Coding Style Guide Cheat-Sheet

LANG	CSS	HTML	JavaScript	Python	Shell	C / C++	Java
Extension	.css	.html	.js	.py	.sh	.c / .cpp	.java
Encoding	utf-8	utf-8	utf-8	utf-8	utf-8	ASCII	utf-8
Shebang Line & Declaration		<!DOCTYPE html>		#!/usr/bin/env	#!/bin/bash	#include <filename.h>	package xxx.xxx import java.lang.Filename
Comment	/* Comment */	<!-- Comment -->	// Comment /* Comment */ /** JSDoc */	"""Comment""" #TODO(Name):Comment #TODO(Email):Comment	# Comment #TODO(mrmonkey):	// Comment /* Comment */	// Comment /* Comment */ /**JavaDoc*/
Indentation	2 spaces, no tab.	2 spaces, no tab.	2 spaces, no tab.	4 spaces, no tab.	2 spaces, no tab.	2 spaces, no tab.	4 spaces, no tab.
Max Line length (in character)	80	80	80	80	80	80	80
Statements At Most (per line)			1	1	1	1	1
Whitespace	Between last selector and declaration block; after property-name's colon; no trailing space.	At the left of "/>" no space around '='	Follow "(", ";" and "," no space around '='	No space in () [] {} or before() [] or before: , ; or after: , ; or around '='	After the close parenthesis of the pattern and before the ;;	No trailing space.	Info:
Semicolons	After every declaration.		At the end of every statement.	No semicolon at the end of lines.	Put "; do", "; then" on the same line as the while, for, if.	At the end of statements and divide parameters.	At the end of statements and divided parameters.
Variable Name			nameLikeThis	name_likethis	namelikethis	nameLikeThis	nameLikeThis
Constant Name			NAME_LIKE_THIS	NAME_LIKE_THIS	NAME_LIKE_THIS	kNameLikeThis	NAME_LIKE_THIS
Function Name			nameLikeThis	nameLikeThis	my_funname	NameLikeThis	

Function Declarations			function fun(a, b) {
			...
			}
				def function(arg):
				...
					[function] fun [()]
					{
					...
					}
						returntype fun(,) {
						...
						}	

Blank line, trailing space; normalized by 'git stripspace'	1 blank line between rules.	1 blank line to separate large logical chunks of code.	1 blank line between logically related pieces of code in group.	2 blank lines between top-level definitions. 1 blank line between method definitions.	1 blank line between code block.	1 blank line after class declaration. 1 blank line after function definition.	1 blank line between members or interfaces of a class; between local variable and its 1st statement.

Other reminder:

CSS:

ID and Class Naming

·Use meaningful or generic ID and class names. Use ID and class names that are as short as possible but as long as necessary.

·Do not concatenate words and abbreviations in selectors by any characters (including none at all) other than hyphens.

Declaration Order

·In alphabetical order.

·Ignore vendor-specific prefixes. Keep sorted multiple vendor-specific prefixes for a certain CSS property.

Selector and Declaration Separation

·Always start a new line for each selector and declaration.

CSS Quotation Marks

·Use single (') rather than double ("") quotation marks for attribute selectors or property values. Do not use quotation marks in URI values (url()).

Exception: If you do need to use the @charset rule, use double quotation marks—single quotation marks are not permitted.

Embedded in the HTML

·<link rel="stylesheet" href="mystyle.css">

More info: http://google-styleguide.googlecode.com/svn/trunk/htmlcssguide.xml

JavaScript:

var	·Always declare variable with var.
Nested functions	·Feel free to use nested functions
delete	·Prefer "this.foo = null", to "delete this.foo".
this	·Only in object constructors, methods, and in setting up closures.
Function Declarations within Blocks	·No. Instead use a variable initialized with a Function Expression to define a function within a block.
for-in loop	·Only for iterating over keys in an object/map/hash.
Strings	·For consistency single-quotes (') are preferred to double-quotes ("). This is helpful when creating strings that include HTML.
Embedded in the HTML	·<script src="myscripts.js"></script>.

More info: http://google-styleguide.googlecode.com/svn/trunk/javascriptguide.xml

Python:

Imports	·Use imports for packages and modules only. Imports should be on separate lines.
Function and Method Decorators	·Use decorators judiciously when there is a clear advantage.
Global variables	·Avoid global variables
Nested/Local/Inner Classes and Functions	·They are fine.
Files and Sockets	·Explicitly close files and sockets when done with them.

Power Features

·Avoid these features, such as metaclasses, access to bytecode, on-the-fly compilation, dynamic inheritance, object reparenting, import hacks, reflection, modification of system internals, etc.

Strings	·Use the format method or the % operator for formatting strings, even when the parameters are all strings.

More info: http://google-styleguide.googlecode.com/svn/trunk/pyguide.html

Shell:

SUID/SGID	·SUID and SGID are forbidden on shell scripts.
Pipelines	·Pipelines should be split one per line if they don't all fit on one line.
Loops	·Put "; do" and "; then" on the same line as the while, for or if.
Variable expansion	·Stay consistent with what you find; quote your variables; prefer "${var}" over "$var".
Command Substitution	·Use $(command) instead of backticks.
Test, [and [[	·[[...]] is preferred over [, test and /usr/bin/[.
Checking Return Values	·Always check return values and give informative return values.
Main	·A function called main is required for scripts long enough to contain at least one other function.
Eval	·eval should be avoided.

More info: http://google-styleguide.googlecode.com/svn/trunk/shell.xml

C/C++:

Function Parameter Ordering	·When defining a function, parameter order is: inputs, then outputs.
Function Calls	·On one line if it fits; otherwise, wrap arguments at the parenthesis.
Default Arguments	·We do not allow default function parameters, except in limited situations as explained below.
Structs vs. Classes	·Use a struct only for passive objects that carry data; everything else is a class.
Variable and Array Initialization	·You may choose between =, (), and {}.
0 and nullptr/NULL	·Use 0 for integers, 0.0 for reals, nullptr (or NULL) for pointers, and '\0' for chars.
Sizeof	·Prefer sizeof(varname) to sizeof(type).
Lambda expressions	·Do not use lambda expressions, or the related std::function or std::bind utilities.
Preincrement and Predecrement	·Use prefix form (++i) of the increment and decrement operators with iterators and other template objects.
Exceptions	·Do not use C++ exceptions.
Return Values	·Do not needlessly surround the return expression with parentheses.

Loops and Switch Statements

·Switch statements may use braces for blocks. Annotate non-trivial fall-through between cases. Empty loop bodies should use {} or continue.

More info: http://google-styleguide.googlecode.com/svn/trunk/cppguide.xml

Java:

One variable per declaration	·Every variable declaration (field or local) declares only one variable: declarations such as int a, b; are not used.
No C-style array declarations	·The square brackets form a part of the type, not the variable: String[] args, not String args[].
Package statement	·The package statement is not line-wrapped. The column limit does not apply to package statements.
Caught exceptions	·Not ignored
Exactly one top-level class declaration	·Each top-level class resides in a source file of its own.

Static members:

·When a reference to a static class member must be qualified, it is qualified with that class's name, not with a reference or expression of that class's type.

Switch statements

·Terminology Note: Inside the braces of a switch block are one or more statement groups. Each statement group consists of one or more switch labels (either case FOO: or default :), followed by one or more statements.

More info: http://google-styleguide.googlecode.com/svn/trunk/javaguide.html