**Become a Front End Developer**

**with Altimetrik | 2nd Edition**

**Documentation**

[Vanessa Martínez]

Index

**Agile Software Development Methodologies** .………………………………..2

**Types of Methodologies** …………………………………………………………………..2

**Scrum** .……………………………………………………………………………….……….2

Roles and process ………………………………………………………………………….3

**Kanban** .……………………………………………………………………………………....3

**Extreme Programming** ……………………………………………………………………...4

Rules of extreme programming …………………………………………………………...4

**Git** ………………………………………………………………………………………………4

**More common git commands** ……………………………………………………………5

**Tags and commits** ………………………………………………………………………….7

**Stash commands** ……………………………………………………………………………7

**Hooks** ………………………………………………………………………………………….7

**Git rebase and squash** …………………………………………………………………….8

**Difference between git merge and git rebase** ………………………………………..9

**Javascript** ……………………………………………………………………………………9

***Agile Software Development Methodologies***

*Agile software development, or “Agile”, refers to a group of development methodologies that anticipates the need for flexibility, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams to the delivery of the finished product. It focuses on the clean and fast delivery of individual pieces or parts of the software and not on the entire application.*

*Some benefits:*

*- The ability to help teams in a complex area while it’s still focused on the delivery of business value.*

*- Improves efficiency throughout the organization as teams work together and understand their specific roles in the process.*

*- Companies can trust on who are using agile methodologies because they can feel they are releasing a high-quality product since testing is performed throughout development.*

*- Provide the opportunity to make changes as needed and alert teams to any potential issues*

