# Is Flash Really Accessible When Interacting through Screen Readers?

Barbara LEPORINI, Fabio PATERNO', Lucio Davide SPANO
*ISTI-CNR*
*Via G. Moruzzi 1, 56124, Pisa, Italy*
*{barbara.leporini, fabio.paterno, lucio.davide.spano}@isti.cnr.it*

**Abstract**. In this paper we report on a study, which aims to assess whether the recent versions of Flash allow designers to obtain really accessible applications when users interact through screen readers. A Flash application has been analysed for this purpose along with various solutions for improving its accessibility. The results show that in some important cases Flash applications still raise important accessibility problems.

**Keywords**. Accessibility, Screen Readers, Flash applications.

## Introduction

Flash is increasingly used for Web and standalone applications due to the offered multi-media features and opportunities. With Flash several enhancements can be made available through hypermedia and multimedia objects: toolbars, animations, interactive games, etc. Thus, it is important that Flash applications be accessible to all users, including those who have to interact through assistive technologies. Several years ago the use of Flash was criticised for the difficulties that it raised for users with disabilities for various reasons [4], such as for lack of alternative textual descriptions. However, in recent years Flash has improved in this respect aiming to provide support for accessibility [1]. Thus, we found it useful to investigate at this time what aspects can be developed in an accessible format in Flash for users who navigate via screen readers. Other investigations have been done in this field [2] and [3] but they do no provide a thorough analysis of such issues.

In the paper we first report on the application that has been used as case study. In particular, we describe its design and implementation, and how the Flash accessibility support has been used. We then report on the accessibility issues as well as possible further solutions considered to solve them. We conclude discussing the main issues detected and providing some indications for solving the current accessibility issues.

## 1. A Case Study: A Museum Application

The application considered describes the tactile room in the Museum of Natural History (Calci, Italy). This room has been designed according to accessibility rules. It allows visitors to touch animal samples and marine and terrestrial plants. The room provides information about the various environments (mountain, river, sea). The Flash application aims to support various types of users: those who would like to visit the museum, and thus receive information about its spatial arrangement; and people who have visited the museum, and would like some more detailed information. In order to improve the learning experience, the interactive application supports a set of interactive games to help learning of the associated contents (recognise an animal, associate an animal to its environment, associate an animal to its taxonomic group).

The application is composed of two main parts: the museum content presentation and games for interactive learning. The museum content UI consists of four parts (see Figure 1): the header, the navigation bar, the content and the footer. While the header and the navigation bar are mainly static, the content part is a dynamic text area, which displays the information about the current section. The content is separated from the UI and stored into XML files through HTML tags so that it can be modified without changing the application code. Consequently, the main content presented is a dynamic text area, which displays the information on the current section loaded dynamically by an Action Script reading the XML file, and is composed of text and links.


**Figure 1.** The Application User Interface.

To implement an interactive and attractive game as well as to test how to make it accessible via an alternative version, we used drag-and-drop function. In order to make the game accessible via keyboard, a hidden alternative version detectable only by the screen reader - embedded in the same graphical version - has been developed in which the available choices have been implemented through radio buttons. For example, for the game "Assign the animal to the right environment", the three choices "sea, wood, and meadow" have been provided by three radio buttons. Thus, the accessible version is just detected by the screen reader and it is not visible.

## 2. Accessibility Problems

Even if accessibility support made available by the Flash environment (i.e. 8 and 9 versions, through the Accessibility panel) was used when developing the application, the resulting UI has not been found completely accessible through screen readers. A Flash application can be launched in standalone version as well as embedded in a Web page as a multimedia object. Unfortunately, at the moment the standalone version cannot be made accessible through a screen reader. In order to make it accessible for blind users, the application must be embedded in an X/HTML page. Consequently, to interact with it a Web browser has to be used. In this perspective, we tested the application by using the screen reader Jaws (http://www.freedomscientific.com/) version 9.0 with the Internet Explorer 6.0 and 7.0. When the screen reader intercepts the Flash application in the X/HTML code, the user is informed through the message: "Flash movie start ...[text, buttons, etc.]... Flash movie end". In short, the main accessibility problems are related to content reading order - i.e. text, buttons and so on are mixed - especially when contents are built through a dynamic way. In detail, the following drawbacks were identified in our study:

- *Content reading order*: as mentioned previously, the main components of the considered interface are (1) header and navigation bar, (2) main dynamic content, and (3) footer. The main content - composed of text, buttons, images and links - is built dynamically by loading the content from an XML-based external file. In Flash Player 8 or higher, the reading order can be partially specified. The most precise means for controlling reading order is to use tab index values [5]. To control reading order the accessibility panel or ActionScript can also be used. In our case, to give a correct order we assigned the Tabindex property by using the accessibility panel available in the Flash development kit. Despite the clear separation of each part (i.e. navigation bar, content, etc.), and the Tabindex property, when the screen reader intercepts the interface the content is somewhat confused. For instance, firstly some buttons belonging to the navigation bar are read, then some text of the main content is rendered, and then some buttons of the navigation bar are encountered again, as they are mixed in with the main content. Also, other menu or navigation bar buttons are recognised after the footer. Generally speaking, the content is mixed, so that the user risks being confused when reading in a sequential way. Figure 2 shows an example of how the screen reader Jaws interprets the application content. The part in bold indicates the kind of control element detected and rendered by Jaws. The information in the square brackets reports the role of that text in the application (e.g. the general title, the current section title, etc.).

