GO on the toilet

episode #3

You don't raise or throw exceptions, you return errors.

There is a lot of code blocks like this in go:

```
fileInfo, err := os.Stat(filename)
if err != nil {
  return fmt.Errorf("could not stat file, err: %v", err)
}
```

Each time you encounter a possibility of error you handle it immediately. You return it and let caller deal with it or you just log it and move on.

In golang errors are normal variables, **`error`** type is an interface with single method **`Error()`**. **`Error()`** returns description of what happened.

Although you can define your own errors, more frequently people just use built-in ones and use messages to describe abnormal situations.

But If you want to check what type of error did you get, you can do it with technique called **type assertion**:

```
func someFunc() error {
     return &SomeError{} //Some defined earlier error
}
func main() {
     err := someFunc()
     if err != nil {
          _, ok := err.(*SomeError)
          if ok {
               fmt.Println("This is SomeError for sure")
          }
     }
}
```

If this looks confusing do not worry, we will talk about type assertions in the future!