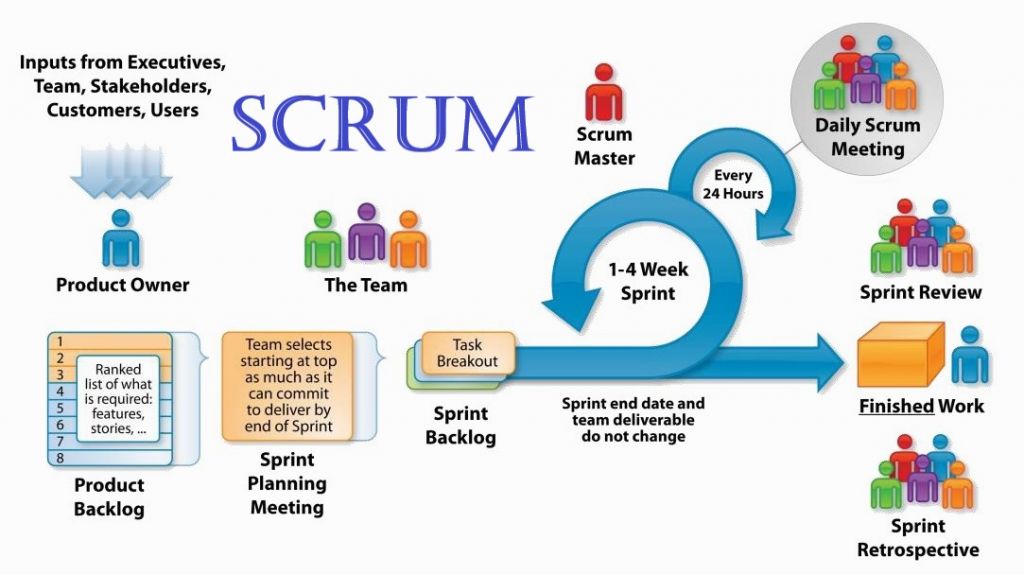
*´.*

***Types of Agile Methodologies***

***Scrum***

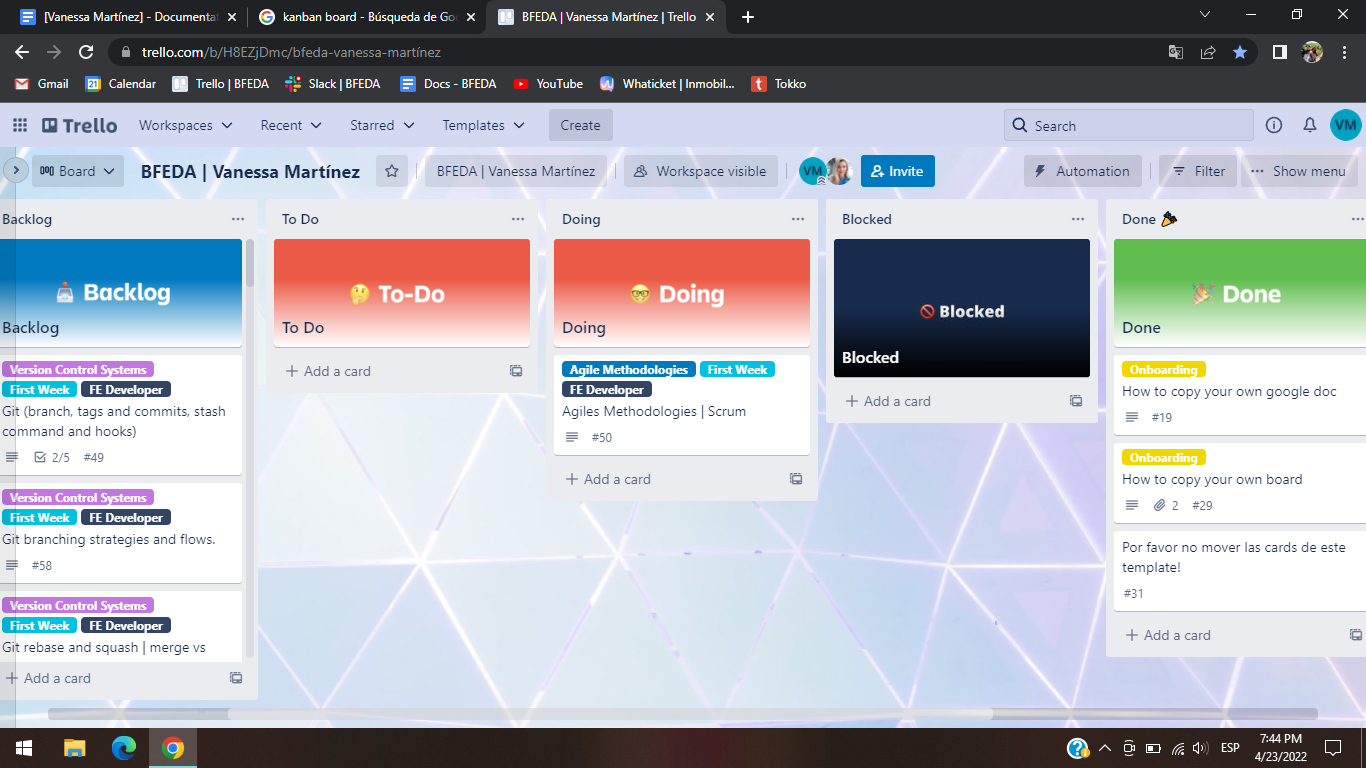
*Scrum is a methodology that helps people, teams and organizations to self-organize, and applies a set of good practices to work together and obtain the best results, generating value by solving complex problems. It replaces a programmed algorithmic approach with a heuristic one (a set of techniques to solve a problem).*

***Roles and process:***

******

***Kanban***

*Kanban is a form to organize tasks with cards on a board that has no limit. You can have a few columns to pass, beginning on the ‘backlog’ column, constantly updated, once the task has begun, you can move the cards to another column called ‘in progress’, or similar, and then through the correspondent columns until it is ‘done’.*

