

AMS Bootcamp Fullstack Developer.

**Spring Core Notebook**



Confidencial

Índice

[1. Personal Notebook 4](#_Toc97116284)

[1.1 Datos alumno 4](#_Toc97116285)

[1.2 Personal Notebook 4](#_Toc97116286)

[2. Spring Framework 5](#_Toc97116287)

[2.1 Objetivos de aprendizaje 5](#_Toc97116288)

[2.2 Introducción a Spring e instalación inicial 6](#_Toc97116289)

[2.2.1 Objetivos 6](#_Toc97116290)

[2.2.2 Recursos formativos 6](#_Toc97116291)

[2.2.3 Posibles preguntas 6](#_Toc97116292)

[2.2.4 Tareas 6](#_Toc97116293)

[2.2.5 Dudas 6](#_Toc97116294)

[2.3 Qué es IoC e Inyección de dependencias 7](#_Toc97116295)

[2.3.1 Objetivos 7](#_Toc97116296)

[2.3.2 Recursos formativos 7](#_Toc97116297)

[2.3.3 Posibles preguntas 7](#_Toc97116298)

[2.3.4 Tareas 7](#_Toc97116299)

[2.3.5 Dudas 7](#_Toc97116300)

[2.4 Ejemplo de IoC e Inyección en Java 8](#_Toc97116301)

[2.4.1 Objetivos 8](#_Toc97116302)

[2.4.2 Recursos formativos 8](#_Toc97116303)

[2.4.3 Posibles preguntas 8](#_Toc97116304)

[2.4.4 Tareas 8](#_Toc97116305)

[2.4.5 Dudas 8](#_Toc97116306)

[2.5 Ciclo de vida de los Beans 9](#_Toc97116307)

[2.5.1 Objetivos 9](#_Toc97116308)

[2.5.2 Recursos formativos 9](#_Toc97116309)

[2.5.3 Posibles preguntas 9](#_Toc97116310)

[2.5.4 Tareas 9](#_Toc97116311)

[2.5.5 Dudas 9](#_Toc97116312)

[2.6 Configuración por Anotación 10](#_Toc97116313)

[2.6.1 Objetivos 10](#_Toc97116314)

[2.6.2 Recursos formativos 10](#_Toc97116315)

[2.6.3 Posibles preguntas 10](#_Toc97116316)

[2.6.4 Tareas 10](#_Toc97116317)

[2.6.5 Dudas 10](#_Toc97116318)

[2.7 Ejemplo de IoC e Inyección en Spring 11](#_Toc97116319)

[2.7.1 Objetivos 11](#_Toc97116320)

[2.7.2 Recursos formativos 11](#_Toc97116321)

[2.7.3 Posibles preguntas 11](#_Toc97116322)

[2.7.4 Tareas 11](#_Toc97116323)

[2.7.5 Dudas 11](#_Toc97116324)

# Personal Notebook

## Datos alumno

* DAS
* Nombre Apellidos
* email
* URL de GitHub corporativo

## Personal Notebook

Este es tu cuaderno de trabajo. En él tendrás que registrar tu aprendizaje, tus avances y tus dudas. ¡Vas a generar tu propio material!

Puede ser una gran herramienta si la utilizas correctamente.

El cuaderno estará compartido con los tutores y el resto de colaboradores para ver tu progreso y ayudarte.

Al final del Bootcamp tendrás que entregarlo como evidencia de tu aprendizaje.



Instrucciones de uso:

El fichero está ubicado en tu carpeta del espacio que tenemos en Teams.

Siempre deberá ser editado en modo online y/o con guardado automático en la nube.

Responde a las preguntas con la información que necesites para dar una respuesta cualificada al formador o a tus compañeros. No es necesario copiar texto solo por “rellenar”. Cita las Fuentes en caso de utilizar otros recursos.

# Spring Framework

## Objetivos de aprendizaje

Objetivo: **Adquirir los conocimientos básicos del framework Spring.**

Conceptos principales a adquirir:

* IoC
* ID
* Anotaciones
* Configuración de Spring
* Instalación de entornos (IDE) en Spring

Lecciones:

1. Introducción a Spring e instalación inicial
2. Qué es IoC e Inyección de dependencias
3. Ejemplo de IoC e Inyección en Java
4. Ciclo de vida de los Beans
5. Configuración por Anotación
6. Ejemplo de IoC e Inyección en Spring

## Introducción a Spring e instalación inicial

### Objetivos

1. **Conocer los motivos para usarlo y sus ventajas.**
2. **Componentes de Spring.**
3. **Tener un entorno de desarrollo (IDE+servidor) listo para trabajar**

### Recursos formativos

<https://www.youtube.com/watch?v=kFIvslQQZ9k&list=PLU8oAlHdN5Blq85GIxtKjIXdfHPksV_Hm>

(PildorasInformáticas videos del 1 al 5)

### Posibles preguntas

¿Qué es Spring?

