Assertion with assert

In this lesson, we will be learning how to use the assert statement.

We'll cover the following

- Syntax
- Learn by Example
- Comparing with if

`assert` is an incredibly useful statement that allows you to put conditions on code execution. It's used to disrupt normal execution when a boolean condition is `false`.

Syntax

The syntax is as follows:

```
assert(condition, optional message)
```

The first argument passed to `assert` can be any expression that reduces to a boolean value. If the expression's value is true, the assertion succeeds, and execution continues. If it's false, the assertion fails and an exception (error) is thrown.

To attach a message to an assertion, add a string as the second argument to `assert`.

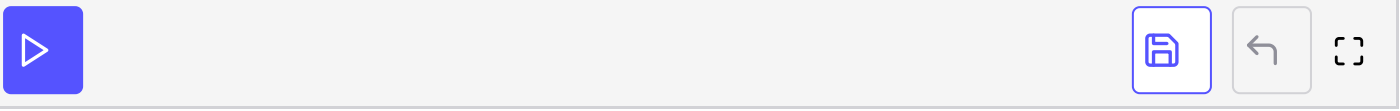
Learn by Example

Let's take a look at an example to get a better understanding of things.

```
main() {
  var variable;
  print(variable);

  assert(variable != null);
```

```
  variable = 5;
  print(variable);
}
```



On **line 2** of the code snippet above, we are declaring a variable, `variable`, without assigning it a value, hence, it is equivalent to `null`.

On **line 5** we are using an assert statement and passing it the expression `variable != null`. As this expression is `false`, the code after the assertion will not be executed and an exception will be thrown.

> **Try it Yourself:** See how the output changes when you modify the expression from `variable != null` to `variable == null`.

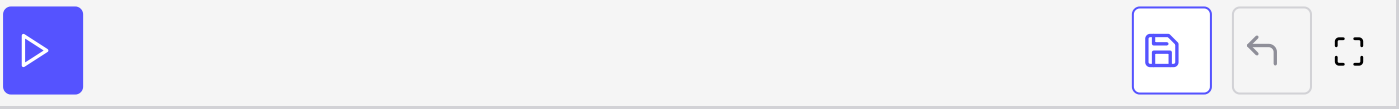
Comparing with `if`

One thing that might come to mind is that `assert` is similar to an `if` statement.

Let's modify the code above so that it uses an `if-else` statement rather than `assert`.

```
main() {
  var variable;
  print(variable);

  if(variable != null){
    print('not null');
  }
  else{
    variable = 5;
    print(variable);
  }
}
```



For one thing, the program continues to run and executes until the last line of code. Furthermore, using an `if` statement results in high levels of nesting of `{}`'s in code. Also, as the code grows and becomes modular, you can put asserts anywhere in any function as needed and count on it to disrupt normal execution when an exceptional condition arises.

With assertion, our discussion on control structures comes to an end. Let’s take a quiz for a quick recap before we move on to the next chapter.