Flash movie start
Play-pause music Alt+3 **button**
cd-rom accessibility Alt+u **button**
The tactile room [The CD main title]
Presentation [current pressed button]
Basic Concepts **button** [navigation bar]
6 **button** [unknown]
Museum of Natural and territorial history [CD subtitle]
The room **button** [navigation bar]
11 **button** [unknown]
Learning by touching [dynamic content title]
games **button** [navigation bar]
"The Nature to hand ....." [dynamic content]
Bibliography **button** [navigation bar]
...
Exit **button** [closes the application]
Flash movie end

**Figure 2**. An excerpt of how the screen reader Jaws describes the application.

- *Images*: the dynamic text area displays the image loaded using the <*img*> tag, but the screen reader does not read the *alt* attribute. In fact, Jaws is not able to detect images in the page or to read their alternative descriptions, whereas those (i.e. labels) associated to button images are appropriately detected.
- *Links*: the dynamic text area displays the links but the screen reader cannot list them, and thus they cannot be used by the visually impaired.
- *Buttons*: Buttons do not present particular accessibility problems. A strange problem occurs for some buttons inexplicably detected by the screen reader as unlabelled. In this case the screen reader provides messages such as "six button", "nine button", but it is not clear what those buttons refer to. These unclear buttons are enumerated by the screen reader (in the same way as when the developer has forgotten to assign a label). In some cases we supposed Jaws recognised the buttons associated with the scroll bar as unlabelled buttons. Unfortunately, this problem also occurs when no scroll bar is in the page.
- *Shortcuts*: Even if shortcuts are assigned to buttons or links, they can not be used. The indications provided in [5] are: “Using the shortcut field on the accessibility panel or the shortcut property in ActionScript is not sufficient for this purpose. Creating a keyboard shortcut requires that a listener event be defined and a script associated with that listener”. However, even in this way when interacting through the screen readers the shortcuts do not work. If the application is used without an active screen reader, with some browsers (e.g. Firefox) the shortcuts work.

In order to overcome the accessibility problems described, we also developed a specific parser to map the HTML structure of the dynamic content of a presentation into Flash UI components (widgets). The main idea was to create a script that dynamically parses the HTML content, creates the corresponding Flash UI components

and sets the accessibility information for images, titles and so on. The size of these components is then set to only one pixel in order to not interfere with the overall layout: they become impossible to see but the screen reader can detect and render them vocally. We implemented a top-down parser that recognizes the subset of HTML tags that defines the text content structure (as titles, paragraphs etc.), the images and hyperlinks. Then, the dynamic components creation proceeds by creating separate specific text fields for various purposes (e.g., title, paragraphs, links, and so on). In particular, in the case of links button elements were introduced to open the target content in the dynamic text area. However, even using this solution, the application is still unusable through screen readers. In fact, the interface components are not read in the correct order: the content with the section buttons and the widgets dynamically added by the script are mixed, even if the tab indexes are correctly set.

## 3. Conclusions

Our experience in the case study has highlighted that relatively complex Flash applications cannot be developed while maintaining proper accessibility. Various accessibility problems still exist for Flash components, probably due to the fact that on the one hand Flash technology is not yet "mature" for hypermedia application accessibility; and on the other hand, that the screen readers might not be able to completely interact with Flash applications. The set of interactions that can be made accessible is quite limited. A more specific API (application programming interface) – such as ARIA (Accessible Rich Internet Applications) for the Web - should be provided for improving the communication between Flash Technology and screen readers. The results discussed herein could be taken into account by Flash and assistive technology developers in order to improve accessibility features for both environments. As previously reported, one of the greatest difficulties observed when interacting through screen readers is related to the reading content order, since the text, links, images, and so on are mixed when read sequentially. For these reasons, we suggest making use of Flash technology for some components of the page, such as the navigation bar - when it is composed of buttons - or for those objects requiring graphical animations.

## Acknowledgments

We thank Martina Lorenzoni for her work on the implementation of the Flash application.

## References

[1] Adobe, Accessibility Resource Center, at http://www.adobe.com/accessibility/
[2] Cantón, P., González, A. L., Mariscal, G., Ruiz, C. (2008). Building accessible Flash applications: An XML-based toolkit. *In LCNS Vol. 5105,* ICCHP Conference, Linz, Austria, 2008, pp. 370-377.
[3] Krüger , M. (2008). Accessible Flash is no Oxymoron: A Case Study in E-Learning for Blind and Sighted Users. *In LNCS Vol. 5105 (2008),* ICCHP conference, Linz, Austria, 2008, 362-36
[4] Nielsen, J.: Making Flash Usable for Users with Disabilities (October 14, 2002),. http://www.useit.com/alertbox/20021014.html
[5] Regan, B. (2005). Best Practices for Accessible Flash Design, August 2005.