PARSER COMBINATORS

SELF INTRO

» Name: Joshua Kaplan

» Interests: 🥨 λ 🍺

» Company: GMO Pepabo

» Service: minne

WHAT ARE PARSER COMBINATORS?

» Higher-order function that takes multiple parsers and /combines/ them into a new parser

» CS theory

» Parsec, Haskell

IN SWIFT

```
struct Parser<A> {
    let run: (inout Substring) throws -> A
}
```

CHARACTERISTICS

» Monadic

» Composable

» Generic

» Immutable

USE

```
let int = Parser<Int> { input in
    let num = input.prefix(while: { $0.isNumber })
    guard let number = Int(num) else { throw ParserError.IntError.notANumber(num) }

    input.removeFirst(num.count)
    return number
}
```

```
func removingLiteral(_ string: String) -> Parser<Void> {
    return Parser<Void> { input in
        guard input.hasPrefix(string) else {
          throw ParserError.StringError.literalNotFound(string[...])
        }
        input.removeFirst(string.count)
    }
}
```

HIGHER ORDER FUNCTIONS

» map

» flatMap (bind, >>=)

» zip

```
struct Coordinate { let x, y: Int }
let str = "1,2"

let coordinateParser = zip(
  int,
  removingLiteral(","),
  int
).map { x, _, y in Coordinate(x: x, y: y) }

let (coordinate, _) = try coordinateParser.run(str[...])

▽ Coordinate
  - x: 1
  - y: 2
```

```swift
func substring(while predicate: @escaping (Character) -> Bool) -> Parser<Substring> {
    return Parser<Substring> { input in
        let p = input.prefix(while: predicate)
        input.removeFirst(p.count)

        return p
    }
}
```

LET'S MAKE ANOTHER PARSER!

```
struct Person { let name: String; let age: Int }
let str = "name: John, age: 90"
```

NAME AND AGE PARSERS

```
let nameParser = zip(
  removingLiteral("name: "),
  substring(while: { $0.isLetter })
).map { _, name in return String(name) }

let ageParser = zip(
  removingLiteral("age: "),
  int
).map { _, age in return age }
```

PERSON PARSER

```
let personParser = zip(
  nameParser,
  removingLiteral(", "),
  ageParser
).map { name, _, age in return Person(name: name, age: age) }

let (person, _) = try personParser.run(str[...])
▿ Person
  - name: "John"
  - age: 90
```

COMPARISON

» By hand

» Scanner

WHY AND WHEN?