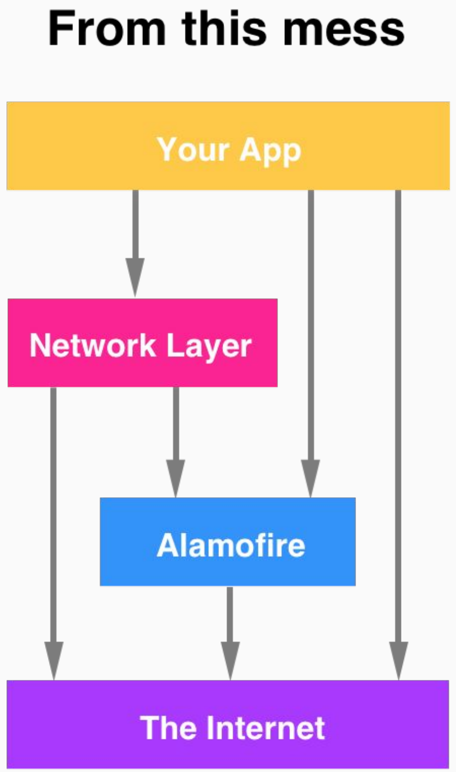
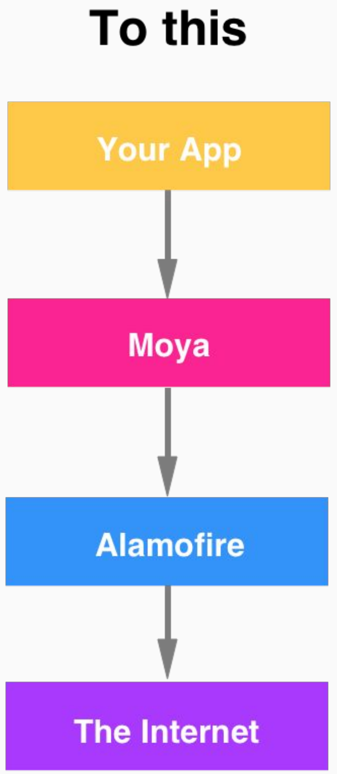
Moya

