DQ2: Validation

[Collapse](https://elearning.uol.ohecampus.com/webapps/discussionboard/do/message?action=list_messages&forum_id=_952804_1&nav=discussion_board&conf_id=_441678_1&course_id=_1470762_1&message_id=_15931459_1)

Top of Form

Consider the situations where input validation is best carried out on the client-side, the server-side or both. Make recommendations that could guide a Web developer.

Bottom of Form

1. Introduction
   1. Fote, 2013
      1. In web applications, there is the client and the server. The “client” is a web browser, like Internet Explorer, Google Chrome, Firefox, etc. The “server” is a web application server at a remote location that will process web requests and send pages to the client. Web applications can contain code that is processed on the client’s browser or on the web server. However, web applications have a disconnected architecture, which means that there is never a live, constant connection between the page displayed in the client’s browser and a web or database server. The majority of the processing will be done at the server and not on the client’s internet browser. When a database needs to be accessed on a server, the web application will post the page back to the web server and server-side code will process the request.
   2. ICIEV, 2012
      1. Form validation is the process of checking the data that a user enters into form fields [3]. There are mainly two types of validation related to web applications done by developers. One is client-side and another is server-side validation. Client-side validation uses a scripting language, like JavaScript or VBScript, to check the data that a user enters into a form before the web browser sends that data to the server for processing. Client-side validation is usually very quick in responsiveness. However, it may not be entirely reliable, given that users may choose to disable JavaScript. On the other hand, server-side validation is somewhat slower that client-side validation and relies on a program located on the server. While server-side validation may be slower, it is very reliable. The user can't disable the server-side validation script [3]. So developer must use server-side validation to validate the data given in the form by the users.
2. Client-side
   1. Fote, 2013
      1. The benefits of client-side processing in an ASP.NET web application are programming languages like C# and VB.NET along with the .NET Framework. Languages like C# and VB.NET sit on top of the .NET framework and have all the benefits of object oriented architectures like inheritance, implementing interfaces and polymorphism.
      2. In contrast to server-side code, client-side scripts are embedded on the client’s web page and processed on the client’s internet browser. Client-side scripts are written in some type of scripting language like JavaScript and interact directly with the page’s HTML elements like text boxes, buttons, list-boxes and tables. HTML and CSS (cascading style sheets) are also used in the client. In order for client-side code to work, the client’s internet browser must support these languages.
      3. There are many advantages to client-side scripting including faster response times, a more interactive application, and less overhead on the web server. Client-side code is ideal for when the page elements need to be changed without the need to contact the database. A good example would be to dynamically show and hide elements based on user inputs. One of the most common examples is input validation and Microsoft’s Visual Studio includes a set of client-side validation controls.
      4. However, disadvantages of client-side scripting are that scripting languages require more time and effort, while the client’s browser must support that scripting language.
   2. (Don’t site, just reword)
      1. Client side programming has to do with the user interface, with which the user interacts.
      2. It’s main tasks are: interacting with the server - sending requests and retrieving data from it, validating input, animation, manipulating UI elements, applying styles, etc. An example of client-side programming is Javascript. Javascript can be used to run checks on form values and send alerts to the user’s browser.
   3. Jaiswal, A. (2014)
      1. Similarly to server-side programming, client-side programming is also the name of the entire program that runs on the client.
         1. Or we can say that client-side programming mostly deals with the user interface with which the user interacts in the web. It is mostly a browser, in the user's machine, that runs the code and is mainly done in any scripting language like JavaScript (or we can use Flash instead of JavaScript or VNScript).
      2. Client-side/Uses
         1. Makes interactive web pages
         2. Make stuffs work dynamically
         3. Interact with temporary storage
         4. Works as an interface between user and server
         5. Sends requests to the server
         6. Retrieval of data from Server
         7. Interact with local storage
         8. Provides remote access for client server program
      3. Languages
         1. JavaScript
         2. VBScript
         3. HTML (Structure)
         4. CSS (Designing)
         5. AJAX
         6. jQuery etc.
3. Server-side
   1. Fote, 2013
      1. There are several server-side technologies that can be used when developing web applications. The most popular is Microsoft’s ASP.NET. In ASP.NET, server-side code uses the .NET Framework and is written in languages like C# and VB.NET. Server-side processing is used to interact with permanent storage like databases or files. The server will also render pages to the client and process user input. Server-side processing happens when a page is first requested and when pages are posted back to the server. Examples of server-side processing are user validation, saving and retrieving data, and navigating to other pages.
      2. The disadvantage of server-side processing is the page postback: it can introduce processing overhead that can decrease performance and force the user to wait for the page to be processed and recreated. Once the page is posted back to the server, the client must wait for the server to process the request and send the page back to the client.
      3. The general rule is to use server-side processing and page postbacks when the client needs to interact with server-side objects like databases, files, etc.
   2. (Don’t site, just reword)
      1. Server-side programming is the general name for programs which run on a Server. Server side programming is about generating dynamic content. Most of the web pages are not static, they search a database in order to show the user personalized information.
      2. Server-side programming can be done in a lot of languages: PHP, ASP, Java and Jsp, Phyton, etc.
      3. This code has to do with: Querying the database, encoding the data into html, inserting and updating information in the database.
   3. Net-informations.com, n.d
      1. In the Server Side Validation, the input submitted by the user is being sent to the server and validated using one of server side scripting languages such as ASP.Net, PHP etc. After the validation process on the Server Side, the feedback is sent back to the client by a new dynamically generated web page.
