# CSS Best Practices

1. Make it readable: we use multiple lines, use plugins to order the properties e.g. csscomb to reorder class properties
2. Agree on naming convention, dash vs. camelCase. We will be sticking with dash naming. e.g. main-content

No names should ever specify properties within the class, all names should be semantically driven and abstract as possible. For example, avoid using:

.border4px this will be hard to maintain down the road.

Instead use

.bd-med

1. Organize the stylesheet with a Top-down structure

Sections of the stylesheet should all be present in the right order as the the page seen.

1. Generic Class (body, a, p, h1, etc...)
2. #header
3. #nav-menu
4. #main-content
5. #component

This should all be clarified and made readable by using comments:

/\* ================= #header ================== \*/

/\*your styles goes here\*/

/\* ================= #main-container ================== \*/

/\*your styles goes here\*/

/\* ================= #footer ================== \*/

/\*your styles goes here\*/

1. Combine Elements together

h1, h2, h3 {

font-family: tahoma;

color: #333;

}

1. Use Multiple classes and more classes than ids

Avoid using ids as much as you can, since the entire application is componentized and those elements can be used all across and they may all be used within the same page.

1. Use shorthand for writing properties
2. Stick to and study well the bootstrap documentation and guidelines in using your styles, use already styles defined in bootstrap before creating your own.
3. Validate your CSS on the go, missing commas and small errors could cause the entire CSS to break.
4. As we approach a responsive design, we are sticking to the Ems as opposed to Pixels.
5. Avoid using contextual styling; i.e. descendant selectors as the entire application is ‘componentized’ and modularized as those components should be able to live sparingly and faultlessly within any container not depending on where they live.
6. Don’t use inline styling
7. Use CSSLint to validate your code before integrating it into the main css file.
8. Do not create widgets unless you revise our widget board/ functional toolkit and check that the new widget required is not present in our widget board, before creating that new widget, an approval for that is required.   
   The widget board is present at: <http://vc1cgr01cgsoa87/ux/>
9. Never touch libraries. Do NOT edit / change in any of the library CSS files e.g. bootstrap css files or infragestics css files or any other library files. Any additions of customizations you should make, should all go within our customized CSS file (in this case it is dsis-theme.css) file.