Desarrollo de Aplicaciones autoconfigurables con SpringBoot

UNRN Universidad Nacional de Río Negro

LIA LABORATORIO DE INFORMÁTICA APLICADA

UNRN

Equipo

LIA LABORATORIO DE INFORMÁTICA APLICADA

- Ale Hualquilican
- Horacio Muñoz
- José Luis Cruz
- Mauro Cambarieri

Cronograma

Inicio	Inicio: 13/06/2023	Finalización 6/07/2023	
Semana	**Dia**	**Fecha**	**Temas**
1	martes	13/6/2023	Módulo 1 : fundamentos y conceptos
	jueves	15/6/2023	Módulo 1 : Instalación ejemplos
2	martes	20/6/2023	Módulo 2 : Modelado, conceptos, definiciones
	jueves	22/6/2023	Módulo 2 : Modelado, persistencia de datos
3	martes	27/6/2023	Módulo 3 :Servicios de Consultas
	jueves	29/6/2023	Módulo 3 :Servicios de Consultas, ejemplos.
4	martes	4/7/2023	Módulo 4 : Proyecto Web - Introducción a API rest
	jueves	6/7/2023	Módulo 4 : Proyecto Web - Introducción a API rest +
			Módulo extra: Herramienta para la generación de una App Web basada en un modelo de clases.

- **Modalidad de aprobación**
  - Completar correctamente los cuestionarios por módulos.
  - Completar una encuestas
  - Entrega del trabajo final (desarrollado en el transcurso del taller)

UNRN

Agenda

LIA LABORATORIO DE INFORMÁTICA APLICADA

- Backend development
- Abstracción
- Java Enterprise Edition
- Arquitectura
- Que es un Framework?
- Que dice Spring de Spring… Spring Framework Overview
- Evolución
- Spring Framework (+ características)
- Beneficios de Spring Framework
- Qué nos aporta Spring
- El contenedor IoC
- Entorno de desarrollo

Backend development

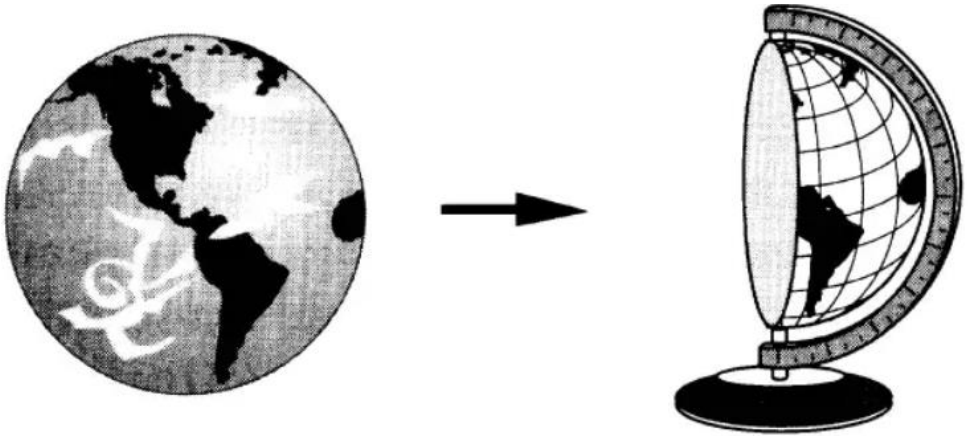
"..es al área lógica de toda la web, a la parte del servidor, dicho de otra manera, a la parte que pertenece al servidor. Esto es lo que asegura que todos los elementos desarrollen la función de forma correcta. Hablamos, por decirlo de alguna forma, de las "tripas" de las webs, lo que no ve el usuario." (https://www.inesdi.com/blog/backend-development-que-es/)

"El backend developer es un perfil profesional muy demandado en estos momentos, el programador que se encarga de que todo funcione correctamente en un sitio web o en una aplicación. Aunque no podamos apreciar su trabajo directamente, sin su labor, ninguna de las apps de nuestro móvil funcionaría." (https://www.edix.com/es/instituto/backend-develope)

."La separación del sistema en front ends y back ends es un **tipo de abstracción** que ayuda a mantener las diferentes partes del sistema separadas. La idea general es que el front end sea el responsable de recolectar los datos de entrada del usuario, que pueden ser de muchas y variadas formas, y los transforma ajustándolos a las especificaciones que demanda el back end para poder procesarlos, devolviendo generalmente una respuesta". Wikipedia

"El **Back End Developer** es la persona encargada de la implementación de un sitio web o aplicación web en todos sus componentes, y se ocupa de diseñar la lógica y las soluciones para que las acciones solicitadas en un sitio o aplicación web sean ejecutadas correctamente." www.soyhenry.com

Abstracción



Abstracción

UNRN

LIA LABORATORIO DE INFORMÁTICA APLICADA

- ¿De qué hablamos cuando hablamos de abstracción?

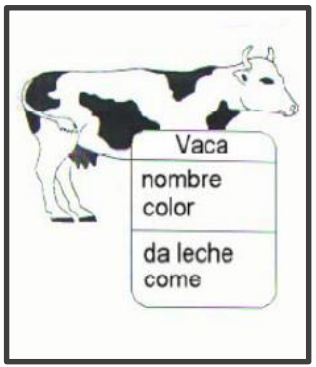
Es el proceso [intencional] de supresión de detalles respecto de un fenómeno, entidad o concepto.

