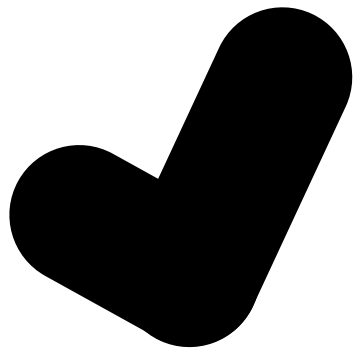# DON'T PRINT,

GO DEBUG!

by Alexander Podkhalyuzin
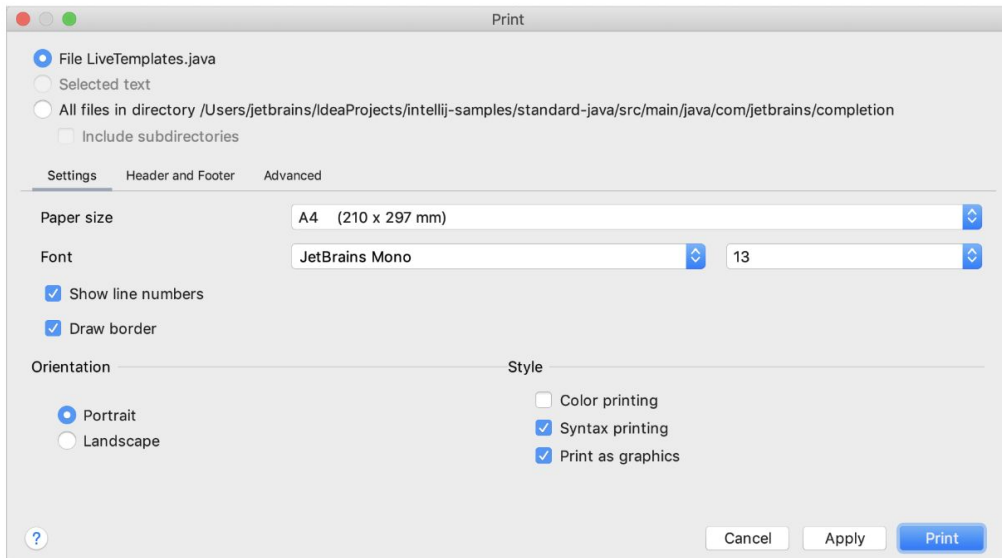彭俐客

How to print?

Casey Newton
@CaseyNewton

...

Here it is: Twitter engineers were told today to *print out* their last 30 to 60 days of code, so they could show it to Elon Musk himself.

Then they were told wait, no, actually, please shred all that code you just printed out.
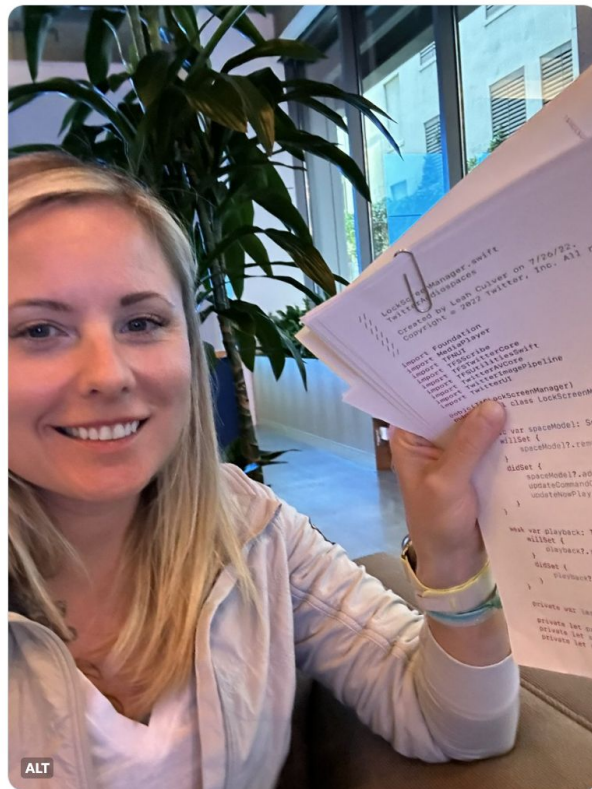
Subscribe to read ➡️

platformer.news/p/elon-takes-o…

How to print?

