PayCore Java Spring Bootcamp [2.3]

Homework 2#

21.01.2022

QUESTION 1: IOC and DI means?

Answer:

Inversion of control is a software design principle. With Ioc, it is aimed to minimize the dependencies of the object instances in the Application by providing management. It can also be explained as the framework for creating and managing the dependencies in your project, instead of the developer.  
  
Except for our code block, the management of everything is controlled by the framework. E.g; We have developed a simple notepad application with Spring. Since Spring is a framework, it will set and manage all the resources by itself. In other words, Spring will start the project and run your code when it's ready. When you create a record for the notebook, it will run your code and when your code is finished it will take over control of the Spring Framework again. This is exactly what is called Inversion of Control. IOC can be implemented with mechanisms such as:

1. Strategy Pattern
2. Service Lacator Pattern
3. Factory Pattern
4. Dependency Injection

The advantages of IOC:

1. It allows a method to be run in isolation from its implementation.
2. Allows you to switch between different implementations easily.
3. Increases program modularity.
4. Makes testing/writing easier as dependencies are minimized.

Dependency injection is a programming principle that aims to free a class or object from dependencies and makes that object as independent as possible.  
  
By applying Dependency Injection; You can ensure that a class can act independently of the object it is dependent on, and you can eliminate the need to make changes despite possible improvements to the code. Dependency injection is a technique that is generally used to write code independent of frameworks.

QUESTION 2: Spring Bean Scopes?

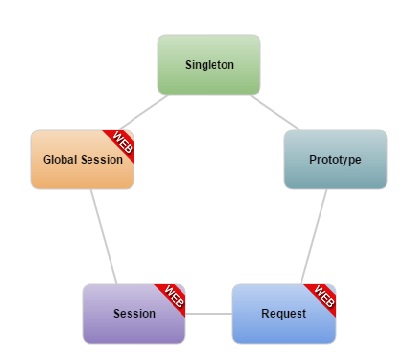
Answer:

Beans:

Objects that form the backbone of our Spring Framework application and are managed by the Spring IOC container are called BEANs. We can consider them as reusable objects.

Scopes: The word Scope has meanings such as scope, area, field of activity, and we can think that our Bean objects have a living space.

Scope Types:



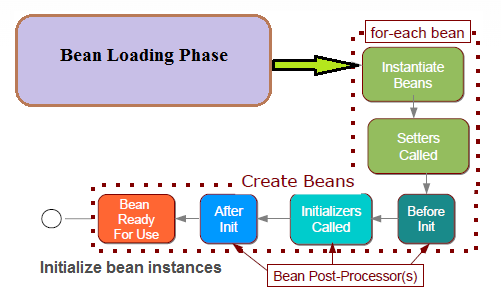
Singleton: Each Bean is a singleton by default, but is generated only once. We can think of it as in Singleton Design Pattern. It is the ability to reuse what is produced once.

Prototype : It is created every time a request is made for the bean in question. A different instance is generated at each creation.

Request : Based on its name, the request bean is created when an HTTP request arrives. An active form is covered at the HTTP request level.

Session : It is created when HTTP request is received in Session Scope Web Applications. An approach similar to Request Scope.

Global Session : It covers the definition of a single Bean in the lifecycle of an HTTP. Valid only in a WEB responsive Spring.



QUESTION 3: What does @SpringBootApplication do?

Answer:

The @SpringBootApplication annotation specifies the application's input method. So the main function of our application is. The application starts with this method.

QUESTION 4: What is Spring AOP ? Where and How to use it?

Answer:

Aspect is a programming paradigm that handles the cross cutting concerns of our applications, and the paradigm's starting point is to find solutions to concerns. The motivation for using this structure is that it complies with principles such as single responsibility and don't repeat yourself.

The following examples can be given to the usage areas of AOP:

* Logging → Logging of requests and responses coming to our service.
* Transaction Management → Performing the return process after the error that will occur in the running code cycle from the receipt of the payment.
* Performance → Calculation of the working times of the methods.
* Validation → User e-mail permission control before the e-mail to be sent.

QUESTION 5: What is Singleton and where to use it?

Answer:

Singleton is the ability to reuse what is produced once for spring beans. It can be generated only once for our each bean.

