The Swift Programming Language

Part 6 – The Optionals

iCodeWave Community

What We’re Going to Learn?

Introduction

Declaration & Initialization

Force Unwrap an Optional

Optional Binding

Nil Coalescing

Use of Guard Statement

Optional Chaining

iCodeWave Community

Introduction to Optionals

Optionals let you indicate the absence of a value for any type at all, without the need for special constants.

Look at box illustration beside. You don’t even know what is inside that box. Even we don’t know if that box is empty.

This is sample analogy of optional.

iCodeWave Community

Declaration & Initialization

How do we do an optional in Swift?

- An optional is declared by adding a question mark (?) After the data type
- Put them directly after data type, **don't** give space or separate the question mark after data type
- Example:

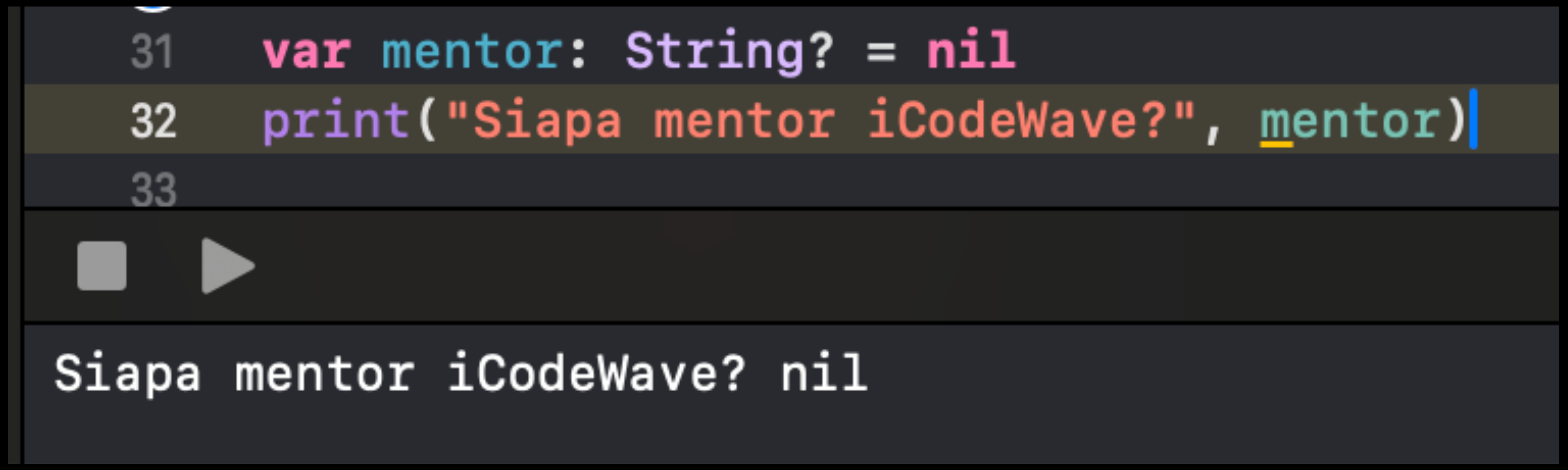
```
var example: Int? = 10 // The value variable is optional with an initial value of 10
```

- Blue text color gives us a clue that "example" variable is a optional
- Pink text color is initialization value of "example" variable
- We can even initialize and treat nil as a value

iCodeWave Community

Declaration & Initialization

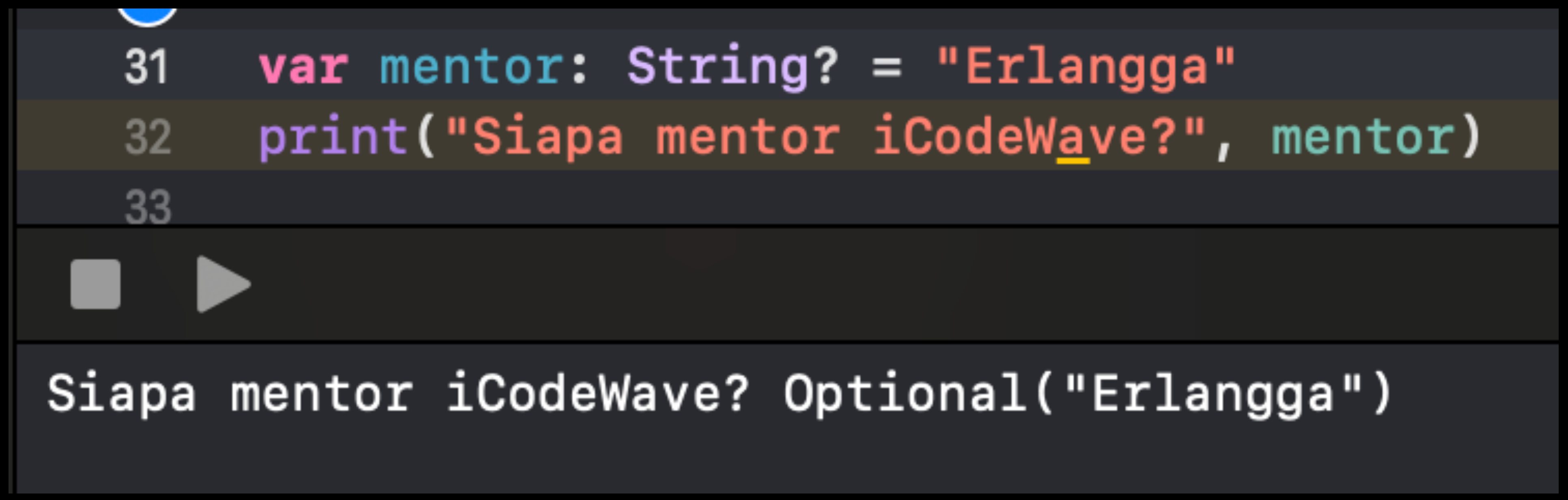
Example codes



```swift
var mentor: String? = nil
print("Siapa mentor iCodeWave?", mentor)
```

```
Siapa mentor iCodeWave? nil
```

Initialization with nil value



```swift
var mentor: String? = "Erlangga"
print("Siapa mentor iCodeWave?", mentor)
```

```
Siapa mentor iCodeWave? Optional("Erlangga")
```

