User Interface Development Best Practices

"A good visual design, followed up with a robust UI architecture and efficient UI development makes products click immediately with users. At Clarice, we believe in the philosophy that 'Looks can kill'. Along with appearance, we focus equally on the user experience through a robust architecture and solid development practices to create efficient customer centric solutions."

Clarice Technologies

www.claricetechnologies.com | July 2012

Best Practices for UI Design & Architecture

"A good foundation often leads to a strong structure"

1. Performance
2. Integrity
3. Usability
4. Scalability
5. Localization & Internationalization
6. Reliability
7. Supportability
8. Security

User Interface
Development
Best Practices

01 Performance

Solution designed should meet specified response/processing times and throughput rates

The solution must utilize appropriate resources in appropriate time when performing the functions (memory, communications, storage)

Identify metrics to measure Performance through metrics

> Design is not just what it looks like and feels like. Design is how it works.
>
> Steve Jobs

02 Integrity

Uniformity in implementation

Cross Browser Compatibility , i.e. solution should work on all browsers

Compliance to standards

Easy to add interfaces and integrate with new or existing systems

XML for data interchange

> Always ensure uniformity in implementation

03 Usability

Implement Application Layering. Loosely couple the Presentation and Business logic components

Architect the solution following some well-defined architecture pattern like MVC2

Wherever applicable, solution must support compliance like 508 for disabled users

Make appropriate user help available for navigations and in case when processing fails in the user interaction

The User Interface should be customizable, i.e. the architecture should support future changes to the User Interface without affecting the business logic components

04 Scalability

The solution must support modifications that strongly increase the system's internal capacity by using Clustering

Horizontal Scalability- The application must scale horizontally, i.e. across multiple hardware platforms

Vertical Scalability- The application must scale vertically in terms of multiple application instances in the same server

The solution must support addition of capacity and users over time without affecting the performance

Horizontal & vertical scalability is a must

05 Localization & Internationalization

Support usage globally in different time zones , date formats & currencies

Label strings must be configurable and not hard-wired

Solution should reflect cultural cohesion, i.e. themes and interface as per the regions of usage

Use of Unicode for some language specific characters

Compliance to Localization and Internationalization standards (L10n & I10n)

06 Reliability

Solution should be architected in such a way that it is available to users at a time when it is needed

The system should maintain a specified level of performance in case of system failures or infringement of its interface

The solution must re-establish a level of performance and recover data directly affected, in the case of a failure

Implement transactional integrity in the solution through ACID (Atomicity, Consistency, Isolation and Durability) properties

> The usability and overall usefulness of any application is governed by how well it performs its functions and how easily those functions are accessed.
>
> Dmitry Fadeyev,
> Smashing Magazine

07 Supportability

Extensibility – To increase the system's functionality or performance to meet future needs

Portability - From one environment to another

Adaptability – To different environments (no software/hardware dependence)

Installability - The system can be initially installed, set up, calibrated, in any specified environment

Reusability - The individual components or services can be reused for developing additional applications with similar functional requirements

System should be extensible to increase it's functionality or performance to meet future needs

08 Security

Solution must be compliant to global security standards / regulations like PCI, SOX etc.

Design a role based access to the solution

Add secured data storage and data transmission features to the solution (data encryption etc.)

Provide secure HTTPS based internet connectivity

Proper configuration management should be designed. Configuration stores should be secured

Identify storage, security and analysis of application log files

Solution must be compliant to global security standards and regulations

Best Practices for UI Development

"While a good design lays out a good foundation for a good UI architecture; good development makes it robust and full-proof"

1. General Coding Practices
2. JavaScript
3. Cross Browser Compatibility
4. Localization & Internationalization
5. Security
6. HTML
7. Performance
8. Cascading Style Sheets (CSS)

User Interface Development Best Practices

01 General Coding Practices

Use precise, descriptive and adequate comments within a Standard Comment Template in the code

No junk “alerts” must be present in the code

Avoid Namespace Collisions by using ad hoc Namespacing, using rewrite tokens, etc.

