

# CocoaPods

CocoaPods is the dependency manager for Swift and Objective-C Cocoa projects. It has almost ten thousand libraries and can help you scale your projects elegantly.

— <https://cocoapods.org>

# O que faz?

- Resolve versões
- Baixa bibliotecas de terceiros
- Integra com seu projeto

<https://cocoapods.org/about>

# Podfile

```
source 'https://github.com/CocoaPods/Specs'

platform :ios, '8.0'

inhibit_all_warnings!
use_frameworks!

target "movile-up-ios", :exclusive => true do
  pod 'Alamofire'
  pod 'Result'
  pod 'TraktModels', :git => 'https://github.com/marcelofabri/TraktModels.git'
end
```

```
target :unit_tests, :exclusive => true do
  link_with 'UnitTests'

  pod 'Nimble'
  pod 'OHHTTPStubs'
end
```

# Principais comandos

`pod install`

`pod update`

`pod update [name]`

# Alamofire

```
import Alamofire

Alamofire.request(.GET, "http://httpbin.org/get")
    .responseJSON { (_, _, JSON, _) in
        println(JSON)
    }
```

# Closures

Closures are self-contained blocks of functionality that can be passed around and used in your code. Closures in Swift are similar to blocks in C and Objective-C and to lambdas in other programming languages.

- <https://developer.apple.com/library/ios/documentation/Swift/Conceptual/SwiftProgrammingLanguage/Closures.html>



```
let names = ["Chris", "Alex", "Ewa", "Barry", "Daniella"]

func backwards(s1: String, s2: String) -> Bool {
    return s1 > s2
}

// função
var reversed = sorted(names, backwards)

// closure
reversed = sorted(names, { (s1: String, s2: String) -> Bool in
    return s1 > s2
})
```

```
// inferência de tipos  
reversed = sorted(names, { s1, s2 in return s1 > s2 } )
```

```
// return implícito  
reversed = sorted(names, { s1, s2 in s1 > s2 } )
```

```
// shorthand argument names  
reversed = sorted(names, { $0 > $1 } )
```

```
// operador  
reversed = sorted(names, >)
```

```
// trailing closure  
reversed = sorted(names) { $0 > $1 }
```

<http://fuckingclosuresyntax.com>

```
class TraktHTTPClient {  
  
    private lazy var manager: Alamofire.Manager = {  
        let configuration: NSURLSessionConfiguration = {  
            var configuration = NSURLSessionConfiguration.defaultSessionConfiguration()  
  
            var headers = Alamofire.Manager.defaultHTTPHeaders  
            headers["Accept-Encoding"] = "gzip"  
            headers["Content-Type"] = "application/json"  
            headers["trakt-api-version"] = "2"  
            headers["trakt-api-key"] = "..."  
  
            configuration.HTTPAdditionalHeaders = headers  
  
            return configuration  
        }()  
  
        return Manager(configuration: configuration)  
    }()  
}
```

```

private enum Router: URLRequestConvertible {
    static let baseURLString = "https://api-v2launch.trakt.tv/"

    case Show(String)

    // MARK: URLRequestConvertible
    var URLRequest: NSMutableURLRequest {
        let (path: String, parameters: [String: AnyObject]?, method: Alamofire.Method) = {
            switch self {
            case .Show(let id):
                return ("shows/\(id)", ["extended": "images,full"], .GET)
            }
        }()

        let URL = NSURL(string: Router.baseURLString)!
        let URLRequest = NSMutableURLRequest(URL: URL.URLByAppendingPathComponent(path))
        URLRequest.HTTPMethod = method.rawValue

        let encoding = Alamofire.ParameterEncoding.URL

        return encoding.encode(URLRequest, parameters: parameters).0
    }
}

```

```
func getShow(id: String, completion: ((Result<Show, NSError?>) -> Void)?) {  
    // ???  
}
```

```
if let show = Show.decode(json) {  
    // ...  
}
```

```
func getShow(id: String, completion: ((Result<Show, NSError?>) -> Void)?) {  
    manager.request(Router.Show(id)).validate().responseJSON { (_, _, responseObject, error) in  
        if let json = responseObject as? NSDictionary {  
  
            if let show = Show.decode(json) {  
                completion?(Result.success(show))  
            } else {  
                completion?(Result.failure(nil))  
            }  
        } else {  
            completion?(Result.failure(error))  
        }  
    }  
}
```



```
func getEpisode(showId: String, season: Int, episodeNumber: Int,  
    completion: ((Result<Episode, NSError?>) -> Void)?) {  
    // ???  
}
```

```

func getEpisode(showId: String, season: Int, episodeNumber: Int,
    completion: ((Result<Episode, NSError?>) -> Void)?) {

    manager.request(router).validate().responseJSON { (_, _, responseObject, error) in
        if let json = responseObject as? NSDictionary {

            if let episode = Episode.decode(json) {
                completion?(Result.success(episode))
            } else {
                completion?(Result.failure(nil))
            }
        } else {
            completion?(Result.failure(error))
        }
    }
}

```

Como remover essa repetição de código?

