Project Development Coding standards

(MEAN-MERN)

Coding standards are **collections of rules and guidelines that determine the programming style, procedures, and methods for a programming language**. Without the coding conventions, every individual in a team will settle their coding styles. It will not be easy to maintain and debug the code in the near future.

What are code quality standards?

Code quality metrics **defines code that is good (high quality) — and code that is bad (low quality)**. This — quality, good, bad — is all subjective. Different teams may use different definitions, based on context.

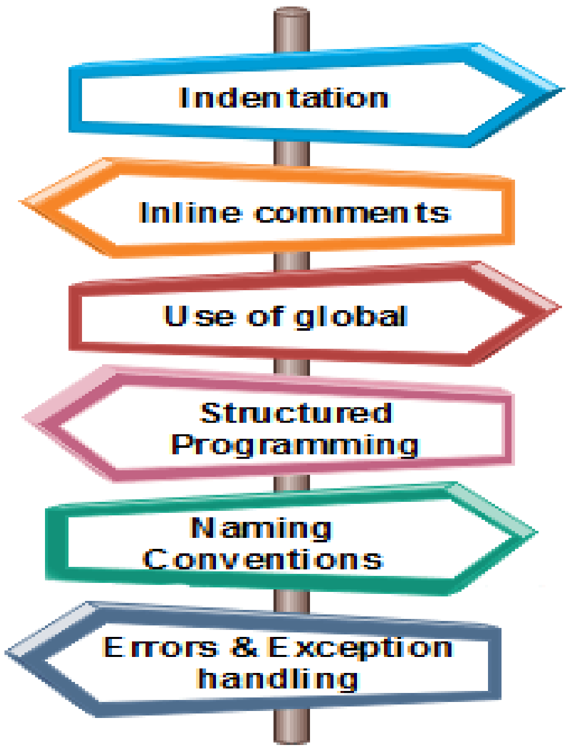
Why are coding standards important?

Coding standards are important for software developers for several reasons: **40% - 80% of the total cost of software is spent on its maintenance**.

Coding standards improve software readability by allowing developers to understand new code faster and better.

General coding standards refers to how the developer writes code, so here we will discuss some essential standards regardless of the programming language being used.

Coding Standards

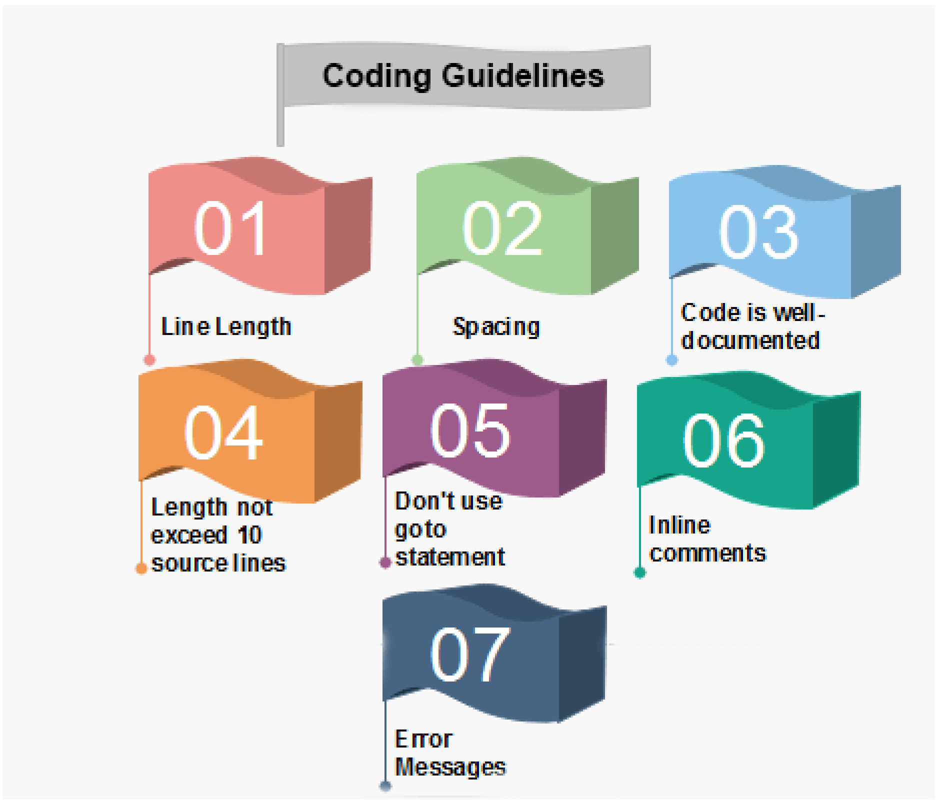


1. **Indentation:** Proper and consistent indentation is essential in producing easy to read and maintainable programs.
   Indentation should be used to:
   - Emphasize the body of a control structure such as a loop or a select statement.
   - Emphasize the body of a conditional statement
   - Emphasize a new scope block
2. **Inline comments:** Inline comments analyse the functioning of the subroutine, or key aspects of the algorithm shall be frequently used.
3. **Rules for limiting the use of global:** These rules file what types of data can be declared global and what cannot.
4. **Naming conventions for global variables, local variables, and constant identifiers:** A possible naming convention can be that global variable names always begin with a capital letter, local variable names are made of small letters, and constant names are always capital letters.

