دانشکده مهندسی کامپیوتر

دانشگاه اصفهان

تکلیف چهارم برنامه‌نویسی دستگاه‌های سیار

استاد درس:

نوید شیرمحمدی

مهروسادات نوحی

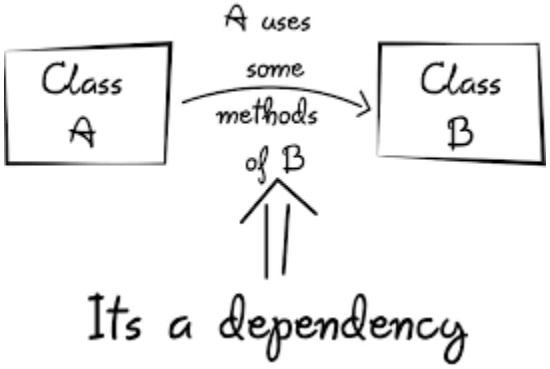
۹۹۳۶۱۳۰۶۱

نیم‌سال دوم تحصیلی ۰۳-۰۲

Dependency Injection (DI)

تزریق وابستگی (Dependency Injection) یک الگوی طراحی است که به وسیله آن، کنترل ایجاد و مدیریت وابستگی‌ها از داخل کلاس‌ها به یک منبع خارجی منتقل می‌شود.به عبارت دیگر نوعی تکنیک کدنویسی است که در آن وابستگی‌ها توسط یک موجودیت خارجی (معمولا به عنوان پارامتر یا مرجع) وارد می‌شوند، به جای این که در یک ماژول قرار بگیرند. این وابستگی‌ها اشیا یا سرویس‌هایی هستند که یک ماژول می‌تواند از آن‌ها استفاده کند. این روش باعث کاهش وابستگی بین کلاس‌ها و افزایش قابلیت تست و نگهداری کد می‌شود.

وقتی کلاس A از برخی عملکردهای کلاس B استفاده می‌کند، گفته می‌شود که کلاس A وابستگی کلاس B دارد.در جاوا، قبل از اینکه بتوانیم از متدهای کلاس‌های دیگر استفاده کنیم، ابتدا باید شیء آن کلاس را ایجاد کنیم یعنی کلاس A باید نمونه‌ای از کلاس B ایجاد کند.

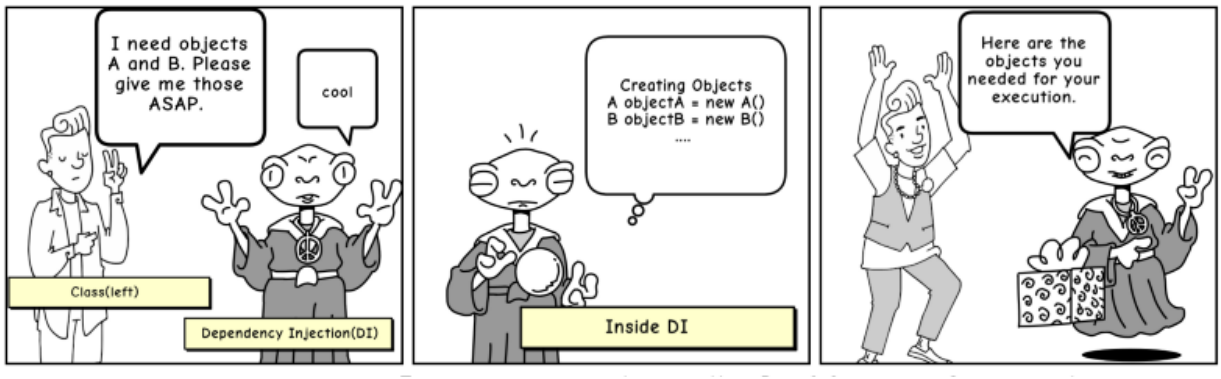


در اینجا یکی از این مشکلات این هست که کلاس‌ها وابستگی شدید بهم پیدا می‌کنند و جدا کردنشان سخت می‌شود. یعنی کلاس‌ها از همدیگه مستقل نیستند.

همچنین تست کردن برنامه هم سخت میشود. فرض کنیم کلاس X رو می‌خوایم تست کنیم، ولی این کلاس به کلاس Y وابسته هست .Y کلاسی هست که عملیات سنگینی انجام میده. مثلاً عملیات اتصال به دیسک یا یک سرویس خارجی. بنابراین اگه اتصال به سرویس خارجی میسر نباشه، ما نمی‌تونیم برنامه رو تست کنیم!

باDI ، خود شی دیگه مسئول فراهم کردن وابستگی‌هایش نیست و وابستگی‌ها از بیرون تزریق می‌شود. این تزریق در زمان Run Time اعمال می‌شود. یعنی زمانی که کد داره اجرا میشه.

بنابراین، به انتقال وظیفه ایجاد شی به شخص دیگری و استفاده مستقیم از وابستگی، تزریق وابستگی می گویند.



اصول و منطق تزریق وابستگی

- Inversion of Control (IoC)

مفهوم IoC به معنای واگذاری کنترل ایجاد و مدیریت وابستگی‌ها به یک فریمورک خارجی است. در اینجا، تزریق وابستگی یکی از راه‌های پیاده‌سازی IoC است. به عبارت دیگر، به جای اینکه یک کلاس خودش وابستگی‌هایش را ایجاد و مدیریت کند، این کار به یک فریمورک یا سپرده می‌شود.

