**Good Programming Practices**

**Writing Clean Code** : **“Programming is an art of telling another human what one wants the computer to do.”-Donald Knuth..**

**Anyone can write a code that computer can understand. An only good programmer can write a code that human can understand.”**

In today’s world, there is a wrong perception that one who knows a number of technologies or latest technologies, is a hero. But believe me, there is no meaning of learning Java, CSS, AngularJS, Perl, Ruby, Python, COBOL if at the end of the day, you are not able to admire your own code. At the end, you become “Jack of all, Master of none”. It’s not just the technology; it’s the code that makes soft powerful.

**Naming conventions :**

**int a = b(x); a,b,x ?**

**now , it is difficult to tell what above expression is doing…..naming should be relevant to the job being performed by code like:**

**int salary = getSalary(Employee);**

* **Class:**  A class name must start with the capital letter then it should follow camel-case i.e. the first letter of every word in the class name would be capital. There is no need of underscore. It should be a noun.
* **Variable:** The rules for the variable name are same as the class except for one change. The variable should start with a small letter. e.g-  
  **Wrong names- Student, Inkjet\_printer, Bookinghistory**  
  **Correct names- student, inkjetPrinter, bookingHistory**
* **Method:** It should be camel-case and for words’ separation, make the first letter of next word capital. It should be a verb. A method should do what its name says.
* Wrong names- void getExpensesHistory(), int setMyAge()  
  Correct names- List getExpensesHistory(), void setMyAge(int age)

**Modularization :** Write reusable code. For example, don’t write salary calculation code everywhere wherever you need the salary of an employee. Just write it in one place, create a method and use it everywhere. You save the line of codes, scale and extension time, creates easy understanding, minimize the chances of errors.

**Keep it simple and small-**(KISS) : In point no. 1, method getSalary should only calculate salary and returns it. It should not do any other thing i.e. calculating a number of paid leaves he took in last one year. You should have a separate method to calculate leaves and call it from getSalary() method if you need it while calculating salary.

**Headers And Revision History:**

**Strongly recommended..**

* Identification of the project of which the program is a part
* Program name
* Author identification which should be human readable and unique
* Date - convention used in different organizations will vary, so be clear as to what this date represents, for example date program initiated, date program passed final validation prior to database lock, etc.
* Revision History

The following 9 rules are important to follow while writing an object oriented code :

1. Use only one level of indentation per method.
2. Don’t use “else” as a keyword! (It is a keyword in Java/C++/C.)
3. Wrap all primitives and strings.
4. Use only one dot per line.
5. Don’t abbreviate.
6. Keep all entities small.
7. Don’t use any classes with more than two instance variables.
8. Use first-class collections.
9. Don’t use any getters, setters, or properties.

**Sources:**<https://dev.to/mohitrajput987/importance-of-writing-clean-code>