When you improve the user experience by loading content through asynchronous requests, make sure to inform the user that an AJAX request is being processed. Without this indicator, the user may give up waiting or wonder why nothing has happened in response to their click

Try to ignore Scriplets in JSPs

Fields in forms should follow tab order (forward + reverse) or Left- to-Right and Top-to-Bottom

Special character inputs should not bepermitted in form fields, unless specified otherwise

Run-time exceptions should be caught and NEVER dumped to the users. (Users should never see any Java stack traces)

Right clicks, Ctrl + C, Ctrl +V, Alt- Back should not be allowed on screens unless client specifically asks for the same

When using select boxes, always display the text ‘Please Select’ when there is no default value selected

If an item is disabled on the screen, its keyboard shortcut should not work

Make sure that you could check/ uncheck a checkbox using spacebar

In general, strings should be left aligned and numeric fields should be right aligned, unless stated otherwise by the client

For JSP forms, both mouse and keyboard should work for navigation through the form

If an item is disabled on the screen, its keyboard shortcut should not work

02 JavaScript

Any JavaScript code that does not write out to the document should be placed within the head of the document. This ensures that the JavaScript function definitions have been loaded by the browser before it is required. It also makes it slightly easier to maintain the JavaScript code if it can always be found in the head of the document

Always wrap inline JavaScript code within comment tags to hide the JavaScript from older non JavaScript aware browsers

Use double quotes (") for HTML attributes and single quotes (') for JavaScript string literals

Add comments to your code to aid maintenance

To avoid confusion always end single JavaScript statements with a semi-colon

Use the NOSCRIPT tag to provide alternative text for JavaScript disabled browsers

Always check for the existence of the image object before attempting to access any image properties

The void () function is not supported by all browsers. Create your own void function

Do not use the JavaScript const keyword, as it is not supported in every browser. Instead use the var keyword with an upper-case variable name

JavaScript functions in libraries (i.e. sourced in HTML files) should not use document.write nor alert() statements, but instead return values to the caller

JavaScript functions can have an arbitrary number of arguments. When this capability is used, the possible use and meaning of these optional arguments must be stated in the function's comment section

Avoid the use of global variables in JavaScript

JavaScript variables, whatever their scope, must be declared with the var keyword

To avoid confusion always end single JavaScript statements with a semi-colon

03 Cross Browser Compatibility

Use tools like Firebug , Firebug Lite etc. for Debugging/stepping through JS

Always test your code on various browsers and ensure that the code looks and behaves same across different browsers. Use tools like IETester for testing on multiple browser versions

Declare a valid doctype. For instance, a “strict” doctype makes IE behave better

Make sure that your code also works with different Document Modes: E.g.: Quirks with IE

Make use of sites like Quirksmode.com for hunting down random browser differences and Browsershots.org to see how your page will be displayed in an assortment of browsers and operating systems

Make use of techniques like test- driven progressive enhancement (e.g.: check support for a given CSS property before using it on page)

> Measuring programming progress by lines of code is like measuring aircraft building progress by weight.
>
> Bill Gates

04 Localization & Internationalization

Every label on the screen should come from a resource file

All messages and errors should come from JS

All date formats should be localized and dates should be adjusted for time zones

Use Unicode as your character encoding to represent text

Avoid slang expressions and obscure phrasing in all text. At the best, they are difficult to translate, at worst they are offensive

For Internationalization, use Foreign language input

If localizing to a Middle Eastern language, such as Arabic or Hebrew, use the right-to-left layout APIs to lay out your application right-to-left

All date formats should be localized and dates should be adjusted for time zones

05 Security

Ensure XSS (Cross-site Scripting) and SQL Injection in your code for vulnerability

Solution must take care of security through session management (including session time-outs) and cookies. Use SSL to protect authentication cookies

Ensure Information Security by ensuring that data is not tampered or altered by unauthorized users

Ensure that sensitive data is protected. Secrets should not be stored unnecessarily and should not be stored in code. Database connections, passwords, keys or other secrets should not be stored in plain text. Sensitive data should also not be stored in persistent cookies

Sensitive screens should not be directly accessible through URLs (without a session)

> Sensitive screens should not be directly accessible through URLs

06 HTML

HTML pages should be viewable and have their animation working on a local client

HTML files encoding should be consistent throughout the web site

HTML pages must at least be XHTML 1.0 Transitional compliant

HTML pages must have a valid Document Type Definition

The pages should validate against the W3C validator (http:// validator.w3.org/)

No deprecated tags (e.g. <font>, , etc.) should be allowed

All JavaScript functions and CSS declarations must be placed within the <head> of the HTML document. In particular, JavaScript functions must not be in the <body>. The only allowed exception will be JavaScript embedded in an HTML file that is included with a SSI #include directive, but then the JavaScript code must be attached to an event handler in order to guarantee correct execution timing when the page is loaded