- کاهش پیوستگی(Decoupling)

وقتی که کلاس‌ها به جای ایجاد وابستگی‌های خودشان، آن‌ها را از بیرون دریافت می‌کنند، پیوستگی بین کلاس‌ها کاهش می‌یابد. این کار باعث می‌شود که تغییر در یک کلاس، نیازی به تغییر در کلاس‌های دیگر نداشته باشد.

- قابلیت تست بالا(Testability)

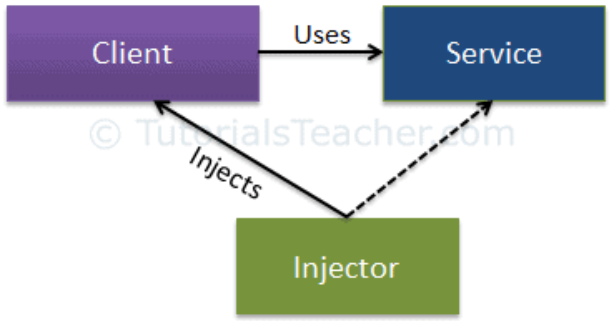
وقتی که وابستگی‌ها از بیرون تزریق می‌شوند، می‌توانیم در تست‌ها از نسخه‌های شبیه‌سازی شده (mock) این وابستگی‌ها استفاده کنیم. این کار باعث می‌شود که تست‌ها دقیق‌تر و مستقل‌تر از وابستگی‌های واقعی باشند.

- مدیریت ساده‌تر وابستگی‌ها

با استفاده از تزریق وابستگی، مدیریت و پیکربندی وابستگی‌ها به صورت مرکزی و خارج از کلاس‌ها انجام می‌شود. این کار باعث می‌شود که بتوانیم به راحتی وابستگی‌ها را تغییر دهیم یا پیکربندی کنیم.

الگوی تزریق وابستگی شامل ۳ نوع کلاس است:

1. **Client Class:** The client class (dependent class) is a class which depends on the service class
2. **Service Class:** The service class (dependency) is a class that provides service to the client class.
3. **Injector Class:** The injector class injects the service class object into the client class.



انواع تزریق وابستگی

همانطور که در بالا مشاهده کردید، کلاس injector سرویس (وابستگی) را به مشتری (وابسته) تزریق می کند. کلاس injector وابستگی ها را به طور گسترده به سه روش تزریق می کند: از طریق سازنده، از طریق یک ویژگی یا از طریق یک روش.

> **Constructor Injection:** In the constructor injection, the injector supplies the service (dependency) through the client class constructor.
>
> **Property Injection:** In the property injection (aka the Setter Injection), the injector supplies the dependency through a public property of the client class.
>
> **Method Injection:** In this type of injection, the client class implements an interface which declares the method(s) to supply the dependency and the injector uses this interface to supply the dependency to the client class.

۱. تزریق از طریق سازنده(Constructor Injection)

در این روش، وابستگی‌ها از طریق سازنده کلاس تزریق می‌شوند. این روش به دلیل اجباری بودن تزریق وابستگی‌ها و جلوگیری از ایجاد شیء بدون وابستگی‌ها، از محبوبیت بیشتری برخوردار است.

۲. تزریق از طریق متد(Setter Injection)

در این روش، وابستگی‌ها از طریق متدهای setter به کلاس تزریق می‌شوند. این روش انعطاف‌پذیری بیشتری دارد و امکان تغییر وابستگی‌ها در زمان اجرا را فراهم می‌کند.

۳. تزریق مستقیم به فیلدها(Field Injection)

در این روش، وابستگی‌ها به طور مستقیم به فیلدهای کلاس تزریق می‌شوند. این روش معمولاً با استفاده از annotations انجام می‌شود و کد را مختصرتر می‌کند.

مزایا:

۱. در تست واحد کمک می‌کند.
۲. گسترش برنامه آسان تر می‌شود.
۳. همچنین منجر به افزایش انعطاف‌پذیری می‌شود.
۴. در نهایت، تزریق وابستگی امکان توسعه همزمان را فراهم می‌کند. دو توسعه‌دهنده می‌توانند به‌طور مستقل کلاس‌هایی را توسعه دهند که از یکدیگر استفاده می‌کنند، در حالی که فقط نیاز به دانستن رابطی دارند که کلاس‌ها از طریق آن ارتباط برقرار می‌کنند.

معایب:

۱. مشتریانی را ایجاد می‌کند که جزئیات پیکربندی را می‌خواهند، که در صورت در دسترس بودن پیش‌فرض‌های آشکار می‌تواند سخت باشد.
۲. ردیابی کد را دشوار می‌کند زیرا رفتار را از ساختار جدا می‌کند.
۳. معمولاً با بازتاب یا برنامه‌نویسی پویا اجرا می‌شود که مانع اتوماسیون IDE می‌شود.
۴. معمولاً به تلاش‌های اولیه بیشتری برای توسعه نیاز دارد.

فریمورک‌های تزریق وابستگی

- Spring (Java)
- Google Guice (Java)
- Dagger (Java and Android)
- Castle Windsor (.NET)
- Unity(.NET)