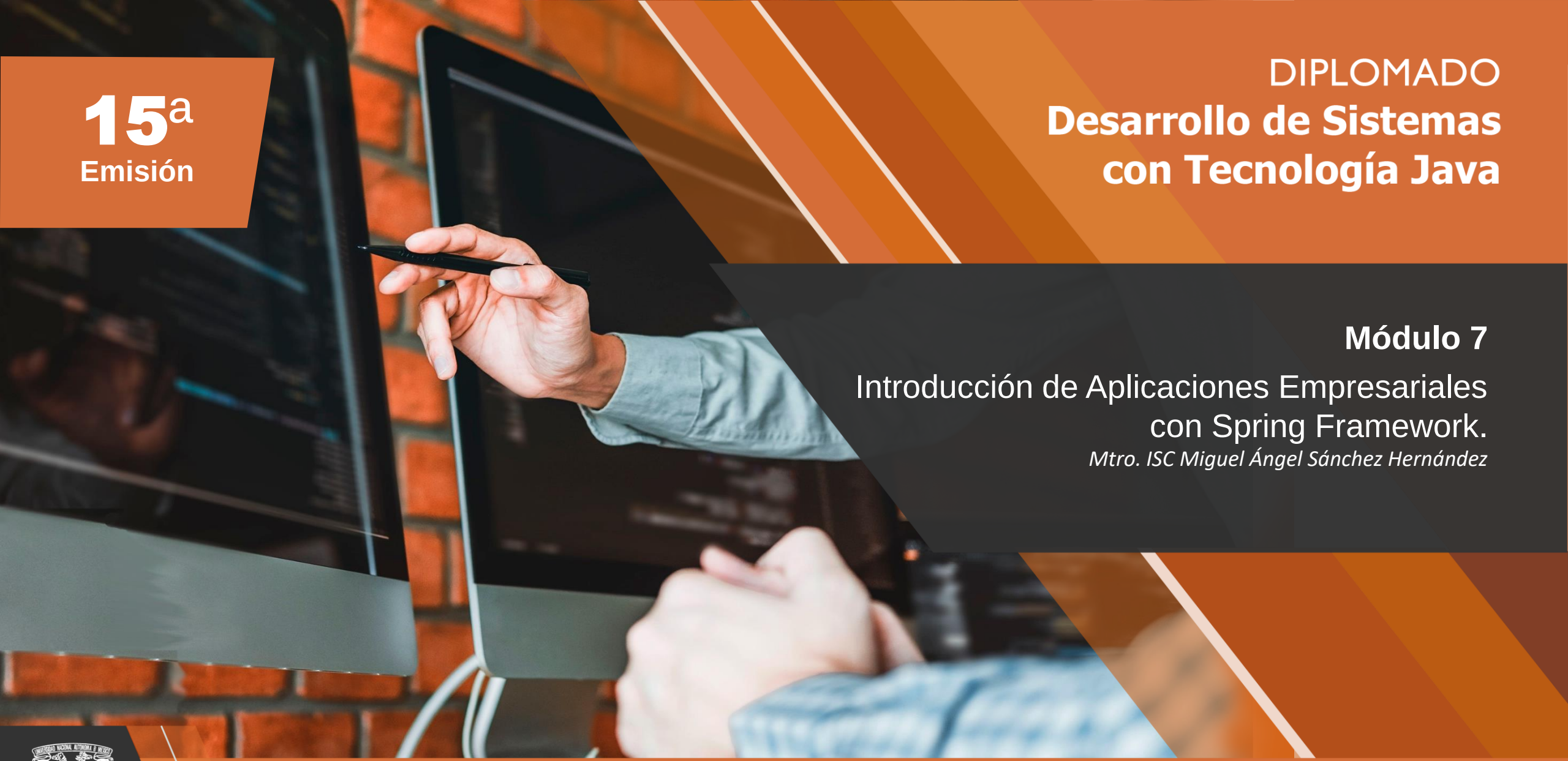


15ª Emisión

DIPLOMADO

Desarrollo de Sistemas con Tecnología Java

Módulo 7

Introducción de Aplicaciones Empresariales con Spring Framework.

Mtro. ISC Miguel Ángel Sánchez Hernández

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
Dirección General de Cómputo y de Tecnologías de información y Comunicación
Dirección de Docencia en TIC

DGTIC

Educación Continua
1971 - 2021

Objetivo

Aprender los diferentes tipos de configuración basadas en java que nos ofrece Spring para un Bean.

Lo que veremos

- Configuraciones basadas en Java
- La vida de un Bean

Configuración del Bean basada en Java

Se analizo como hacer las configuraciones de un bean en XML, ahora lo haremos con configuración basada en Java.