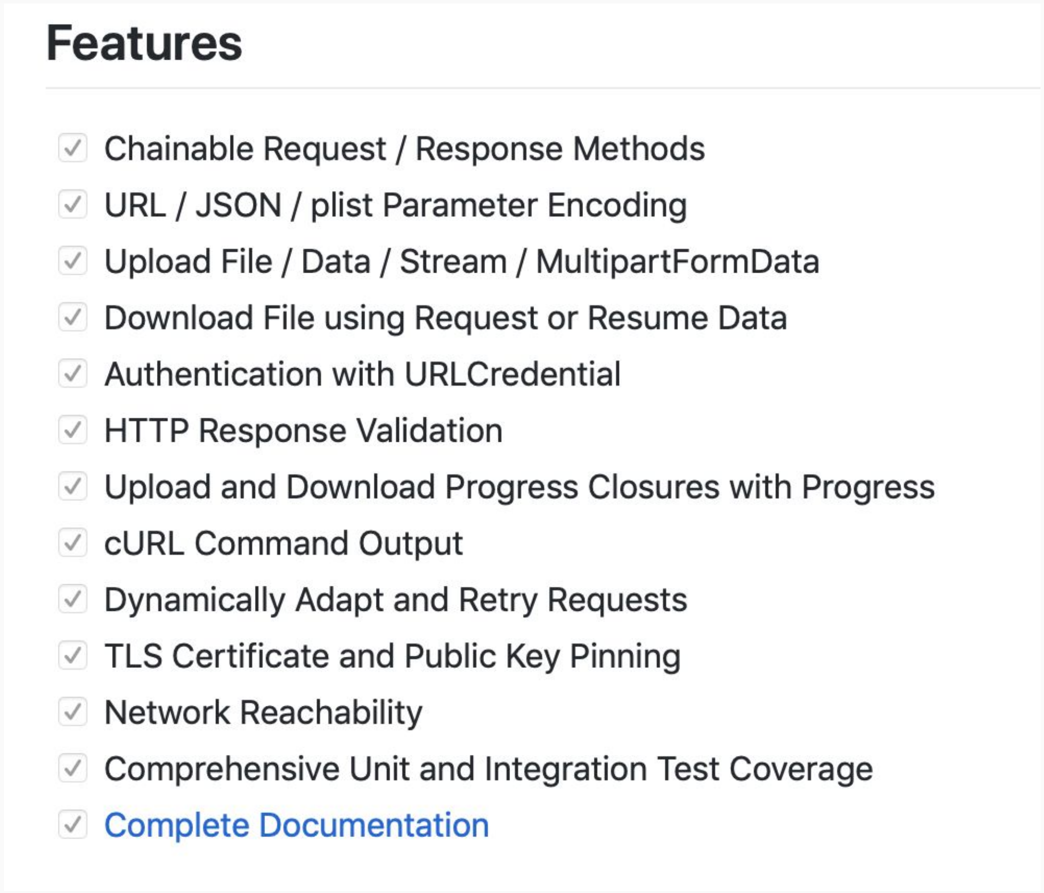
A Network Abstraction Layer

Motivation


From this mess
Your App
Network Layer
Alamofire
The Internet


To this
Your App
Moya
Alamofire
The Internet

Alamofire



Features

- ✓ Chainable Request / Response Methods
- ✓ URL / JSON / plist Parameter Encoding
- ✓ Upload File / Data / Stream / MultipartFormData
- ✓ Download File using Request or Resume Data
- ✓ Authentication with URLCredential
- ✓ HTTP Response Validation
- ✓ Upload and Download Progress Closures with Progress
- ✓ cURL Command Output
- ✓ Dynamically Adapt and Retry Requests
- ✓ TLS Certificate and Public Key Pinning
- ✓ Network Reachability
- ✓ Comprehensive Unit and Integration Test Coverage
- ✓ Complete Documentation