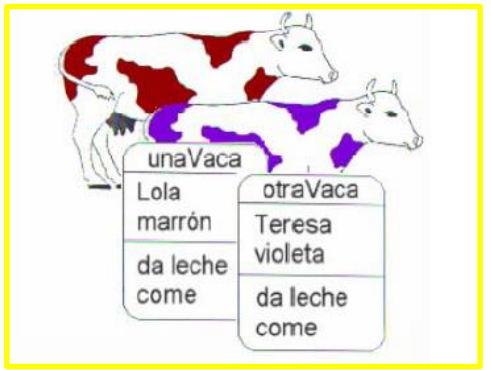
El objetivo es concentrarse en los aspectos más significativos

Algunas definiciones:

- Un concepto general formado a partir de la extracción de aspectos comunes tomadas de ejemplos específicos.

- El proceso mental donde las ideas son separadas de los objetos concretos





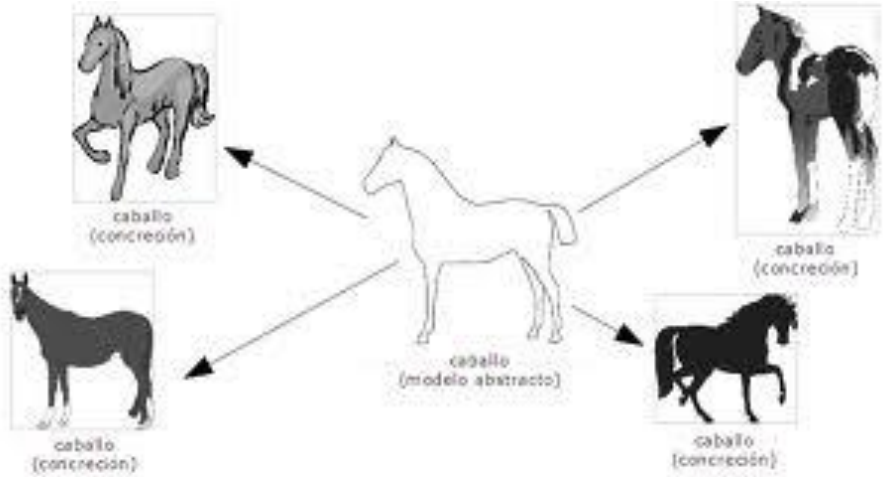
Abstracción

UNRN

LIA LABORATORIO DE INFORMÁTICA APLICADA

Abstraer determinadas características de las entidades de nuestro dominio, para modelar un concepto mas general que las abarque

Lo que buscamos es separar efectivamente los objetos concretos de la idea que queremos representar.

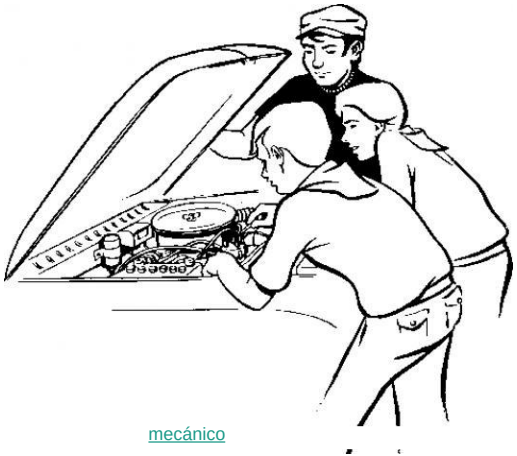


FACIL!!!!

