Development Process for constructor injection

1. Define the dependency interface and class

```java
package com.flynaut.constructorInjectionPro;

public interface Coach {
    String getDailyWorkout();
}
```

```java
package com.flynaut.constructorInjectionPro;

import org.springframework.stereotype.Component;

@Component
public class CricketCoach implements Coach{

    @Override
    public String getDailyWorkout(){
        return "Practice!!!!";
    }
}
```

2. Create a DemoController

```java
package com.flynaut.constructorInjectionPro;

import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.RestController;

@RestController
public class DemoController {

    private Coach myCoach;

    public DemoController(Coach theCoach){
        myCoach=theCoach;
    }

    @GetMapping("/dailyWorkout")
    public String getDailyWorkout(){
        return myCoach.getDailyWorkout();
    }
}
```

@Component Annotation

- Marks the class as Spring Bean
- A spring bean is just a class which is managed by the SpringContainer
- Also makes the bean available for Dependency Injection

CI Behind the Scenes:

How spring will process our application?

BTS Spring will create an instance of our Coach class

How?

```
Coach theCoach = new CricketCoach();
DemoController demoController = new
DemoController(theCoach);
```

& this is how constructor injection occurs.

Spring is more than just IOC and DI.

It provides features like

1. REST APIs
2. Security
3. Database interactions or transactions

- Component Scanning
  SC scans for the component classes
  Spring will scan all java classes with annotation @Component.

```
@SpringBootApplication
```

Enables:

- Auto Configuration
- Component Scanning
- Additional Configurations

It is composed of following annotations:

@EnableAutoConfiguration -> Enables SB’ s Auto Configuration Support

@ComponentScan -> Enables the component scanning

@Configuration -> able to register some extra beans with @Bean

BTS:

Creates application context & registers all beans

Starts the embedded server

Setter Injection

Inject dependencies by calling setter methods in our class

```
package com.flynaut.constructorInjectionPro;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class DemoController {
    // defining the private field for the dependency
    private Coach myCoach;

    @Autowired
    public void setCoach(Coach theCoach){
        myCoach=theCoach;
    }

    @GetMapping("/dailyWorkout")
    public String getDailyWorkout(){
        return myCoach.getDailyWorkout();
    }
}
```

BTS of Setter Injection:

Coach theCoach = new CricketCoach();

DemoController demoController = new DemoController();

demoController.setCoach(theCoach);

Field Injection:

```java
package com.flynaut.constructorInjectionPro;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class DemoController {
    // defining the private field for the dependency
    //Field Injection
    @Autowired
    private Coach myCoach;


    @GetMapping("/dailyWorkout")
    public String getDailyWorkout(){
        return myCoach.getDailyWorkout();
    }
}
```

Bean Scopes

Scope is nothing but the lifecycle of a bean

Like

1. How long does the bean live?
2. How many instances are going to be created?

Default bean scope is singleton.

What is singleton?

- **SC creates only one instance of the bean by default.**

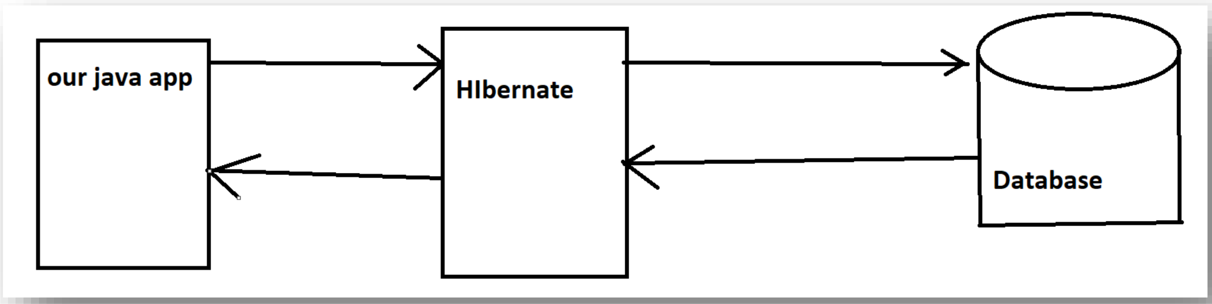
Prototype*

Request*, Session*

Hibernate/JPA

What is hibernate?

- A framework used for saving/persisting java objects in a database
- We can also retrieve data from database.