**

***“Scrum is a technique and Kanban is a specific form to apply it”***

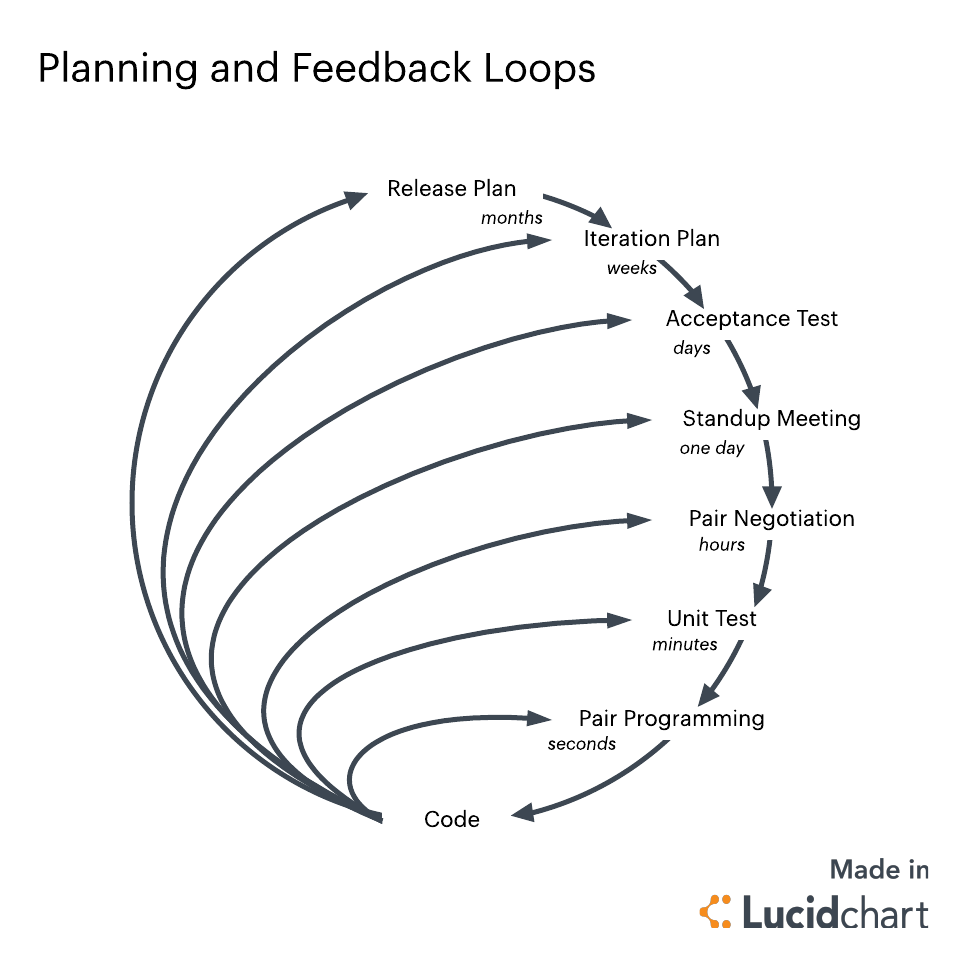
***Extreme Programming (XP)***

*Extreme programming is a framework that aims to produce higher quality software. XP is the most specific of the agile frameworks regarding appropriate engineering practices.*

*XP has 5 values:*

* *Simplicity*
* *Communication*
* *Feedback*
* *Courage*
* *Respect*

***Rules of XP:***

**

***Git***

*Git is a function that controls code’s versions in distributive form that is an important part of daily programming. It allows you to have a complete version history in a fast and free way, this system works with branches and these branches can grow up in different directions from the main one to try out new functionalities.*

***More common git commands***

* ***Git clone:*** *It’s a command that makes an identical copy of the latest version of a project in your local workspace.*

*git clone <https://name-of-the-repository-link>*

* ***Git branch:*** *It’s used to create, list and delete branches.*

*To create one locally, you use: git branch <branch-name>*

*To push it into the remote repository, you use: git push -u <remote> <branch-name>*

