Try Catch

Let's learn how to actually perform exception handling using try catch.

We'll cover the following

What are `try` and `catch` in R language?

`try` is a keyword that represents the handling of exceptions caused by errors or warnings during the execution of code. Using `try` we can avoid the unwanted termination of code abruptly however, the error message is still displayed.

Syntax for `try`

```
try(<exceptionCause>)
# Pass `try` the object that might throw an exception
```

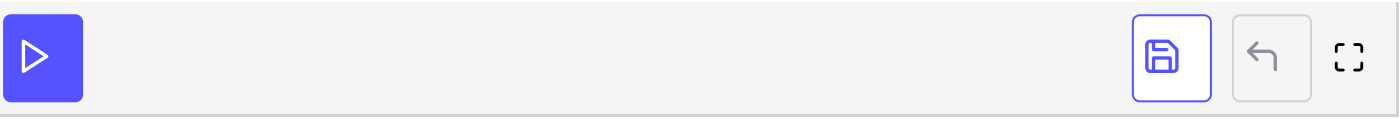
Let’s have a look at an example:

```
Sum <- function(myNumber1, myNumber2)
{
  return(myNumber1 + myNumber2)
}

# Driver Code
Sum("a", 1)
Sum(1, 2)
```

In this code, **Line number** 7 throws an exception and the program halts, which is why the **Line number 8** is not executed even though there are no errors in this line.

```
Sum <- function(myNumber1, myNumber2)
```

```
{
  return(myNumber1 + myNumber2)
}

# Driver Code
try(Sum("a", 1))
try(Sum(1, 2))
```



However, now that we have wrapped the function `Sum()` in `try()`, the code is not halted abruptly. It continues execution but displays the specified error.

What if we do not want to display the compiler error but our specified notification. For this, we will add the catch block.

> A **try block** is the block of code in which exceptions occur and a **catch block** catches and handles try block exceptions.

Syntax for `tryCatch()`

```
result <- tryCatch(
{
    expression
},
warning = function(w)
{
    warningHandlerCode
},
error = function(e) {
    errorHandlerCode
},
finally = {
    cleanup code
}
```

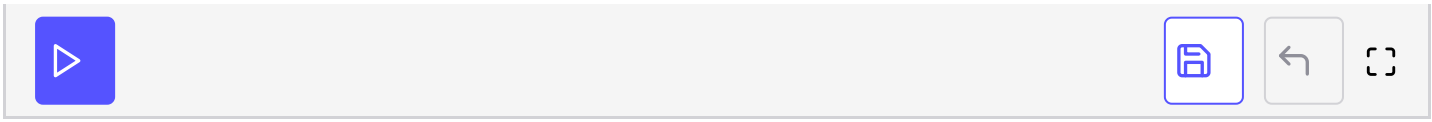
The first parameter is the expression or the object that might throw the error/warning. The second and third parameter involves the resolution of the error/warning depending on the type of message thrown. `error` is evaluated when an error occurs, and `warning` is evaluated when a warning occurs.

The last parameter `finally` is an expression that is evaluated before returning or exiting the `tryCatch()` block. `finally` is always executed, irrespective of whether

an error/warning is thrown or not. We may or may not include this as need be.

Let's have a look at an example:

```
tryCatch(
  sqrt("a"),
  error = function(e)
  {
    print("Error --> Cannot find square root of a character")
  },
  warning = function(w)
  {
    print("Warning --> Cannot find square root of a character")
  },
  finally = {
    print("tryCatch ended")
  }
)
```



Here, **Line number 2** is the expression that throws the exception. Since the exception thrown in this case is an **error**, the function at **Line number 3** is executed and the following statement is printed

```
[1] "Error --> Cannot find the square root of a character"
```

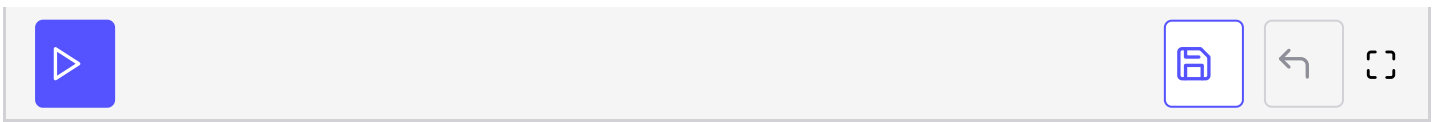
In the last step the `finally` parameter is evaluated and

```
[1] "tryCatch ended"
```

is printed on the console.

Now, let's modify our example using `tryCatch()`:

```
Sum <- function(myNumber1, myNumber2)
{
  return(myNumber1 + myNumber2)
}

# Driver Code
tryCatch(Sum("a", 1), error = function(e) {print("Objects being added do not have same data type")
tryCatch(Sum(1, 2), error = function(e) {print("Objects being added do not have same data type")})
```



Let’s do a quick exercise on implementing try and catch in R language.