# Structs exported fields

## Exported structs and fields

Like functions if you would like to export a struct from a package for another package or module to use then the first letter of the struct name must be capitalized, otherwise it is not exported and cannot be used.

```
type ComplexNum struct {
	Real int
	Complex int
}
```

When you are initializing a struct you can use named fields to assign value to the fields such as:

```
c1 := structs.ComplexNum{Real: 10, Complex: 20}
```

Keep in mind that the field must also be exported in order for key-value assignment, otherwise, you cannot assign to it and will receive the default value.

### Special case: Struct are in the same go file

If the struct definition is within the same go file then whether or not the struct / field itself is exported it doesn't matter, you can use it within the same go file. Assign the key-value pair regardless of whether the field is actually exported.

```
type complexNum struct {
	real int
	complex int
}

func main() {
	c1 := complexNum{real: 10, complex: 30}
	fmt.Println(c1)
}
```

### Special case: Struct are in the same package

Very similar to the previous case, if the struct are defined within the same package, you can just use it whether or not the struct / field itself is exported. You can also assigned to any unexported field, it doesn't not matter.

---

Revision #1
Created 2023-05-14 12:29:07 UTC by Tamarine
Updated 2023-05-14 12:36:30 UTC by Tamarine