Initialization with real value (this case is String). Pay attention to output, there's an Optional output

iCodeWave Community

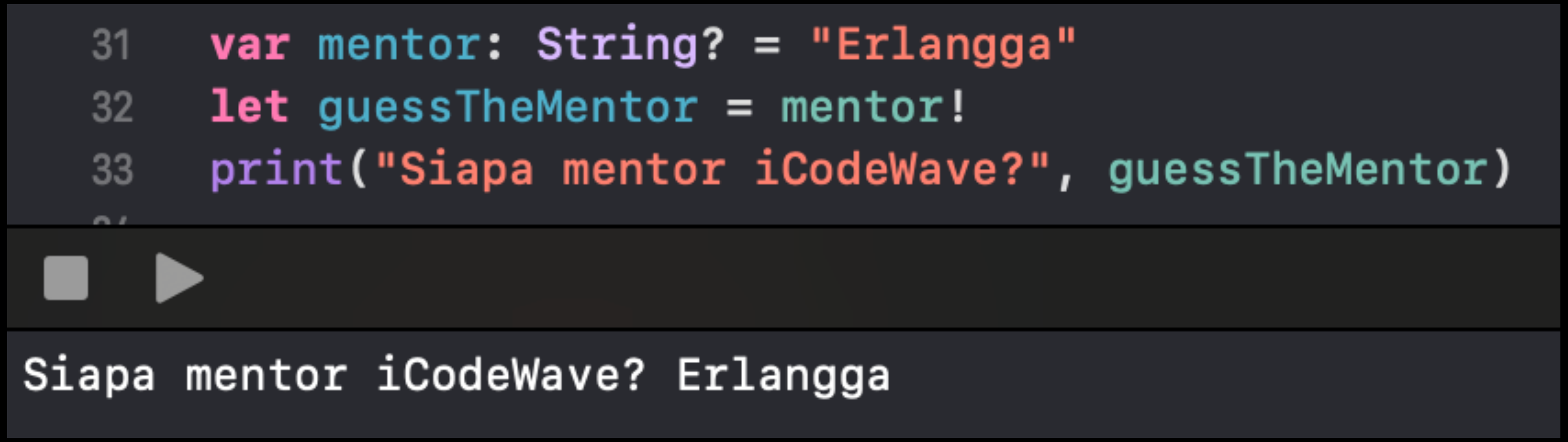
Force Unwrap an Optional

Handling optionals, Part 1 – The risky way

- The process of extracting the value in an optional
- We can use exclamation mark ! to get rid of Optional value
- Use ! after a variable to mark them as force unwrap
- Keep in mind that forcing unwrap a value will return error if value given is nil

Force Unwrap an Optional

Example codes

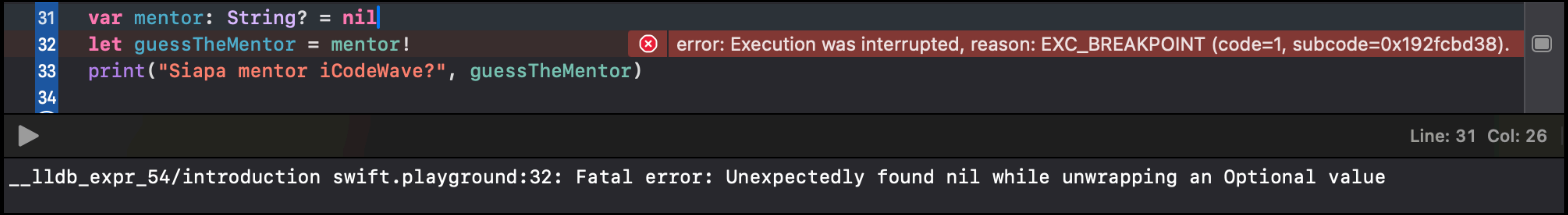


```
var mentor: String? = "Erlangga"
let guessTheMentor = mentor!
print("Siapa mentor iCodeWave?", guessTheMentor)
```

```
Siapa mentor iCodeWave? Erlangga
```

At a glympse, we just get rid of Optional output printed on our Debug area

This is what we get if we force unwrap an optional that has value of nil on it



```
var mentor: String? = nil
let guessTheMentor = mentor!        error: Execution was interrupted, reason: EXC_BREAKPOINT (code=1, subcode=0x192fcbd38).
print("Siapa mentor iCodeWave?", guessTheMentor)
```

Line: 31 Col: 26

```
__lldb_expr_54/introduction swift.playground:32: Fatal error: Unexpectedly found nil while unwrapping an Optional value
```