# Generics!

```
func swapTwoInts(inout a: Int, inout b: Int) {  
    let temporaryA = a  
    a = b  
    b = temporaryA  
}  
  
var someInt = 3  
var anotherInt = 107  
swapTwoInts(&someInt, &anotherInt)  
// "someInt is now 107, and anotherInt is now 3"
```

```
func swapTwoValues<T>(inout a: T, inout b: T) {  
    let temporaryA = a  
    a = b  
    b = temporaryA  
}
```

```
var someInt = 3  
var anotherInt = 107  
swapTwoValues(&someInt, &anotherInt)  
// someInt is now 107, and anotherInt is now 3
```

```
var someString = "hello"  
var anotherString = "world"  
swapTwoValues(&someString, &anotherString)  
// someString is now "world", and anotherString is now "hello"
```

```

func getJSONElement<T: JSONDecodable>(router: Router,
    completion: ((Result<T, NSError?>) -> Void)?) {

    manager.request(router).validate().responseJSON { (_, _, responseObject, error) in
        if let json = responseObject as? NSDictionary {

            if let value = T.decode(json) {
                completion?(Result.success(value))
            } else {
                completion?(Result.failure(nil))
            }
        } else {
            completion?(Result.failure(error))
        }
    }
}

```

```
func getShow(id: String, completion: ((Result<Show, NSError?>) -> Void)?) {  
    getJSONElement(Router.Show(id), completion: completion)  
}
```

```
func getEpisode(showId: String, season: Int, episodeNumber: Int,  
    completion: ((Result<Episode, NSError?>) -> Void)?) {  
  
    let router = Router.Episode(showId: showId, season: season,  
                                number: episodeNumber)  
    getJSONElement(router, completion: completion)  
}
```



```
func getPopularShows(completion: ((Result<[Show], NSError?>) -> Void)?) {  
    // ??  
}
```

```
func getSeasons(showId: String,  
    completion: ((Result<[Season], NSError?>) -> Void)?) {  
    // ??  
}
```

```
func getEpisodes(showId: String, season: Int,  
    completion: ((Result<[Episode], NSError?>) -> Void)?) {  
    // ??  
}
```

# map

```
var result = []  
for x in elements {  
    result.append(f(x))  
}
```

```
result = elements.map(f)
```

# filter

```
var result = []  
for x in elements {  
    if shouldBeIncluded(x) {  
        result.append(x)  
    }  
}
```

```
result = elements.filter(shouldBeIncluded)
```

# reduce

```
var result = initialValue
for x in elements {
    result = operation(result, x)
}
```

```
result = elements.reduce(initialValue, operation)
```

```
if let jsonArray = responseObject as? [NSDictionary] {  
    let values = jsonArray.map { T.decode($0) }.filter { $0 != nil }.map { $0! }  
    completion?(Result.success(values))  
} else {  
    completion?(Result.failure(error))  
}
```

jsonArray.map retorna [T?]

filter { \$0 != nil } remove os nils

.map { \$0! } retorna [T]

<https://www.weheartswift.com/higher-order-functions-map-filter-reduce-and-more/>

```
class ShowsViewController {  
  
    private let httpClient = TraktHTTPClient()  
    private var shows: [Show]?  
  
    func loadShows() {  
        httpClient.getPopularShows { [weak self] result in  
            if let shows = result.value {  
                println("consequiu")  
                self?.shows = shows  
                self?.collectionView.reloadData()  
            } else {  
                println("oops \(result.error)")  
            }  
        }  
    }  
}
```

# Carregamento de imagem

```
pod 'Kingfisher'
```

<https://github.com/onevc/Kingfisher/>

```
class ShowCollectionViewCell: UICollectionViewCell {

    @IBOutlet weak var posterImageView: UIImageView!
    @IBOutlet weak var nameLabel: UILabel!

    private var task: RetrieveImageTask?

    func loadShow(show: Show) {
        let placeholder = UIImage(named: "poster")
        if let url = show.poster?.fullImageURL ?? show.poster?.mediumImageURL ?? show.poster?.thumbImageURL {
            posterImageView.kf_setImageWithURL(url, placeholderImage: placeholder)
        } else {
            posterImageView.image = placeholder
        }

        nameLabel.text = show.title
    }
}
```



```
override func prepareForReuse() {  
    super.prepareForReuse()  
    task?.cancel()  
    task = nil  
    posterImageView.image = nil  
}  
}
```

```
override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject?) {  
    if segue == Segue.shows_to_show {  
        if let cell = sender as? UICollectionViewCell,  
            indexPath = collectionView.indexPathForCell(cell) {  
  
            let vc = segue.destinationViewController as! ShowViewController  
            vc.show = selectedShows[indexPath.row]  
        }  
    }  
}
```

9:41 AM100%

< Shows

House of Cards



HOUSE of CARDS

★★★★★

9.0

STORYLINE

Ruthless and cunning, Congressman Francis Underwood and his wife Claire stop at nothing to conquer everything. This wicked political drama penetrates the shadowy world of greed, sex and corruption in modern D.C.

SEASONS



Season 3

13 episodes

★★★★★


8.1

9:41 AM100%

< Shows

House of Cards

SEASONS




Season 3

13 episodes

★★★★★

8.1




Season 2

13 episodes

★★★★★

8.8



Season 1

13 episodes

★★★★★

9.0

GENRES

drama


thriller

DETAILS

9:41 AM100%


< Shows

House of Cards



★★★★★

8.1




Season 2

13 episodes

★★★★★

8.8



Season 1

13 episodes

★★★★★

9.0

GENRES

drama

thriller

DETAILS

**Network:** Netflix

**Status:** returning series

**Aired Episodes:** 39

**Started in:** 2013

**Country:** US

35

```
pod 'TagListView'  
pod 'FloatRatingView', :git => 'https://github.com/strekfus/FloatRatingView.git'  
pod 'BorderedView'  
pod 'OverlayView', :git => 'https://github.com/marcelofabri/OverlayView.git'
```