F# - OPTIONS

The **option** type in F# is used in calculations when there may or may not exist a value for a variable or function. Option types are used for representing optional values in calculations. They can have two possible values − **Some***x* or **None**.

For example, a function performing a division will return a value in normal situation, but will throw exceptions in case of a zero denominator. Using options here will help to indicate whether the function has succeeded or failed.

An option has an underlying type and can hold a value of that type, or it might not have a value.

Using Options

Let us take the example of division function. The following program explains this −

Let us write a function div, and send two arguments to it 20 and 5 −

```
let div x y = x / y
let res = div 20 5
printfn "Result: %d" res
```

When you compile and execute the program, it yields the following output −

```
Result: 4
```

If the second argument is zero, then the program throws an exception −

```
let div x y = x / y
let res = div 20 0
printfn "Result: %d" res
```

When you compile and execute the program, it yields the following output −

```
Unhandled Exception:
System.DivideByZeroException: Division by zero
```

In such cases, we can use option types to return Some *value* when the operation is successful or None if the operation fails.

The following example demonstrates the use of options −

Example

```
let div x y =
   match y with
   | 0 -> None
   | _ -> Some(x/y)

let res : int option = div 20 4
printfn "Result: %A " res
```

When you compile and execute the program, it yields the following output −

```
Result: Some 5
```

Option Properties and Methods

The option type supports the following properties and methods −

Property or method	Type	Description
None	'T option	A static property that enables you to create an option value that has the **None value**.
IsNone	bool	Returns **true** if the option has the **None** value.
IsSome	bool	Returns **true** if the option has a value that is not **None**.
Some	'T option	A static member that creates an option that has a value that is not **None**.
Value	'T	Returns the underlying value, or throws a NullReferenceException if the value is **None**.

Example 1

```
let checkPositive (a : int) =
   if a > 0 then
      Some(a)
   else
      None

let res : int option = checkPositive(-31)
printfn "Result: %A " res
```

When you compile and execute the program, it yields the following output −

```
Result: <null>
```

Example 2

```
let div x y =
   match y with
   | 0 -> None
   | _ -> Some(x/y)

let res : int option = div 20 4
printfn "Result: %A " res
printfn "Result: %A " res.Value
```

When you compile and execute the program, it yields the following output −

```
Result: Some 5
Result: 5
```

Example 3

```
let isHundred = function
   | Some(100) -> true
   | Some(_) | None -> false

printfn "%A" (isHundred (Some(45)))
printfn "%A" (isHundred (Some(100)))
printfn "%A" (isHundred None)
```

When you compile and execute the program, it yields the following output −

```
false
true
```

```
false
```

Loading [MathJax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js