iCodeWave Community

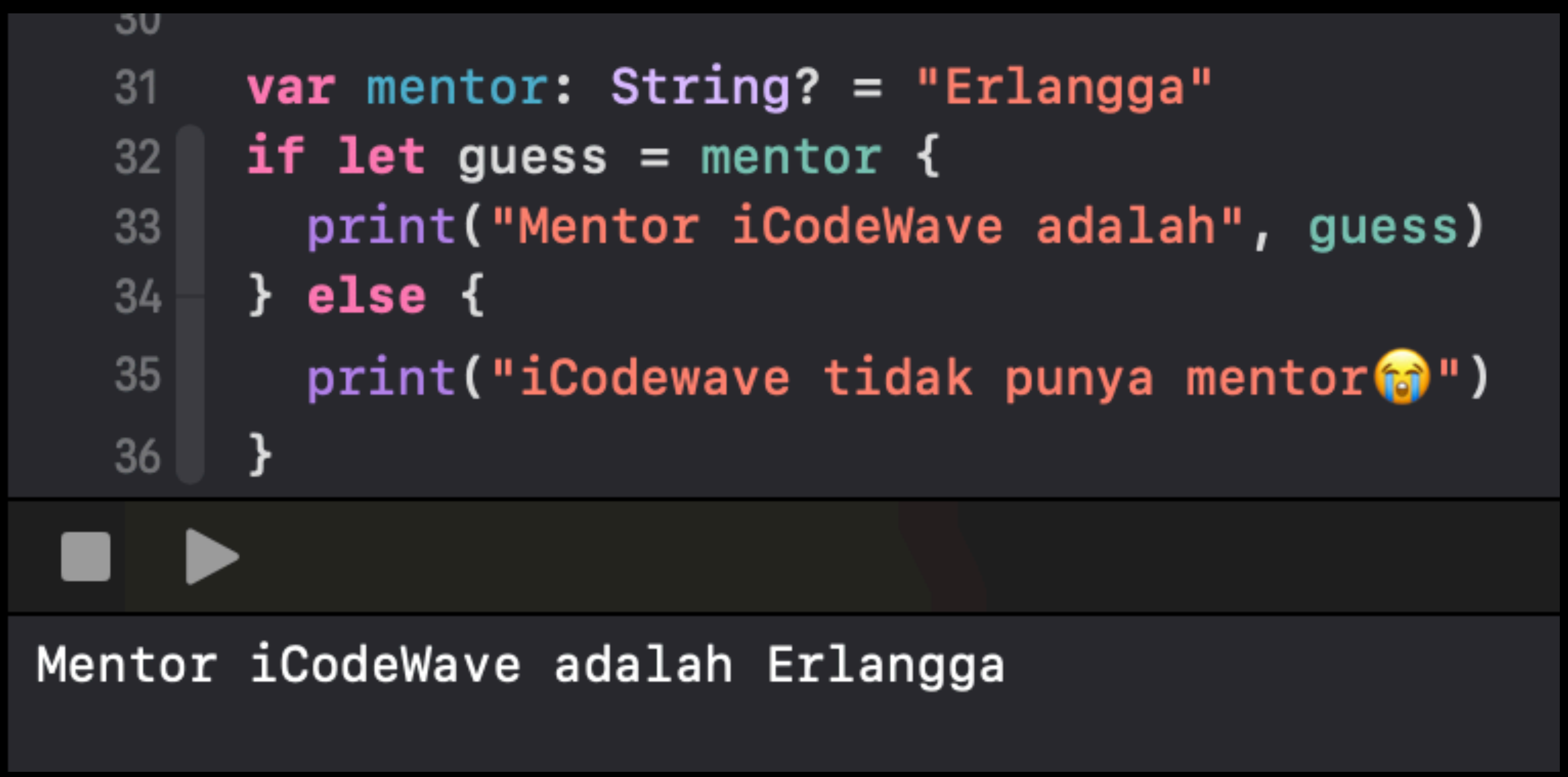
Optional Binding

Handling optionals, Part 2 – The looonger wayyy

- Optional binding is used to decipher values from optional into new constants or variables
- This allows us to use the optional value safely
- This way is safer than our part 1 (force unwrap a value)

Optional Binding

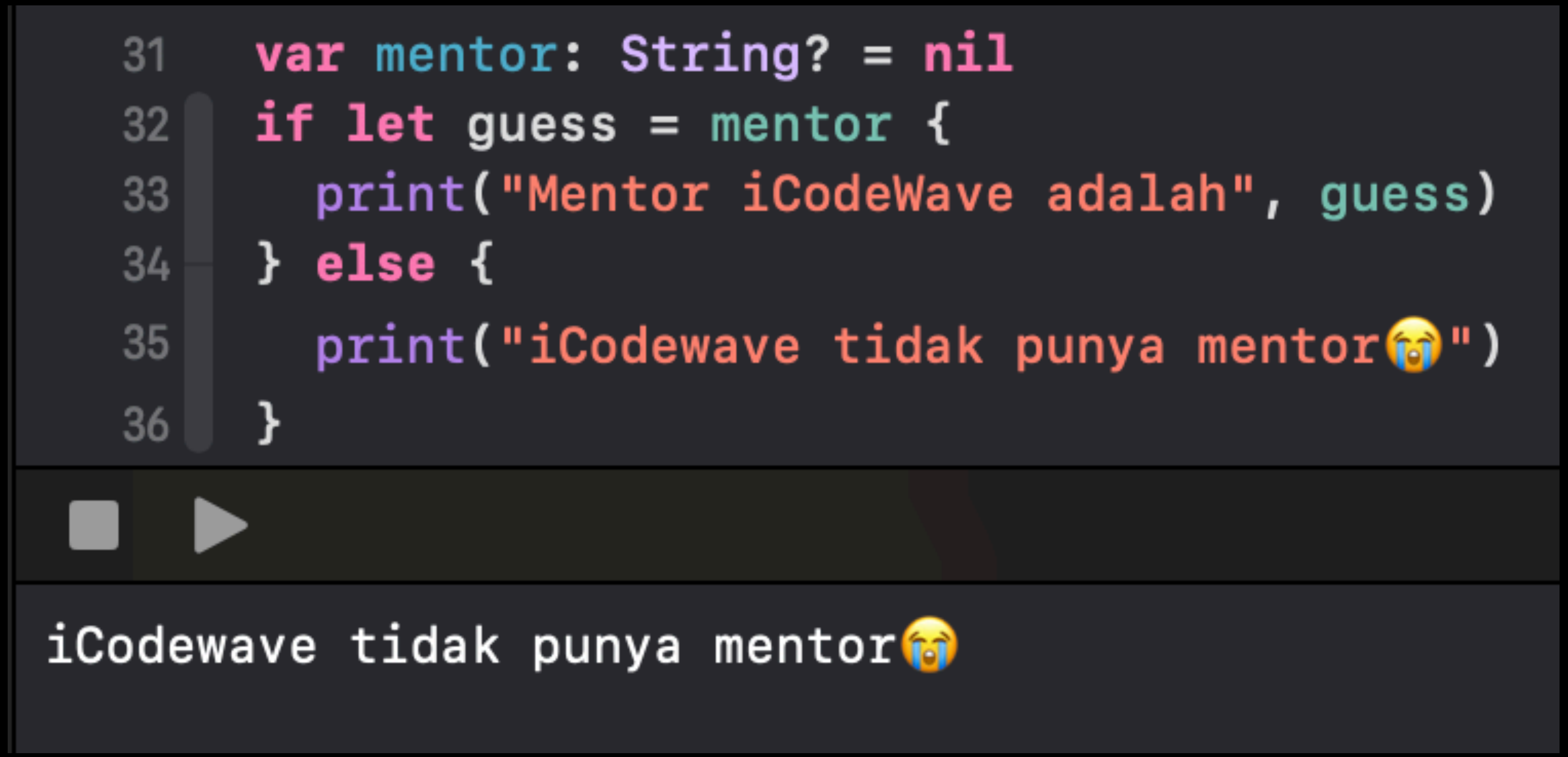
Example codes



```swift
var mentor: String? = "Erlangga"
if let guess = mentor {
  print("Mentor iCodeWave adalah", guess)
} else {
  print("iCodewave tidak punya mentor😭")
}
```

```
Mentor iCodeWave adalah Erlangga
```

Value is not nil (String)



```swift
var mentor: String? = nil
if let guess = mentor {
  print("Mentor iCodeWave adalah", guess)
} else {
  print("iCodewave tidak punya mentor😭")
}
```

```
iCodewave tidak punya mentor😭
```

