**Spring IoC container**

 The container will create the objects, wire them together, configure them, and manage their complete life cycle from creation till destruction

**What is Bean ?**

objects that form the backbone of your application

 instantiated, assembled, managed by the Spring IoC container

**What is Spring Framework? OLL for reduce com**

Spring is a powerful open-source, loosely coupled, lightweight, [java framework](https://www.interviewbit.com/java-interview-questions/) meant for reducing the complexity of developing enterprise-level applications

loosely coupled **: components are weakly associated (have breakable relationship) with each other**

### What is a Spring configuration file?

A Spring configuration file is basically an XML file that mainly contains the classes information and describes how those classes are configured and linked to each other. The XML configuration files are verbose and cleaner.

For working with Libraries we use the : Spring.io website click project then spring framework

Copy jar file and paste them into the junit file inside the POM.xml and inside <dependcies > tag

Spring Core Framework (Application Context)

Vehicle obj = new car()

Assume we want to use bike so we need to change right side and this is also called dependecy we wanna remove the dependecy so what we should do is using spring framework

Vehicle obj = (Vehicle) context.getBean(“Vehicle”) ;

GetBean is a method which blongs to beanFactory(For Small company) and application Context (For Big Enterprise)

**\*\*\* Import app context or bean factory at the top of your java file \*\*\***

ApplicationContext context = new ClassPathApplicationContext(**spring.xml**);

Where you define beans



### How is the configuration meta data provided to the spring container?

**XML-Based configuration:**

**Make xml file and define the bean inside the XML file**

**You have to use the defenition (definition of bean in this case)**

**<beans>**



**<bean id = “vehicle” class = “com.navin.Tulesko.(car , bike , or the other classes)**

**</beans>**

**Annotation-Based configuration:** Instead of the XML approach, the beans can be configured into the component class itself by using annotations on the relevant class, method, or field declaration.



Vehicl obj = (vehicle)context.getBean(car)



Inside the class of car we are going to use the anotation

@component import it at the top



Class car {

}

**<bean id = “vehicle” class = “com.navin.Tulesko.(car , bike , or the other classes) remove it from configuration file**



But the spring should know that we are using anotation based configuration

<Context-component-scan base-package = “**tulesko package**‘’ ></Context-component-scan>

**Java-based configuration or @bean Method:**Spring Framework introduced key features as part of new Java configuration support. This makes use of the **@Configuration** annotated classes and **@Bean** annotated methods

Tyre t = (tyre) context.getBeans(‘’Tyre’’) ;

**Inside the configuration file spring.xml we define the value**

<bean id = ‘’tyre” class = “com.navin.tulesko.tyre”>



<property name = “brand” value = “hello”> Tyre = [brand = “hello”]

</bean>

**What are the bean scopes available in Spring?**



The Spring Framework has five scope supports. They are:

* **Singleton:** The scope of bean definition while using this would be a single instance per IoC container.
* **Prototype:** Here, the scope for a single bean definition can be any number of object instances.
* **Request:**The scope of the bean definition is an HTTP request.
* **Session:** Here, the scope of the bean definition is HTTP-session.
* **Global-session:** The scope of the bean definition here is a Global HTTP session

You can assign a value either by set the name of bean or you can costruct the bean

Create the constructor by right clicking make it automatically

Ctrl space make construcotr arg for you and it will assign the value you pass to the object

### What is the use of @Autowired annotation?

It is going to check the configuration type for the proprty we instantiated and pick the value of the class associated with that id.

Class car{

@autowired



Privae Tyre tire ;



}

<bean id = tyre class =”tulesko.tyre”></bean>



**The Spring container** is at the core of the Spring Framework. The container will create objects, wire them together, configure them, and manage their complete life cycle from creation till destruction. The Spring container uses DI to manage the components that make up an application. These objects are called Spring Beans, which we will discuss in the next chapter.

