While Loops

Let's take a look at some of the key features of the while loop.

WE'LL COVER THE FOLLOWING

A `while` loop is slightly different from the previously visited `for` loop. Instead of relying on a defined range, it runs as long as a **boolean expression** holds **true**.

The Structure

The template for a `while` loop is fairly simple:

```
while(boolean expression) {
  set of operations
};
```

The logic of this structure lies in the `boolean expression`. At some point, the value of the expression has to change from *true* to *false*. From this, we can make the deduction that the expression will contain a variable whose value updates in the body of the loop until it turns the `boolean expression` into a *false* one.

We can think of this variable or value as the iterator in a `for` loop.

> **Warning**: If the boolean expression of the `while` always holds *true*, the loop will keep running forever, resulting in a potential!

In order to change the value of the `boolean expression`, a part of it must be

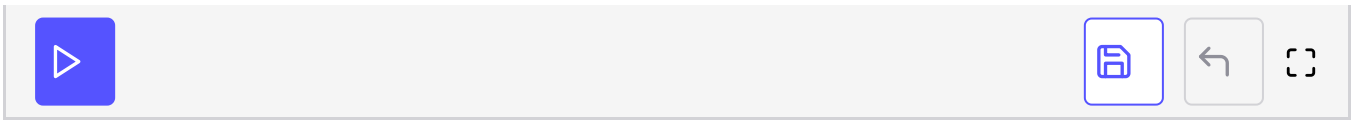
mutable. Hence, a good practice is to stick to mutable entities when dealing with `while` loops.

The Syntax

Let's write a simple `while` loop which prints integers from `0` to `10`:

```
let i = ref(0); /* A mutable iterator */

while(i <= ref(10)){
  print_int(i^);
  print_string(" ");
  i := i^ + 1;
};
```



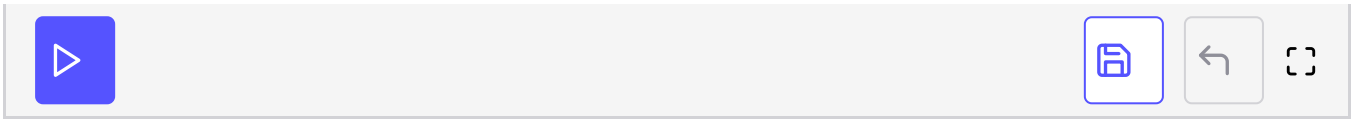
Nothing too tricky going on here. In each iteration, `i` is compared to `ref(10)` and incremented using the `^` operator.

Alternatively, we could write the boolean expression as `i^ <= 10`.

With `while` loops, we can also give the iterator a larger step than just `1`:

```
let i = ref(0); /* A mutable iterator */

while(i <= ref(10)){
  print_int(i^);
  print_string(" ");
  i := i^ + 3; /* A step of 3 */
};
```



Similar to the `for` loop, the `while` structure also has a **unit** return type. Hence, it wouldn't be too wise to return values from it.

There you have it. We're done with loops in ReasonML. To reinforce all the concepts we've learned here, there is a quiz up ahead.