Value is a nil

Nil Coalescing Operator

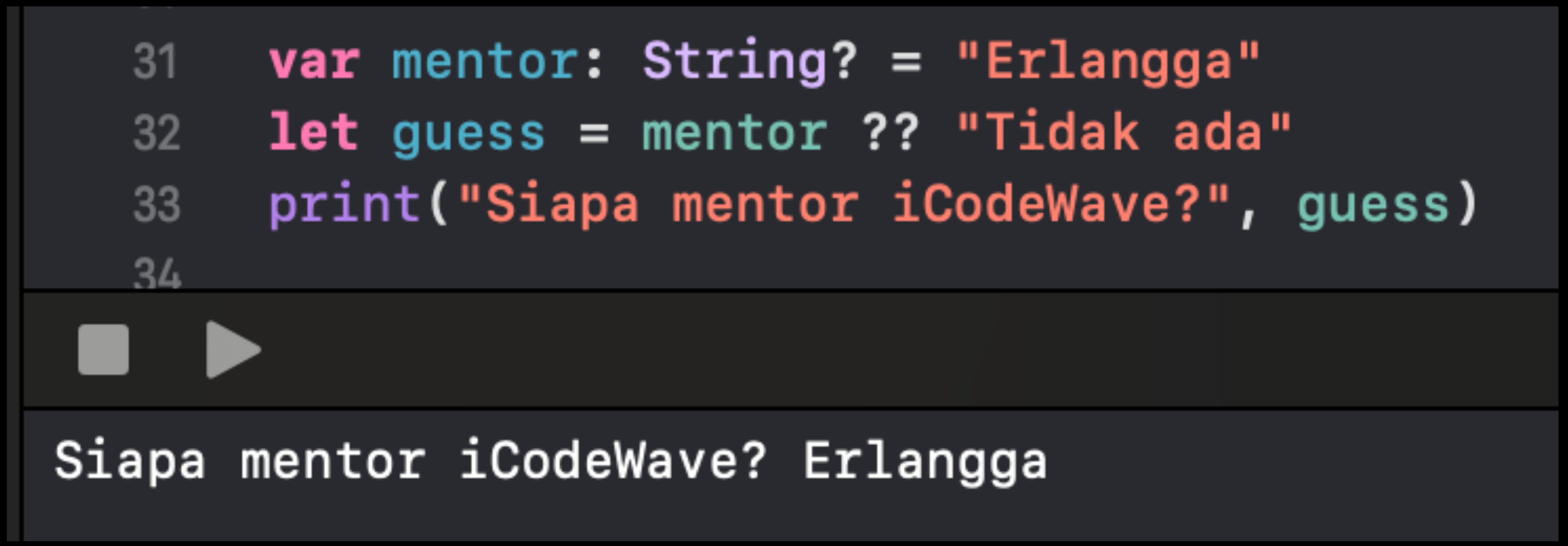
Handling optionals, Part 3 – The s1mpl3st w4y.

- Nil coalescing operator (??) Used to give the default value to optional if nil
- This allows us to provide an alternative if the optional value is nil
- Be careful to not swap between nil coalescing and ternary operator, because both have similar syntax
- The syntax: value ?? default value
- Pink text color indicates your value that contains optional
- Yellow text color is a nil coalescing operator sign
- Blue text color is providing default value (depend on value’s data type) when value is nil

iCodeWave Community

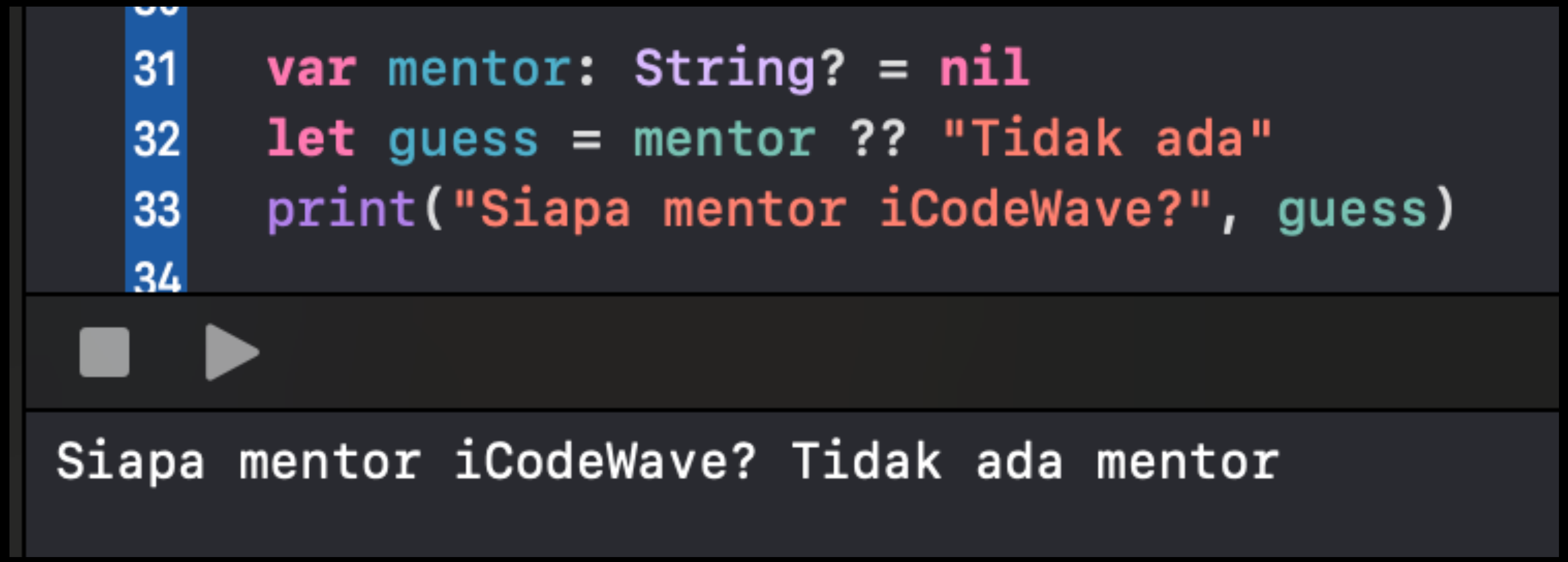
Nil Coalescing Operator

Example codes



```
var mentor: String? = "Erlangga"
let guess = mentor ?? "Tidak ada"
print("Siapa mentor iCodeWave?", guess)
```

```
Siapa mentor iCodeWave? Erlangga
```



```
var mentor: String? = nil
let guess = mentor ?? "Tidak ada"
print("Siapa mentor iCodeWave?", guess)
```

```
Siapa mentor iCodeWave? Tidak ada mentor
```

