Go packages

WASA: Web and Software Architecture

Enrico Bassetti

Packages

Packages

- Go program is a composition of packages.
- The main package is `main`. The initial function is `main()` inside the `main` package.
- Packages are imported using the package path, e.g. `encoding/json` or `fmt`
- Each package must be in a dedicated directory.
- A *module* is a group of packages.

Use packages

```go
package main

import (
  "fmt"
  "math/rand"
)

func main() {
  fmt.Println("My favorite number is", rand.Intn(10))
}
```

Standard library vs external

- Go provides a *standard library* with common packages: https://pkg.go.dev/std
- You can create packages inside your project too
- Also, you can use external packages or create packages to share with others.

Create packages inside your project

Inside your project directory, `go mod init some-module` will create a module (`go.mod` and `go.sum`).

From that moment on, you can create sub-modules by creating sub-directories for packages and import them using `some-module/sub-dir`.

Create packages inside your project

```
main.go
go.mod
go.sum
package1/functions.go
package1/other-things.go
package1/subpackage/file.go
```

What is inside `package1/` files

Example: *package1/functions.go*:

```go
package package1

// Note that we need to use the uppercase letter as the
// first letter to make this function public, otherwise
// it will be available only inside package1
func Dummy() {}
```

What is inside `main.go`

Example: *`main.go`*:

```go
package main

import "some-module/package1"

func main() {
  package1.Dummy()
}
```

External modules/packages

To use external modules, you'll need to "get" them first.

```
$ go get gopkg.in/yaml.v2
```

This gets the *gopkg.in/yaml.v2* package. To use it:

```
package main

import "gopkg.in/yaml.v2"

func main() {
  yaml.SomeFunction()
}
```