

The option Type

Let's learn about what the option variant type is in Reason.

WE'LL COVER THE FOLLOWING ^

- What is the `option` Type?
- The Structure
- The Syntax

What is the `option` Type?

In theory, Reason does not have **nullable** types. Such types can have a value of *null* or in other words, nothing.

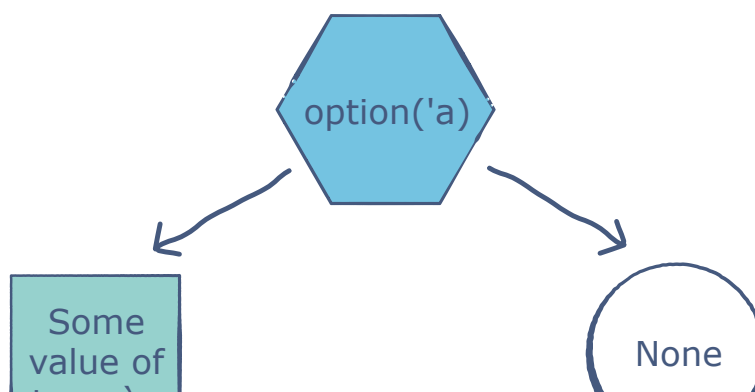
We'll never see a null `int` or `string` object. They must always contain some value.

However, sometimes, we need to cater for an identifier that has a null value. Reason's solution to this problem is the `option` type.

The `option` is a variant type which allows an object to have some value or no value at all.

The Structure

An `option` consists of two parts:



The Syntax

Here's how the components discussed above would translate into Reason:

```
type option('a) =  
  | None  
  | Some('a)
```



In syntactic terms, null is represented by `None` in Reason. `Some` can be interpreted as some actual value of the `option` variable we're creating.

The `option` can be seen as a variant that has the `None` and `Some` constructors.

As a result, we can assign the `None` constructor to any variable:

```
let num = None;  
Js.log(num);
```



In case we want to assign some other value to the variable, we have to call the `Some` constructor.

We can use a `switch` expression to define what happens when `Some` is called:

```
type option('a) =  
  | None  
  | Some('a);  
  
let check = true;  
  
let num =  
  switch(check) {  
    | true => Some(12)  
    | false => None  
  };  
  
switch (num) {  
  | None => "Null"  
  | Some(num) => string_of_int(num)  
};  
  
Js.log(num);
```





As we can see, `None` and `Some` both return different strings. The value of `num` is being decided by a boolean, `check`.

In the next lesson, we'll study an advanced data structure which also uses variants.