iCodeWave Community

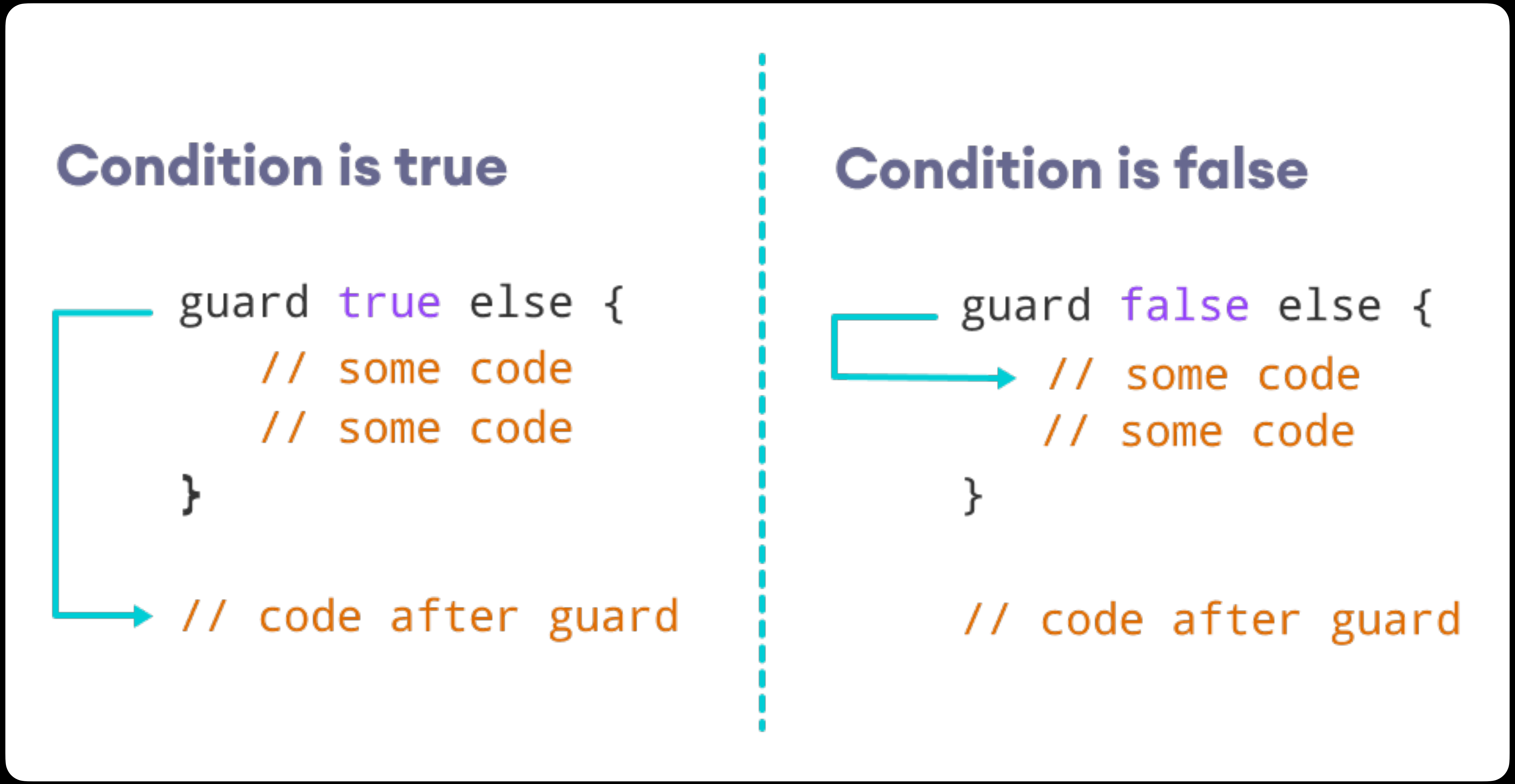
Use of Guard Statement

Handling optionals, Part 4 – Whoa, another way?

- A guard statement is used to check if a condition is met
- Otherwise, the program is out of the current scope
- Another way used to handle optional value
- The guard statement is similar to the if statement with one major difference, the if statement runs when a certain condition is met
- However, the guard statement runs when a certain condition is not met
- Guard **works best** only in function, otherwise you should return a fatalError() if you put outside function

Use of Guard Statement

Block execution of Guard



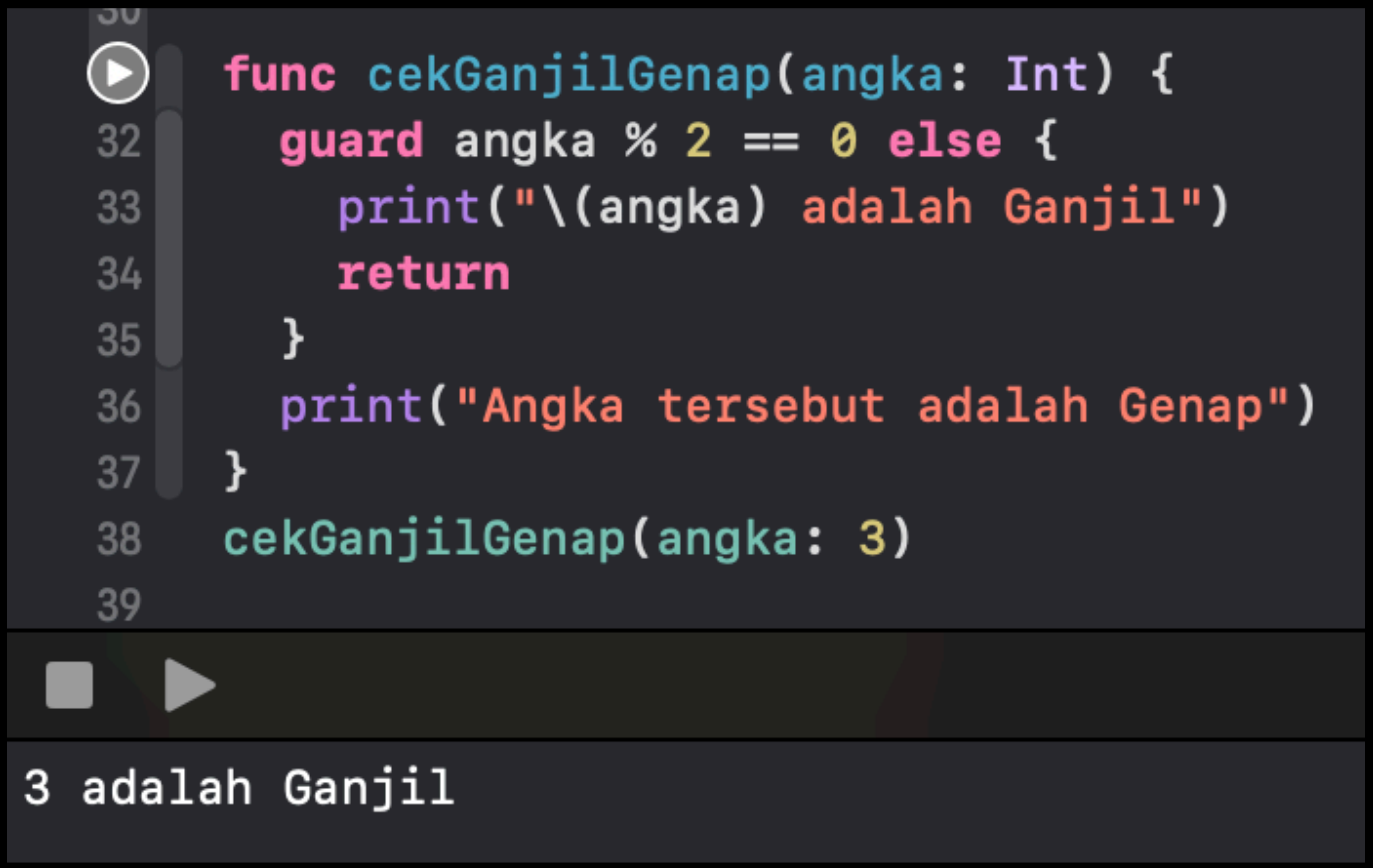
```
guard true else {
    // some code
    // some code
}

// code after guard
```

```
guard false else {
    // some code
    // some code
}

// code after guard
```