Ocuparemos ahora anotaciones como @Bean, anotadas debajo de métodos y todos dentro de una clase con la anotación @Configuration. Todas estas definiciones de los beans serán una relación directa a los objetos reales que componen nuestra aplicación. Pueden ser objetos para una capa de servicio, acceso a datos (DAO) etc.

Objetivos de la práctica siguiente

Los objetivos de la practica son:

- Aprender como indicar al contenedor de Spring que registre un bean por configuración basada en Java.
- Como resolver el problema de tener beans de alcance singleton.
- Combinar configuraciones basada en Java con configuraciones XML
- Indicarle a Spring que descubra los beans que se crean en el contexto de aplicación.

Ejercicio 12: Configuración basada en Java (@Configuration,@Bean)

Objetivos de la práctica siguiente

Los objetivos de la practica son:

- Crear una clase POJO con @Component para obtener beans DAO.
- Ocupar @Autowired para brindar un servicio con DAO.
- @Autowired en un método con el atributo required.
- @Autowired en el constructor.
- Resolviendo ambigüedad en @Autowired con @Primary y @Qualifier
- @Qualifier por constructor, propiedad y método.

Ejercicio 13: POJO, @Autowired

¿Qué es Spring Boot?

Spring es una Framework muy eficiente y potente, pero su configuración y preparación para aplicaciones en producción lleva mucho tiempo y esfuerzo.

Es donde entra Spring Boot, la configuración automática de Spring Boot ha reducido drásticamente la cantidad de configuración explícita (ya sea con XML o Java) necesaria para crear una aplicación.

También permite distribuir nuestra aplicación Web en un archivo .jar, esto es embebiendo el servidor de aplicaciones web(Tomcat) dentro del jar.

¿ Que es Spring MVC ?

Con Spring MVC nos ayudara a construir aplicaciones de base Web, teniendo una flexibilidad y acoplamiento ligero como lo que garantiza el Framework de Spring.

Model-View-Controller (MVC), este patrón de diseño desacopla la lógica de negocios de la interfaz de usuario, separando los roles del modelo, vista y controlador en una aplicación.

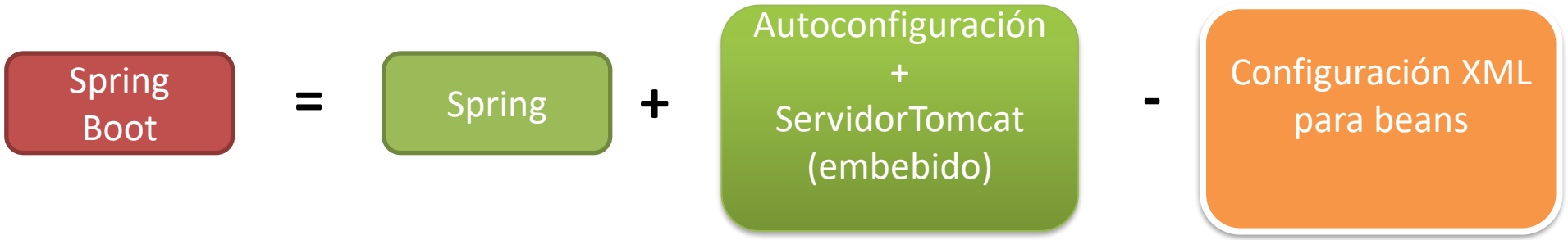
En una aplicación Spring MVC, los modelos generalmente consisten en objetos de dominio que son procesados por capa de servicio y la capa de persistencia, las vistas suelen ser paginas HTML o plantillas JSP escritas con el estándar Java.

Nos permite definir servicios web RESTful (se verán más adelante).

Diferencia entre Spring, Spring Boot y MVC

Podemos concluir entonces lo siguiente:

Spring es un Framework que nos permite crear aplicaciones empresariales en Java, este a su vez está constituido por varios módulos, entre los cuales “spring-web, spring-webmvc” son los encargados de poder construir y ocupar aplicaciones de base Web. Dado que la configuración es tediosa y complicada ocupamos Spring Boot para la conformación automática de nuestra aplicación empresarial.



Starters en Spring Boot

Las dependencias que manejamos en Spring, se proporcionan de una manera muy fácil con Spring Boot, estas dependencias las llamaremos starters, estas se agregan en Maven, o en Gradle.

Esto nos permite ocupar dependencias de Spring o terceros, estos están configurados para tener una minimización en la configuración. Básicamente los podemos dividir en:

- Application Starters
- Production Starters
- Technical Starters

Algunos Starters en Spring Boot

Application Starters:

- spring-boot-starter-thymeleaf: Para construir aplicaciones Web con el patrón MVC ocupando Thymeleaf
- spring-boot-starter-web-services: Ocupar servicios Web.

Production Starters:

- spring-boot-starter-actuator: Podemos monitorear y administrar nuestra aplicación.

Technical Starters:

- spring-boot-starter-tomcat: Utilizar Tomcat como un servidor embebido de Servlet.