QUESTION 6: What is Spring Boot Actuator and Where to use it?

Answer:

Spring Boot Actuator automatically activates production-ready features (health check, disk usage, heap dump etc.) of applications and offers a structure that allows interacting with different HTTP endpoints.  
If we want Spring Boot Actuator to be active in the project, the following Maven dependency block must be added to the pom file:

<dependency>

<groupId>org.springframework.boot</groupId>

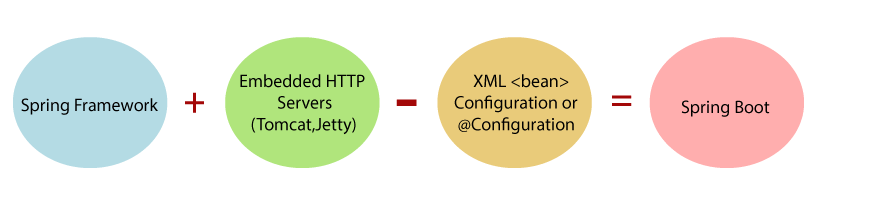
<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

QUESTION 7: What is the primary difference between Spring and Spring Boot?

Answer:

Spring Boot is a project that is built on the top of the Spring Framework. It provides an easier and faster way to set up, configure, and run both simple and web-based applications.



Spring library benefits:

* Spring library can be applied to all layers of a web application.
* Provides Loose Coupling and easy testability.
* Supports XML and Annotation configurations.
* Singleton and Factory It has the ability to eliminate classes.
* Supports declarative programming.

Spring Boot benefits:

* Creates stand-alone applications.
* Comes with Tomcat, Jetty or Undertow as embedded.
* Doesn't need XML configuration.
* LOC (Lines of Lines of) Code).
* Easy to start.
* Simple to customize and manage.

Spring Boot is not a library, it is a ready-made Spring-based project starter. With features such as automatic configuration, it saves you from writing long code and allows you to get rid of unnecessary configurations.

QUESTION 8: Why to use VCS?

Answer:

Version Control System is a system that records the changes we make on a document (software project, office document…) step by step and allows us to store and manage it in a repository on the internet if you want. Git, SVN, BitKeeper, Mercurial are examples of version control systems.

Advantages of Using Version Control Systems:

1. Considering that there is more than one person working in the projects, it accelerates the development.
2. While developing on the project, it allows us to compare our old and new code to understand where we are at.
3. It allows us to return to the old code record in case we encounter an error in the project.
4. It facilitates the process in new projects to be developed based on open source projects.

QUESTION 9: What are SOLID Principles ? Give sample usages in Java?

Answer:

SOLID software principles; It is a set of principles put forward by Robert C. Martin, which ensures that the developed software is flexible, reusable, maintainable and understandable, prevents code duplication.

S — Single Responsibility principle:

A class (object) can only be changed for one purpose, which is the responsibility imposed on that class, so a class (which can also be reduced to a function) has only one job to do.

O — Open-Closed Principle:

A class or function should preserve properties that already exist and should not allow changes. That is, it should not be changing its behavior and should be able to acquire new features.

L — Liskov Substitution Principle:

We should be able to use subclasses in place of their derived (superior) classes without making any changes to our code.

I — Interface Segregation Principle:

Instead of gathering all the responsibilities on a single interface, we should create more customized interfaces.

D — Dependency Inversion Principle:

Dependencies between classes should be as low as possible, especially high-level classes should not depend on lower-level classes.  
  
The advantages when we write our code with SOLID principles:

1. It is ensured that the software we develop easily adapts to the requirements in the future.
2. It allows us to easily add new features without the need for a change in the code.
3. Allows the minimum change on the code despite new requirements.
4. Constant corrections on the code, such as rewriting or even rewriting. It also minimizes the loss of time caused by problems.

QUESTION 10: What is RAD model ?

Answer:

Rapid Application Development (RAD) is a set of methods applied to software development processes and supporting the realization of these processes as quickly as possible. The main objectives of this method, which supports the rapid completion of processes; can be explained as high speed, high quality and low cost. This methodology for increasing the quality of software development processes and reducing costs is a current method as it also saves time.