VIP: recuerda que tienes que responder a estas preguntas de forma que te ayude a explicárselas al formador o a los compañeros!!

Es un framework, es una alternativa a JEE más simple y ligera

Y un framework es un entorno de trabajo compuesta por reglas y herramientas.

Indica los motivos por los que usar Spring

El uso de menos recursos lo que lo hizo más ligero y rápido.

Simplifica el acceso a datos.

permite la separación de módulos de un programa java para que todas aquellas modificaciones se realicen en un solo lugar.

Partes del framework (módulos y que hacen)

Se divide en 5 módulos

Contenedor central

realiza todo el trabajo relativo a la creación de beans,

a la lectura de archivos config.

Realiza el manejo de propiedades y dependencias

uso del context como almacén de los beans en memoria.

Infraestructura

Se hace referencia a la Programación Orientada Aspectos.

Maneja todo lo referente a las transacciones, logueos, a la seguridad de las aplicaciones.

Acceso a Datos

Permite el acceso a datos con menos código con la ayuda de las helper clases.

Este módulo trae submódulos:

Jdbc

Transactions y este se divide en OXM y JMS

ORM object relational Mapping: acceder a datos con un enfoque orientado a objetos

permite un alta integración con hibernate

java Messaging service

WEB/MVC

Viene con 4 módulos que nos van a permitir el acceso web remoto, la programación distribuida. Nos va permitir también la integración de otras tecnologias como jsf y struts.

Pruebas

nos va permitir realizar pruebas durante el desarrollo de los programas.

Para ello usa Mock Objects que son objetos simulados que podemos crear durante el desarrollo de un programa java para hacer pruebas

¿Qué significa acoplamiento?

¿Si quiero enviar mensajería que tengo que usar?

Se usa java messaging service

¿Qué tipo de aplicaciones puede crear con Spring?

Aplicaciones web.

Aplicaciones con acceso a base de datos

Aplicaciones móviles, escritorio.

### Tareas

Montar un entorno de desarrollo para trabajar con Spring (IDE a usar Eclipse XXX, Tomcat 8, Maven, …)

Describir los pasos a realizar (de forma general)

Descargar e instalar el jdk y el jre de java.

Verificar a traves de cmd si se está usando la versión java instalada

Descargar e instalar el tomcat.

Cambiar de puerto si en caso está en uso

Descargar eclipse.

Instalar esta version de eclipse enterprise for java and web developers

Durante la instalación nos pedirá seleccionar una carpeta para nuestro espacio de trabajo.

Podemos crearla o utilizar la que viene por defecto.

Esperamos que termine la instalación.

Procederemos arrancar tomcat

En la zona inferior clicaremos en servers y luego en el enlace que dice “no server are available”

Se muestra una ventana en la cual elegiremos la versión de tomcat que hemos instalado

En la siguiente ventana se nos pedirá la ubicación de tomcat y la versión de java que utiliza eclipse.

Al seleccionar la ubicación de tomcat realizarlo como administrador sino aparece un mensaje de que no encuentra tomcat

Si todo está bien y no hay mensajes de error, presionar en finalizar.

Descargar e instalar librerías de Spring

Dirigirse a google y Descargar los jar files y descomprimirlos.

Luego nos dirigimos a eclipse para cambiar la perspectiva ya que tenemos la vista Enterprise Empresarial.

Ya en eclipse. Crearemos un java project donde importaremos la librerías que hemos descargado (jar files)

Para importar dentro del proyecto crearemos una carpeta donde iran las librerias

Copiaremos las librerías en esa carpeta.

Ahora tendremos que indicarle al proyecto que utilice esas librerías.

Click derecho en la carpeta y seleccionar properties->java build path->libraries->addJars.

Luego desplegar la carpeta libs y seleccionar todas las librerías. Agregar y finalizar.

Nota: No usar STS hasta que no se indique en los cursos. La tarea es montar el entorno sobre un “Eclipse estándar” y añadir las librerías y servidores correspondientes

### Dudas

¿?

¿?

VIP: registra las dudas que tengas para la sesión con el formador!!

¿?

¿?

¿?

¿?

