View Code

Reutilizando ao máximo as suas views

Leonardo A Piovezan

Como construir views?

Storyboards

XIBs

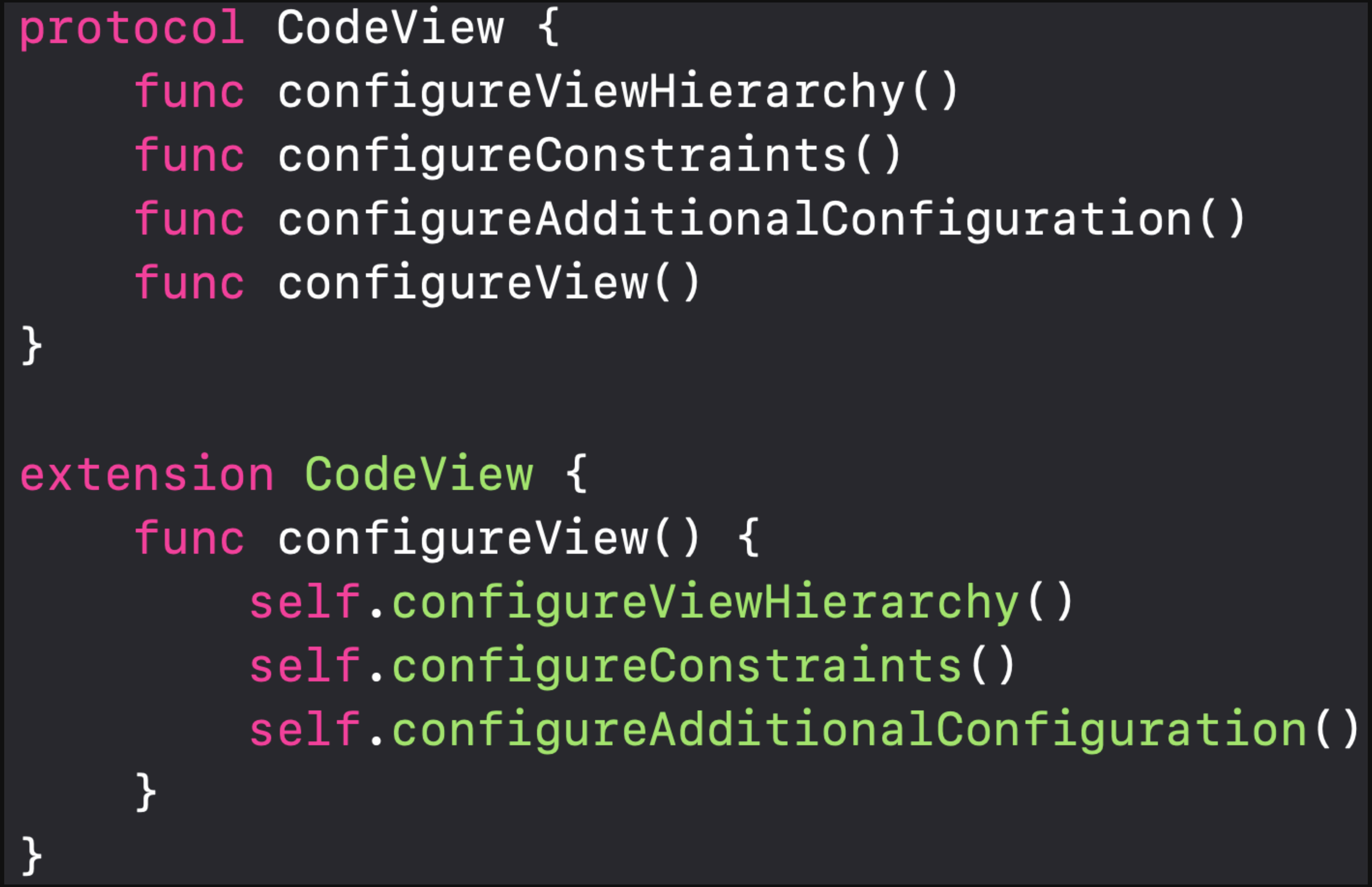
Código

Organização

Adicionar views

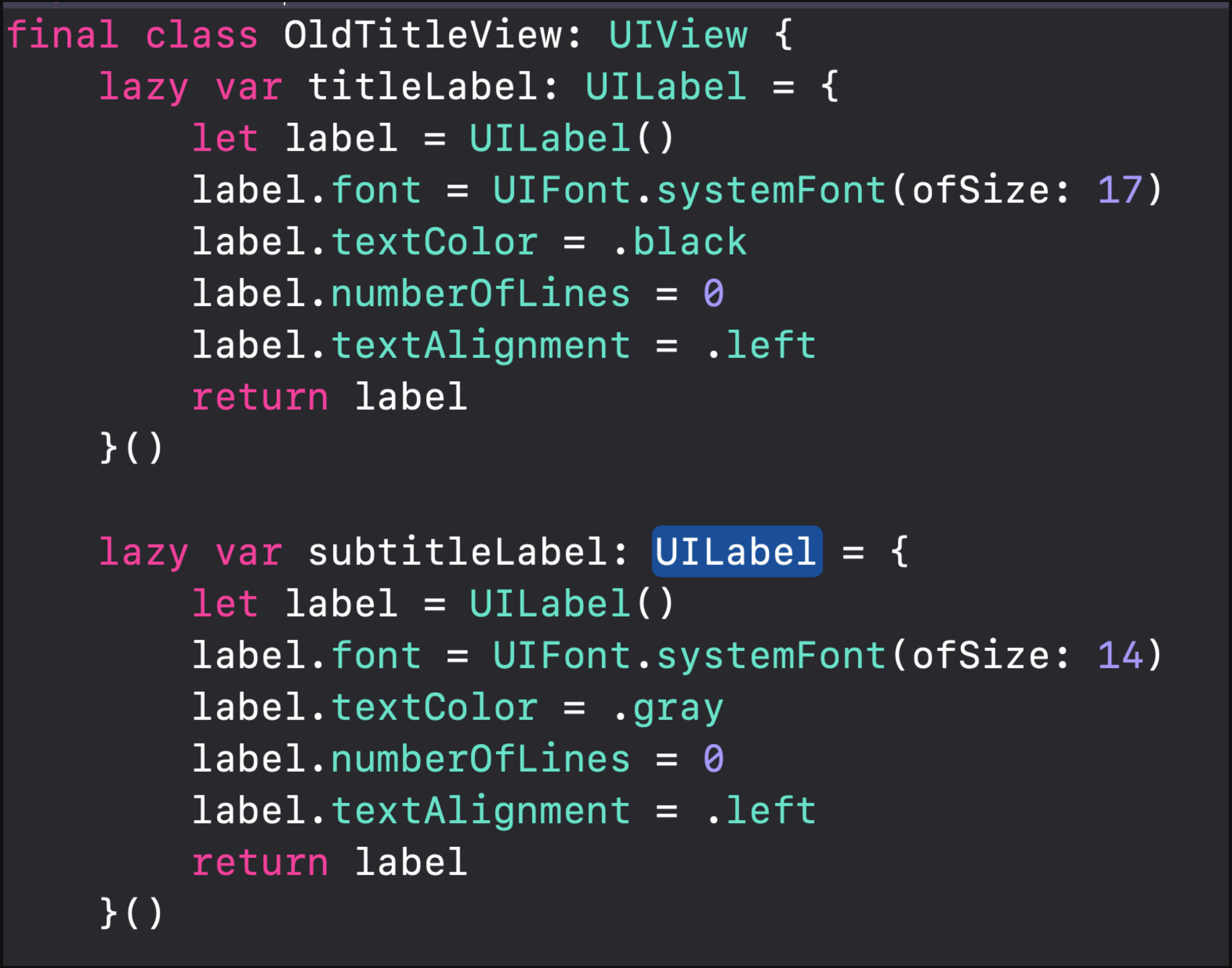
Colocar constraints

Estilizar as views



```swift
protocol CodeView {
    func configureViewHierarchy()
    func configureConstraints()
    func configureAdditionalConfiguration()
    func configureView()
}

extension CodeView {
    func configureView() {
        self.configureViewHierarchy()
        self.configureConstraints()
        self.configureAdditionalConfiguration()
    }
}
```



```swift
final class OldTitleView: UIView {
    lazy var titleLabel: UILabel = {
        let label = UILabel()
        label.font = UIFont.systemFont(ofSize: 17)
        label.textColor = .black
        label.numberOfLines = 0
        label.textAlignment = .left
        return label
    }()

    lazy var subtitleLabel: UILabel = {
        let label = UILabel()
        label.font = UIFont.systemFont(ofSize: 14)
        label.textColor = .gray
        label.numberOfLines = 0
        label.textAlignment = .left
        return label
    }()
```

```swift
extension OldTitleView: CodeView {
    func configureViewHierarchy() {
        self.addSubview(self.stackView)
        self.stackView.addArrangedSubview(self.titleLabel)
        self.stackView.addArrangedSubview(self.subtitleLabel)
    }

    func configureConstraints() {
        self.stackView.snp.makeConstraints { make in
            make.edges.equalToSuperview()
        }
    }

    func configureAdditionalConfiguration() {}
}
```

```swift
protocol LabelDescriptor {
    var font: UIFont { get }
    var textColor: UIColor { get }
    var numberOfLines: Int { get }