   4. Jaiswal, A. (2014)
      1. Or we can say that server-side programming must deal with dynamic content. It runs on the server. Most web pages are not static since they deal with searching databases.
      2. Uses
         1. It processes the user input
         2. Displays the requested pages
         3. Structure web applications
         4. Interaction with servers/storages
         5. Interaction with databases
         6. Querying the database
         7. Encoding of data into HTML
         8. Operations over databases like delete, update.
      3. Languages:
         1. PHP
         2. ASP.NET (C# OR Visual Basic)
         3. C++
         4. Java and JSP
         5. Python
         6. Ruby on Rails and so on.
4. Recommendations
   1. (Don’t site, just reword)
      1. In terms of Security, server-side scripts are more secure than client-side scripts. For example, when a user accesses his bank account online, the server side-script communicates with the client using encryption. A client-side script consists of plain text and runs on the client’s browse allowing hackers to take a peek at the the code and steal private information from the user’s computer.
      2. In terms of Compatibility, the main issue with client-side scripts are the problems with OS’s and web browsers compatibility. Programming a website involves users having multiple computer software, so bugs in the code or compatibility issues with browsers may occur. Server-side scripts being run on the server reduces the number of bugs and compatibility issues since the code is run on a single server using a single language and hosting software.
   2. (Net-informations.com, n.d.)
      1. It is better to validate user input on Server Side because you can protect against the malicious users, who can easily bypass your Client Side scripting language and submit dangerous input to the server.
      2. In the Client Side Validation you can provide a better user experience by responding quickly at the browser level. When you perform a Client Side Validation, all the user inputs validated in the user's browser itself. Client Side validation does not require a round trip to the server, so the network traffic which will help your server perform better. This type of validation is done on the browser side using script languages such as JavaScript, VBScript or HTML5 attributes.
      3. For example, if the user enter an invalid email format, you can show an error messeage immediately before the user move to the next field, so the user can correct every field before they submit the form.
      4. Mostly the Client Side Validation depends on the JavaScript Language, so if users turn JavaScript off, it can easilly bypass and submit dangerous input to the server . So the Client Side Validation can not protect your application from malicious attacks on your server resources and databases.
      5. As both the validation methods have their own significances, it is recommended that the Server side validation is more SECURE!
   3. Greenspan, 2013)
      1. Client-side rendering means JavaScript running in the browser produces HTML or manipulates the DOM. The benefit is you can update the screen instantly when the user clicks, rather than waiting a few hundred milliseconds at least while the server is contacted to ask what to display. Sites where you mostly navigate and view static content can get away with mostly server-side rendering, but I can’t think of any website that doesn’t do any client-side rendering. Any portion of a page that’s animated or highly interactive (a drag-able slider, a sortable table, a dropdown menu) almost certainly uses client-side rendering.
      2. The reason server-side and client-side rendering aren’t typically mixed is that they are typically done in different programming environments, usually in different languages. For example, almost all websites render the overall page layout and the content, such as blog posts, on the server, generating HTML in a language like Ruby, PHP, or Python. On the other hand, a chat widget or an image gallery might be written entirely in client-side JavaScript, while a slider or a rich text editor certainly would be.
      3. For websites that blend static, navigable content and app-like interactivity (or for companies that seek to provide an added measure of sanity for their engineers and designers), this divide becomes a huge pain. This is why Google Plus does all rendering on the client. Quora cleverly does almost all rendering on the server (including when you, say, click a “Follow” button), but only the updated parts are sent to the client. (Particular widgets like the editor and “Promote” slider that provide instant feedback are clearly client-side.) In an effort to increase responsiveness and flexibility, many sites now render Ruby templates on the server and Backbone templates on the client for different parts of the same user interface. AirBnB has experimented with running client-side JavaScript templates on the server.
      4. In the absence of such a solution, programmers usually end up arguing either that client-side rendering isn’t very important (because most sites or apps consist of large “pages” or “screens” and a round-trip to the server is acceptable when navigating between them), or that server-side rendering isn’t very important (because an all-client app just needs a loading screen and a little extra attention to search engine optimization). These arguments are motivated by the pain of having to deal with both types of rendering in the same application when they involve different languages and APIs.
      5. More examples of major sites and their rendering strategies: Facebook FB +0.33% is mostly PHP but certain parts of the UI are pure JavaScript — the chat, the comment form, probably the photo gallery. Gmail is written in something like C++ or Java, and I’m not sure where the rendering takes place, but I’d guess it’s mostly or completely on the client (given the progress bar it displays while it loads). The old or “basic” Gmail view is presumably server-side rendered. Github’s Markdown implementation is written in C and runs on the server. StackOverflow has two Markdown implementations, one for live preview on the client and one for the server.
   4. Stackoverflow
      1. security and validation should always be present at the server side (sometimes duplicated in the client).
      2. the client-side should contain only UI-logic. No business logic.
      3. logically, everything that accesses a database should be on the server.
      4. Javascript should be only used to manipulate the UI of the page. You can also do certain validations using it, however, there must be corresponding validation on the server-side. For doing any data manipulation, applying business logic, etc you should always use server side code.
      5. Server-side
         1. User authentication
         2. Business logic (deciding what to show to which users, calculations)
         3. Database access
5. Conclusion

