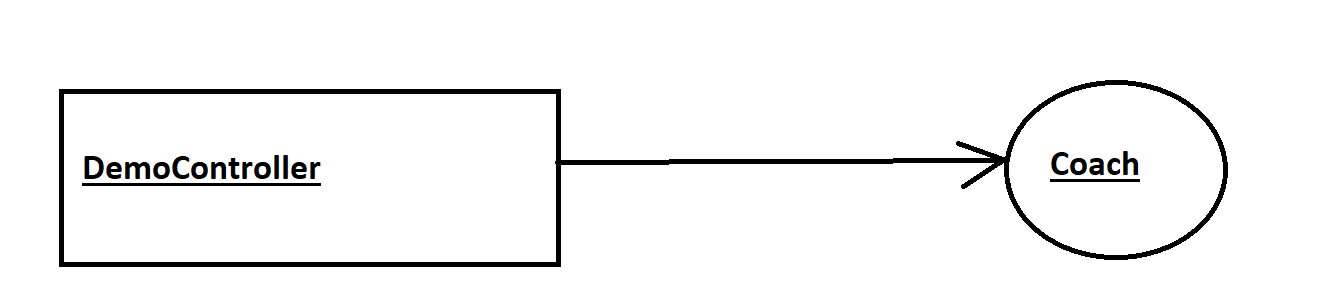


Demo Example:



1. Coach will provide daily workouts
2. The DemoController wants to use the Coach

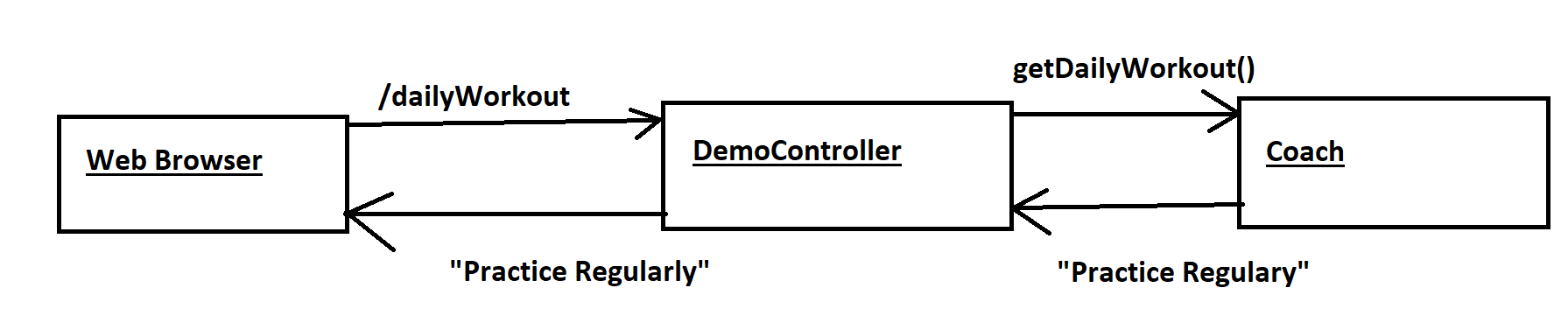
* Helper : Coach
* This is a dependency

1. We need to inject this dependency into Controller

What is Spring Autowiring?

* For dependency injection, spring uses autowiring
* Autowiring Example

Injecting a Coach interface



Development Process:

Step1: Define Interface and the class

package com.flynaut.injection.demo3;  
  
public interface Coach {  
 String getDailyWorkout();  
}

package com.flynaut.injection.demo3;  
  
import org.springframework.stereotype.Component;  
  
@Component  
public class CricketCoach implements Coach{  
 @Override  
 public String getDailyWorkout() {  
 return "Practice Regularly";  
 }  
}

Step2: Create DemoController

Step3: Create a constructor in our class for injection.

Step4: Add @GetMapping for /dailyWorkout

package com.flynaut.injection.demo3;  
  
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 {  
  
 private Coach myCoach;  
  
 @Autowired  
 public DemoController(Coach theCoach){  
 myCoach=theCoach;  
 }  
  
 @GetMapping("/dailyWorkout")  
 public String getDailyWorkout(){  
 return myCoach.getDailyWorkout();  
 }  
  
  
}

BTS of Constructor Injection:

Spring will create an instance of our Coach

How?