**How to kill a task in CMD :**

netstat -a -o -n

taskkill /PID 5948 /F

**JAVA PERESISTANCE API (jpa)**

Is a mechanism to mapping all the presistance and object relational mapping and functions

**SHORT CUT TO CREATE CONSTRUCTOR**

Alt + insert

**jpa always need zero arg constructor**

We need to give the jpa an id so it will know how to store enitites in database and how to retrive them

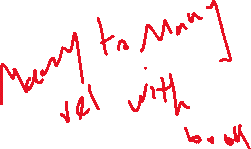
**HOW TO ASSIGN REALATIONSHIP FOR JPA**

**How to write many to many relation ship :**

**@manytomany**

**@JoinTable (name = entity 1\_entity2) , JoinCloumns = @JoinClumn(name = entity1\_id),**

**InverseJoinCloumn = @JoinCloumn (name = enity2\_id)**



**Private Set <Author> Authors**



@ManyToMany  
@JoinTable(name = "author\_book",joinColumns = @JoinColumn(name = "book\_id"),  
 inverseJoinColumns = @JoinColumn(name = "author\_id"))  
private Set<Author> authors;

\*\*\*At the other side of the relationship we are using

@ManyToMany(mappedBy = "authors")

Abbove indicates that the book is mapped by authors

bEquality of the object id properties that we added to database record when we are going to change or override this method if the id had different name it going to set that id to the id that we have

@Override  
public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
  
 Book book = (Book) o;  
  
 return id == book.id;  
}

**Hibernate** is a Java framework that **simplifies the development of Java application to interact with the database**. It provides a framework for mapping an object-oriented domain model to a relational database

CrudRepository is **a Spring Data interface for generic CRUD operations on a repository of a specific type**. It provides several methods out of the box for interacting with a database. We are going to create the interface and spring JPA reposiory going to implement It for us. It is providing a lot of pre build function like find by id , delete , save and etc

Creating Repositories For Our Classes

Graphical user interface, text

Description automatically generated

Initialization of some data for Our database

CommandLineRunner is **a simple Spring Boot interface with a run method**. Spring Boot will automatically call the run method of all beans implementing this interface after the application context has been loaded.

public class BootStrapData implements CommandLineRunner {

@Override  
public void run(String... args) throws Exception {  
  
 System.*out*.println("Started in Bootstrap");  
  
 Publisher publisher = new Publisher();  
 publisher.setName("SFG Publishing");  
 publisher.setCity("St Petersburg");  
 publisher.setState("FL");

Graphical user interface, text, application, chat or text message

Description automatically generated



@Id  
@GeneratedValue(strategy = GenerationType.*AUTO*)  
private Long id;

What is interface mapping?

In Interface mapping, **you define what interfaces that will be communicating with each other**

**Create an interface with methods for mapping between objects**. Add a @Mapper annotation to the interface. The method signature will generally need to have a similar structure to this

One to many relationship

@OneToMany

@JoinCloumn(name = “”)

Many to one

@ManyToOne

ManyToMany

@ManyToMany  
@JoinTable(name = "author\_book",joinColumns = @JoinColumn(name = "book\_id"),  
 inverseJoinColumns = @JoinColumn(name = "author\_id"))  
private Set<Author> authors;

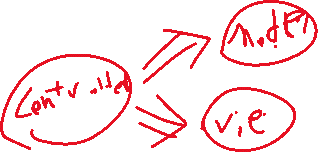
Hibernate generate the sql ddl statement To go out and create database table based on our Jpa anotation

**Hibernate is one of the popular implementations of JPA.**

1. Hibernate understands the mappings that we add between objects and tables. It ensures that data is stored/retrieved from the database based on the mappings.
2. Hibernate also provides additional features on top of JPA.

Try so hard to be successful try try try trt

What is mvc ? model view controller



**DispatcherServlet is** **the front controller**. The DispatcherServlet's job is to send the request on to a Spring MVC controller. A controller is a Spring component that processes the request.

Request come in dispathcher look at handeling mapping invoke controller mapper return model and it will passed to the template engine and the html rendered back

A screenshot of a computer

Description automatically generated with medium confidence

Anotate controller with @controller

This will generate a spring bean as a controller in spring mvc

To map method to http request use @RequestMapping

Thymleaf is natural template engine You can view template in your browser with it

You have to add the appropiriate dependency related to thymleaf to the pom first

Then you are going to add the <html lang="en" xmlns:th="http://www.thymeleaf.org">

At the top of your html file then you can use many options that thymleaf provided inside the tags you have in html file <tr th:each="book : ${books}">  
 <td th:text="${book.id}">123</td>  
 <td th:text="${book.title}">Spring in Action</td>  
 <td th:text="${book.publisher.name}">Wrox</td>  
</tr>

