Roma, 20 Dicembre 2022

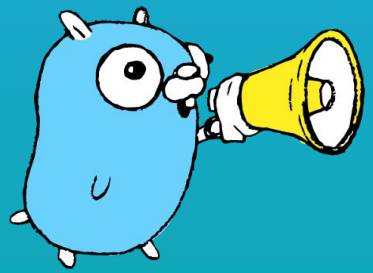
_ BLACK MAGIC _

The black magic behind the blank identifier

Enrico Candino

SUSE

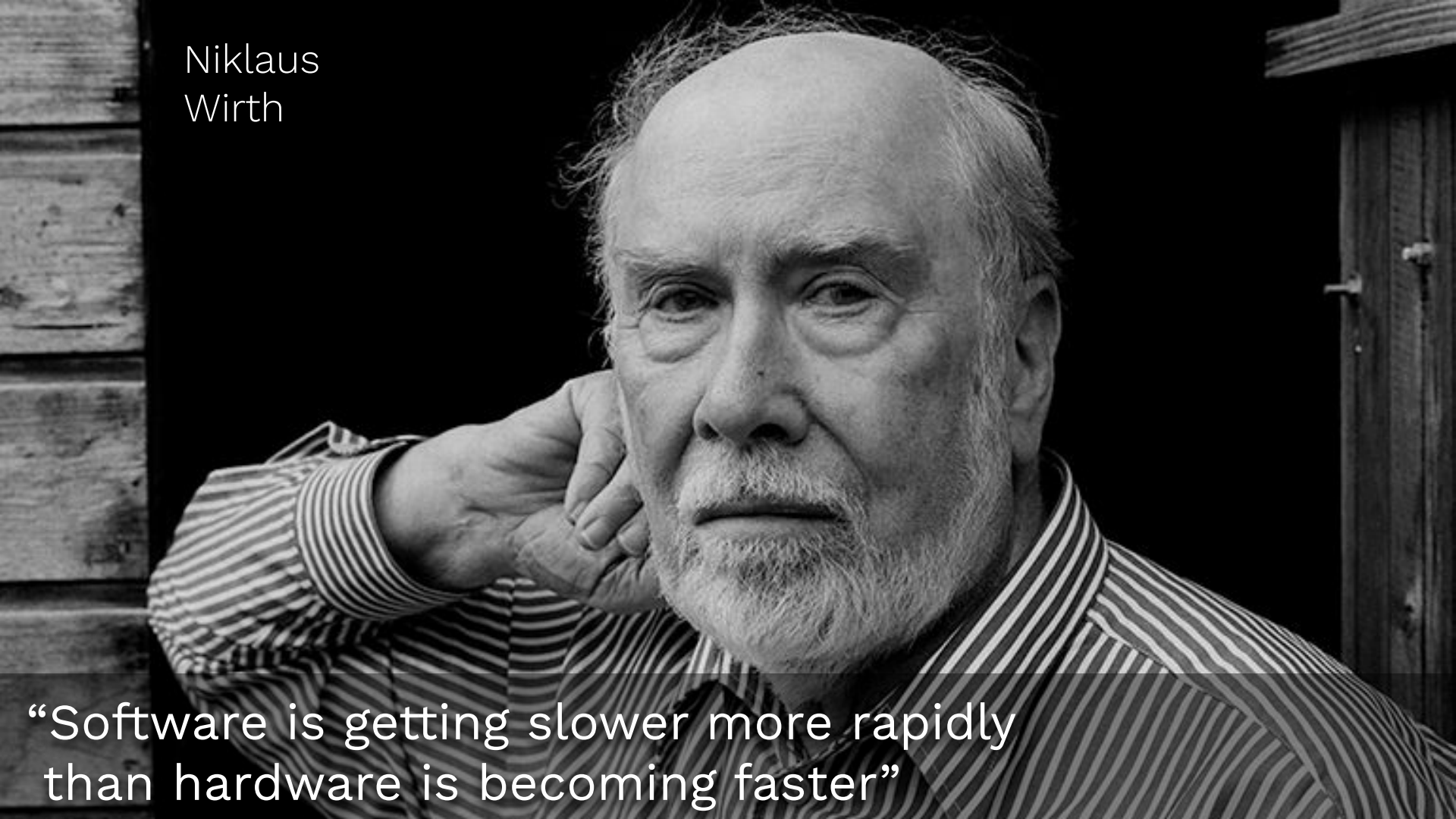
@enrichman



Agenda

01 Definition

Identifier


Niklaus
Wirth
“Software is getting slower more rapidly
than hardware is becoming faster”

