Chapter 8
UI Patterns Support on RIA's Development

ABSTRACT

In Chapter 8, a review of UI patterns supported by using non-JavaScript-based frameworks and JavaScript-based frameworks for RIA's development is presented. For this purpose, the UI Patterns are classified in Transitional Patterns, Lookup Patterns, and Feedback Patterns. Additionally, this chapter presents a series of applications samples depicting the use of not only UI controls but also non-visual functions or classes (as applicable) for implementing the supported UI patterns. These applications are intended to clarify the differences and similarities between UI controls and UI patterns. Finally, the chapter presents a comparative analysis that summarizes the review of the capabilities of the RIA frameworks in the context of rich UI design both for JavaScript-based frameworks and non-JavaScript-based frameworks.

1. INTRODUCTION

RIAs (Rich Internet Applications) design involves two main stages: 1) application structure design and 2) GUIs (Graphic User Interfaces) design which, in turn, is composed of three subtasks: a) layouts design, b) GUI controls design and c) interactions design (Neil, 2009).

Layouts design covers the arrangement of visual elements, i.e., GUI controls across the screen. In this sense, there are common ways to arrange the RIA's GUI controls, namely canvas, vertical and horizontal layouts. In fact, most of the RIA frameworks consider these standard layouts; thereby, Web 2.0 community has outlined a set of screen patterns that can be used at this design level, and they can be implemented as a combination of GUI controls containers such as panels, forms and the aforementioned layouts.

GUI controls design covers the design of the GUI controls that allows applications to communicate the required user interactions. Typically, RIA frameworks offer both a set of simple GUI controls acting as application building blocks and a set of look and feel customization mechanisms. However, most of the RIA frameworks also offer the possibility of adding custom GUI controls built from simple controls.

In another hand, interactions design implies the usage of patterns aimed at achieving effective user experiences, i.e., rich interactions. In this sense, most of the RIA frameworks offer simple GUI
controls that natively implement interaction patterns, e.g., progress bar (progress indicator pattern) and accordion (expand/collapse pattern) in Adobe Flex™ or JavaFX™. However, some interaction patterns require more development time and effort in order to be implemented. Therefore, it is not easy to understand the differences between GUI controls and interaction patterns. Interaction design patterns are commonly used to ease the interaction between users and Web 2.0 applications by bringing more intuitive and responsive user experiences, i.e., rich user experiences; in short, they allow developers to encourage users to engage with applications. The most popular interaction patterns from the point of view of the authors are described below.

- **Brighten and Dim**: it is actually considered a pair of interaction design patterns intended to focus user attention on an area of the GUI by brightening there at the same time that the rest of the GUI is dimmed to indicate that is not in use. The most common technique to achieve this consists on displaying a non-dimmed overlay over a dimmed background which is called lightbox effect.

- **Expand/Collapse**: it allows keeping additional contents hidden until the user needs them; it is also useful in cases where the content length exceeds the length of the area of the GUI intended to display them.

- **Animation**: it is a means for clarifying user interactions by mimicking object transitions from the real world. This pattern is closely related to the drag and drop interaction pattern because it is commonly used to clarify where objects came from and are going.

- **Spotlight**: it is one of the major mechanisms for notifying users about changes occurred on applications. It consists on temporarily focus the user attention on changes in the GUI even when these changes are explicitly triggered by users.

- **Auto Complete**: it allows users to save time on filling out forms which is over-used among Internet search engines such as Google™, Bing™ and Yahoo!™ with the aim of optimizing searches by providing real-time term suggestions for creating searches starting from previous entries. At this point, this pattern is known as live suggest.

- **Live Preview**: in the context of form submission, it is used to prevent mistakes by providing real-time feedback about form inputs. It can also be used in the context of e-commerce as a means for showcasing customizable products such as clothes and cars.

- **Progress Indication**: it is a means for keeping the user informed about the progress of a lengthy operation running in background; it is also used to simply indicate the user that must wait for an operation to be completed; i.e., as an indication of a busy application.

- **Periodic Refresh**: it is useful to maintain the content of an application up-to-date; in this sense, the new content can be loaded either automatically or on demand as is broadly used by some microblogging websites such as Twitter™ and Facebook™.

According to the above, this chapter is aimed at reviewing the support for implementing the interaction design patterns outlined above among popular non-JavaScript-based and JavaScript-based RIA frameworks not only in terms of GUI controls but also in terms of non-visual classes. This systematic review is finally summarized in a comparative analysis of the capabilities of the RIA frameworks in the context of rich GUI design. In addition, the differences and similarities between GUI controls and interaction design patterns are intended to be explained by using sample appli-
Related Content

**Software Modernization of Legacy Systems for Web Services Interoperability**
Chia-Chu Chiang (2008), *Encyclopedia of Internet Technologies and Applications* (pp. 551-557).
[www.igi-global.com/chapter/software-modernization-legacy-systems-web/16903?camid=4v1a](www.igi-global.com/chapter/software-modernization-legacy-systems-web/16903?camid=4v1a)

**Internet Gambling**
Mark Griffiths and Adrian Parke (2008), *Encyclopedia of Internet Technologies and Applications* (pp. 228-234).
[www.igi-global.com/chapter/internet-gambling/16858?camid=4v1a](www.igi-global.com/chapter/internet-gambling/16858?camid=4v1a)

**Quality of Service Architectures**
[www.igi-global.com/chapter/quality-service-architectures/16885?camid=4v1a](www.igi-global.com/chapter/quality-service-architectures/16885?camid=4v1a)

**A Taxonomy of Online Game Security**
Kuen Park and Heejo Lee (2008), *Encyclopedia of Internet Technologies and Applications* (pp. 606-611).
[www.igi-global.com/chapter/taxonomy-online-game-security/16910?camid=4v1a](www.igi-global.com/chapter/taxonomy-online-game-security/16910?camid=4v1a)