Algo que hacemos dia a dia

La abstracción es completamente subjetiva y dependiente del contexto

Ejemplo: concepto de automóvil y diferentes observadores



mecánico

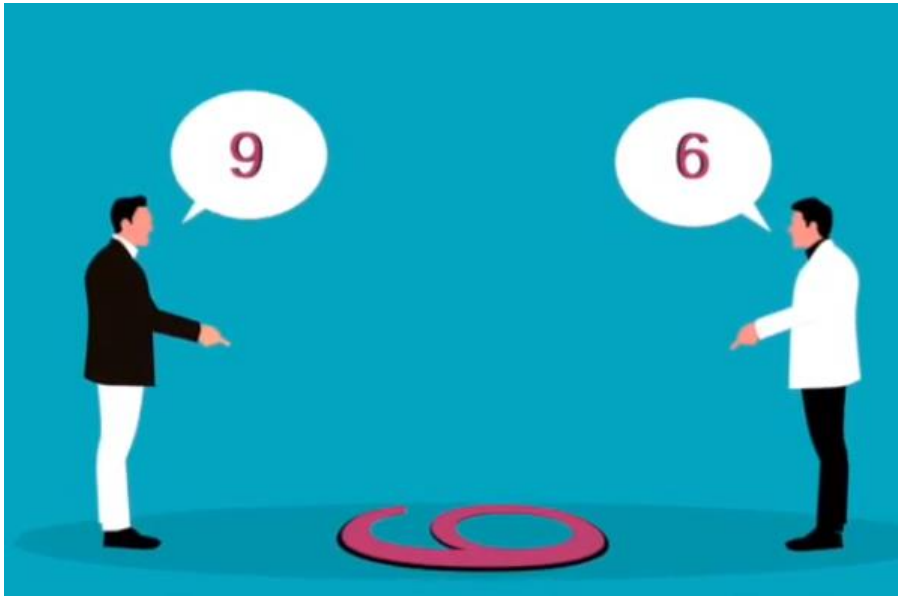
trasporte

Abstracción

UNRN

LIA LABORATORIO DE INFORMÁTICA APLICADA

La abstracción es completamente subjetiva y dependiente del contexto



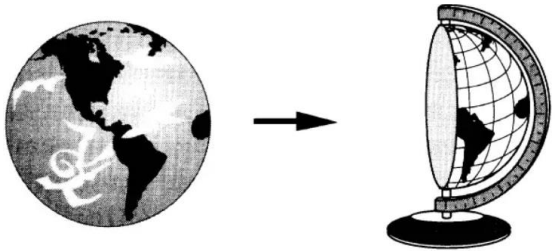
Conceptos básicos - Modelo

UNRN

LIA LABORATORIO DE INFORMÁTICA APLICADA

¿Qué es un modelo?

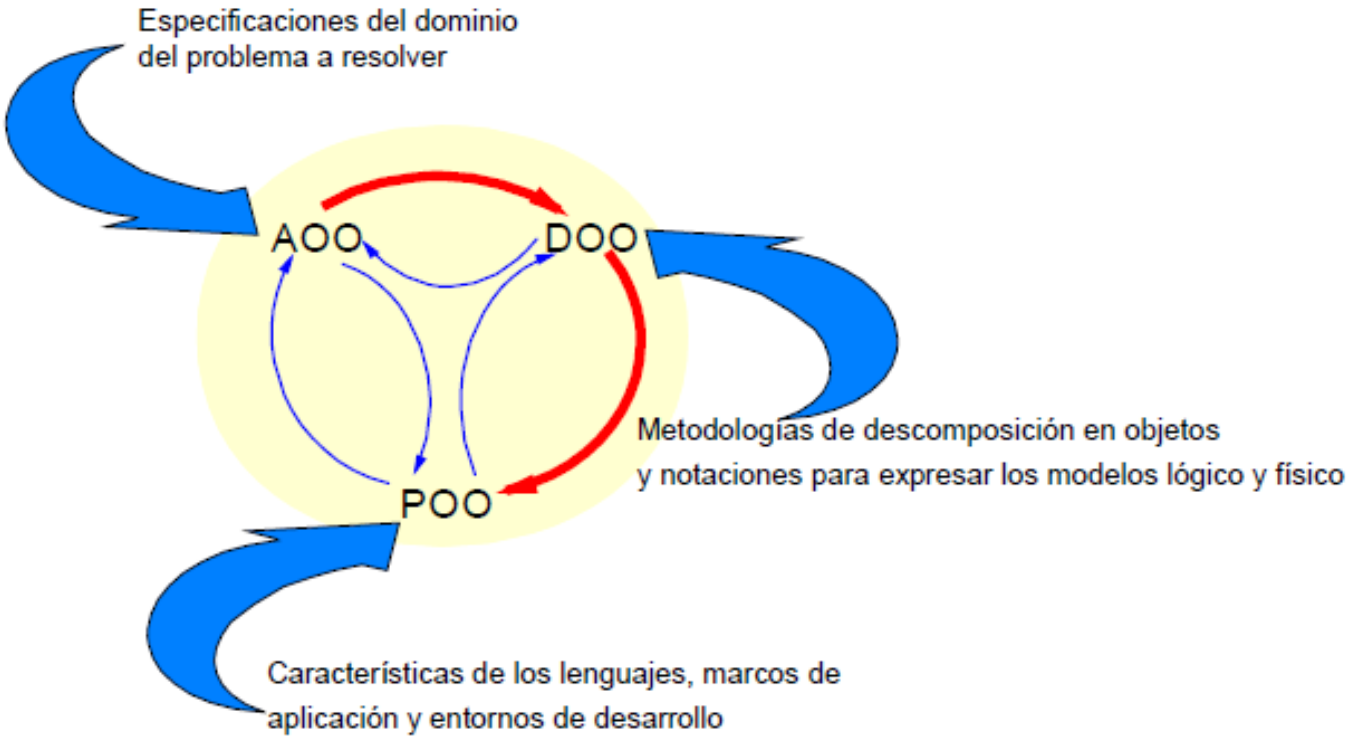
Es una versión simplificada de algún fenómeno o entidad del mundo real



¿Qué significa modelar?

- Es un proceso de abstracción (simplificación)
- Tomamos una versión reducida de lo que queremos representar. Sólo especificamos aquellas cosas que son relevantes.
- Subjetividad. El modelo nunca es 100% fiel a lo que estamos representando