> People should never feel like a failure when using technology. Like the customer, the user is always right. If software crashes, it is the software designer's fault. If someone can't find something on a web site, it is the web designer's fault... The big difference between good and bad designers is how they handle people struggling with their design. Technology serves humans. Humans do not serve technology.
>
> Joshua Porter

07 Performance

To maximize the user experience, it's imperative that performance is optimized for web sites. Handling UI components effectively can drastically impact the rendering time of UI pages. Some of the best practices used by us in the past are listed below:

Minimize HTTP requests by using concepts like combined files, CSS sprites, image maps and inline images

Caching: Try to minimize the server hits by caching information as much as possible. Leverage browser and proxy caching

Put stylesheets at the top and scripts at the bottom

Make JavaScript and CSS external

Avoid using document.write() to fetch external resources. Instead, declare resources directly in HTML markup

Minimize the number of iframes in HTML pages

Eliminate unnecessary cookies and even if you are using it, keep cookie sizes as low as possible to minimize the impact on the user response time

Don't scale images in HTML, i.e. don't use a bigger image than you need just because you can set the width and height in HTML

Optimize browser rendering by effective use of CSS. Use standard CSS properties, if possible. Remove any inline style blocks containing CSS that is not used by the current page

Eliminate unnecessary cookies & put stylesheets at the top

08 Cascading Style Sheets (CSS)

CSS files must validate against the W3C validator (http://jigsaw.w3.org/ css-validator/)

Try to use an external style sheet as it helps to separate content from presentation and makes organization easier

The use of HTML tables for content layout is deprecated and should be avoided where CSS provides an equivalent solution

Tables should be used to present tabular data, not for page layout, wherever possible

Avoid the use of CSS hacks (filters based on browsers' bugs) as they are not future proof and are ugly. Use conditional comments instead in the document's head to source another CSS file or specify inline CSS

If a page specific style is needed, i.e. on only one page, this style's definitions must not be within the project-wide CSS files but defined inline within the <head> of the document. The only allowed exception of CSS within HTML markup is style="display:none" to hide a container as soon as possible at page load time Units for color, fixed sized font, border size, margin, padding, etc., must be consistent across the style sheet (e.g. em vs. %, color names vs. hexadecimal, px vs. pt). Units must be specified unless the value is 0

Link styles must be specified in the correct order (link, visited, hover, active)

The layout of the pages, and the CSS structure (e.g. class names), should be documented

Use hyphens instead of underscores as old browsers may not support underscores. For better backward compatibility, use hyphens instead

> There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies. The first method is far more difficult.
>
> Joshua Porter

Conclusion

"At Clarice, our focus has always been on the User Interfaces for products. Having worked on multiple UI development projects in various UI technologies and varied domains, there have been numerous learning experiences from these projects. They have been captured in form of best practices, which should be implemented by UI architects and developers to make their designs and codes extremely efficient, scalable and robust. While doing this, we believe that they would also bring immense value to customers and help reduce the overall cost and effort in building UI components."

Clarice Technologies

Capabilities

We are a one-stop shop for design & development of Web based solutions as well as Mobile applications. The various horizontals that we have experience with allows us to identify and apply these user interface development best practices to their best capacity. Our depth of experience in the mobile technology development as well as web based products, enables us to provide our customers the best UI design as well as robust development necessary for any application.

Product Engineering Expertise

Apps for iOS, Android, and Windows phone platforms

HTML5/CSS, JavaScript, JQuery, Ext JS

GWT, Flash/Flex, Silverlight, Template engines, CMS

Clarice Technologies

info@claricetechnologies.com | +91.20.4078.9520 | www.claricetechnologies.com

User Experience Expertise

Information Architecture, Interaction & Visual Design

Enterprise and consumer product user interfaces and RIAs

Total user experience for target audience

Customers

Clarice Technologies has helped design and engineer a broad range of world class products like:

Private Cloud infrastructurefor Android device sync

The Tap n Tap UI system for Android Tablets,complete with built in applications

Multiple iPhone and iPad applications for the world's top Graphics Software Company

Consumer and enterprise management solution using HTML5 for a large multinational chip manufacturing company

HTML5 application interfacing with hardware for controlling key parameters

Dashboard for CIOs covering Risk Management and Compliance Management for one the biggest security technology company

New UI system for desktop and mobile products for a leading anti-virus and internet security company

UI redesign partners for a big Indian retail bank

Corporate website, several major brand websites and internet TV platform for a leading TV channel company

Clarice Technologies

info@claricetechnologies.com | +91.20.4078.9520 | www.claricetechnologies.com