*To list or view it, you use: git branch or git branch --list*

*To delete it, you use: git branch -d <branch-name>*

* ***Git Checkout:*** *It’s mostly used for switching from one branch to another and for checking out files and commits.*

*git checkout <name-of-your-branch>*

*You can also use another command to create and switch a branch at the same time:*

*git checkout -b <name-of-your-branch>*

* ***Git Status:*** *It gives you all the information about branches and files, gathering information like the if the preset branch is up to date, if there is anything to commit, pull or push, if there are files created, modified or deleted.*

*git status*

* ***Git Add:*** *If we create, make a change or erase files, those changes will be on our workspace and won’t be in the next commit. To save it we have to use this command to include those changes into the next commit.*

*To add a file: git add <file>*

*To add everything at once: git add -A*

* ***Git Commit:*** *It’s a command used to save changes we have done after a specific task or issue on our local workspace. We need to write a short message to explain what we did in the source code.*

*git commit -m "commit message"*

* ***Git Push:*** *It sends your changes to the remote server which already are committed.*

*git push <remote> <branch-name>*

*If your branch is new and you need to upload it too, you can use: git push --set-upstream <remote> <name-of-your-branch> or git push -u origin <branch\_name>*

* ***Git Pull:*** *It’s used to get updates from the remote repository and applies the latest changes in your local.*

*git pull <remote>*

* ***Git Revert:*** *It’s used to undo changes in a safe way.*

*First we have to see our commit history, using: git log -- oneline*

*Then specify the hash code of the commit that we want to undo: git revert 3321844*

*Finish pressing shift + q to exit.*

*Really, this command doesn't delete the commit from the history, just create a new “revert” one, you can still see every single move that you did.*

* ***Git Merge:*** *when everything is working ok, it’s used to integrate your feature branch with all of its commits back to the dev branch.*

*Before merging, you have to switch to the dev branch: git checkout dev*

*Then update it using: git fetch*

*At the end, merge your feature branch into dev: git merge <branch-name>*

***Tags and Commits***

*Tags is a feature used to specify a point on a repository. It’s mostly used to keep the release version of a repo, using this tag you are able to move it to an earlier version.*

*git status*

*git add <file>*

*git commit -m “short message”*

*git tag v-1.0 <the correspondent version>*

*git tag -n*

*A* ***tag*** *can be created to a specific* ***commit*** *too, to create it, follow the commands bellow:*

*git status*

*git add <file>*

*git commit -m “short message”*

*git log -- oneline*

***Stash command***

*We can use* ***git stash*** *when we want to temporarily record our current state of the working directory, saving our local modifications away, to manage this copy on another side, here we can make the changes needed without damaging the main and feeling overwhelmed, when everything is fine, you can put it back on the original one.*

***Hooks***

*They are scripts that run every time a certain important action occurs in a repo. Exist two groups of these hooks:*

* ***Client-side hooks:*** *are activated by operations such as committing and merging.*
* ***Server-side hooks:*** *run on network operations such as receiving pushed commits.*

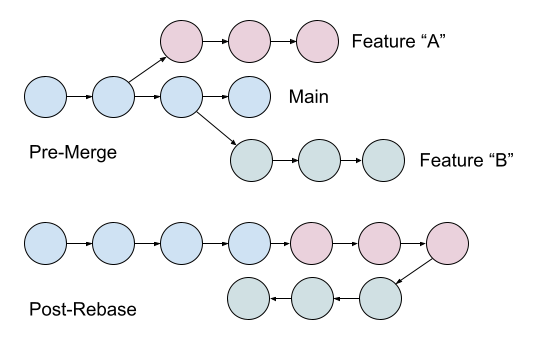
*The most useful local hooks:*

* *pre-commit*
* *prepare-commit-msg*
* *commit-msg*
* *post-commit*
* *post-checkout*
* *pre-rebase*