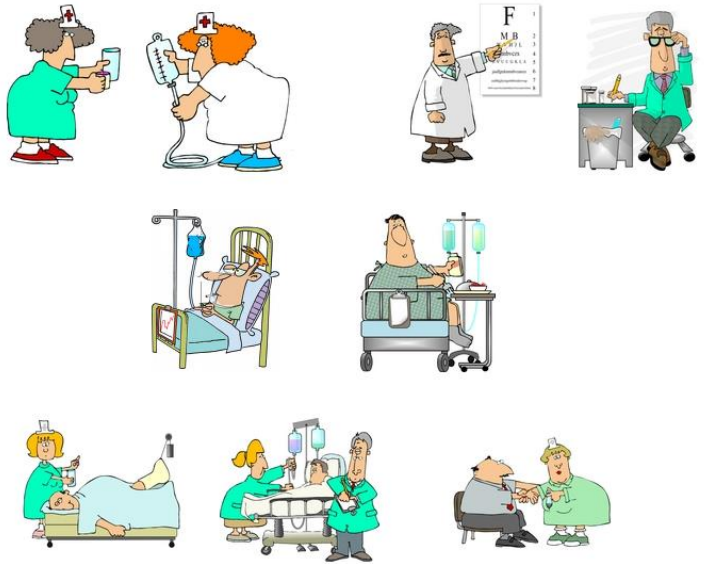
3 Etapas
Análisis – Diseño – Programación



Como se relacionan AOO- DOO-POO?

Análisis Orientado a Objetos(AOO)

Esta orientado a definir el problema, intentando el mejor entendimiento con el cliente



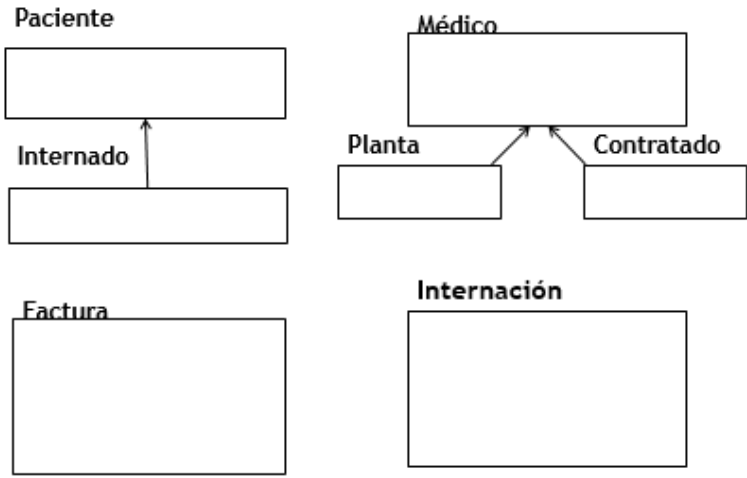
Preguntas que se deben plantear

- ¿Cuál es el comportamiento que se desea en el sistema?
- ¿Qué objetos existen en el sistema?
- ¿Cuáles son las misiones de los objetos para llevar a cabo el comportamiento deseado del sistema?

Como se relacionan AOO- DOO-POO?

Diseño Orientado a Objetos

Esta orientado a construir un modelo que permita razonar sobre el problema y guiar la implementación

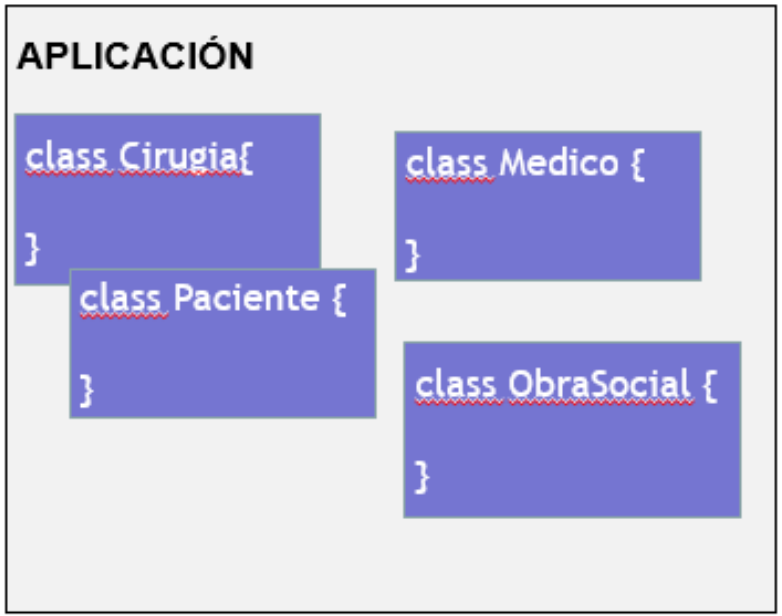


Preguntas que se deben plantear
- ¿Qué clases de objetos existen y como se relacionan ?
- ¿Dónde debería declararse y construirse cada clase y objeto?
- ¿A qué procesador debería asignarse un proceso, y para un procesador dado,
cómo deberían planificarse sus múltiples procesos?

Como se relacionan AOO- DOO-POO?

Programación Orientado a Objetos

Esta orientado a la implementación del diseño en un lenguaje de Programación y a las herramientas necesarias para el desarrollo



Tecnologías Java

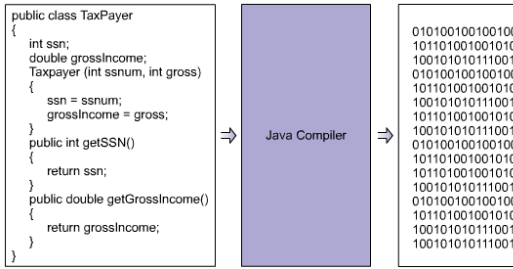
La plataforma JAVA está compuesta por 3 elementos:

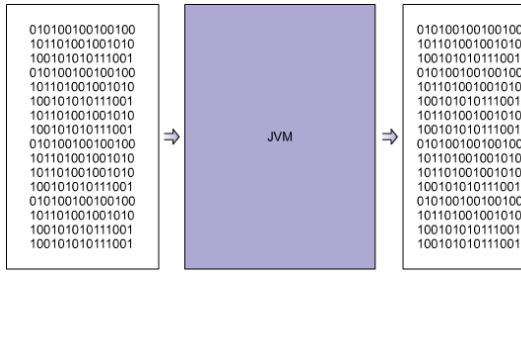
- **El Lenguaje de Programación**
- **Una Máquina Virtual**

Es el fundamento de la plataforma Java. Posee las siguientes características:

a) la Máquina Virtual está implementada para distintos SO y hardware, permitiendo así compatibilidad binaria de aplicaciones java. Las aplicaciones funcionan en forma consistente sobre distintas implementaciones.

b) la Máquina Virtual provee un control estricto del código binario que ejecuta, permitiendo una ejecución segura de código no-confiable



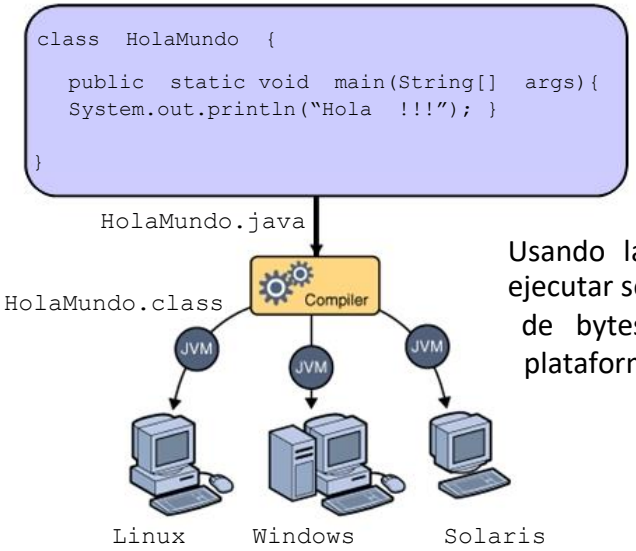


- **Una Interface de Programación estándar y amplia (API's)**

Provee un amplio soporte para implementar aplicaciones, desde librerías para escribir interfaces de usuario, criptografía, conectividad, internacionalización, etc

La Plataforma Java

Existen diferentes Máquinas Virtuales Java para los diferentes sistemas operativos. Los mismos archivos .class pueden ejecutar en los distintos sistemas operativos, como Microsoft Windows, Solaris, Linux o Mac, sin ninguna compilación previa.



Usando la JVM, la misma aplicación es capaz de ejecutar sobre las distintas plataformas => el código de bytes (.class) es independiente de la plataforma.

Tecnologías Java

La plataforma JAVA está compuesta por 3 elementos:

- **JEE(Java Entreprise Edition):** está diseñada para programar y ejecutar aplicaciones empresariales, caracterizadas por ser multiusuario y distribuidas. El procesamiento de estas aplicaciones se realiza en un servidor. Usualmente son aplicaciones web.

- **Java Platform, Standard Edition (Java SE) :** está diseñada para programar y ejecutar aplicaciones de escritorio JAVA. Típicamente son programas que se ejecutan en una PC. Está compuesta por el JRE y el JDK.

- **Java Embedded:** está diseñada para programar y ejecutar aplicaciones para dispositivos con recursos de cómputo limitados, como pueden ser teléfonos celulares, palms, pdas, etc. Estos dispositivos cuentan con poca memoria RAM, pantallas muy chicas inclusive algunos carecen de ellas, la conexión de red puede ser intermitente, etc.

https://docs.oracle.com/en/java/index.html

Plataforma

Java Enterprise Edition

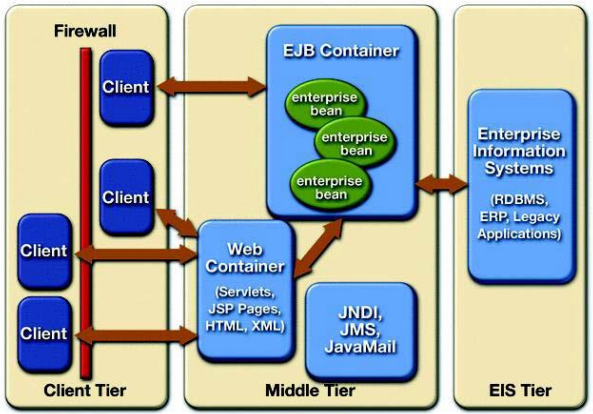
Java EE (*Enterprise edition*) es una especificación Java para el desarrollo de aplicaciones empresariales.