Coach theCoach = new CricketCoach();

DemoController DemoController = new DemoController(theCoach);

This way Constructor Injection Happens.

* Component:

Makes a class candidate for dependency injection

Marks the class as a Spring Bean

Spring Bean is just a class which is managed by Spring

Also makes the bean available for DI

* Component Scanning

It scans for component classes

Spring will scan our java classes for annotation @Component

@Qualifier

@Primary

* @SpringBootApplication
* It Enables
* AutoConfiguration
* ComponentScanning
* Additional Configuration

It is Combination of following annotations

1. @EnableAutoConfiguration
2. @ComponentScanning
3. @Configuration

Setter Injection

Inject dependencies by calling the setter method in our class

BTS:

Coach theCoach= new CricketCoach();

DemoController demoController = new DemoController();

demoController.setCoach(theCoach);

Bean Scopes:

Scope = LifeCycle of Bean

Default Scope is singleton

Singleton – SC creates only one instance of the bean by default.

Prototype\*

Session\*

Request\*

BeanLifeCycle:

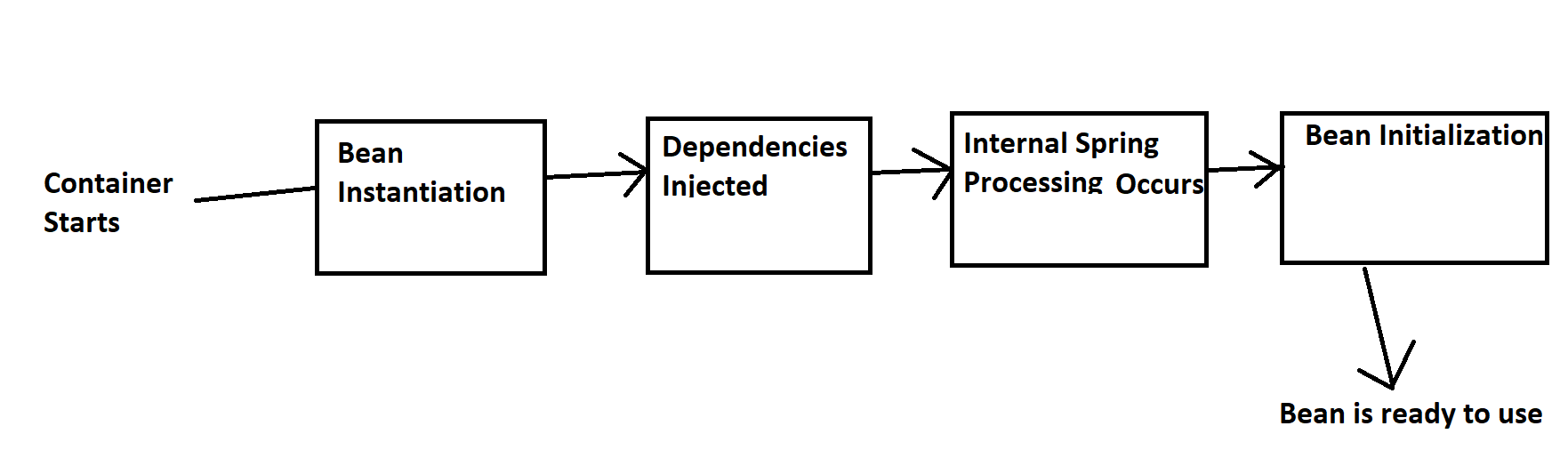
1. Bean Instantiation

* Spring creates object of the bean
* This happens when it calls the constructor of the class.

1. Dependencies injection

* Spring sets all the dependencies that bean needs
* How? -> DI

1. Internal Spring Processing
2. Bean Initialization
3. Bean is ready to use



Hibernate/JPA

Benefits of using hibernate

What is hibernate?

What is JPA?

Benefits of JPA

Field Injection