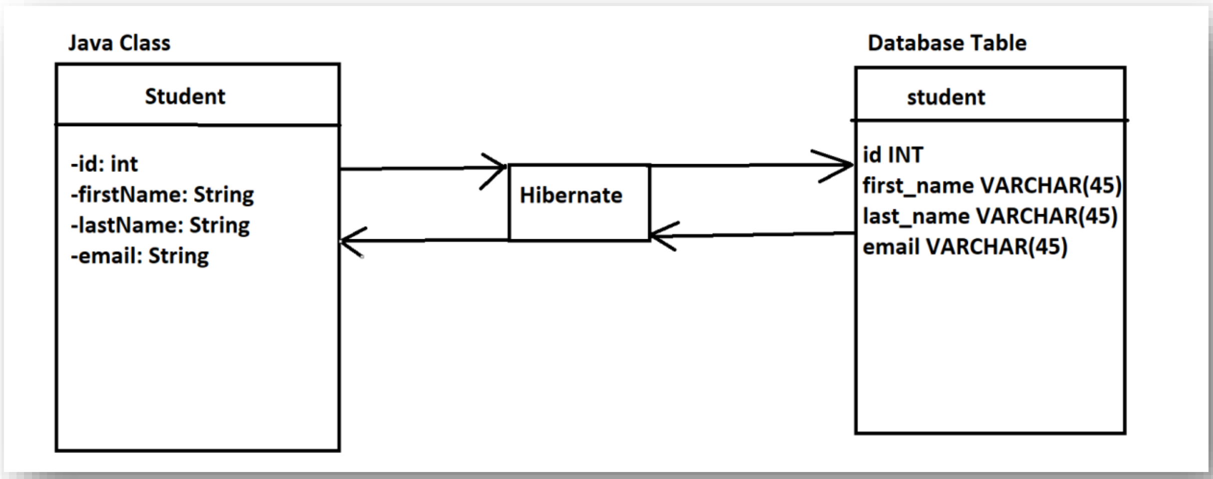


Advantages :

1. It handles all the low level sql code
2. Minimizes the amount of JDBC code we have to develop
3. Hibernate also provides the Object-to-relational mapping(ORM)

ORM

- As a developer all we need to do is tell hibernate how our java class or object maps to the database.

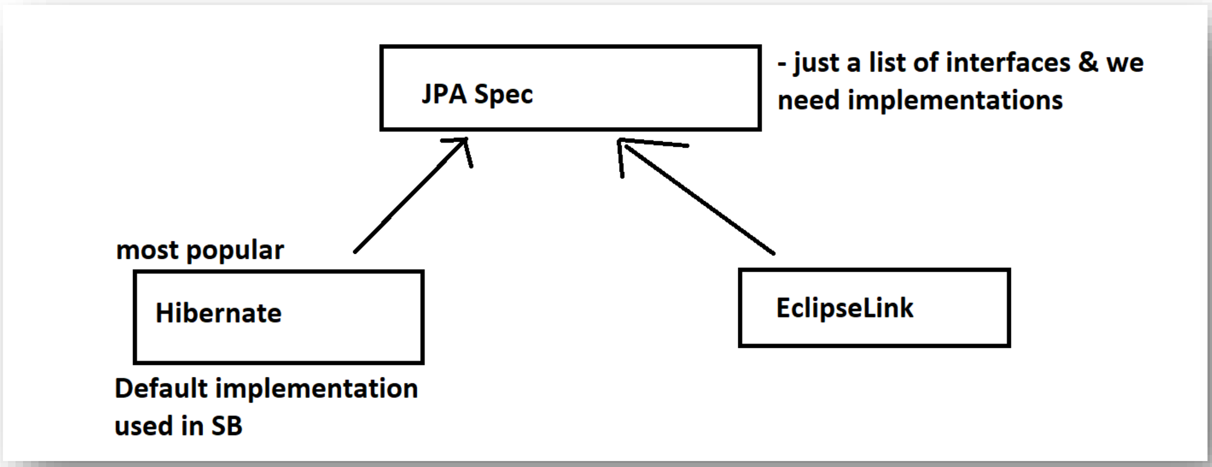


- What we will do is map this java class to the given table & we will set up one-to-one mapping between fields and actual columns in the database.

What is JPA?

- Jakarta Persistence API …… previously known as Java Persistence API
- Standard API for ORM
- It is only a specification
  Which defines set of interfaces
  But requires an implementation to be usable

JPA vendor implementations(OpenJPA, DataNucleus, hibernate, EclipseLink)


JPA Spec
- just a list of interfaces & we need implementations
most popular
Hibernate
Default implementation used in SB
EclipseLink

Advantages of using JPA:

1. By having standard API, we are not locked to vendor implementation
2. We can switch the vendor implementation

EntityManager

Saving a java object with JPA

// create a java object and saving it with JPA

Student theStudent = new Student(“Krishna” ,” Yadav” ,” ky@gmail.com”);

//save it to database

entityManager.persist(theStudent);

- BTS hibernate is the implementation of JPA
  But here JPA with the hibernate does all the work for us in background.