## Qué es IoC e Inyección de dependencias

### Objetivos

* **Comprender los conceptos de Ioc e ID**
* **motivos por los cuales se usán**
* **ventajas e incovenientes…**
* **conceptos de "Contenedor", "Patrón", etc..**

### Recursos formativos

* (Recomendado) <https://www.youtube.com/watch?v=-Cs1HN6pEg4>

(pildorasInformáticas) (videos del 6 al 12)

* (complementario) <https://www.youtube.com/watch?v=jURvf96Qaq8>
* (complementario) <https://www.youtube.com/watch?v=UXeoZ6YA6LM&list=PLF6521742961719FC&index=3&t=0s>

### Posibles preguntas

* ¿Qué es IoC? ¿e ID?
* IoC: Es delegar a un tercero el control de la construccion y el manejo de objetos. En nuestro caso el encargado de esto es el framework(spring).
* Es modificar el orden en el cual se van llamando las clases. Es ir en contra de la manera tradicional como normalmente se harían las cosas en cuanto a las llamadas, a las invocaciones de una clase a otra, de la instanciación de una clase a otra.
* ID: responde a la pregunta de ¿cómo hago la IoC? Se aplica la técnica de la ID que es pasarle la instancia, ya sea por constructor o método, la clase que necesita.
* ¿Es lo mismo IoC que ID?
* Están bastante relacionados, es difícil conseguir uno sin el otro, pero no son lo mismo. Entonces la IoC es invertir la iniciativa de las clases y para hacer esto se utiliza la técnica llamada inyección de dependencias.
* ¿Qué es el principio Hollywood?
* Es una forma de diseñar sistemas informáticos en la que los componentes del sistema interactúan entre si a través de mensajes, en lugar de compartir estados datos directamente.
* ¿IoC/ID están dentro de los principios SOLID?
* Están relacionados en el sentido de que ambos buscan mejorar la calidad del código y reducir la complejidad de las aplicaciones ya que uno de los principios solid (Principio de Inversión de Dependencias) establece que las dependencias deben depender de abstracciones, no de implementaciones concretas. IoC implementa también este principio ya que las dependencias se definen en términos de interfaces y abstracciones en lugar de implementaciones concretas.
* ¿y qué son los SOLID?

Son una serie normas o recomendaciones que guían un poco la forma de programar y diseñar nuestros sistemas. SOLID es una regla mnemotécnica para recordar los nombres de los 5 principios, los cuales fueron impulsados por Rober C Martin. Pero el nombre lo puso Michael feathers. Los 5 principios son los siguientes:

Single Responsibility. Responsabilidad única

Open Closed. Abierto Cerrado

Liskov Substitution. Sustitución de LIskov

Interface Segregation. Segregación de la Interfaz

Dependency Inversion. Inversión de la Dependencia.

* ¿Qué siginifica acoplamiento?
* Se refiere a la excesiva dependencia entre partes o clases de un sistema informático.
* ¿Con spring tenemos un alto o bajo acoplamiento?
* Con spring se puede lograr un bajo acoplamiento. Esto se debe a que spring promueve la separación de responsabilidades y fomenta el uso de interfaces y abstracciones para la comunicación entre componentes.
* ¿Qué es un POJO?
* El POJO (Plain Old Java Object) es una clase completamente independiente del framework que no extiende ni implementa nada particular. Y puede ser utilizado por múltiples propósitos.

### Tareas

NA

### Dudas

¿?

## Ejemplo de IoC e Inyección en Java

### Objetivos

* **Comprender los conceptos de IoC y DI utilizando únicamente el lenguaje Java**

### Recursos formativos

* <https://www.youtube.com/playlist?list=PLG1t8jaLbxA8NfCZwlHcrXhB_Ih1WTCyS>

(video 4 y 16)

### Posibles preguntas

NA

### Tareas

Entregar un ejemplo operativo y funcional donde se explique el funcionamiento de IoC e ID usando únicamente clases java (sin usar ningún framework tipo Spring o EJB3.0)

### Dudas

* ¿?

## Ciclo de vida de los Beans

### Objetivos

* + **comprender el concepto de Bean.**
  + **cómo se pueden añadir propiedades y valores a los Bean mediante configuraciones adicionales (anotaciones).**
  + **trabajar con beans con ambas configuraciones**

### Recursos formativos

* (Recomendado) <https://www.youtube.com/watch?v=-Cs1HN6pEg4>

(pildorasInformáticas) (videos del 9 al 14)

