GO on the toilet

episode #17

Testing time dependent feature

Imagine that you want to test a feature where passing time plays the key role. Let it be a transaction that must be committed before timeout. To get current time you would probably use **`time.Now()`** which returns an instance of **`time.Time`**. The problem is that invoking **`time.Now()`** in your code makes it impossible to test. Each time you invoke **`go test`**, you'll receive a different time.
It would be desirable to be able to time travel in tests. Set some time at the beginning, advance the clock, and assert a condition. One way to do it is to assign **`time.Now`** to a variable.

```go
// production code
var timeNow = time.Now

func function() {
   fmt.Println(timeNow())
}
```

```go
// test code
func TestFunction(t *testing.T) {
    curr := time.Date(2012, 1, 1, 12, 0, 0, 0, time.Local)
    timeNow = func() time.Time { return curr }
    defer func() { timeNow = time.Now }()
    function() // printed time will be always January 1st 2012
}
```

Production code doesn't use **`time.Now`** directly, but rather **`timeNow`** alias. In tests we can switch **`timeNow`** with any time we want. There are downsides to this approach:

1. **`timeNow`** behavior must be reverted after test execution (deferred in example)
2. because **`timeNow`** is global multiple tests using it cannot run concurrently

We can do better:

```go
// production code
type Clock interface {
    Now() time.Time
}

type realClock struct {}
func (realClock) Now() time.Time {
   return time.Now()
}

func function(clock Clock) {
   fmt.Println(clock.Now())
}
```

```go
// test code
type fakeClock struct { curr time.Time}
func (fc *fakeClock) Now() time.Time {
   return fc.curr
}


func TestFunction(t *testing.T) {
    curr := time.Date(2012, 1, 1, 12, 0, 0, 0, time.Local)
    clock := &fakeClock{curr:curr}
    function(clock)
}
```

Now the time provider is injected. We do not rely on a global variable anymore. Thanks to the fact that type injected into the function is **`Clock`** which is an interface we can satisfy this dependency in tests with any type that implements **`Now()`** method.