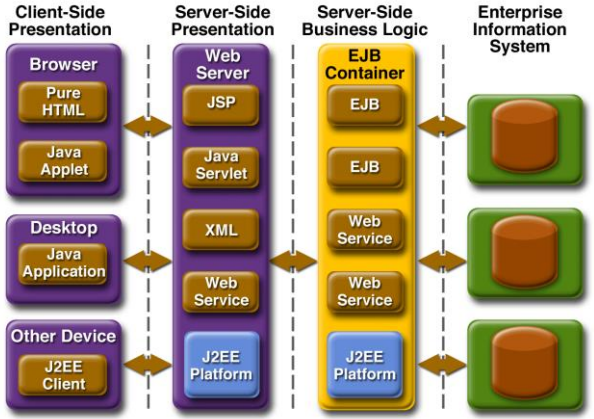
Java EE especifica una serie APIs tales como JDBC, RMI, e-mail, JMS, servicios web, XML, … que todos los programas deben cumplir, es decir un estándar que todos los fabricantes deben cumplir, esto hace que las aplicaciones sean portables e independientes de la plataforma

https://www.oracle.com/java/technologies/appmodel.html

Plataforma
Java Enterprise Edition

- La arquitectura JEE implica un modelo de aplicaciones distribuidas en diversas capas o niveles (tier).
- La capa cliente admite diversas tipos de clientes (HTML, Applet, aplicaciones Java, etc.).
- La capa intermedia (middle tier) contiene subcapas (el contenedor web y el contenedor EJB).
- La tercera capa dentro de esta visión sintética es la de de aplicaciones backend como ERP, EIS, bases de datos, etc





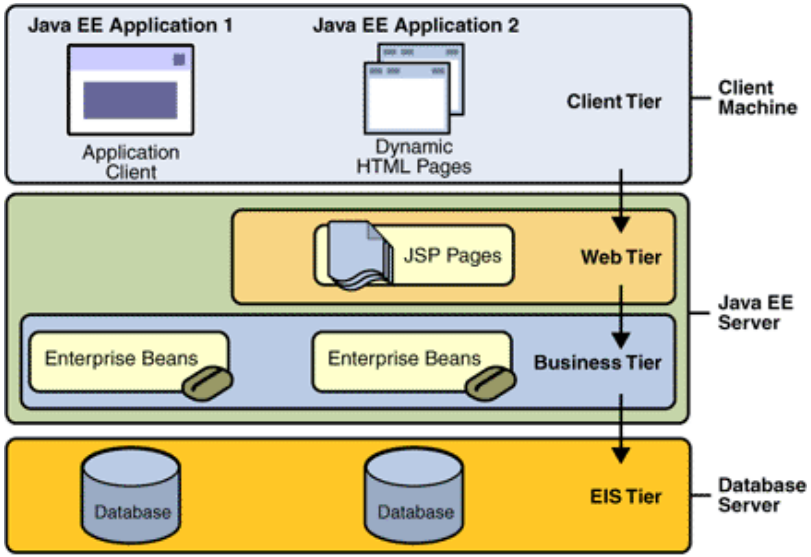
Enterprise Application Model

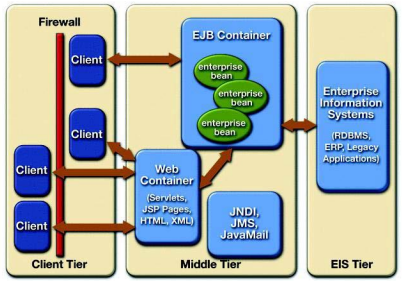
https://www.oracle.com/java/technologies/appmodel.html

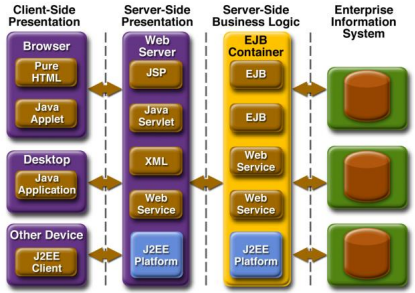
Plataforma
Java Enterprise Edition

- **Arquitectura JEE**: entorno de ejecución estandarizado

La visión de la arquitectura es un esquema lógico, no físico.
Cuando hablamos de capas nos referimos sobre todo a servicios diferentes (que pueden estar físicamente dentro de la misma máquina)







Enterprise Application Model

https://www.oracle.com/java/technologies/appmodel.html

UNRN

LIA LABORATORIO DE INFORMÁTICA APLICADA

Que es un Framework?

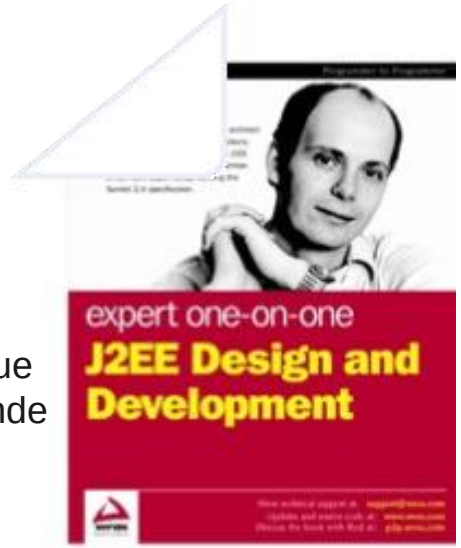
“Un entorno de trabajo (del inglés framework) o marco de trabajo es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar”.(wikipedia)

“En el desarrollo de software, un entorno de trabajo es una estructura conceptual y tecnológica de asistencia definida, normalmente, con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. ” ”.(wikipedia)

Que es un Framework?

“Una forma especialmente valiosa de aprovechar los componentes existentes, ya sean de terceros o desarrollados internamente, es construir dentro de un marco. Un marco es una arquitectura genérica que constituye la base de aplicaciones específicas dentro de un dominio o área tecnológica.”.(Rod Johnson,)

“Los Frameworks son concretos, no abstractos.
Se puede tomar un framework existente y construir una aplicación con él añadiendo código adicional. Normalmente consiste en implementar interfaces o subclases de clases del framework.”