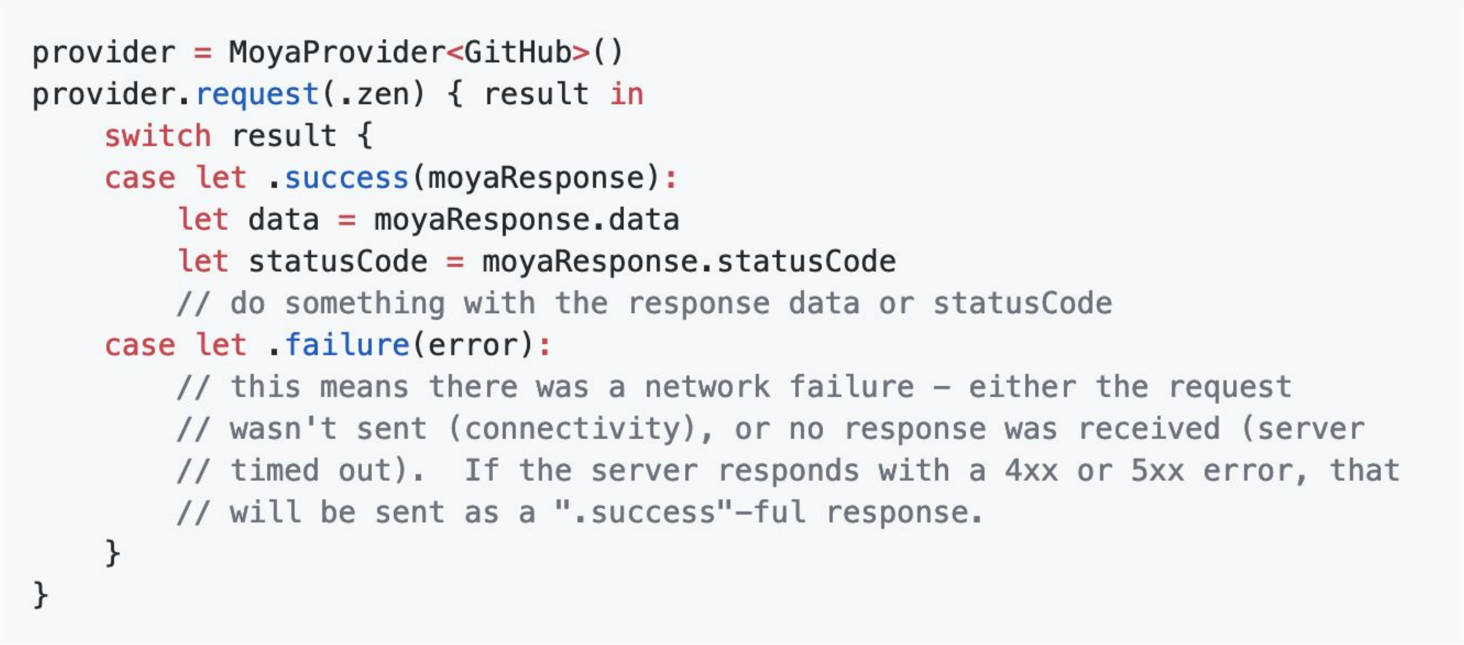
ALAMOFIRE

Elegant Networking in Swift

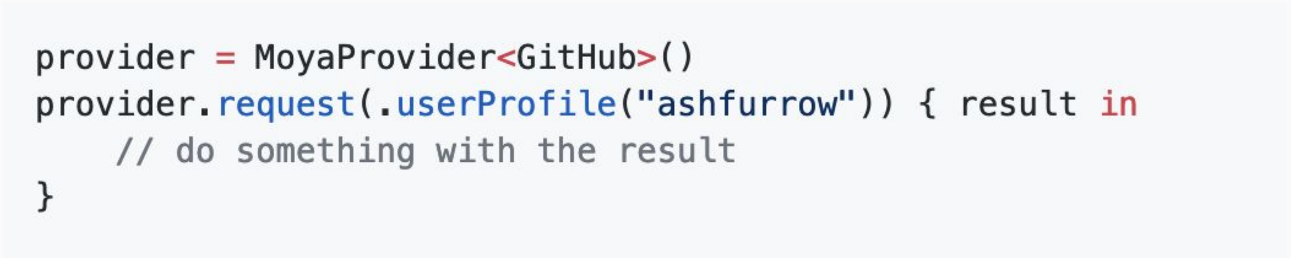
Moya

- Open Source https://moya.github.io/
- Entwickelt in Swift
- Alamofire abstrahiert URLSession, mit Moja abstrahiert man auch URLs und ihre Parameter
  - Unterstützung des Compilers bei der Bildung von URLs
  - Benutzung durch Verwendung von Enums
  - Sehr gute Testbarkeit
- Optionale Unterstützung durch ReactiveExtensions
  - ReactiveSwift
  - RxSwift
- Nutzung des ‘Codable’-Protokolls für das Mapping

Beispiel

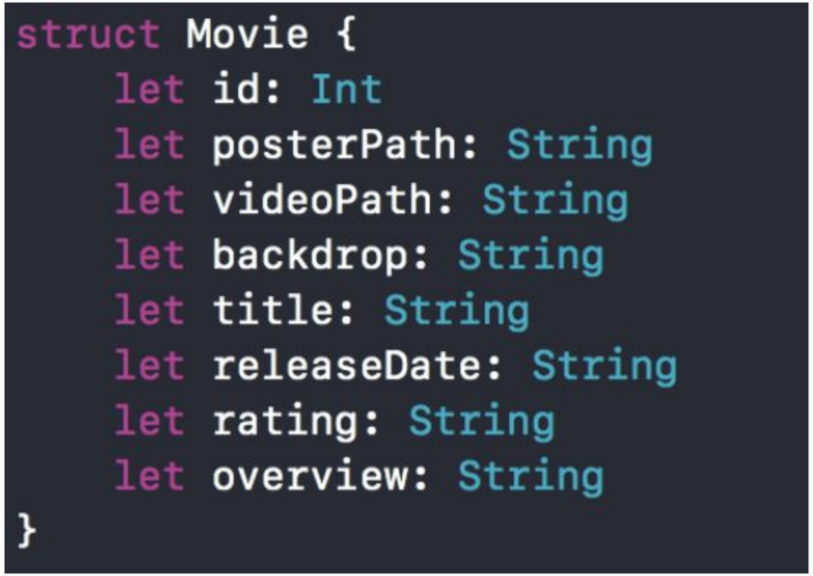


```swift
provider = MoyaProvider<GitHub>()
provider.request(.zen) { result in
    switch result {
    case let .success(moyaResponse):
        let data = moyaResponse.data
        let statusCode = moyaResponse.statusCode
        // do something with the response data or statusCode
    case let .failure(error):
        // this means there was a network failure - either the request
        // wasn't sent (connectivity), or no response was received (server
        // timed out).  If the server responds with a 4xx or 5xx error, that
        // will be sent as a ".success"-ful response.
    }
}
```