How to debug in intelij

How to address the bin in defining application context

Graphical user interface, text, application

Description automatically generated

<bean id = "helloWorld" class = "com.example.demo.classes.HelloWorld" init-method = "init"  
 destroy-method = "destroy">  
 <property name = "message" value = "Hello World!"/>  
</bean>

public void getMessage(){  
 System.*out*.println("Your Message : " + message);  
}  
public void init(){  
 System.*out*.println("Bean is going through init.");  
}  
public void destroy() {  
 System.*out*.println("Bean will destroy now.");  
}

What is the output ?

remember that the destroy method alaways get implemented while the bean get eliminated

Spring - Bean Life Cycle

with **initmethod** and/or **destroy-method** parameters. The init-method attribute specifies a method that is to be called on the bean immediately upon instantiation. Similarly, destroymethod specifies a method that is called just before a bean is removed from the container.

public class ExampleBean implements InitializingBean {

public void afterPropertiesSet() {

// do some initialization work

}

}

public class ExampleBean implements DisposableBean {

public void destroy() {

// do some destruction work

}

}

What is the call back method ?

A callback method in java is a method that gets called when an event (call it E) occurs. Usually you can implement that by passing an implementation of a certain interface to the system that is responsible for triggering the event E , example:

public class CallbackImpl implements Callback {

**BeanPostProcessor**

This bean provide a call back method that you can implement and by that you can provide order for bean execution like

postProcessAfterInitialization

postPeoccessBeforeInitialization

public Object postProcessAfterInitialization(Object bean, String beanName)  
 throws BeansException {  
  
 System.*out*.println("AfterInitialization : " + beanName);  
 return bean; // you can return any other object as well  
}

**Bean definition Inheritance** You can define a parent bean definition as a template and other child beans can inherit the required configuration from the parent bean.

<bean id = "helloWorld" class = "com.tutorialspoint.HelloWorld">

<property name = "message1" value = "Hello World!"/>

<property name = "message2" value = "Hello Second World!"/>

</bean>

<bean id ="helloIndia" class = "com.tutorialspoint.HelloIndia" parent = "helloWorld">

<property name = "message1" value = "Hello India!"/>

<property name = "message3" value = "Namaste India!"/>

</bean>

**Why do we do dependency injection ? and what are the two methods for doing the dependenct injection?**

When writing complex application at java we have to make the classes dependent as possible so it gonna help programmers when they want to reuse these classes and do the unit testing withuot any problem, there are two method in this case first method is using class constructor and the second one would be using setter method , this the example

public class TextEditor {

private SpellChecker spellChecker;

public TextEditor(SpellChecker spellChecker) {

this.spellChecker = spellChecker;

}

}

For fetching information in spring we need to creat a service so we start to build the java classes

And this how we fetch data in to console to use it later in our web application

Text

Description automatically generated



Hey spring start this when the application started by this annotation

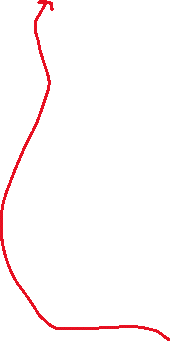
**Parsing information in java is possible by csv library :**

**Website:** <https://commons.apache.org/proper/commons-csv/index.html>

This library is using to convert it to object and parsing it .

Graphical user interface, text, application

Description automatically generated



Iterable<CSVRecord> records = CSVFormat.*DEFAULT*.withFirstRecordAsHeader().parse(in);



@scheduled annotation run a method in a regular basis

Check this web for more info : https://www.baeldung.com/spring-scheduled-tasks

For example when you want to get informations from specific uri every day assume you want to updated number of people getting corona every day so you can specify this annotatin to do that

It has different options like ; **corn** you can specify the time **example : @scheduled(corn =**

**“\* \* \* \* \* \*”**



**Second mintue hour**

**Enablescheduling :** grab the informations used in @scheduled annotation by @EnableScheduling in java application

**Text

Description automatically generated**

Text

Description automatically generated

**A proxy :** server is **a bridge between you and the rest of the internet**. Normally, when you use your browser to surf the internet, you connect directly to the website you're visiting.

What is the porpocess of controller ?

controller methods are **the final destination point that a web request can reach**. After being invoked, the controller method starts to process the web request by interacting with the service layer to complete the work that needs to be done.

**How controller interact with html page to print the value**

Text

Description automatically generated Text

Description automatically generated