Dependency Inject trong C#

|  |
| --- |
| Dependency Injection (DI) Là một design pattern được [ASP.Net](http://asp.net/) hỗ trợ. Đây là một kỹ thuật để hiện thực hóa Inversion of Control Pattern (có thể coi nó là một design pattern riêng cũng được). Các module phụ thuộc (dependency) sẽ được inject vào module cấp cao. Có thể hiểu 1 cách đơn giản như sau:   * Các module không giao tiếp trực tiếp với nhau, mà thông qua interface. Module cấp thấp sẽ implement interface, module cấp cao sẽ gọi module cấp thấp thông qua interface. * Ví dụ: Để giao tiếp với database, ta có interface IDatabase, các module cấp thấp là XMLDatabase, SQLDatabase. Module cấp cao là CustomerBusiness sẽ chỉ sử dụng interface IDatabase. * Việc khởi tạo các module cấp thấp sẽ do DI Container thực hiện. Ví dụ: Trong module CustomerBusiness, ta sẽ không khởi tạo IDatabase db = new XMLDatabase(), việc này sẽ do DI Container thực hiện. Module CustomerBusiness sẽ không biết gì về module XMLDatabase hay SQLDatabase. * Việc Module nào gắn với interface nào sẽ được config trong code hoặc trong file XML. * DI được dùng để làm giảm sự phụ thuộc giữa các module, dễ dàng hơn trong việc thay đổi module, bảo trì code và testing.   => Đó là lý do tại sao cần sử dụng Dependency Injection |

|  |
| --- |
| Mô tả :  Ví dụ: ta có 1 Class A cần 1 dependency thì cơ chế đó sẽ tự động Inject (bơm) vào Class A  Để có thể thiết kế được cơ chế như thế thì khá là phức tạp nên hiện nay có rất nhiều Library hỗ trợ **Dependency Inject**  **Yêu cầu :** khi viết class ta phải thiết kế làm sao có thể đưa 1 Dependency vào 1 Class (gọi là đưa các đối tượng ngoài vào lớp ) |

**1. Dịch vụ là gì?**

Trong lập trình, **dịch vụ** có thể hiểu là một lớp (class) thực hiện một nhiệm vụ cụ thể nào đó. Ví dụ:

* Dịch vụ gửi email (EmailService).
* Dịch vụ xử lý dữ liệu người dùng (UserService).
* Dịch vụ ghi log (LoggingService).

Dịch vụ này thường được dùng xuyên suốt ứng dụng, thay vì tạo đối tượng mỗi khi cần, bạn sẽ dùng **Dependency Injection** để quản lý và cung cấp nó.