***Git rebase and squash***

***Rebase:*** *When we have 2 different features branches to include in the main one, mostly you will get conflict errors. If a feature branch doesn’t consider the other one, bugs cloud appear in a feature branch.*

*Using git rebase, it will change the base of a branch, not create another original point, just putting that on a different point in time.*

**

*To get it, use something like this:  
git checkout featureA*

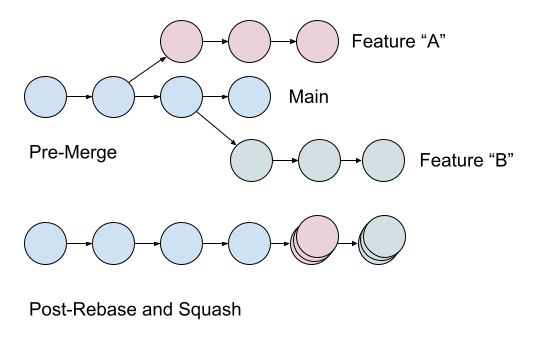
*git fetch origin*

*git rebase origin/main Rebase the latest "main" to "Feature A"*

*git checkout main*

*git merge featureA*

***Squash:*** *It’s used to merge all your commits of one feature branch into a single commit, and then you can put it at the end of the main branch.*

**

*To get it, first you have to rebase and then add:  
git merge featureA --squash Squash "Feature A" to end of main*

***Difference between git merge - git rebase***

* *Rebase reapplies commits of the top of another base branch (rewrite it).*
* *Merge joins two or more development histories together (preserved history as it happened).*

***Javascript - Part 1***

***Javascript*** *is a dynamic programming language used for web development.*

*With Javascript you will have more versatility to create style declarations, supporting math calculations, extracts contents from another website, and more.*

*It has more features we can explain, but by understanding the basic constructs you can begin to use it, and learn more and more while you are working with it.*

***Javascript syntax***

*Refers to the set of rules that determines how JavaScript programs are constructed: variables, operator, expression, keyword, comments, data types, functions, etc.*

***Console in Javascript***

*It shows the informacion of the web you are playing at the moment. It’s an object used to access the web console of the browser.*

*The console.log ( ) function is used to print any kind of variables defined before in it or to just print any message that needs to be displayed to the user.*

*The console.table ( ) function is used to display data in tabular form, taking data (can be an array or an object) and an additional parameter.*

***Data types in Javascript***

1. ***Number:*** *Numerical value.*
2. ***String:*** *Sequences of characters that represents a value.*
3. ***Boolean:*** *Logical values, true or false.*
4. ***Null:*** *Denote a null value. (Remembering JS is case-sensitive, null isn’t the same than NULL).*
5. ***Undefined:*** *Represents non-existence of data.*
6. ***Symbol:*** *Unique values created from string keys.*
7. ***Object:*** *it is a related data collection, consists in variables and functions, when these are inside objects, they are known as properties and methods.*
8. ***Array:*** *It is a set of related data by positions associated by one variable.*

***Variables in Javascript***

*Variables are containers for storing data values. They are identificators, inasmuch as JS differences uppercase and lowercase, you have a lot of characters to use, from ‘A’ to ‘Z’ and from ‘a’ to ‘z’, underscore and symbols to begin a variable, it can be followed by numbers.*

***3 forms to declare a variable in Javascript***

| *var* | *let* | *const* |
| --- | --- | --- |
| * *It’s used in all JavaScript code from 1995 to 2015.* * *It creates a function scoped variable.* * *Can be reassigned.* * *Can be redeclared.* * *It’s hoisted to the top of its executions (global or function) and initialized as undefined.* | * *Added to JavaScript in 2015, in ES6.* * *It creates a block - scope variable.* * *Can be reassigned.* * *Cannot be redeclared.* * *It’s hoisted to the top of its executions (global or block) and left uninitialized.* | * *Added to JavaScript in 2015, in ES6.* * *It creates a block - scope variable.* * *Cannot be reassigned.* * *Cannot be redeclared.* * *It’s hoisted to the top of its executions (global or block) and left uninitialized.* |

