

Why Dependency Injection ?

Dependency injection is a form of moving the control of specifying dependencies away from classes. Away where? The idea is to move it to a piece of code that assembles the class to be *injected*.

Frameworks :

- Swinject
- Resolver
- Cleanse
- Weaver

Why Swinject ?

- Usual way requires a lot of setup code before writing a single unit test. Swinject has a less verbose method for this called autoregister, which is provided by a separate library in the Swinject organization named **SwinjectAutoregistration**.
- Swinject takes it one step further by leveraging Swift generics
- Swinject uses a popular pattern among DI frameworks: a **Dependency Injection (DI) Container**. This type of pattern keeps the resolution of your dependencies simple, even as code complexity increases
- Apps can grow big and things can get complex easily. You could end having hundreds of dependencies. For which a function to create and inject its dependency would be required.
- However by leveraging Swift generics, Swinject is capable of creating an easy way to resolve dependencies without having to create a specific function for each


dependency to do so and then calling that named function. It resolves dependency in a consistent way. It also does so by using an interface which is highly readable or *fluent*

- Luckily for us `SwinjectStoryboard` offers functionality that extends `Swinject` to do so

Without DI :

```
class ViewController: UIViewController {  
  
    lazy var requestManager: RequestManager? = RequestManager()  
  
}
```

With DI :



```
protocol Serializer {  
    func serialize(data: AnyObject) -> NSData?  
}  
  
class RequestSerializer: Serializer {  
    func serialize(data: AnyObject) -> NSData? {  
        ...  
    }  
}  
  
class DataManager {  
    var serializer: Serializer? = RequestSerializer()  
}
```

Links :

- <https://medium.com/onfido-tech/dependency-injection-in-ios-and-swift-using-swinject-and-swinjectstoryboard-e4e59a48c1ce>
- <https://github.com/Swinject/Swinject>
- <https://www.youtube.com/watch?v=sHLd1BuWoaU&t=687s>
- <https://medium.com/@JoyceMatos/dependency-injection-in-swift-87c748a167be>

Key Takeaways swinject DI :

- DependencyInjection Time consuming in development but cleaner code. (Early stages)
- Swinject is the most convenient framework if we go with DI, still has a wide range of features. Needs more time to understand the framework for better use.
- code will be coupled to the dependency injection framework we use (or more generally how you decide to implement the DI pattern)
- If taken to an extreme it can be more work than would justify the benefit
- Works better in testing, easier to do xctest.

Sample App :

<https://koenig-media.raywenderlich.com/uploads/2018/04/BitcoinAdventurer.zip>