Bootstrap:

Bootstrap is a front-end framework for rapid development of Web applications and Web sites. Bootstrap is based on HTML, CSS, JAVASCRIPT's.

## Why Bootstrap?

* **Mobile First:** since Bootstrap 3 onwards, the framework includes libraries throughout the mobile device priority of style.
* **Browser support:** All major browsers support Bootstrap.
* **Easy to use:** as long as you have a basic knowledge of HTML and CSS, you can begin to learn Bootstrap.
* **Responsive design:** Bootstrap Responsive CSS capable of adaptive desktops, tablets and mobile phones.

**For more information about:**

* It developers create simple interface provides a unified solution.
* It includes a powerful built in components, easy to customize.
* It also provides Web-based customization.
* It is open source.

**Bootstrap package contents**

* **The basic structure:** Bootstrap provides the basic structure of the link style background with a grid system. This will be explained in detail Bootstrap basic moiety.
* **CSS:** Bootstrap comes with the following features:

1. Global CSS settings,
2. Define the basic HTML element styles,
3. Scalable class,
4. An advanced grid system. This section will explain in detail the Bootstrap CSS.

* **Components:** Bootstrap contains a dozen reusable components for creating images, drop-down menus, navigation, alert box popup box, and so on.
* **JavaScript widget:** Bootstrap contains a dozen custom jQuery plugin. You can include all plugins, these plugins can also contain individually.
* **Customization:** You can customize Bootstrap components, LESS variables and jQuery plugin to get your own version.

Bootstrap HTML coding Standard

## grammar

1. With two spaces instead of tabs (tab) - This is the only way to ensure a consistent approach to show in all environments.
2. Nested elements should be indented once (i.e. two spaces).
3. For the definition of property, make sure that all use double quotes, never use single quotes.
4. Do not self-closing (self-closing) element tail add a slash - clearly stated that this is optional.
5. Do not omit optional end tag (closing tag) (for example, </li> or </body>).

## HTML5 Document type

For each of the first line of HTML pages Add a standard mode (standard mode) statement, this can ensure that you have a consistent display in each browser.

**Structure:**

**<!DOCTYPE html>**

**<html>**

**<head>**

**</head>**

**</html>**

## Language property

According to the HTML5 specification:

It is strongly recommended to specify the language attribute html root element, so as to set the correct language for the document. This will help to speech synthesis tools should be used to determine their pronunciation, translation to

## IE compatibility mode

IE support through specific <meta> tag to determine the current version of IE rendering pages should be used. Unless there is a strong special needs, otherwise it is best to set the **edge mode,** thereby notifying the IE using the latest models it supports.

## Character Encoding

By explicitly declare the character encoding, the browser can be secured quickly and easily determine page content rendering. The advantage of this is to avoid the use of character entities mark (character entity) in HTML, which is consistent with the entire document encoding (generally use UTF-8 encoding).

<head>

<meta charset="UTF-8">

</head>

## Attribute Order

HTML attribute should be in order of priority in the order given below, to ensure code readability.

* class
* id , name
* data-\*
* src , for , type , href
* title , alt
* aria-\* , role

## Boolean type attribute

Boolean attribute may not be assigned at the time of declaration. XHTML specifications for its assignment, but the HTML5 specification is not required.

Boolean attribute element if the value is true, if no value is false.

If the attribute exists, its value must be the empty string or [...] attribute canonical name, and do not add whitespace at the beginning and end.

Format:

<input type="text" disabled>

<input type="checkbox" value="1" checked>

<select>

<option value="1" selected>1</option>

</select>

## JavaScript generated label

Tags generated by JavaScript so that content becomes difficult to find, edit, and reduce performance.

