaps, remember the resolution and scalability issues. With Flash Lite 2.0 images can be easily loaded during run time to a specified level, so even though it requires a little work, there is a solution to it.
3. Keep the amount of vector gradients, alphas, and curves to a minimum.
4. Keep the amount of action-script loops and the amount of code executed in each loop to a minimum.
5. Keep the amount of simultaneous animations on the screen to a minimum, and animate one or two objects at a time.
6. Even though this sounds boring, avoid, at least for the time being, music/sounds if possible.

10 Conclusions

Flash Lite — when used correctly — has true potential in improving the user experience of mobile services. With Flash Lite the mobile UI designers and developers have better chances than ever before to realize their visions about good user interfaces.

The message of this document can be summarized as follows: Get crazy with Flash Lite — now is your time to show your stuff, the audience is waiting! But also remember the small rules discussed in this document. This way you can immediately achieve much more satisfied users, create a better Flash Lite community, and make your learning curve steeper when designing with Flash Lite. Now, please, go on and impress us.

11 References

- [1] [Flash Lite for S60 - an emerging global ecosystem](http://www.forum.nokia.com) available at www.forum.nokia.com
- [2] Adobe/Macromedia
<http://www.adobesystems-macromedia.com/>
- [3] Flash Lite
<http://www.adobesystems-macromedia.com/products/flashlite/>
- [4] Adobe Press Release: Nokia Signs Licensing Agreement with Macromedia
http://www.adobe.com/macromedia/proom/pr/2005/nokia_flashtechnology.html
- [5] DoCoMo
<http://www.nttdocomo.com/>
- [6] Adobe Flash 8 Professional
<http://www.adobe.com/products/flash/flashpro/>
- [7] Flash Lite 2 Update for Flash Professional 8
<http://www.macromedia.com/support/flash/downloads.html#flash8pro>
- [8] Adobe's Flash Lite Nokia Discussion Forum
<http://www.adobe.com/cfusion/webforums/forum/categories.cfm?forumid=68&catid=472>
- [9] Symbian
<http://www.symbian.com>
- [10] Advantages of creating a .sis file, thread in the Flash Lite Nokia discussion Forum
<http://www.adobe.com/cfusion/webforums/forum/messageview.cfm?catid=472&threadid=922676&enterthread=y>
- [11] Device Profile Updates
http://www.macromedia.com/software/flash/download/device_profiles/
- [12] Macromedia Flash Lite Content Development Kit
<http://www.adobe.com/devnet/devices/flashlite.html#cdk>
- [13] Getting Started Developing for Flash Lite
http://www.adobe.com/devnet/devices/flashlite_gettingstarted.html
- [14] Minimal Fonts Guide
<http://miniml.com/fonts/guide/index.htm>
- [15] Mobile Web Best Practices
<http://www.w3.org/TR/mobile-bp/>
- [16] IDC White Paper: Addressing Growing Handset Complexity with Software Solutions
http://www.adobe.com/mobile/news_reviews/articles/2005/idc_whitepaper.pdf

Companies that have contributed to this document

- [1] Aniway
<http://www.aniway.fi>
- [2] Blueskynorth
<http://www.blueskynorth.com>

[3] Flashcell
<http://www.flashcell.com/home.cfm>

[4] Idean Creative
<http://www.ideancreative.com>

Downloads

[1] Works of Blueskynorth
<http://blueskynorth.com/BSN3/portfolio/index.html>

[2] Manhattan application by Justin Everett-Church
<http://www.infinitumdesign.com/mobile/manhattan.html>

12 Evaluate this resource

Please spare a moment to help us improve documentation quality and recognize the resources you find most valuable, by [rating this resource](#).