    var alignment: NSTextAlignment { get }
}

extension LabelDescriptor {
    var numberOfLines: Int {
        return 0
    }

    var alignment: NSTextAlignment {
        return .left
    }
}
```

```swift
enum LabelStyle {
    case title
    case subtitle
    case custom(descriptor: LabelDescriptor)

    var descriptor: LabelDescriptor {
        switch self {
        case .title:
            return TitleDescriptor()
        case .subtitle:
            return SubTitleDescriptor()
        case .custom(let descriptor):
            return descriptor
        }
    }
}
```

```swift
class Label: UILabel {
    required init(style: LabelStyle) {
        super.init(frame: CGRect.zero)
        let descriptor = style.descriptor
        self.setPropertiesWith(descriptor: descriptor)
    }

    required init?(coder aDecoder: NSCoder) {
        fatalError("init(coder:) has not been implemented")
    }

    private func setPropertiesWith(descriptor:
        LabelDescriptor) {
        self.font = descriptor.font
        self.numberOfLines = descriptor.numberOfLines
        self.textAlignment = descriptor.alignment
        self.textColor = descriptor.textColor
    }
}
```

```swift
final class LabelFactory {
    static func getLabelFor(style: LabelStyle) -> UILabel {
        return Label(style: style)
    }
}
```

```swift
final class TitleView: UIView {
    private lazy var titleLabel: UILabel = {
        return LabelFactory.getLabelFor(style: .title)
    }()

    private lazy var subtitleLabel: UILabel = {
        return LabelFactory.getLabelFor(style: .subtitle)
    }()
```

Fontes

Centralizar

Cores

Margens

```swift
struct Style {
    struct Colors {
        static let purple: UIColor = .purple