**Grammar**

* All declaration statement ends with a semicolon.
* We use two spaces instead of tabs (tab) – as This is the only way to ensure a consistent approach to show in all environments.
* When the selector is grouped separately on a line selector.
* For code readability, in front of the block each statement is braced to add a space.
* Braced block statement alone makes the trip.
* Each statement in the statement after inserts a space.
* In order to obtain more accurate error reporting, each statements are on separate line.
* For property values separated by commas, each comma is inserted into a space.
* We didn’t use rgb(), rgba(), hsl(), hsla() or rect() behind the *internal* values comma insert a space.
* For the color attribute value or parameter it’s omitted less than 1 in front of the decimal 0.
* Hexadecimal values are all lowercase, for example, #fff. When scanning a document, lowercase characters easy to distinguish, because their form is more easily distinguishable.
* Used a short form of the hexadecimal value, for example, with #fff instead of #ffffff .
* Added double quotes for the selection of a property, for example, input[type="text"] .
* 0 units are specified value to avoid, for example, with a margin: 0; instead of margin: 0px;.

**Declaration order**

Property-related statements are grouped and arranged in the following order:

1. Positioning
2. Box model
3. Typographic
4. Visual

As the positioning can remove the element from the normal document flow, and also cover the box model related to style, so in the first row. The box model in second place, because it determines the size and position of components. Other properties affect only the internalcomponents or does not affect the first two groups attribute, so at the back.

## Don’t use

And <link> compared tag, @import command much slower, not only increases the number of additional requests, but also lead to unexpected problems. There are several alternatives:

* Using multiple <link> element
* By Sass or Less like CSS preprocessor multiple CSS files are compiled into one file
* By Rails, Jekyll or other system provided the CSS file merge feature

## With attributes prefix

When using vendor-specific attributes prefixed by indentation, which allows the value of each property are aligned in the vertical direction, it is easy to multi-line editing.

## Single-line rule declaration

## For styles that **contain only a statement**,**** in order to facilitate legibility and quick editing, we recommend that the statement on the same line. For a number of styles with a declaration or statement should be divided into multiple lines.

## Shorthand property declarations

## In the need to explicitly set the value of all cases, we should try to limit the use of shorthand property declarations. Situation commonly abused shorthand property declared as follows:

1. padding
2. margin
3. font
4. background
5. border
6. border-radius

In most cases, we do not need to specify all values ​​for the attribute declaration abbreviated form. For example, HTML heading elements only on the set top and bottom margins value, therefore, when necessary, just covering these two values ​​can be. Excessive use the short form of the property declaration can lead to confusing code, property value and would bring unnecessary overlap causing unexpected side effects.

## Less and Sass nested

Avoid unnecessary nesting. This is because although you can use the nest, but that does not mean you should use nested.

## Note

## The code is written and maintained by the people. Please make sure that your code can be self-describing, well-commented and easy to understand others. Good code comments can convey context and purpose of the code. Do not simply reiterate component or class name.

## class name

* class names can appear only lowercase characters and dashes (dashe) (not underlined, nor hump nomenclature). Dash should be used for the relevant class named (like namespaces) (for example, .btn and .btn-danger ).
* Avoid excessive arbitrary shorthand. .btn representatives *button,* but .s not express any meaning.
* class name should be as short and clear meaning.
* Use meaningful names. Organized or purposeful use of the name, do not use expressions (presentational) name.
* Based on the nearest parent class or basic (base) class as a prefix for the new class.
* Use .js-\* class to identify the behavior (as opposed to style), and do not contain these class to the CSS file.

## Selector

* For common elements use class, so conducive to optimize rendering performance.
* For components often avoid using the property selectors. Browser performance will be affected by these factors.
* Selectors as short as possible, and try to limit the number of elements of the selector, it is recommended not to exceed 3.
* Only when it is necessary to limit in the last class of the parent element (ie descendant selector) (For example, do not use the class with the prefix - namespace prefix is similar).

## Code Organization

* Component-unit organization code.
* Development of consistent annotation specification.
* Use consistent whitespace separated into blocks of code, so conducive to scan large documents.
* If more than one CSS file, in the form of spin-off assembly rather than the page, because the page will be reorganized, and the assembly will be moved.