Type Errors

In this lesson, we'll see how Reason deals with inconsistencies in types.

WE'LL COVER THE FOLLOWING

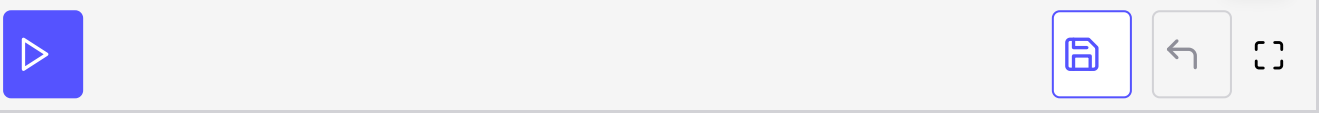
Mixing Different Types

Reason ensures that a value of a particular type only interacts with values of the same type.

Inconsistency in Arithmetic

If we try and mix different types in a computation, the compiler will throw a **type exception**:

```
Js.log(100 *. 5.5); /* int and float */
```



In the code above, the compiler will not recognize the second argument in the multiplication, throwing this error as a result:

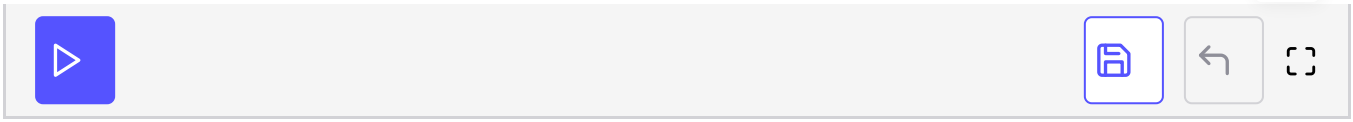
```
Error: This expression has type:
 int
But somewhere wanted:
 float
```

Inconsistency in Comparison

The same principle holds for comparison operators. The types in the

The same principle holds for comparison operators. The types in the expression must be consistent:

```
Js.log("b" == 'b'); /* Js.log("b" == String.make(1,'b')) will return true */
```



These possible exceptions can be avoided if we keep our data types consistent. We'll soon see that we can explicitly define the type of a data object, which makes the code much safer.

By now, we've learned about all the different data types available to us.

Before moving on to the next section, let's take a short quiz to recap what we've learned here.