July 20, 2018

LUNCHBOX - A LIGHTWEIGHT DEPENDENCY INJECTION FRAMEWORK WRITTEN IN SWIFT FOR iOS DEVELOPMENT

HP INC

LunchBox – A lightweight dependency injection framework written in Swift for iOS Development

LunchBox is a lightweight dependency injection framework written in Swift. It helps manage dependencies by providing a centralized location for instantiating new objects and removing much of the boilerplate code around instantiation by taking advantage of generics in Swift.

Dependency injection is a common topic of discussion in programming but utilizing it in an efficient and readable way can be challenging. Many frameworks have been developed for using dependency injection to its fullest potential across most programming languages, though the options for Swift and iOS development are limited. Additionally, these options can be overly complex with unnecessary overhead or reduced readability.

The largest advantage LunchBox has is its simple and readable design. It's able to reduce the amount of actual code needed to integrate dependency injection into the project by utilizing generics and type-inference to assume the types of needed objects without specification. This allows for less boilerplate code in factory methods and easier project maintenance should constructor signatures change. It is also extremely lightweight, making it easy to understand, learn, and modify. Additionally, it is largely self-contained, allowing for easy integration or removal.

LunchBox works by associating types with factory methods and storing these pairs in a 'Box'. Each factory receives an optional list of parameters as well as a reference back to the box from which it can ask for other objects. From this, LunchBox can construct the object and store it for future reference. It utilizes a depth-first search to ensure no circular dependencies and handles such errors cleanly by utilizing optionals in Swift. Additionally, the framework provides an 'Injectable' class to abstract the box and provide clean inline dependency injection, giving it a similar look and feel to other dependency injection frameworks that may be more familiar.

Disclosed by Grey Leman, HP Inc.