***Scope of a variable***

* ***Global Variable:*** *it’s when you declare a variable outside a function and it will be available to others codes.*
* ***Local Variable:*** *it’s when you declare a variable inside a function and it will be just available inside this function.*

***Comments in Javascript***

*Comments are notes in the source code of a program that can be used to write important things about the code, explaining it and making the code more readable, and also be used to prevent execution.*

***Functions in Javascript***

*Functions are the block of code designed, used to do particular tasks, they are used when something calls it, it can call functions several times, so we can say functions are reusables. To write it you can implement the same rules of variables.*

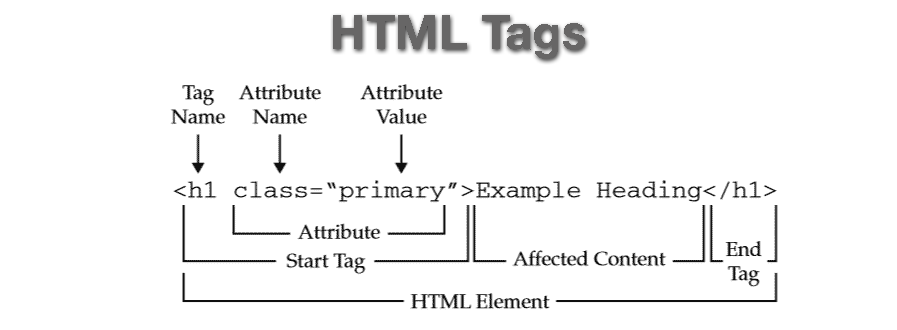
***HTML***

*Hyper Text Markup Language, as its name says, is a markup language used for creating web pages and applications, describing the structure of these last one, with the help of styling.*

***HTML Elements***

*It’s an individual component of an HTML file. In an HTML file, everything written within tags are termed as HTML elements. It’s defined by a start tag, some content, and an end tag*

*<tagname> words </tagname>*

**

*Example:*

*<!DOCTYPE html>*

*<html>*

*<head>*

*<title> write the title </title>*

*</head>*

*<body>*

*<h2> something like a subtitle</h2>*

*<p>This is a paragraph tag</p>*

*<p style="color: purple">The style is attribute of paragraph tag</p>*

*</body>*

*</html>*

***HTML tags***

*HTML Tags are always written in lowercase letters, HTML5 includes 142 tags. Exists different categories, here are some of them:*

***Basic tags***

*<!DOCTYPE> : defines the document type*

*<html> : defines an HTML document*

*<head> : contains metadata/information for the document*

*<title> : defines a title for the document*

*<body> : defines the document's body*

*<h1> to <h6> : defines HTML headings*

*<p> : defines a paragraph*

*<br> : inserts a single line break*

*<hr> : defines a thematic change in the content*

*<!--...--> : defines a comment*

***Unclosed tags***

* *<br> : br stands for break line, it breaks the line of the code.*
* *<hr> : hr stands for Horizontal Rule to put a line across the page.*

***Meta - tags***

* *<!DOCTYPE> : defines the document type.*
* *<title> : defines a title for the document.*
* *<link> : defines the relationship between a document and an external resource (most used to link to style sheets)*
* *<style> : defines style information for a document.*
* *<meta> : defines metadata about an HTML document.*

***Text tags***

*<p>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, <strong>, <em>, <abbr>, <acronym>, <address>, <bdo>, <blockquote>, <cite>, <q>, <code>, <ins>, <del>, <dfn>, <kbd>, <pre>, <samp>, <var>, <br>*

***Link tags***

*<a>, <base>*

***Image and object Tags***

*<img>, <area>, <map>, <param>, <object>*

***List tags***

*<ul>, <ol>, <li>, <dl>, <dt>, <dd>*

***Table tags***

*table, tr, td, th, tbody, thead, tfoot, col, colgroup, caption*

***Form tags***

*form, input, textarea, select, option, optgroup, button, label, fieldset, legend*