Advantages of RAD

* Fast decision making,
* Saving time,
* Low Cost,
* Quick clarification of requests by eliminating communication disruptions.

Disadvantages of RAD

* Complete failure can occur if claims are incorrectly specified or defined. The small size of the RAD Team makes possible warnings from 3rd and 4th parties impossible...
* It is possible that the resulting prototype will not be usable as a result of the clarified demands. Clarifying needs may necessitate a different approach or technological need, making the developed prototype completely obsolete.

QUESTION 11: What is Spring Boot starter? How is it useful?

Answer:

The starter APIs defined in the dependency are used to start the Springboot application. Each operation type is defined by springboot as a starter from within dependencies. These APIs are designed to make developers' work easier. And also they are making our project to run faster.

QUESTION 12: What is Caching ? How can we achive caching in Spring Boot ?

Spring Framework provides caching in a Spring Application, transparently. In Spring, the cache abstraction is a mechanism that allows consistent use of various caching methods with minimal impact on the code. Caching is a part of temporary memory (RAM). It lies between the application and persistence database. It stores the recently used data that reduces the number of database hits as much as possible. In other words, caching is to store data for future reference.

To enable Spring Cache, it is sufficient to add @EnableCaching annotation to any configuration class.

QUESTION 13: What & How & Where & Why to logging ?

Answer:

Logging is a method of storing the runtime status of an application in a systematic, controllable, readable way. We can use logging in development, debug and test processes.

* Logging should be systematic.
* Logging should be controllable.
* Logging should describe the state of an application.

Systematic Structure

Logging should be in a systematic approach. Let's not forget that our aim is not to produce/output information according to our pleasure. A strategy should be determined for logging activities, and which information to be used for logging should be decided. Logging can be done for debugging, for daily maintenance of the application, for system administrators to observe the performance of the application/system or for many different purposes. Therefore, the systematic structure may be different according to our logging strategy.

Controllable  
Of course, we will write the necessary codes in our application to do the logging process. These codes will be subject to the controls applied for other codes in our application. Logging code can be bad or well written like any code. If the codes written for the logging process are good, the quality of our application will increase. Poorly written code can cause poor performance.

Application status

The information generated/resulted from logging can be useless and useless if it is not decided what to log properly. In order for this information to be efficient, it should be determined properly what and which situations will be logged. Our goal should be to explain/identify/log every necessary state of the system.

Advantages of Logging

Fast debugging: When we encounter a problem, log records will show us where the problem originates. A well-written logging code will enable us to find the root cause of the problem in a shorter time, which will save us time in debugging.

Easy maintenance: With a well-written logging structure, our application will be easy to maintain/continue. We know that logging codes are not only kept for debug. These log records will give us information about the system and in the light of this information, system maintenance will be easier.  
  
History: Logging information can be kept in a desired directory, in a desired file name, for example, as a historical appendix, as yilmaz\_log\_05\_11\_2021. In terms of content, we can keep it in different formats. These logged files are kept retrospectively, and can be kept against any problem or a different situation.

Cost and Time Savings: With a well-written logging structure, time and cost savings such as fast debugging and easy maintenance can be achieved.

Disadvantages of Logging

Poorly designed logging structure can cause poor system performance.  
Useless/unnecessary outputs are useless. Bad output can cause confusion.  
Logging will require extra code to be written to our application, so it can be costly and time consuming in the above 2 cases.

QUESTION 14: What is Swagger? Have you implemented it using Spring Boot?

Answer:

To define swagger we need to explain API first. The interface developed for the use of functions and produced data of an application in other applications is called API. One of the most important needs in Web API development is the need for documentation. Because what the API methods do and how they are used should be clear in the documentation. Manually writing API documentation is both difficult and impossible to keep up to date. Somehow it is necessary to produce this documentation up-to-date. Here swagger comes to our rescue.

An important purpose of Swagger is to provide an interface for RestApis. This allows both humans and computers to see, examine and understand the features of RestApis without accessing the source code.  
To include swagger in our Maven project, we use these codes:

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger2</artifactId>

<version>2.9.2</version>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger\_ui</artifactId>

<version>2.9.2</version>

</dependency>