+ E-mail article
+ Printable version

Design 101 for developers

by Michael Koch

Let's face it: Rich Internet applications (RIAs) are still somewhat of a novelty. They are cool, and they make heads turn, but they don't always live up to their promises or potential — to increase brand awareness, sales, or user productivity. Many RIAs could be so much more effective and user friendly if only their presentation designs were as usable and sophisticated as their underlying data models. Here are a few design-oriented ideas that you can easily apply to your existing processes to make a good RIA even better (read: more usable).

It's all about the experience

Whether you work on your own or together with a designer, as a developer you are responsible for a piece of your application's user experience. The more natural an RIA behaves and the more pleasing it is to the eye, the more usable the app and the more engaging the experience.

Ideally, RIAs should provide users with the same rich interface and seamless experiences they get from interacting with desktop applications — the navigation should be intuitive, the behavior should be consistent, available actions and operations should be transparent, and the user should be in control, always. If a feature or design element doesn't benefit the user or support the user's goal, it doesn't belong in your RIA. This includes gratuitous scripted presentations and splash screens, which may be cool to look at but often amount to little more than exhibitions of technical prowess that have users hunting for the Skip Intro link. Experience matters, but it has to support and lead to the successful completion of user tasks.

Know your users

To be usable and effective, your RIA must speak to your users — to their needs and expectations, and in their language. Be sure to design and develop your apps with your users in mind. Ease of use is not the only distinguishing mark of a usable app; it also must support your users' goals, expectations, and capabilities.

Naturally, your best intentions of building a successful and engaging RIA might never materialize if you ignore the importance of capturing usability and accessibility requirements and fail to budget for usability testing early on and throughout the development process.

You don't have to invest in expensive laboratory rentals and materials to learn about your users' motivations and behaviors. Visit your target users in their everyday environments and observe what they actually do to discover patterns of usage. Create scenarios and profiles that focus on users' goals and constraints to help define the strategic decisions about and conceptual framework of your app. Create prototypes early and test them on your typical users (prototypes can be hand-drawn), or ask potential users about their experiences with comparable online or offline apps. Find out what elements of your application will be most relevant for each user type and what conventions to follow, to make information appear in a natural and logical order.

User feedback can provide a wealth of new ideas for improving the functionality and interface design of your application. You know that, of course; the challenge is to communicate to stakeholders how this input will improve the usability of your application and to get them to sign off on a user-centered design process. Put together a sound usability plan and be sure to get all stakeholders to sign off on the usability requirements and testing early on. It might be a tough sell, but investing some resources in usability research and testing methodologies that fit your budget is well worth the effort. After all, building an application right the first time will reduce the overall development and maintenance costs in the long run.

Support your users' goals

The whole purpose of having RIAs is to enable users to be more productive than they were with traditional browser-based services. RIAs possess the functionality to interact with and manipulate data rather than just visualize and present it. Yet, more complex desktop-like interaction models that incorporate drag-and-drop support are still the exception rather than the norm in the fast-growing RIA market. Wouldn't it be nice if users could rearrange the order of their wish list or to-do items with a single drag-and-drop action? Or if they could drag an item to a shopping cart or trash bin instead of first having to select the item or a check box and then click Add to Cart or Delete?

In general, acting directly on an object is usually the most effective and intuitive way to affect it. If a chart needs to be resized, enable users to click its edge and drag it outward. If a user wants to read more than the teaser of an article, let them expand the space they're already in rather than taking them elsewhere.

In form-based applications, you can prevent user input errors by limiting the amount of data entry required, saving repetitious data, and supporting the automatic completion of input data. You can further assist users by designing your application so that it recognizes various input options (for example, 800.333.1212 and 800-333-1212), rather than requiring users to comply with an arbitrary entry format.

As a rule, organize information and functionality to reflect users' primary goals and tasks and enhance their feeling of competence and confidence in the task at hand.

Go easy on the chrome

Macromedia Flash and Adobe Flex make building RIAs a breeze. Using these software solutions, you can easily drag and drop your way to a rich user interface. However, this doesn't mean you should jam-pack the look and feel of your application with all kinds of bells and whistles and complex interaction models just because it's easy and you can.

Use a simple and clear design that makes your application easy to use and increases users' comfort level. Keep surface controls that are key to user tasks at the top level, and hide less commonly used functionality until users need it, and then expose it naturally. Use panels to group related information clearly and to create clear and independent units that help users focus on specific tasks and relationships, and express hierarchies and relationships through visual design.

Avoid the excessive use of eye candy, animation, sound, and interface chrome. Excessive use of chrome and effects only distracts and confuses users and diminishes the usability of your application. Remove all interface items that are not necessary for the task at hand, and limit the use of animation or sound to when the user interface is changing (for example, when a modal menu appears from the side and obscures the user's original view) — to gently alert users as to what just happened and to keep them focused, engaged, and in control.

Unless you have a design background or are working with a designer, it also doesn't hurt to read up on color theory or consult online resources that can help you with your color and text choices and options. Try to find other applications or interactive sites that exhibit the qualities that you are looking for, and learn from them. Remember, you don't want to overwhelm users with complex designs and clutter. You want to communicate information, and your design choices should never compete with the content; instead, they should facilitate the flow of information in effective, visual, and meaningful ways. If an item can be dragged, add a shadow. If a container is meant to hold objects (like a cart), use an inset icon. When using controls, use visual cues that are clear but understated. Don't focus on the functions of your applications; focus on what's important to the user — the content.

Be consistent

Like desktop applications, RIAs should be consistent in their use of visual cues, interaction patterns, navigation conventions, and features, including terminology, layout, color, and behavior. Consistency lends an air of trust and predictability to your RIA that makes users feel comfortable to explore the application.

RIAs don't care much which browser you use. This doesn't mean, however, that RIAs shouldn't be aware of their context. Print, Back, Next, and Refresh are very familiar to web users. Many RIAs frustrate users because they seem to lose their data when the Back or Refresh button in a browser is pressed. RIAs should include code that is aware of and responsive to browser history, as well as a clearly marked exit that enables users to return to the browser if the browser controls are hidden.

RIAs have so much potential, and with the expressive power of Flash and Flex you can take them as far as your skills permit. But if you really want to stand out, you may want to remember the old saying that less is more, especially when it comes to putting a rich user interface on sophisticated data models.

‹ Back

Michael Koch is a technical communicator with a penchant for tools and gadgets that make his life easier and more enjoyable.