BASIC CODING STANDARD

- **IF STATEMENT**
  To make a block with a conditional, it must be done in the following way:

```
if ($condition){
  ...
}
```

  if you need to make a conditional inside a view, remember that you must use the blade templates

```
@if ($condition)
    ...
@elseif ($second_condition)
    ...
@endif
```

  Do not forget the indentation within the conditional sequences

- **STRING**
  When possible prefer string interpolation above sprintf and the . operator.

```
// Good
$greeting = "Hi, I am {$name}.";
// Bad
$greeting = 'Hi, I am ' . $name . '.';
```

- **WHITESPACE**

  Statements should have to breathe. In general always add blank lines between statements, unless they're a sequence of single-line equivalent operations.

  Don't add any extra empty lines between {} brackets.

```
// Bad
if ($foo) {
// whitespace
```

```
    $this->foo = $foo;

// whitespace
}
```

- **VALIDATION**
  When using multiple rules for one field in a form request, avoid using |, always use array notation. Using an array notation will make it easier to apply custom rule classes to a field.

```
// good

        'email' => ['required', 'email'],

// bad

        'email' => 'required|email',
```

- **NOTATION TO VIEWS, CLASSES AND METHODS**

  Views, classes and methods must use camelCase

```
// good

        class OpenSourceController

        {}

// bad

        class opensourcecontroller

        {}

        class open_source_controller

        {}
```

- **VARIABLES**

  Variables must use low_case

```
// Good

$short_string = "Hi, I am {$name}.";

// Bad

$shortString = "Hi, I am {$name}.";
```

COMMENTS

The following structure is used for comments

- If the comment is a single line use //

Example

```
// this is a one-line comment
```

- If you need to explain a lot, you can use a comment block. Notice the single * on the first line:

```
/*
* This is a
* multi-line
* comment
* /
```