        static let blue: UIColor = .blue
        static let gray: UIColor = .gray
        static let white: UIColor = .white
    }

    struct Margins {
        static let smallMargin: CGFloat = 8
        static let mediumMargin: CGFloat = 16
        static let largeMargin: CGFloat = 24
    }

    struct Fonts {
        static func getMainFontWith(size: CGFloat) -> UIFont {
            return UIFont.systemFont(ofSize: size)
        }
    }
}
```

```swift
struct TitleDescriptor: LabelDescriptor {
    var font: UIFont = Style.Fonts.getMainFontWith(size: 20)
    var textColor: UIColor = Style.Colors.purple
}
```

```swift
struct SubTitleDescriptor: LabelDescriptor {
    var font: UIFont = Style.Fonts.getMainFontWith(size: 14)
    var textColor: UIColor = .gray
}
```

E a viewcontroller?

```swift
class CustomViewController<CustomView: UIView>:
    UIViewController {
    var customView: CustomView! {
        return self.view as? CustomView
    }

    override func loadView() {
        self.view = CustomView()
    }
}
```

```swift
final class SignUpViewController:
    CustomViewController<SignUpViewScreen> {
    override func viewDidLoad() {
        super.viewDidLoad()
    }
}
```

```swift
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?

    func application(_ application: UIApplication,
        didFinishLaunchingWithOptions launchOptions:
        [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        self.window = UIWindow(frame: UIScreen.main.bounds)
        self.window?.rootViewController =
            SignUpViewController()
        self.window?.makeKeyAndVisible()
        return true
    }
}
```

ViewCode é melhor?

linkedin.com/in/leonardo-augusto-piovezan

leonardoapiovezan@gmail.com

github.com/LeonardoPiovezan