What does an underscore and interface name after keyword var mean?

Asked 8 years ago Active 2 years, 3 months ago Viewed 14k times



From http://golang.org/src/pkg/database/sql/driver/types.go:

79



27

27

```
type ValueConverter interface {
    // ConvertValue converts a value to a driver Value.
    ConvertValue(v interface{}) (Value, error)
}

var Bool boolType

type boolType struct{}

var _ ValueConverter = boolType{} // line 58

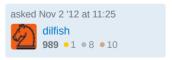
func (boolType) String() string { return "Bool" }

func (boolType) ConvertValue(src interface{}) (Value, error) {....}
```

I known that ValueConverter is an interface name. Line 58 seems to declare that boolType implement interface ValueConverter, but is that necessary? I deleted line 58 and the code works well.

syntax interface go underscores





The blank identifier ___ can be used to strictly provide the keys in a struct too. See this for reference – Vallie Nov 18 '19 at 12:05

Report this ad

2 Answers

Active Oldest Votes



It provides a static (compile time) check that booltype satisfies the valueconverter interface. The _ used as a name of the variable tells the compiler to effectively discard the RHS value, but to type-check it and evaluate it if it has any side effects, but the anonymous variable per se doesn't take any 116 process space.



It is a handy construct when developing and the method set of an interface and/or the methods implemented by a type are frequently changed. The construct serves as a guard against forgetting to match the method sets of a type and of an interface where the intent is to have them compatible. It effectively prevents to go install a broken (intermediate) version with such omission.







It seems like you are creating a dummy value of type valueconverter, assigning a new booltype object to it and then discarding it (which is the meaning of the underscore in go, as in for , elt := range myRange { ...} if you are not interested in the index of the enumeration).

27

My guess is that it simply correspond to a static check to ensure that the struct booltype does implement the valueconverter interface. This way, when you change the implementation of boolType, the compiler will complain early if you broke the implementation of valueConverter interface as it will be unable to cast your new booltype to this interface.

(I)

answered Nov 2 '12 at 11:47

