



Level 2-2

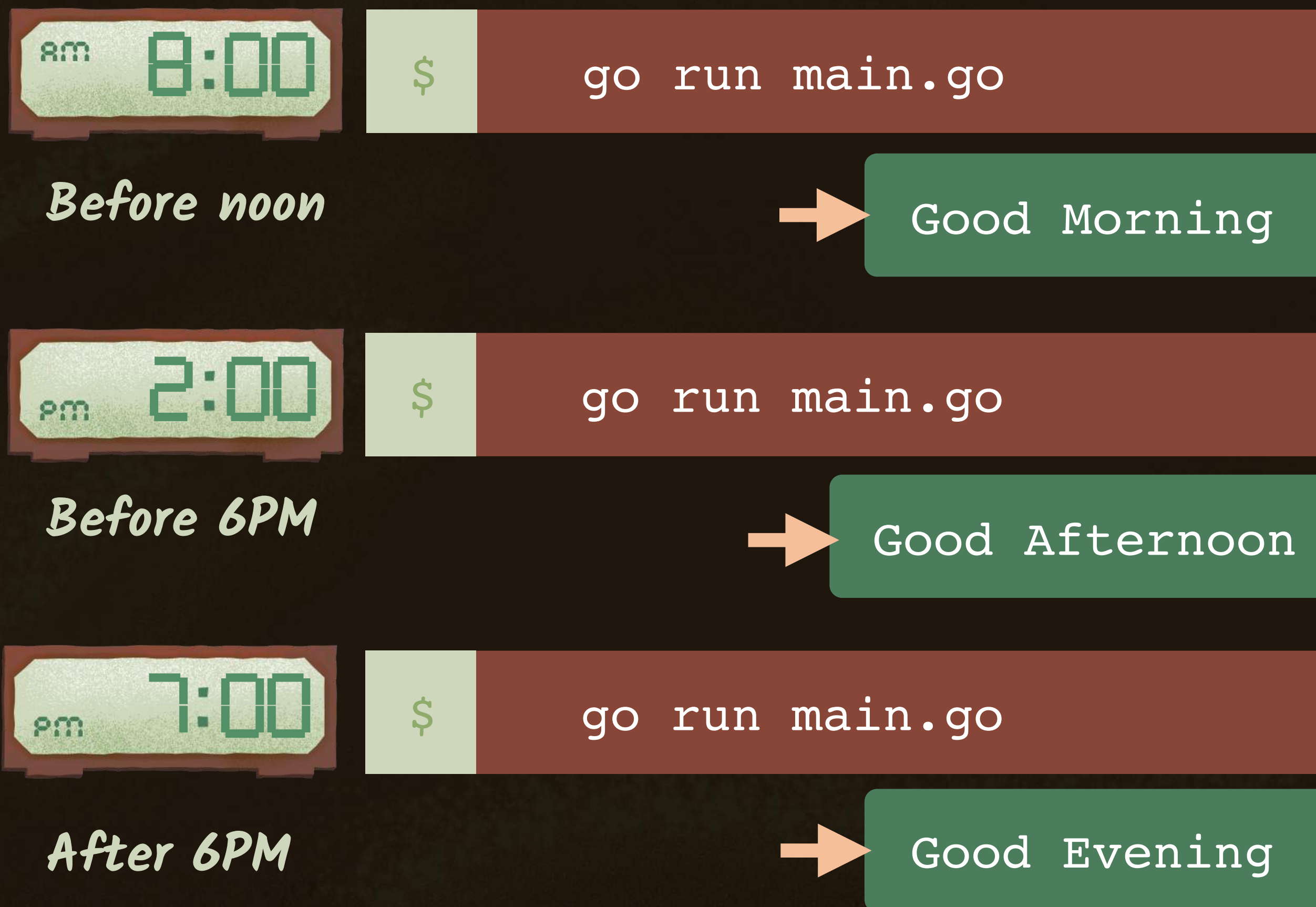
Underneath the Tracks

Functions

ON TRACK
with
GOLANG

The Greeting Program

Let's write a program that **outputs a greeting message**. The message will be different depending on which hour of the day we run our program.



Grabbing Current Time

We'll start by grabbing the **current hour** of the day.

src/hello/main.go

```
package main
```

```
import (  
    "fmt"  
    "time"  
)
```

Import package from standard library.

```
func main() {  
    hourOfDay := time.Now().Hour()  
}
```

Assign return value to a new variable using type inference.

The getGreeting() Function Signature


Named functions in Go must have a **name**, followed by **any arguments** they expect, and end with the **data type** they return (if any). This is commonly referred to as a **function signature**.

src/hello/main.go

...

```
func main() {  
    hourOfDay := time.Now().Hour()  
    greeting := getGreeting(hourOfDay)  
}
```

*How do we find out which
data type this is?*

```
func getGreeting(hour ) string {  
}
```

*Calling the new function passing the current
hour of the day as its single argument*

ANIMATION: func getGreeting shows
first, THEN string, THEN (hour)

*Expected return type
for this function*

The Return Data Type

One way to find out the **data type returned by a function** from the Go standard library is by referring to the official documentation.

src/hello/main.go

...

```
func main() {  
    hourOfDay := time.Now().Hour()  
    greeting := getGreeting(hourOfDay)  
}
```

```
func getGreeting(hour int) string {  
  
}
```

<http://go.codeschool.com/go-time-package>

```
func (t Time) Format(layout string) string  
func (t *Time) GobDecode(data []byte) error  
func (t Time) GobEncode() ([]byte, error)  
func (t Time) Hour() int  
func (t Time) ISOWeek() (year, week int)  
func (t Time) In(loc *Location) Time
```

According to the docs for the time package, Hour() returns a value of type int.

The complete function signature

Returning From `getGreeting()`

We'll return a different greeting message based on the hour of the day passed as argument.

src/hello/main.go

...

```
func getGreeting(hour int) string {  
    if hour < 12 {  
        return "Good Morning"  
    } else if hour < 18 {  
        return "Good Afternoon"  
    } else {  
        return "Good Evening"  
    }  
}
```

*We can nest ifs
inside else clauses.*

ANIMATION: each of these
two line pairs comes in
separately

Returning From `getGreeting()`

We run our program and see a different output depending on the current time of day.

src/hello/main.go

...

```
func main() {  
    hourOfDay := time.Now().Hour()  
    greeting := getGreeting(hourOfDay)  
    fmt.Println(greeting)  
}  
  
func getGreeting(hour int) string {  
  
}
```

