Coding Standards, Following Conventions, Naming, Clean Code

Code_Quality Code_Quality_2 Code_Quality_3 Good_Code

> *Any fool can write code that a computer can understand. Good programmers write code that humans can understand*
>
> *– Martin Fowler*

NAMING

BAD

```
function my_function() {

}

function yourFuntion() {

}

function function1() {

}

function function2() {

}

function this_is_the_name_of_a_very_long_function_and_is_written_by_me() {

}

function ab() {

}
```

```
function displayName() {

}

function getGender() {

}
```

SPACES

```
lifewithoutspaces
```

```
life_without_spaces
// snake_case
```

```
lifeWithoutSpaces
// camelCase
```

```
life-without-spaces
// kebab-case
```

BAD

```
dOntINvent_YOURown_casE
```

UNIFORMITY

BAD

```
function setName() {

}

function set_email() {

}
```

GOOD

```
function setName() {

}

function setEmail() {

}
```

```
function set_name() {

}

function set_email() {

}
```

SPACING & READABILITY

BAD

```
function function1() {
    doing set A 1
    doing set A 2
    doing set B 1
    doing set B 2
}
function function2() {

}
```

```
function function3() {

}
```

GOOD

```
function function1() {
    doing set A 1
    doing set A 2

    doing set B 1
    doing set B 2
}

function function2() {

}

function function3() {

}
```

COMMENTS

BAD

```
// this class set the background color of an element to red
.redBox {
    background-color: red;
}

/**
If a number is not divisible by 2 then it is called odd number
To find a number is odd or not we divide it by 2
  - if the remainder is 1 then it is a odd number
  - if the remainder is 0 then it is a even number
**/
function isOddNumber() {

}
```

GOOD

```
// Styles for the comment box used for Posts and Videos
.mainCommentBox {

}

/**
email & mobile - mandatory
dob & city - optional
**/
function checkRegistrationFields() {
```

```
}
```

Basics of the Unix Philosophy

The ‘Unix philosophy’ originated with Ken Thompson's early meditations on how to design a small but capable operating system with a clean service interface. It grew as the Unix culture learned things about how to get maximum leverage out of Thompson's design. It absorbed lessons from many sources along the way.

- Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new features.
- Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
- Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
- Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

Print Odd & Even Number from 1-20

- Basic Increments https://codepen.io/nrupuld/pen/PrQxYm?editors=1011
- Basic Increments (Params) https://codepen.io/nrupuld/pen/wLyQaP?editors=1011
- Looping Modulus https://codepen.io/nrupuld/pen/gNvQPR?editors=1011
- Looping Modulus Single Function https://codepen.io/nrupuld/pen/vqdQXg?editors=1011
- Looping Modulus Generalised Function https://codepen.io/nrupuld/pen/rEJQjE?editors=1011
- Looping Modulus Generalised Abstraction https://codepen.io/nrupuld/pen/vqdQem?editors=1011