LU09b - IoC (Inversion of Control) & DI (Dependency Injection)

Beim Prinzip IoC geht es darum, einem Framework o. Ä. von Aussen Teile des eigenen Codes zu kontrollieren.

Es gibt verschiedene Entwurfsmuster, die dafür geeignet sind. Unter anderem:

- Strategie-Entwurfsmuster (Strategy design pattern)
- Service-Locator (Service locator)
- Fabrikmethode (Factory method)
- Dependency Injection

Bei Dependency Injection wird die Initialisierung eines Objekts nicht im abhängigen Objekt sondern zum Beispiel mittels zentralen Konfiguration ausgelagert.

Spring Boot erlaubt folgende Arten von Injection für Beans-Klassen (@Component, @Service, @Repository, @Controller, ...)

Field Injection

```
public class ProjectApiController implements ProjectApi {

    @Autowired
    private final JwtUtil jwtUtil;
    ...
}
```

Anmerkungen:

- Mühsamer zu testen
- Felder können nicht zugleich `final` sein

Constructor Injection

```
public class ProjectApiController implements ProjectApi {

    private final JwtUtil jwtUtil;

    @Autowired
    public ProjectApiController(JwtUtil jwtUtil) {
        this.jwtUtil = jwtUtil;
    }
    ...
}
```

Anmerkungen:

- `@Autowired` ist optional, sofern es nur einen Konstruktor gibt.

Setter Injection

```
public class ProjectApiController implements ProjectApi {

    @Autowired
    private final JwtUtil jwtUtil;
    ...
}
```

Anmerkungen:

- Falls das Feld veränderbar oder optional ist

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/de/modul/ffit/3-jahr/java/learningunits/lu09/b

Last update: **2025/11/04 01:35**