The Go Programming Language Specification

Version of June 29, 2022

Identifiers

Identifiers name program entities such as variables and types.
An identifier is a sequence of one or more letters and digits.
The first character in an identifier must be a letter.

https://go.dev/ref/spec

```
identifier = letter { letter | unicode_digit } .
```

```
a
_x9
ThisVariableIsExported
αβ
```

Keywords

break	default	func	interface	select
case	defer	go	map	struct
chan	else	goto	package	switch
const	fallthrough	if	range	type
continue	for	import	return	var

Operators and punctuation

+	&	+=	&=	&&	==	!=	(	)
-	|	-=	|=	||	<	<=	[	]
*	^	*=	^=	<-	>	>=	{	}
/	<<	/=	<<=	++	=	:=	,	;
%	>>	%=	>>=	--	!	...	.	:
	&^		&^=		~			

Predeclared identifiers

```
Types:
    any bool byte comparable
    complex64 complex128 error float32 float64
    int int8 int16 int32 int64 rune string
    uint uint8 uint16 uint32 uint64 uintptr

Constants:
    true false iota

Zero value:
    nil

Functions:
    append cap close complex copy delete imag len
    make new panic print println real recover
```

The blank identifier: _

The *blank identifier* is represented by the underscore character _

It serves as an anonymous placeholder instead of a regular (non-blank) identifier and has special meaning in declarations, as an operand, and in assignment statements.

The blank identifier can be assigned or declared with any value of any type, with the value discarded harmlessly*.

It represents a write-only value to be used as a placeholder where a variable is needed but the actual value is irrelevant.

Hello world!

https://play.golang.com/p/DN2vHVC-w75

```
package main

import (
    "fmt"
)

func main() {
    msg := "Hi GolangRoma!"
    fmt.Println("Hello, playground")
}
```

Output

```
./prog.go:8:2: msg
declared but not used

Go build failed.
```

Hello _

https://play.golang.com/p/biv5vCbzKu0

```
package main

import (
    "fmt"
)

func main() {
    _ = "Hi GolangRoma!"
    fmt.Println("Hello, playground")
}
```

Output

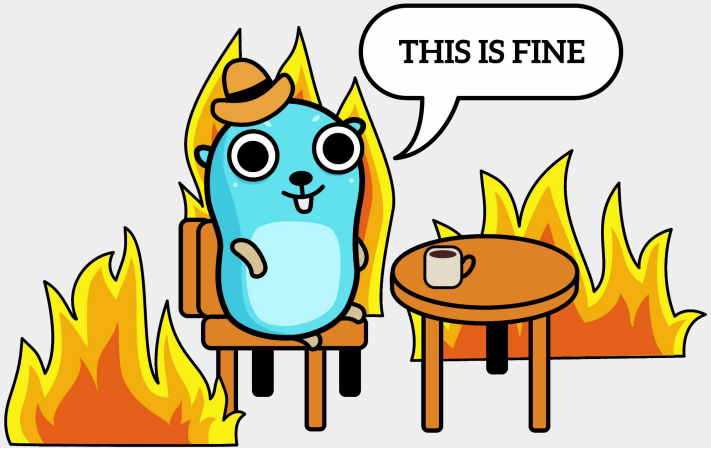
```
Hello, playground

Program exited.
```

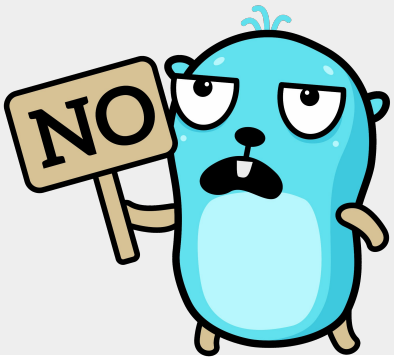
Assignments

```
arr := []string{"one", "ciu", "tri"}

for i, elem := range arr {
    fmt.Printf("%d) %s\n", i, elem)
}

for _, elem := range arr {
    fmt.Println(elem)
}
```

