Swift

Origem da Linguagem

- Criada pela Apple para ser uma linguagem de desenvolvimento para suas plataformas
- Foco em códigos enxutos, com menos erros e com grande poder de expressão
- Seu desenvolvimento começou em julho de 2010, com seu lançamento oficial em 2014.
- Atualmente encontra-se na versão 4.0

Influências

Swift foi influenciada por diversas linguagens, entre elas podemos citar:

- Objective-C
- Rust
- Haskell
- Ruby
- Python
- C++

Classificação

Linguagem compilada e multiparadigma:

- Orientada a objetos
- Imperativa
- Funcional

Com tipagem :

- Estática
- Forte
- Com inferência de tipo

Comparação

- Foco do desenvolvimento em utilizar o que há de melhor em cada linguagem
- Códigos menores, mas que podem ser complexos de ler
- Possui diversas caracteristicas visando redução dos erros mais comuns
- Diversas expressividades particulares (Tipos opcionais e protocolos)
- Alto desempenho, principalmente com relação ao Objective-C

Códigos em Swift

```swift
var carros: String[] = ["Fusca", "Fiat 147", "Opala", "Outros"]
for Carro in carros {
   println(Carro)
}

if let typed = readLine() {
  if let num = Int(typed) {
      print(num)
  }
}
```

```swift
//Swift
func factorial(n: UInt32) -> UInt64 {
    if n == 0 {
        return 1
    } else {
        return n * factorial(n - 1)
    }
}
```

Swift vs Python

```python
# Python
name = "Felipe" # string variable
name = 42       # would run
n = 42          # currently an int
d = 42.0        # currently a float
```

```swift
// Swift
var name = "Felipe" // string
name = 42           // Error
var n = 42          // int
var d = 42.0        // double
```

```python
#Python
def compareMinMax(a,b):
    if a > b:
        return (b,a)
    else :
        return (a,b)

a,b = compareMinMax(10,20)
```

```swift
// Swift
func compareMinMax(a: Int, b: Int) -> (min: Int, max: Int) {
    if a > b {
        return (b, a)
    } else {
        return (a, b)
    }
}
var (a, b) = compareMinMax(10,20)
```

Swift vs Java

```swift
//Swift
print("Hello, world!")
```

```java
#Java
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world!"); }}
```

```swift
//Swift
class Nome {
    var name: String
    init(name: String) {
        self.name = name
    }
}

let aClass = SomeClass(name: "Felipe")
let bClass = aClass
bClass.name = "Rodrigo"
print(aClass.name) // "Rodrigo"
print(bClass.name) // "Rodrigo"
```

```java
//Java
public class SomeClass {
    public String name;
    public SomeClass(String name) {
        this.name = name; }}
public class Program {
    public static void main(String[] args) {
        final SomeClass aClass = new SomeClass("Felipe");
        final SomeClass bClass = aClass;
        aClass.favLang = "Rodrigo";
        System.out.println(aClass.name); // "Rodrigo"
        System.out.println(bClass.name); // "Rodrigo"
    }}
```

Protocolos

```
protocol Drivable {
    var topSpeed: Int { get }
}

protocol Reversible {
    var reverseSpeed: Int { get }
}

protocol Transport {
    var seatCount: Int { get }
}

struct Car: Drivable, Reversible, Transport {
    var topSpeed = 150
    var reverseSpeed = 20
    var seatCount = 5
}
```

- Escopo que define métodos e propriedades que uma classe ou estrutura deve ter
- Possui características de heranças e interfaces
- Múltiplos protocolos
- Métodos padrões e opcionais
- Programação orientada a protocolos

Tipos Opcionais

```swift
let possibleNumber:String = 123
let convertedNumber = Int(possibleNumber) // returns an optional value
if convertedNumber != nil {
    print("convertedNumber value: \(convertedNumber)")  // "convertedNumber value: Optional(123)"
    print("convertedNumber value: \(convertedNumber!)") // "convertedNumber has an integer value of 123"
}
let d:Int? = Int(possibleNumber)
let e:Int  = Int(possibleNumber)!
let f:Int! = Int(possibleNumber)

print(d) // "Optional(123)"
print(e) // "123"
print(f) // "123"
let noNumber = "hello"
let g:Int! = Int(noNumber)

print(g)  // "nil"
print(g!) // fatal error: unexpectedly found nil while unwrapping an Optional value
winSize = house?.windows?.get(2)?.size
```