iCodeWave Community

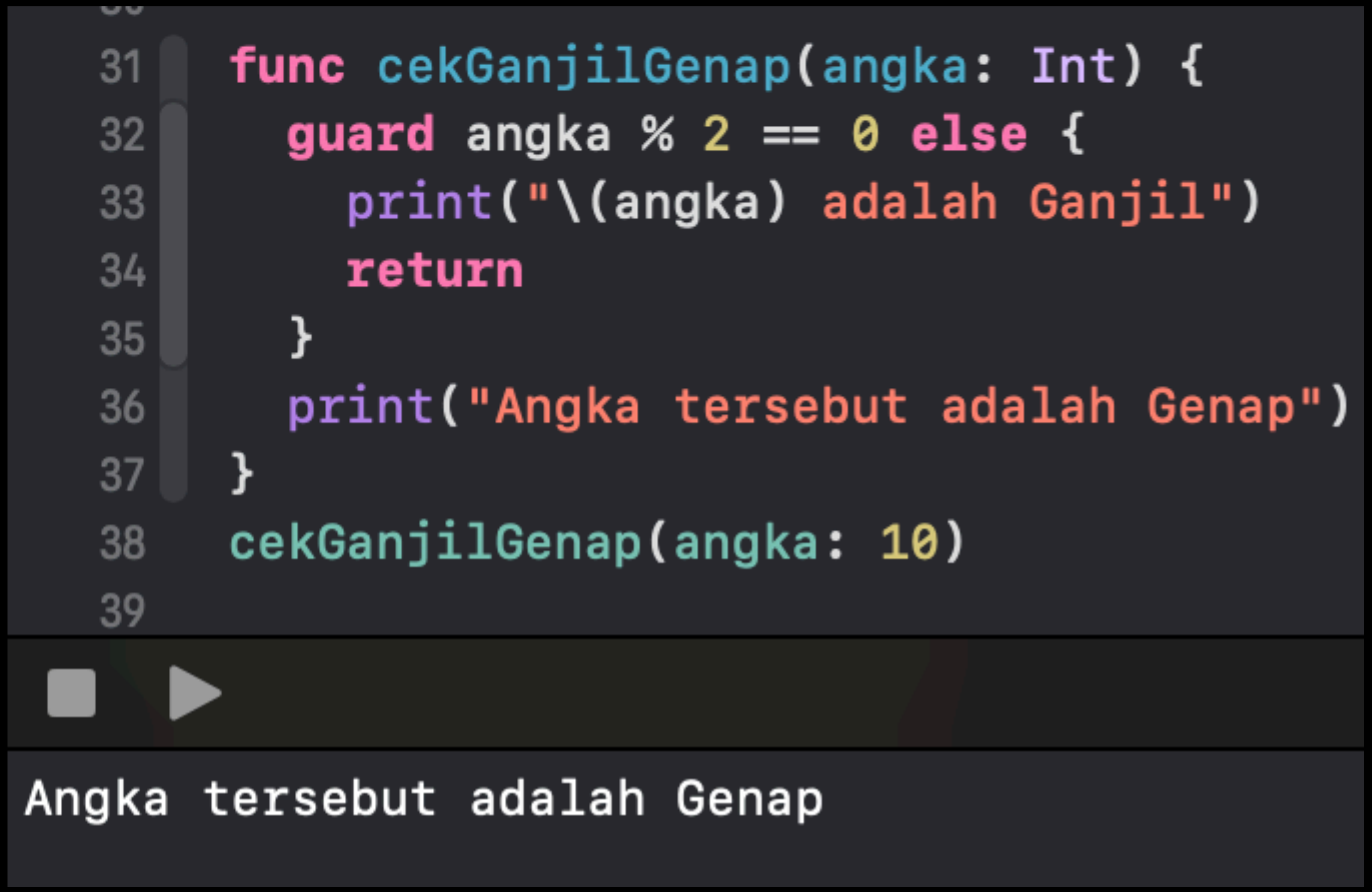
Use of Guard Statement

Example codes – Part 1



```swift
func cekGanjilGenap(angka: Int) {
  guard angka % 2 == 0 else {
    print("\(angka) adalah Ganjil")
    return
  }
  print("Angka tersebut adalah Genap")
}
cekGanjilGenap(angka: 3)
```

```
3 adalah Ganjil
```



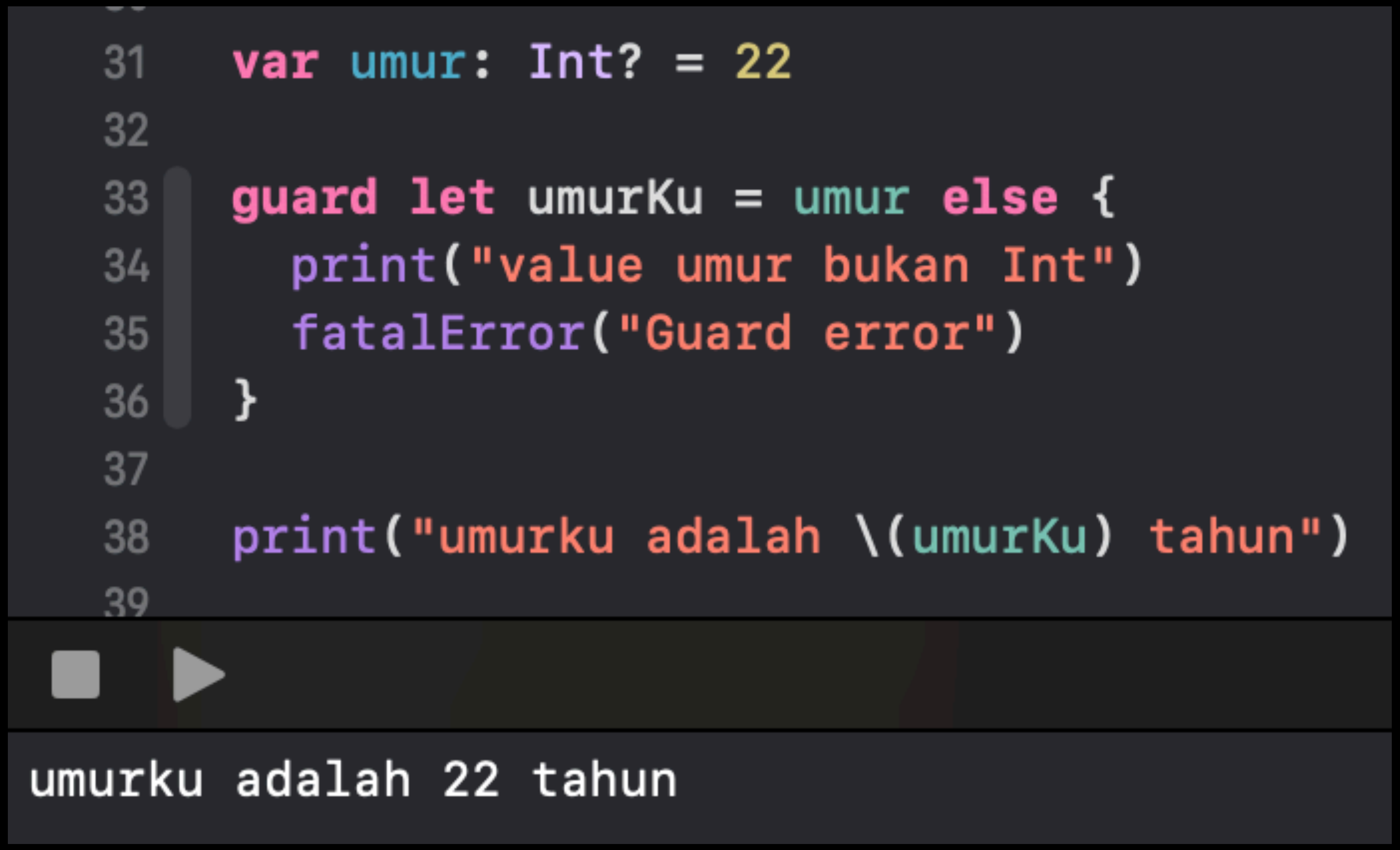
```swift
func cekGanjilGenap(angka: Int) {
  guard angka % 2 == 0 else {
    print("\(angka) adalah Ganjil")
    return
  }
  print("Angka tersebut adalah Genap")
}
cekGanjilGenap(angka: 10)
```

```
Angka tersebut adalah Genap
```