***Scripting tags***

*script, noscript*

***Input types***

*Those are an important part of an element that defines the information field.*

* *text : defines a one-line text input field*
* *password : defines a one-line password input field*
* *submit : defines a submit button to submit the form to server*
* *reset : defines a reset button to reset all values in the form*
* *radio : defines a radio button which allows selecting one option*
* *checkbox : defines checkboxes which allow selecting multiple options form*
* *button : defines a simple push button, which can be programmed to perform a task on an event*
* *file : defines to select the file from device storage*
* *image : defines a graphical submit button*

*And in HTML5 we also have:*

* *color : defines an input field with a specific color.*
* *date : defines an input field for selection of date.*
* *datetime-local : defines an input field for entering a date without a time zone.*
* *email : defines an input field for entering an email address.*
* *month : defines a control with month and year, without a time zone.*
* *number : defines an input field to enter a number.*
* *url : defines a field for entering URL*
* *week : defines a field to enter the date with week-year, without a time zone.*
* *search : defines a single line text field for entering a search string.*
* *tel : defines an input field for entering the telephone number.*

***Difference between HTML and XHTML***

* *XHTML is a stricter version of HTML*
* *XHTML is support by all major browsers*
* *<!DOCTYPE>, <html>, <head>, <title>, and <body> are mandatory*
* *Elements must always be properly nested, closed and in lowercase, tags should appear in pairs.*
* *Attribute names must always be in lowercase*
* *Attribute values must always be quoted*
* *Attribute minimization is forbidden*

***HTML attributes***

*In HTML attributes are special words that give additional information about the element or modify it, it can define the behavior of the element, always apply it with the start tag and its name and value pair. The attributes name and value are case - sensitive and you can write 2 or more attributes in the same element giving space between each one.*

*<element attribute\_name="value">content</element>*

***Global attributes***

*Exists different types and categories of attributes, here are some the global one:*

* *Accesskey : specifies a shortcut key to activate/focus an element*
* *class : specifies one or more classnames for an element (refers to a class in a style sheet)*
* *contenteditable : specifies whether the content of an element is editable or not*
* *data-\* : used to store custom data private to the page or application*
* *dir : specifies the text direction for the content in an element*
* *draggable : specifies whether an element is draggable or not*
* *hidden : specifies that an element is not yet, or is no longer, relevant*
* *id : specifies a unique id for an element*
* *lang : specifies the language of the element's content*
* *spellcheck : Specifies whether the element is to have its spelling and grammar checked or not*
* *style : specifies an inline CSS style for an element*
* *tabindex : specifies the tabbing order of an element*
* *title : specifies extra information about an element*
* *translate : specifies whether the content of an element should be translated or not*

***Accessibility***

*Designing and creating web pages, it’s always important to have accessibility in mind, giving a nice experience to all audiences to navigate and interact with your site, and to make your code semantic. The goal is to make your content accessible to visually-impaired or blind users.*

***Semantic HTML***

*Basically it refers to the correct use of HTML elements for their real purpose, semantic HTML gives context to screen readers making the element focusable.*

*Examples of* ***non-semantic*** *elements: <div> and <span> - Tells nothing about its content.*

*Examples of* ***semantic*** *elements: <form>, <table>, and <article> - Clearly defines its content.*

***SEO - DOM***

***CSS***

***Specificity***

***Box Model***

***Javascript - Part 2***

*.*

***Hoisting in Javascript***

*It’s the default behavior of moving all the declarations at the top, no matter where functions and variables are declared, their scope regardless of whether their scope is global or local. JavaScript only hoists declarations, not the initializations.*

***DOM in Javascript***

***Scope in Javascript***

*It’s an important concept that manage the visibility of variables, exists 3 types:*