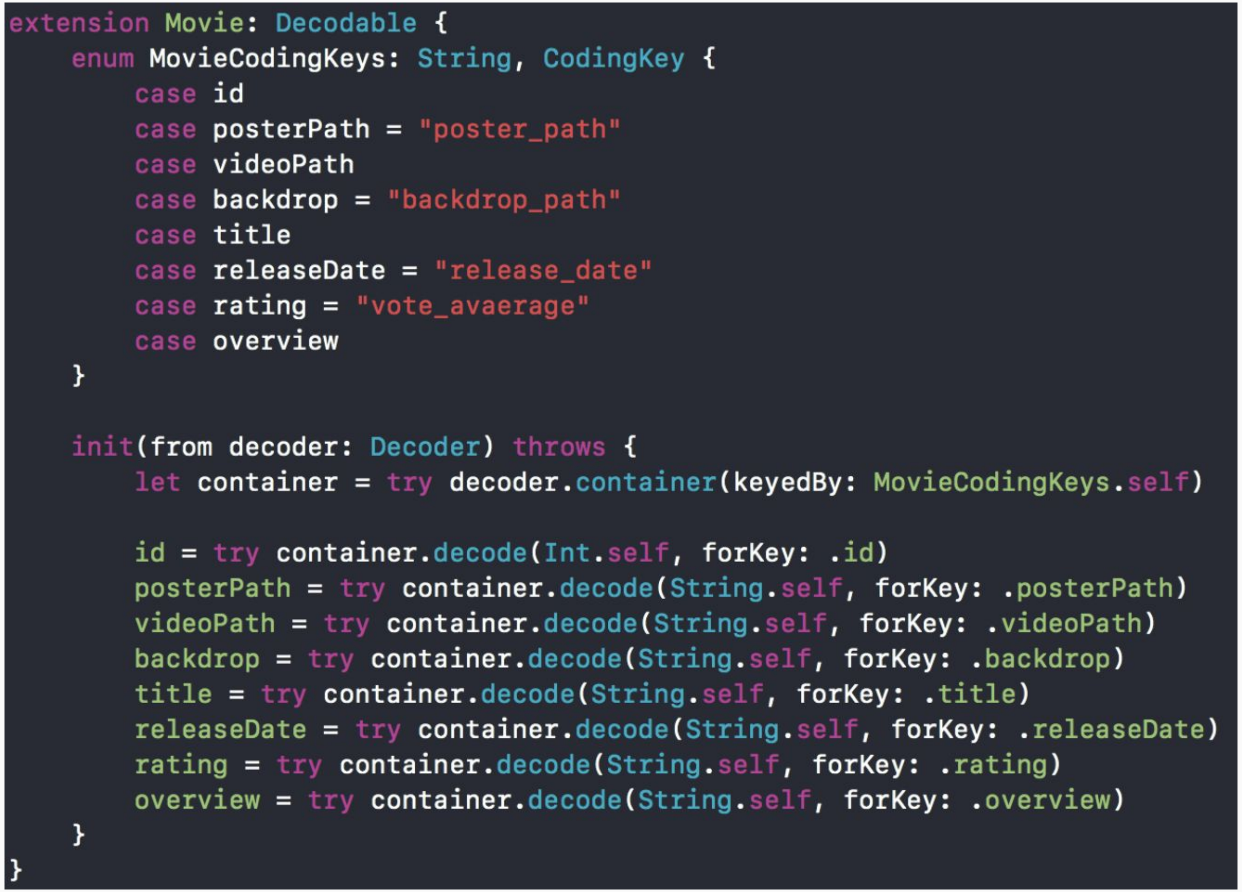
```swift
provider = MoyaProvider<GitHub>()
provider.request(.userProfile("ashfurrow")) { result in
    // do something with the result
}
```

Codable Protocol - Beispiel



```swift
struct Movie {
    let id: Int
    let posterPath: String
    let videoPath: String
    let backdrop: String
    let title: String
    let releaseDate: String
    let rating: String
    let overview: String
}
```

Codable Protocol - Beispiel



```swift
extension Movie: Decodable {
    enum MovieCodingKeys: String, CodingKey {
        case id
        case posterPath = "poster_path"
        case videoPath
        case backdrop = "backdrop_path"
        case title
        case releaseDate = "release_date"
        case rating = "vote_avaerage"
        case overview
    }

    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: MovieCodingKeys.self)

        id = try container.decode(Int.self, forKey: .id)
        posterPath = try container.decode(String.self, forKey: .posterPath)
        videoPath = try container.decode(String.self, forKey: .videoPath)
        backdrop = try container.decode(String.self, forKey: .backdrop)
        title = try container.decode(String.self, forKey: .title)
        releaseDate = try container.decode(String.self, forKey: .releaseDate)
        rating = try container.decode(String.self, forKey: .rating)
        overview = try container.decode(String.self, forKey: .overview)
    }
}
```

Nützliche Tutorials

- https://medium.com/flawless-app-stories/getting-started-with-moya-f559c406e990
- https://www.raywenderlich.com/5121-moya-tutorial-for-ios-getting-started
- https://medium.com/@vsemenchenko/writing-network-layer-with-moya-for-swift-3aa039a6e693
- https://github.com/Moya/Moya/tree/master/docs/Examples
- https://github.com/Moya/Moya/blob/master/docs/Examples/Basic.md