* <https://www.youtube.com/playlist?list=PLG1t8jaLbxA8NfCZwlHcrXhB_Ih1WTCyS>

### Posibles preguntas

* ¿Qué es un Bean?
* Es una instancia de un clase que se configura en el applicationContext.xml o contenedor.xml para proporcionar funcionabilidad a la aplicacion. Dentro de estos archivos config.xml se puede crear y configurar beans segun sea necesario.
* ¿Cuál es la diferencia entre crear un Bean por constructor y por un setter?
* La diferencia entre crear un bean por constructor y por un setter es la forma en que se inyectan las dependencias. La inyección por constructor es adecuada cuando todas las dependencias son requeridas y no se pueden establecer posteriormente, mientras que la inyección por un setter es adecuada cuando las dependencias no son necesarias en el momento de crear el bean y se pueden establecer posteriormente. Otra diferencia es la mutabilidad de los beans, ya que los creados por constructor son inmutables y los creados por setters son mutables, esto es que pueden cambiar sus dependencias inyectadas en cualquier momento.
* ¿Qué significa el concepto de convención sobre configuración? (pon ejemplos)
* ¿Cuáles son los distintos scopes (ámbitos) de un Spring Bean? ¿Cuál es el scope por defecto?
* El scope de un bean se refiere a la duracion de vida que este tiene dentro del contenedor de spring. Los mencionados en el video son el patron de diseño singleton y prototype pero hay mas.
* Tenemos Request, Session, Global Session y Application
* El scope por defecto es singleton el cual solo crea una instancia unica del objeto del bean y la reutilizará cada vez que se solicite ese bean, mientras que si se define con el ámbito Prototype, Spring creará una nueva instancia del objeto del bean cada vez que se solicite.

### Tareas

NA

### Dudas

¿en que caso se podria usar DI por constructor y por setters?

¿se usa siempre singleton?

¿Por qué crear diferentes instancias de un objeto con prototype?

## Configuración por Anotación

### Objetivos

* **configurar Spring mediante anotaciones**
* **Conocer que existe la opción de configuración por XML**

### Recursos formativos

* (Recomendado) <https://www.youtube.com/watch?v=-Cs1HN6pEg4>

(pildorasInformáticas) (videos del 15 al 25)

* <https://www.youtube.com/playlist?list=PLG1t8jaLbxA8NfCZwlHcrXhB_Ih1WTCyS>
* [CONFIGURACIÓN DE SPRING: XML vs. ANOTACIONES vs. JAVACONFIG - YouTube](https://www.youtube.com/watch?v=yWDxgzDZzwo)
* lecturas: https://www.adictosaltrabajo.com/2013/10/30/tutorial-basico-spring-3/

### Posibles preguntas

* ¿en qué fichero se configura?
* Se configura en el archivo xml llamado applicationContext. Aqui con la etiqueta component scan se le dice al archivo xml que se usara java annotations
* ¿Cómo se identifican la clases/bean que estarán bajo Spring?
* Se identifican con la anotacion @component linea arriba de la declaracion de la clase. Dentro de ella entre comillas se le da el ID del bean. gracias a esta anotacion se registra este bean en el contenedor de Spring
* ¿Qué significa la propiedad autowiring?
* Signfica que posee un mecanismo que inyecta las dependencias automaticamente.
* ¿Cómo funciona? (enumerar los distintos tipos)
* Spring busca en el contenedor de beans aquellos objetos que satisfagan las dependencias requeridas por una clase y las inyecta automáticamente en la clase.
* La propiedad autowiring se puede configurar en tres modos diferentes: byName, byType y constructor. Con byName, Spring busca un objeto cuyo nombre coincida con el atributo de la dependencia de la clase y lo inyecta. Con byType, Spring busca un objeto cuyo tipo sea compatible con la clase y lo inyecta. Y con constructor, Spring busca un constructor que tenga argumentos compatibles con los tipos de las dependencias y los inyecta
* ¿Qué anotaciones tenemos para el autowiring?
* Tenemos: @autowired @qualifier @resource @inject
* ¿Qué diferencias tienen?
* @autowired:es la anotación más comúnmente utilizada en Spring para realizar la inyección de dependencias. Se puede aplicar a un constructor, método setter o directamente a un campo de clase.
* @qualifier: esta anotación se utiliza en conjunción con @Autowired para especificar el nombre o el identificador único del bean que se debe inyectar en el componente. Se puede aplicar a un campo o a un parámetro de un constructor o método
* @resource: esta anotación se puede utilizar para inyectar dependencias por nombre, y se puede aplicar a un campo o a un método setter
* ¿Puedes indicar otro tipo de anotaciones de spring? (respuesta esperada: Component, Controller, Repository,Service)

@configuration reemplazan a lo archivos xml que anteriormente se utilizaban para configurar el framework

@bean son la especificaion de los metodos que se suelen poner dentro de un @configuration. Los @bean no se pueden usar a nivel de clases para ellos usamos los @component

### Tareas

### Dudas

¿?

## Ejemplo de IoC e Inyección en Spring

### Objetivos

* **comprender los conceptos de IoC e ID utilizando únicamente el framework Spring**

### Recursos formativos

* lecturas: https://www.adictosaltrabajo.com/2013/07/25/spring-container-inyeccion-dependencias/

### Posibles preguntas

NA

### Tareas

* Realiza un ejercicio usando la versión actual de Spring [5.3.25](https://mvnrepository.com/artifact/org.springframework/spring-core/5.3.25) (sin usar Spring Boot) parecido a los vistos en los recursos formativos donde se ve el uso de las distintas anotaciones, configuraciones… de spring (se valorará la originalidad y que el ejercicio no sea igual al del vídeo).

### Dudas

¿?