Johnson es un especialista informático australiano que creó Spring Framework y cofundó SpringSource, donde se desempeñó como director ejecutivo hasta su adquisición en 2009 por parte de VMware

UNRN

LIA LABORATORIO DE INFORMÁTICA APLICADA

Que es un Framework?

“Los Frameworks suelen ser específicos de un dominio o una tecnología” .

Adoptar un buen framework que se adapte bien puede reducir drásticamente el tiempo de desarrollo de un proyecto. Los problemas de diseño problemas de diseño más difíciles pueden haberse resuelto, basándose en las mejores prácticas reconocidas. Gran parte de la ejecución del proyecto se dedicará a rellenar los huecos, lo que no debería implicar tantas decisiones de diseño difíciles.

“A good framework simplifies application code”

UNRN

Que dice Spring de Spring…

LIA LABORATORIO DE INFORMÁTICA APLICADA

Spring Framework Overview

Spring facilita la creación de aplicaciones empresariales Java. Proporciona todo lo necesario para adoptar el lenguaje Java en un entorno empresarial, con soporte para Groovy y Kotlin como lenguajes alternativos en la JVM, y con la flexibilidad necesaria para crear muchos tipos de arquitecturas en función de las necesidades de una aplicación. A partir de Spring Framework 6.0, Spring requiere Java 17+.

Spring es de código abierto. Cuenta con una comunidad grande y activa que proporciona información continua basada en una amplia gama de casos de uso en el mundo real. Esto ha ayudado a Spring a evolucionar con éxito durante mucho tiempo.

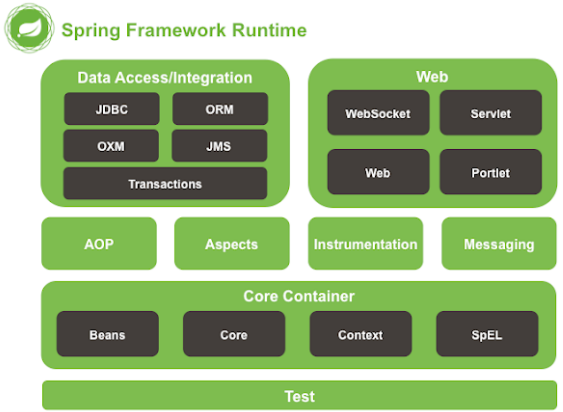
https://docs.spring.io/spring-framework/reference/overview.html

Que dice Spring de Spring…

What We Mean by "Spring"

El término "Spring" significa cosas distintas. Puede utilizarse para referirse al propio proyecto Spring Framework, que es donde empezó todo. Con el tiempo, se han creado otros proyectos Spring sobre Spring Framework. La mayoría de las veces, cuando la gente dice "Spring", se refiere a toda la familia de proyectos.

Spring Framework está dividido en **módulos**. Las aplicaciones pueden elegir qué módulos necesitan. En el núcleo se encuentran los módulos de contenedor central(core container), que incluyen un modelo de configuración y un mecanismo de inyección de dependencias. Además, de soporte básico para diferentes arquitecturas de aplicaciones, como mensajería, datos transaccionales , persistencia, y web.



https://docs.spring.io/spring-framework/reference/overview.html

Evolución

2002 Spring Framework	2004 Spring 1.0	2006 Spring 2.0	2009 Spring 3.0	2013 Spring 4.0	2017 Spring 5.0	2022 Spring 6.0

JDK Version

Spring Framework 5.3.x: JDK 8-21

Spring Framework 6.0.x: JDK 17-21

https://spring.io/projects

https://github.com/spring-projects/spring-framework/wiki/Spring-Framework-Versions

UNRN

Que dice Spring de Spring…

LIA LABORATORIO DE INFORMÁTICA APLICADA

Design Philosophy

1. Proporciona opciones a todos los niveles. Spring permite posponer las decisiones de diseño tanto como sea posible..

2. Adaptarse a diversas perspectivas

3. Mantiene una sólida compatibilidad con versiones anteriores.

4. Diseño cuidadoso de la API. El equipo de Spring dedica mucho tiempo y esfuerzo a crear API intuitivas y que se mantengan a lo largo de muchas versiones y muchos años.

5. Establecer altos estándares de calidad del código.

https://docs.spring.io/spring-framework/reference/overview.html#overview-philosophy

Contexto

¿Qué es Spring?

Inicialmente, un ejemplo hecho para el libro “J2EE design and development” de Rod Johnson, que defendía **alternativas** a la “visión oficial” de **aplicación JavaEE basada en EJBs**

https://spring.io/projects/spring-framework

Contexto

¿Qué es Spring?

Liviano y no intrusivo

Su principal razón:

- Inyección de dependencias ID
- Inversión de Control IoC
- Programación orientada a aspectos
- Flexible, modular.