* ***Block scope:*** *defines a scope for variables declared using let and const.*
* ***Function scope:*** *defines a scope for variables declared using var, let and const.*
* ***Global scope:*** *The global scope is the outermost scope, it is accessible from any inner scope. Variables declared Globally have Global Scope.*

***Strict Mode in Javascript***

*Strict mode is a feature added in ES5, it was created to reduce the random behavior and increment the detectability of poorly written code. These made the code much more secure and maintained a high standard of coding in general. Using the strict-mode it was seen that developers could now write highly optimized programs.*

*use strict*

**Bibliography**

* <https://www.techtarget.com/searchsoftwarequality/definition/agile-software-development>
* <https://www.cprime.com/resources/what-is-agile-what-is-scrum/#:~:text=Agile%20software%20development%20refers%20to%20a%20group%20of%20software%20development,%2Dorganizing%20cross%2Dfunctional%20teams>.
* <https://www.scrum.org/resources/what-is-scrum>
* <https://lucidspark.com/blog/different-types-of-scrum-meetings>
* <https://www.agilealliance.org/glossary/xp/#q=~>(infinite~false~filters~(postType~(~'post~'aa\_book~'aa\_event\_session~'aa\_experience\_report~'aa\_glossary~'aa\_research\_paper~'aa\_video)~tags~(~'xp))~searchTerm~'~sort~false~sortDirection~'asc~page~1)
* <https://www.lucidchart.com/blog/what-is-extreme-programming>
* <https://www.freecodecamp.org/news/10-important-git-commands-that-every-developer-should-know/>
* <https://openwebinars.net/blog/que-es-git-y-para-que-sirve/>
* <https://linuxhint.com/add-tag-specific-commit-git/>
* <https://git-scm.com/docs/git-stash>
* <https://www.perforce.com/blog/vcs/git-beyond-basics-stashing-work-progress#:~:text=Using%20the%20git%20stash%20command,more%20easily%20switch%20between%20branches>.
* <https://www.atlassian.com/git/tutorials/git-hooks#:~:text=Git%20hooks%20are%20scripts%20that,in%20the%20development%20life%20cycle>.
* <https://medium.com/swlh/squash-and-rebase-git-basics-5cb1be1e0dac#:~:text=With%20%E2%80%9Csquash%E2%80%9D%2C%20you%20can,they're%20now%20just%201>.
* <https://betterprogramming.pub/differences-between-git-merge-and-rebase-and-why-you-should-care-ae41d96237b6#:~:text=Reading%20the%20official%20Git%20manual,it%20happened%2C%20rebase%20rewrites%20it%20>.
* <https://www.freecodecamp.org/news/what-is-javascript-definition-of-js/>
* <https://www.geeksforgeeks.org/javascript-syntax/>
* <https://drive.google.com/file/d/11Qd_2a9YfHq7Yt4IGLXwWRs6OFpSu-6o/view>
* <https://www.w3schools.com/js/js_syntax.asp>
* <https://www.w3schools.com/js/js_comments.asp>
* <https://www.w3schools.com/js/js_variables.asp>
* <https://www.w3schools.com/js/js_functions.asp>
* <https://www.w3schools.com/js/js_arrays.asp>
* <https://www.w3schools.com/js/js_functions.asp>
* <https://www.javatpoint.com/what-is-html>
* <https://www.w3schools.com/html/html_elements.asp>
* <https://dmitripavlutin.com/javascript-scope/#2-block-scope>
* <https://www.w3schools.com/js/js_htmldom.asp#:~:text=The%20DOM%20defines%20a%20standard,and%20style%20of%20a%20document.%22>
* <https://www.w3schools.com/tags/ref_byfunc.asp>
* <https://www.javatpoint.com/html-tags>
* <https://www.w3schools.com/html/html_form_input_types.asp>
* <https://www.javatpoint.com/html-form-input-types>
* <https://www.codecademy.com/article/accessibility-and-html>