Switches, Leetcode

Tom Arrell

February 17

Last week...

Last week (2 weeks ago) we covered the basics of Git.

We also had a look at the role of GitHub and its relation to Git.

Today

- Switch statements
  - Case
  - Default
  - Challenge
- Leetcode

Switch Statements

A switch statement is a short way to write a series of if statements. It will execute the a single branch – the first statement in the list which is equal to the "condition."

Switch Example

```
...

func main() {
  color := "blue"

  switch color {
  case "blue":
    fmt.Println("The color of my car")
  case "red":
    fmt.Println("The color of my hat")
  case "green":
    fmt.Println("The color of my grass")
  default:
    fmt.Println("I have nothing of this color")
  }
}
```

Switch Condition

If you omit the condition in the switch statement, the first branch that matches with `true` will be selected.

Switch Truthy Example

```
...

func main() {
  color := "blue"

  switch {
  case true: // Will always print here
    fmt.Println("The color of my car")
  case false:
    fmt.Println("The color of my hat")
  case color == "blue":
    fmt.Println("The color of my grass")
  default:
    fmt.Println("I have nothing of this color")
  }
}
```

Switch Break

If you want to break early from a switch statement, you can use the `break` statement.

```
...

func main() {
  color := "blue"

  switch {
  case true: // Will always print here
    fmt.Println("The color of my car")
    if color == "blue"
      break;
    }
    fmt.Println("The color is not blue")
  default:
    fmt.Println("I have nothing of this color")
  }
}
```

Switch Fallthrough

```
v := 42
switch v {
case 100:
    fmt.Println(100)
    fallthrough
case 42:
    fmt.Println(42)
    fallthrough
case 1:
    fmt.Println(1)
    fallthrough
default:
    fmt.Println("default")
}
// Output:
// 42
// 1
// default
```

Switch Fallthrough

A fallthrough statement must be the last thing in the case.

The following will not work.

```
switch {
case f():
    if g() {
        fallthrough // Does not work!
    }
    h()
default:
    error()
}
```

Fallthrough also does not work in a type switch.

Multiple Cases

You can trigger the same case with multiple values by using a comma separated list.

```
func colors(color string) bool {
    switch color {
    case "blue", "red", "green":
        return true
    }
    return false
}
```

Noop Case

Sometimes you want certain cases to do nothing.

```
func pluralEnding(n int) string {
    ending := ""

    switch n {
    case 1:
    default:
        ending = "s"
    }

    return ending
}

fmt.Sprintf("foo%s\n", pluralEnding(1))  == "foo"
fmt.Sprintf("bar%s\n", pluralEnding(2))  == "bars"
```

Leetcode

Leetcode is a platform for you to improve your programming skills by solving challenges.

https://leetcode.com

Now we'll pick a couple of problems and go through solving them together in Go.

lesson 9, fin

If you had any trouble, now is the time to ask for help!

Questions?