After you configure the printing options and click **Print**, you will see the system **Print** dialog. There, you can select the printer you want to use.

plan

plan

```
First steps
```

plan

1 First steps

2 Run to cursor

plan

1 First steps

2 Run to cursor

3 Don't debug. Print!

plan

1 First steps

2 Run to cursor

3 Don't debug. Print!

4 Attach to process

plan

1. First steps

2. Run to cursor

3. Don't debug. Print!

4. Attach to process

5. Debug Goroutines

plan

1 First steps

2 Run to cursor

3 Don't debug. Print!

4 Attach to process

5 Debug Goroutines

6 Superpower!

plan

What's instead?

Debug !

First steps!

```go
package main

import (

    "fmt"

)

// Book represents a book in the library.
type Book struct {    4 usages
    ID             int
    Title          string
    Author         string
    PublicationYear int
}

// Library represents a library.
type Library struct {    6 usages
    books      []Book
    lastBookID int
```

*LibraryApp.Run()

plan

Try to be faster than debugger

Shout the 'total' value!

```go
package main

import "fmt"

func sum(numbers []int) int {  1 usage
    total := 0

    // Debugging: Print the initial value of total
    fmt.Println( a...:"Initial total value:", total)

    for _, num := range numbers {
        total += num
    }

    // Debugging: Print the final value of total
    fmt.Println( a...:"Final total value:", total)
```

plan

Rarely you still need to print...

Almost 9 years ago, while in Scala, I had an epic:

SCL-8025   Created by Ilan Godik about 9 years ago
Updated by Alexander Podkhalyuzin almost 9 years ago

👁 Visible to issue readers ⌄

Standard scala library unrecognized on sbt project creation on 2.11.4

Scenario:

Created an sbt project, the scala version defaulted to 2.11.4 in the generated build.sbt.

The fix was this one:

```
73   −   def byte(i : Int) = bytes(pos) & 0xFF
78   +   def byte(i : Int) = {
79   +     val b = bytes(pos)
80   +     if (b >= 0) b.toInt
81   +     else b.toInt + 256
82   +   }
```

Such cases are just once per 10 years

For the rest, please, **debug!**

plan

attach-process.go ✕

```go
package main

import (
    "fmt"
    "log"
    "net/http"
    "os"
    "time"

    "github.com/gorilla/mux"
)

const (
    readTimeout  = 5    1 usage
    writeTimeout = 10   1 usage
    idleTimeout  = 120  1 usage
)
```

indexHandler(w http.ResponseWriter, r *http.Request)

plan

label-goroutines.go ✕

```go
package main


func main(){

}
```

main()

Is it allowed to use AI in your company?

Do you actually use AI?

plan

1   First steps

2   Run to cursor

3   Don't debug. Print!

4   Attach to process

5   Debug Goroutines

6   Superpower!

plan

1 First steps

2 Run to cursor

3 Don't debug. Print!

4 Attach to process

5 Debug Goroutines

6 Superpower!

Version control ∨
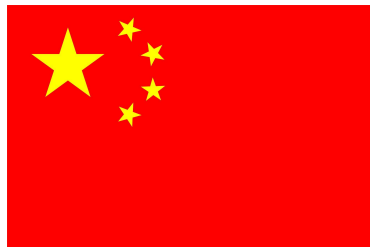
Add Configuration... ∨

Project ∨

main.go ✕

□ full-line-completion-demo ~/Gola
  > go.mod
    main.go
  > External Libraries
  > Scratches and Consoles

```go
1   package main
2
3   func main() {
4
5   }
6
```

main()

full-line-completion-demo > main.go

4:1   LF   UTF-8   Tab

GoLand / CSE in China Mainland
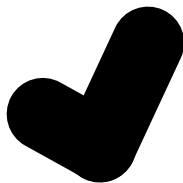
GoLand / CSE in China Mainland

彭俐客 (Alex)

Scan the QR code to add me as friend

Thank you!

Alexander Podkhalyuzin
彭俐客

彭俐客 (Alex)

Scan the QR code to add me as friend