Assignments

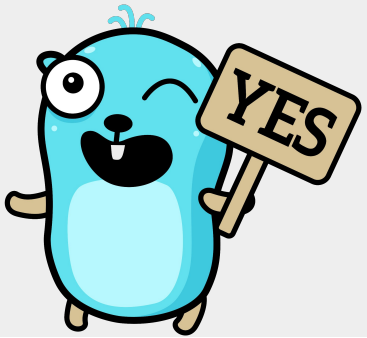
```
res, _ := http.Get(requestURL)
fmt.Printf("status code: %d\n", res.StatusCode)
```





Assignments

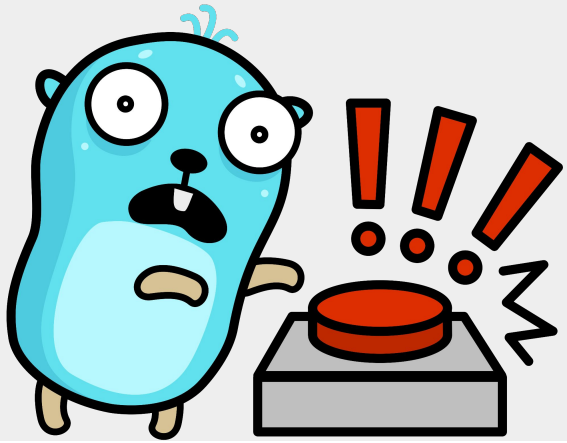
```
res, err := http.Get(requestURL)
if err != nil {
    fmt.Printf("error making http request: %s\n", err)
    os.Exit(1)
}

fmt.Printf("status code: %d\n", res.StatusCode)
```



Assignments

```
func main() {
    s := getString(42)
    fmt.Println(s)
}

func getString(v any) string {
    return v.(string)
}
```

```
panic: interface conversion: interface {} is int, not
string

goroutine 1 [running]:
main.getString(...)
     /tmp/sandbox2016348655/prog.go:13
main.main()
     /tmp/sandbox2016348655/prog.go:8 +0x2e
```



Assignments

```
func main() {
    s := getString(42)
    fmt.Println(s)
}

func getString(v any) string {
    s, _ := v.(string)
    return s
}
```

Assignments

```
_ = 3 + 5

_ = 3 + doSomethingStrange()

outerLoop:
    {
    }

_:
    {
    }
```

Declarations

```
type MyReader struct{}

var _ io.Reader = (*MyReader)(nil)


*MyReader does not implement io.Reader (missing Read method)


func (r *MyReader) Read(p []byte) (n int, err error) {
    return 0, nil
}
```

Side Effects

```
import (
    "database/sql"
)

func main() {
    db, err := sql.Open("postgres", connectionString)
    if err != nil {
        panic(err)
    }
    // do something
}
```

```
panic: sql: unknown driver "postgres" (forgotten import?)
```

Side Effects

```
import (
    "database/sql"
    _ "github.com/lib/pq"
)

func main() {
    db, err := sql.Open("postgres", connectionString)
    if err != nil {
        panic(err)
    }
    // do something
}
```

Side Effects

```
import (
    "database/sql"
    _ "github.com/go-sql-driver/mysql"
)

func main() {
    db, err := sql.Open("mysql", connectionString)
    if err != nil {
        panic(err)
    }
    // do something
}
```

Blank magic!

Side Effects

```
import (
    "database/sql"
    _ "github.com/lib/pq"
)

func main() {
    db, err := sql.Open("postgres", connectionString)
    if err != nil {
        panic(err)
    }
    // do something
}
```

Side Effects

https://github.com/lib/pq/blob/master/conn.go#L59-L61

```
func init() {
    sql.Register("postgres", &Driver{})
}
```

https://pkg.go.dev/database/sql#Register

```
func Register(name string, driver driver.Driver)

type Driver interface {
    Open(name string) (Conn, error)
}
```

Demo time!