Unit 3—Lesson 1: Optionals

nil

```
struct Book {
  let name: String
  let publicationYear: Int
}

let firstHarryPotter = Book(name: "Harry Potter and the Sorcerer's Stone",
                            publicationYear: 1997)
let secondHarryPotter = Book(name: "Harry Potter and the Chamber of Secrets",
                             publicationYear: 1998)

let books = [firstHarryPotter, secondHarryPotter]
```

nil

```
let unannouncedBook = Book(name: "Rebels and Lions",
                           publicationYear: ???)
```

nil

```
let unannouncedBook = Book(name: "Rebels and Lions",
                           publicationYear: 0)
```

nil

```
let unannouncedBook = Book(name: "Rebels and Lions",
                           publicationYear: 2017)
```

nil

```
let unannouncedBook = Book(name: "Rebels and Lions",
                           publicationYear: nil)
```

```
struct Book {
  let name: String
  let publicationYear: Int?
}

let firstHarryPotter = Book(name: "Harry Potter and the Sorcerer's Stone",
                            publicationYear: 1997)
let secondHarryPotter = Book(name: "Harry Potter and the Chamber of Secrets",
                             publicationYear: 1998)

let books = [firstHarryPotter, secondHarryPotter]

let unannouncedBook = Book(name: "Rebels and Lions", publicationYear: nil)
```

Specifying the type of an optional

```
var serverResponseCode = 404
```

```
var serverResponseCode = nil
```

‘nil’ requires a contextual type

```
var serverResponseCode: Int? = 404
```

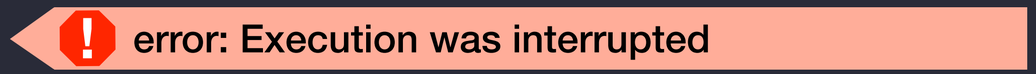
```
var serverResponseCode: Int? = nil
```

Working with optional values

Force-unwrap

```
if publicationYear != nil {
  let actualYear = publicationYear!
  print(actualYear)
}

let unwrappedYear = publicationYear!
```



Working with optional values

Optional binding

```
if let constantName = someOptional {
  //constantName has been safely unwrapped for use within the braces.
}
```

```
if let unwrappedPublicationYear = book.publicationYear {
  print("The book was published in \(unwrappedPublicationYear)")
}
else {
  print("The book does not have an official publication date.")
}
```

Functions and optionals

Return values

```
let string = "123"
let possibleNumber = Int(string)
```

```
let string = "Cynthia"
let possibleNumber = Int(string)
```

Functions and optionals

Defining

```
func printFullName(firstName: String, middleName: String?, lastName: String)
```

```
func textFromURL(url: URL) -> String?
```

Failable initializers

```
struct Toddler {
  var birthName: String
  var monthsOld: Int
}
```

Failable initializers

```
struct Toddler {
  var birthName: String
  var monthsOld: Int

  init?(birthName: String, monthsOld: Int) {
    if monthsOld < 12 || monthsOld > 36 {
      return nil
    } else {
      self.birthName = birthName
      self.monthsOld = monthsOld
    }
  }
}
```

Failable initializers

```
let possibleToddler = Toddler(birthName: "Joanna", monthsOld: 14)
if let toddler = possibleToddler {
  print("\(toddler.birthName) is \(toddler.monthsOld) months old")
} else {
  print("The age you specified for the toddler is not between 1 and 3 yrs of age")
}
```

Optional chaining

```
class Person {
  var age: Int
  var residence: Residence?
}

class Residence {
  var address: Address?
}

class Address {
  var buildingNumber: String?
  var streetName: String?
  var apartmentNumber: String?
}
```

Optional chaining

```
if let theResidence = person.residence {
  if let theAddress = theResidence.address {
    if let theApartmentNumber = theAddress.apartmentNumber {
      print("He/she lives in apartment number \(theApartmentNumber).")
    }
  }
}
```

Optional chaining

```
if let theApartmentNumber = person.residence?.address?.apartmentNumber {
  print("He/she lives in apartment number \(theApartmentNumber).")
}
```

Implicitly Unwrapped Optionals

```
class ViewController: UIViewController {
  @IBOutlet weak var label: UILabel!
}
```

Unwraps automatically

Should only be used when need to initialize an object without supplying the value and you’ll be giving the object a value soon afterwards

Unit 3, Lesson 1

Lab: Optionals.playground

Open and complete the exercises in `Lab - Optionals.playground`