iCodeWave Community

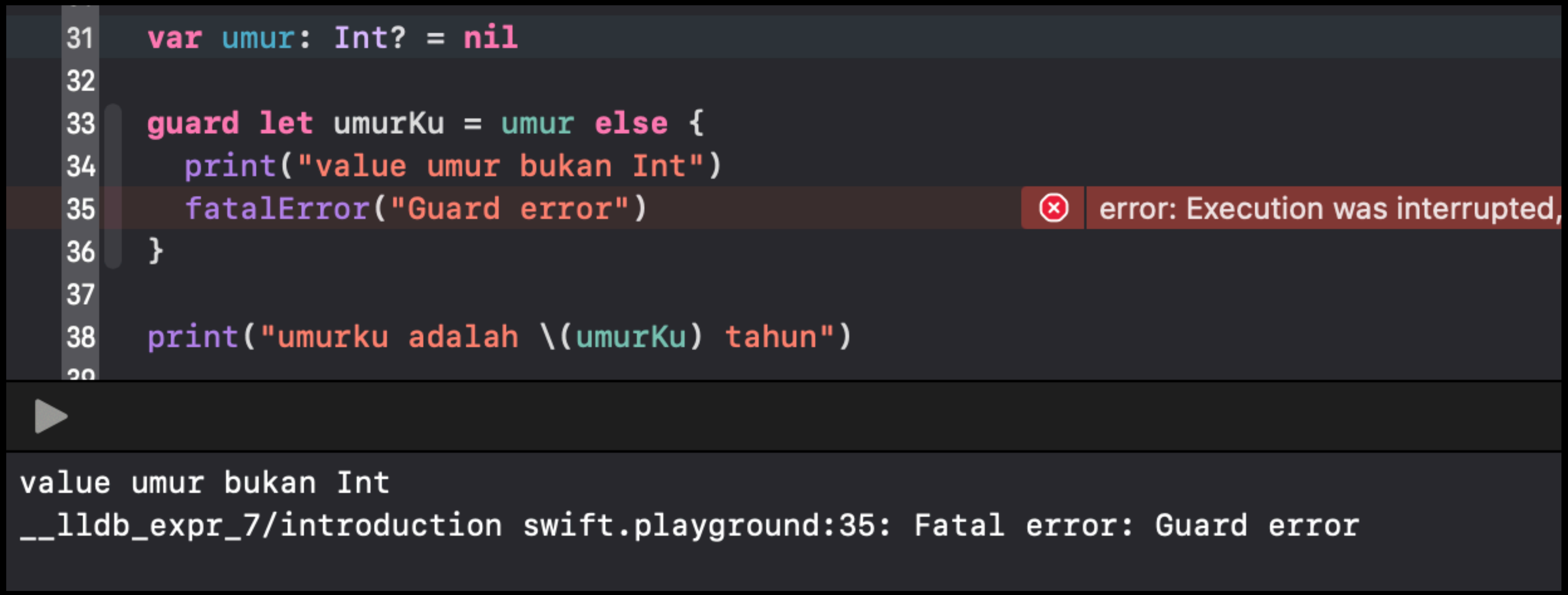
Use of Guard Statement

Example codes – Part 2



```
var umur: Int? = 22

guard let umurKu = umur else {
  print("value umur bukan Int")
  fatalError("Guard error")
}

print("umurku adalah \(umurKu) tahun")
```

```
umurku adalah 22 tahun
```



```
var umur: Int? = nil

guard let umurKu = umur else {
  print("value umur bukan Int")
  fatalError("Guard error")          error: Execution was interrupted,
}

print("umurku adalah \(umurKu) tahun")
```

```
value umur bukan Int
__lldb_expr_7/introduction swift.playground:35: Fatal error: Guard error
```