References:

Fote, B. (2013) ‘Client-Side vs. Server-Side Code: What’s the Difference?’, *Seque* [Online]. Available from: <http://www.seguetech.com/blog/2013/05/01/client-side-server-side-code-difference> (Accessed: 14 June 2014)

Greenspan, D. (2013) ‘Web Development: What Are The Trade-Offs Of Client-Side Rendering Vs. Server-Side Rendering?’, *Forbes* [Online]. Available from: <http://www.forbes.com/sites/quora/2013/07/10/web-development-what-are-the-tradeoffs-of-client-side-rendering-vs-server-side-rendering/> (Accessed: 14 June 2014)

Jaiswal, A. (2014) ‘Client-Side vs Server-Side Programming Languages’, *C# Corner* [Online]. Available from: <http://www.c-sharpcorner.com/UploadFile/2072a9/client-side-vs-server-side-programming-languages/> (Accessed: 14 June 2014)

ICIEV (2012) 'Automated Server-Side Form Validation', *2012 International Conference On Informatics, Electronics & Vision (ICIEV)*, p. 61, Publisher Provided Full Text Searching File, EBSCOhost, (Accessed: 15 June 2014)

Net-informations (n.d.) Client side validation Vs server side validation [Online]. Available from: <http://net-informations.com/faq/asp/validation.htm> (Accessed: 14 June 2014)