Terminology

```
selector {
    property: value;
}
```

The whole thing above is called "a style" or "a style definition". "Style name" may be used as a synonymous for "selector".
"Property-value pair" needs a better name.

Naming selectors

- The most important thing to have in mind is the content nature of the HTML document, not its presentation. Selector names should describe the content.
- Avoid presentation-specific words in the name, like "blue", "text-gray", or "light-box". What happens when you decide to change the color? The class "blue" is actually red?
- Use full descriptive words. Abbreviating a word may save you a few milliseconds to type initially, but may make your code harder to read, costing you more time down the road. Why crypting "product" to "prod" or the so common "txt" (just spell it out as it is, "text")? Being consistent in this will save you the time spent thinking (or trial-error-ing) how exactly you abbreviated a word five days ago. (OK, OK, there are exceptions, you can use "url" instead of "uniform-resource-locator")
- Names are lowercase. There are browser problems with case sensitivity, so you're safe always lowercasing.
- Once again - use names that describe the content ("footer", "navigation", etc), rather than the presentation ("blue", "left", "big"...)

Case

Start the class names with lowercase letters and start each embedded word in the name with a capital. HTML elements are in lower case in anticipation of XHTML.

- Home
- Blog
- CV
- About
- Contact

Ordering of declarations

One of most common bugbears is with the ordering of declarations, a lot of the CSS .The best way to order declarations is alphabetically, it makes it far easier to scan through and visually locate a specific declaration, especially in large rules

```
ul#nav_primary li a {
border:0;
color:#999;
display:block;
float:left;
font-size:1.2em;
font-weight:bold;
padding:0 10px;
text-decoration:none;
text-transform:uppercase; }
```

Spacing, Punctuation and Brackets

Generally spacing promotes readability, so let's take advantage of it.

Empty lines between style definitions

Allow the separate style definitions to breathe, by adding one (and only one) empty line between them.

1 tab = 4 spaces

Indentation also helps with the readability of the code. So put every property-value pair on a new line and indent it with 4 spaces from the beginning of the line. You can use a tab for indentation, it doesn't matter, this is just a personal preference.

Space after colons

Visually emphasis on the separation between a property and value, by adding a space after the colon, like:

```
.class {
   prop: value;
}
```

and not

```
.class {
```

```
   prop:value;
}
```

or

```
.class {
   prop:      value;
}
```

Curly brackets

Curly brackets embrace all property-value pairs in a style definition. The opening bracket must be on the same line as the selector identifier and the closing bracket on a new line.

Example

```
.header {
   font-size: 2em;
}
.navigationBar,
.content,
.header i {
   font-size: 0.8em;
   border: solid black 1px;
   border-left: solid black 5px;
   border-right: solid black 5px;
}
```

Structure

When several style definitions share some styles and are of the same family, their common styles should be defined in one place.

Example

Common message styles:

```
.normal-message,
.warning-message,
.error-message {
   border-style: solid;
   border-width: 1px;
   padding: 5px;
   position: relative;
   width: 500px;
}
```

Specific message styles:

```
.normal-message {
```

```
   background-color: #E5EDF9;
   border-color: #4171B5;
}
.warning-message {
   background-color: #FFEFA7;
   border-color: #FFB30F;
}
.error-message {
   background-color: #FFD1D1;
   border-color: #FF0000;
}
```

Shorthands

Use shorthand properties to keep all parts of a style type on a single line.

For example:

```
margin: 5px 0 4px 10px;
```

Is much more efficient than:

```
margin-top: 5px;
margin-right: 0;
margin-bottom: 4px;
margin-left: 10px;
```

Combining properties onto a single line using shorthand properties means that your CSS will be easier to understand and edit.

Sections

Clearly divide your style-sheet into specific sections by using a separator like this:

```
/*
=======================================
           Global Styles
=======================================
*/
```