You can also use guard without function, but you must return an fatalError

iCodeWave Community

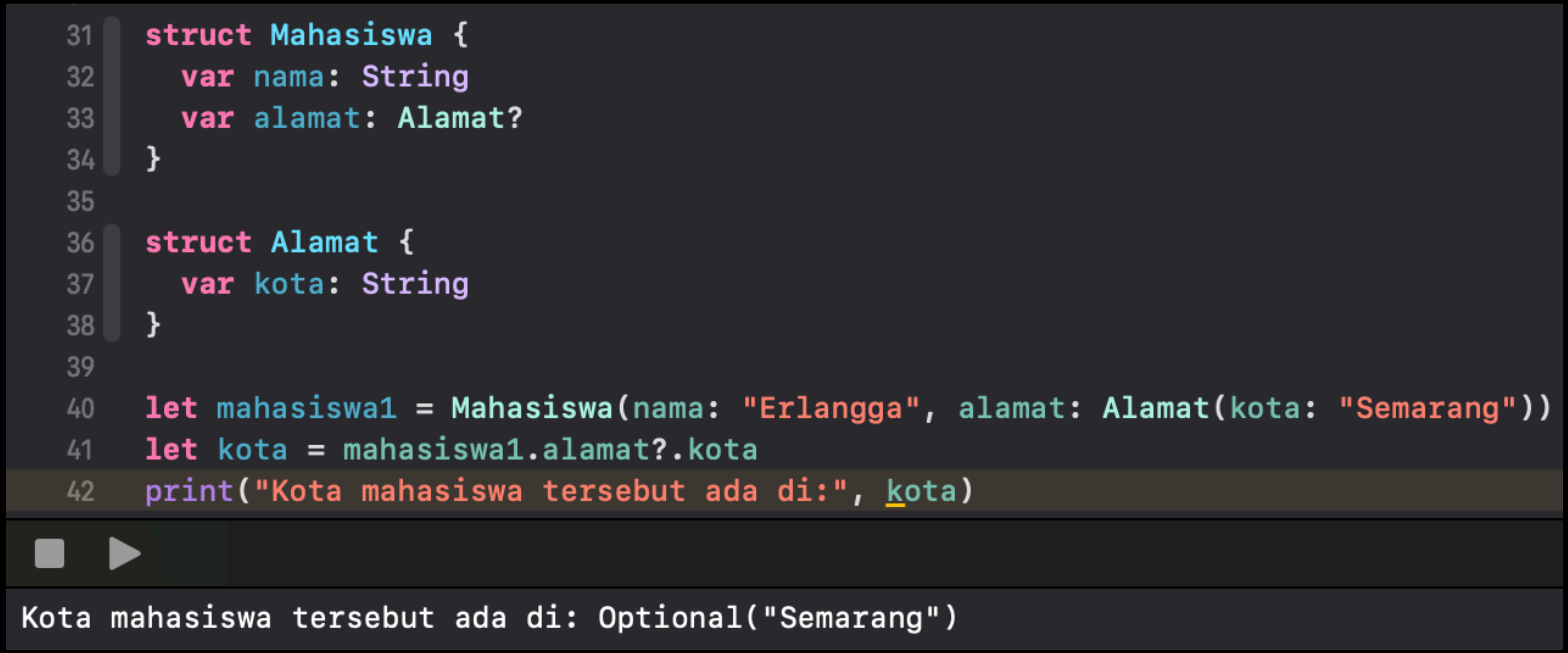
Optional Chaining

Chains like a chaining messages

- Optional chaining allows you to call methods, access properties, or call subscripts on optional that may have a value of nil
- If optional has a value of nil, the call will return nil
- Syntax example: `let orang1 = mahasiswa?.name`

Optional Chaining

Example code #1 – Simple chains

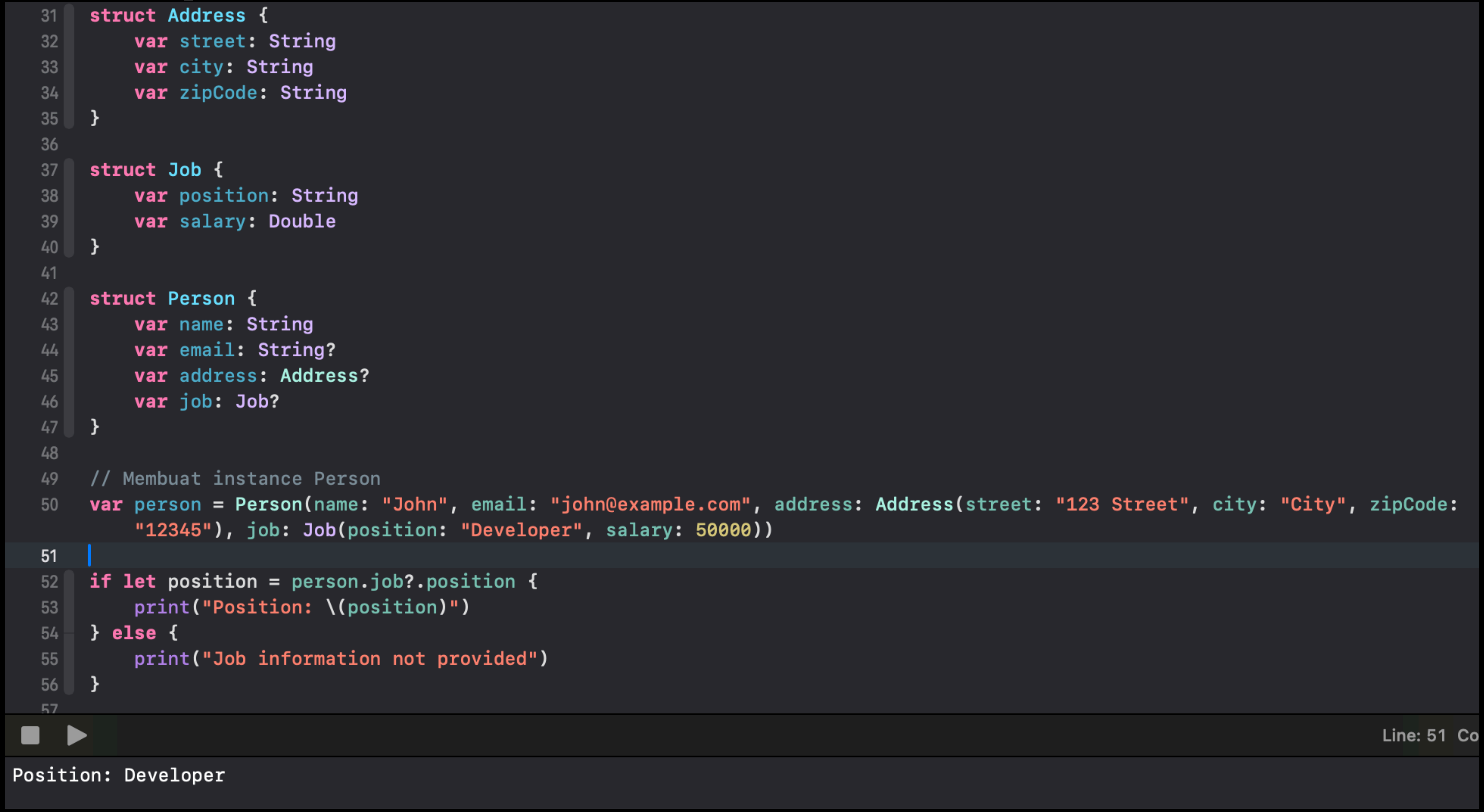


```swift
struct Mahasiswa {
  var nama: String
  var alamat: Alamat?
}

struct Alamat {
  var kota: String
}

let mahasiswa1 = Mahasiswa(nama: "Erlangga", alamat: Alamat(kota: "Semarang"))
let kota = mahasiswa1.alamat?.kota
print("Kota mahasiswa tersebut ada di:", kota)
```

```
Kota mahasiswa tersebut ada di: Optional("Semarang")
```

Optional Chaining

Example code #1 – Chain to chain to chain



```swift
struct Address {
    var street: String
    var city: String
    var zipCode: String
}

struct Job {
    var position: String
    var salary: Double
}

struct Person {
    var name: String
    var email: String?
    var address: Address?
    var job: Job?
}

// Membuat instance Person
var person = Person(name: "John", email: "john@example.com", address: Address(street: "123 Street", city: "City", zipCode:
    "12345"), job: Job(position: "Developer", salary: 50000))

if let position = person.job?.position {
    print("Position: \(position)")
} else {
    print("Job information not provided")
}
```

Line: 51 Co

```
Position: Developer
```

iCodeWave Community

Thanks For Your Attendance Today!

iCodeWave Community