

# Source Code Slides with **reveal**

---

Michael McLoughlin

May 18, 2019

# Hello World

```
package main

import "fmt"

func main() {
    fmt.Println("Hello World!")
}
```

# Hello World

```
package main
```

```
import "fmt"    ◀ Provides formatting functions
```

```
func main() {  
    fmt.Println("Hello World!")  
}
```

# Hello World

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    fmt.Println("Hello World!")    ◀ To stdout
```

```
}
```

Is this *generic* enough?

# The **Greeting** type

```
// Greeting represents a polite word or sign  
// of welcome or recognition.  
type Greeting struct {  
    Salutation string  
    Who        string  
}
```

# The **Greeting** type

```
// Greeting represents a polite word or sign  
// of welcome or recognition.  
type Greeting struct {  
    Salutation string  
    Who        string  
}
```

◁ Example: "Hi"

# The **Greeting** type

```
// Greeting represents a polite word or sign  
// of welcome or recognition.  
type Greeting struct {  
    Salutation string  
    Who        string    ◁ Greeting recipient  
}
```



# Writing Greetings

We implement the `io.WriterTo` interface.

```
// WriteTo writes the greeting to w.  
func (g Greeting) WriteTo(w io.Writer) (int64, error) {  
    n, err := fmt.Fprintf(w, "%s, %s!\n",  
        g.Salutation, g.Who)  
    return int64(n), err  
}
```

# Writing Greetings

We implement the `io.WriterTo` interface.

```
// WriteTo writes the greeting to w.  
func (g Greeting) WriteTo(w io.Writer) (int64, error) {  
    n, err := fmt.Fprintf(w, "%s, %s!\n",  
        g.Salutation, g.Who)  
    return int64(n), err  
}
```

◀ Why int64?

# main, reimagined

```
func main() {  
    g := Greeting{  
        Salutation: "Hello",  
        Who:        "World",  
    }  
    if _, err := g.WriteTo(os.Stdout); err != nil {  
        log.Fatal(err)  
    }  
}
```

# main, reimagined

```
func main() {  
    g := Greeting{  
        Salutation: "Hello",  
        Who:        "World",  
    }  
    if _, err := g.WriteTo(os.Stdout); err != nil {  
        log.Fatal(err)           ◀ Always handle errors  
    }  
}
```

# main, reimagined

```
func main() {  
    g := Greeting{  
        Salutation: "Hello",           < So configurable  
        Who:        "World",           < Much wow  
    }  
    if _, err := g.WriteTo(os.Stdout); err != nil {  
        log.Fatal(err)  
    }  
}
```

<https://github.com/mmccloughlin/reveal>