Coding Guidelines

General coding guidelines provide the programmer with a set of the best methods which can be used to make programs more comfortable to read and maintain. Most of the examples use the C language syntax, but the guidelines can be tested to all languages.

The following are some representative coding guidelines recommended by many software development organizations.



1. Line Length: It is considered a good practice to keep the length of source code lines at or below 80 characters. Lines longer than this may not be visible properly on some terminals and tools. Some printers will truncate lines longer than 80 columns.

2. Spacing: The appropriate use of spaces within a line of code can improve readability.

Example:

Bad:
```
cost=price+(price*sales_tax)
fprintf(stdout ,"The total cost is %5.2f\n",cost);
```

Better:
```
cost = price + ( price * sales_tax )
fprintf (stdout,"The total cost is %5.2f\n",cost);
```

3. The code should be well-documented: As a rule of thumb, there must be at least one comment line on the average for every three-source line.

4. The length of any function should not exceed 10 source lines: A very lengthy function is generally very difficult to understand as it possibly carries out many various functions. For the same reason, lengthy functions are possible to have a disproportionately larger number of bugs.

5. Inline Comments: Inline comments promote readability.

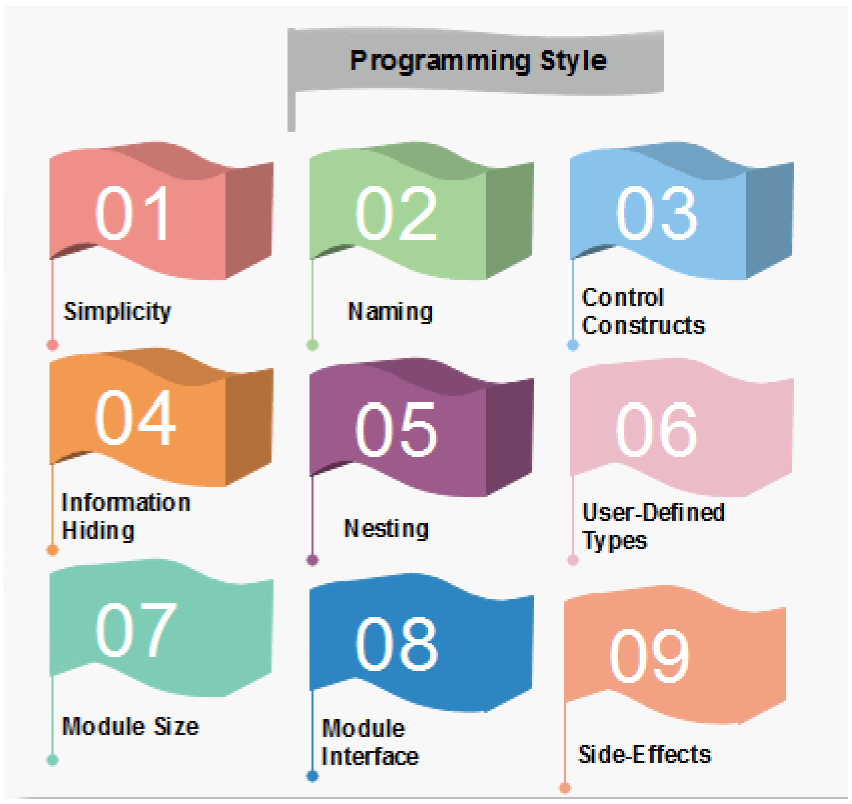
7. Error Messages: Error handling is an essential aspect of computer programming. This does not only include adding the necessary logic to test for and handle errors but also involves making error messages meaningful.

Programming Style

Programming style refers to the technique used in writing the source code for a computer program. Most programming styles are designed to help programmers quickly read and understands the program as well as avoid making errors. (Older programming styles also focused on conserving screen space.) A good coding style can overcome the many deficiencies of a first programming language, while poor style can defeat the intent of an excellent language.

The goal of good programming style is to provide understandable, straightforward, elegant code. The programming style used in a various program may be derived from the coding standards or code conventions of a company or other computing organization, as well as the preferences of the actual programmer.

Some general rules or guidelines in respect of programming style:



1. **Clarity and simplicity of Expression:** The programs should be designed in such a manner so that the objectives of the program is clear.
2. **Naming:** In a program, you are required to name the module, processes, and variable, and so on. Care should be taken that the naming style should not be cryptic and non-representative.
3. **For Example:** a = 3.14 * r * r
   area of circle = 3.14 * radius * radius;
4. **Control Constructs:** It is desirable that as much as a possible single entry and single exit constructs used.
5. **Information hiding:** The information secure in the data structures should be hidden from the rest of the system where possible. Information hiding can decrease the coupling between modules and make the system more maintainable.
6. **Nesting:** Deep nesting of loops and conditions greatly harm the static and dynamic behavior of a program. It also becomes difficult to understand the program logic, so it is desirable to avoid deep nesting.
7. **User-defined types:** Make heavy use of user-defined data types like enum, class, structure, and union. These data types make your program code easy to write and easy to understand.
8. **Module size:** The module size should be uniform. The size of the module should not be too big or too small. If the module size is too large, it is not generally functionally cohesive. If the module size is too small, it leads to unnecessary overheads.

9. **Module Interface:** A module with a complex interface should be carefully examined.
10. **Side-effects:** When a module is invoked, it sometimes has a side effect of modifying the program state. Such side-effect should be avoided where as possible.