“…su foco principal es la inversión del control mediante la inyección de dependencias”

UNRN

Motivación

LIA LABORATORIO DE INFORMÁTICA APLICADA

Ampliamente aceptado y demanda

Facilidad de uso y productividad

Inyección de dependencias y modularidad

Integración con tecnologías actuales

Comunidad y soporte:

Arquitectura empresarial

Flexibilidad y extensibilidad

Testing y calidad del código

Aprendizaje de conceptos clave

Spring Framework (+ caracteristicas)

Es muy popular por:

El enfoque de inyección de dependencia de Spring fomenta la escritura de código comprobable

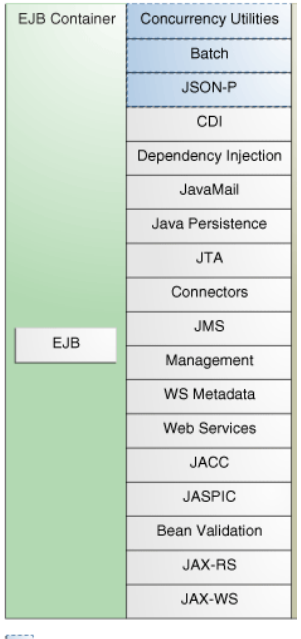
fácil de usar, pero con potentes capacidades de gestión de transacciones de bases de datos

Spring simplifica la integración con otros marcos Java como JPA/Hibernate ORM, Struts/JSF/etc. marcos web

Framework Web MVC de última generación para crear aplicaciones web

Spring vs. JavaEE estándar

Spring es un *framework* alternativo al *stack* de tecnologías estándar en aplicaciones JavaEE. Nació en una época en la que las tecnologías estándar JavaEE y la visión "oficial" de lo que debía ser una aplicación Java Enterprise tenían todavía muchas aristas por pulir. Los servidores de aplicaciones eran monstruosos devoradores de recursos y los EJB eran pesados, inflexibles y era demasiado complejo trabajar con ellos



Qué nos aporta Spring

Desde un punto de vista amplio, Spring se puede ver como un soporte que nos

POJO

Estereotipos configurables:

Servicios Enterprise

Inyección de dependencias

+

Blueprints

Plain Old Java Object. Un POJO es una sigla creada por Martin Fowler, Rebecca Parsons y Josh MacKenzie en septiembre de 2000 y utilizada por programadores Java para enfatizar el uso de clases simples y que no dependen de un framework en especial.(Wikipedia.)

UNRN

La filosofía de Spring

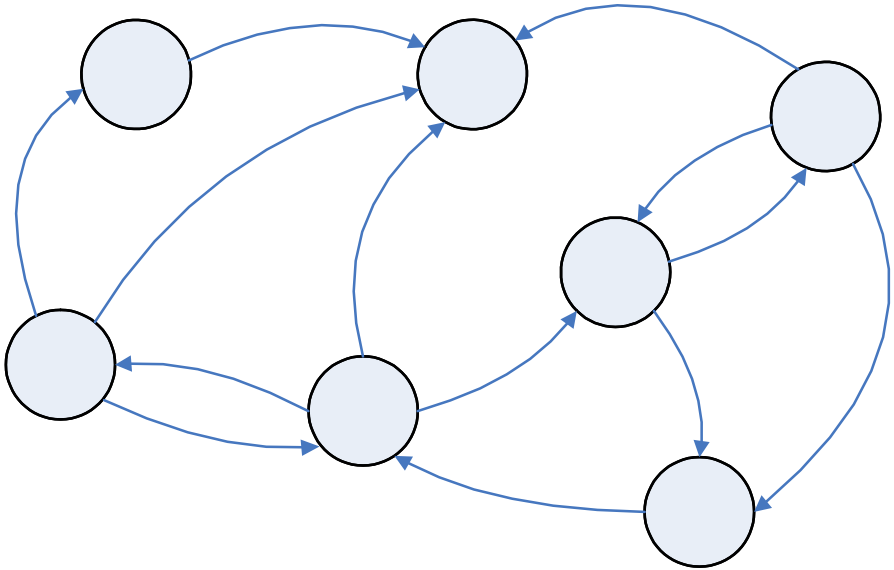
LIA LABORATORIO DE INFORMÁTICA APLICADA

- Utilizar entonces, **los objetos de negocio** que deberían ser **POJOs**
- **Inyección** como medio de resolver **dependencias** (colaboración entre objetos)
- Cuando ya hay algo que funciona, incorpóralo a tu solución, no “reinventes la rueda” (integrar tecnología de terceros).

Programa Orientado a Objetos

¿Qué es un programa en el paradigma OO?

Un conjunto de objetos que colaboran enviándose mensajes

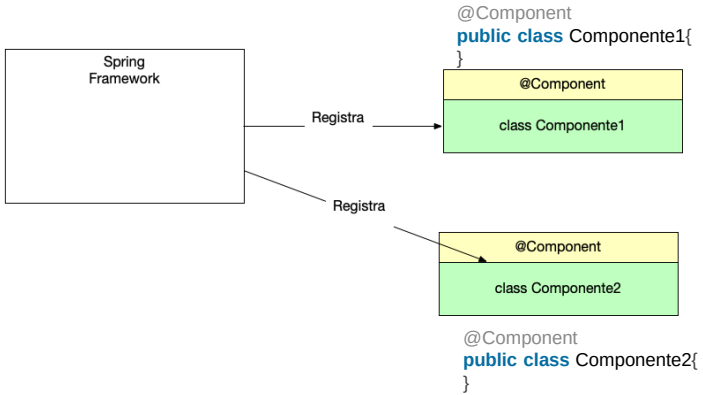


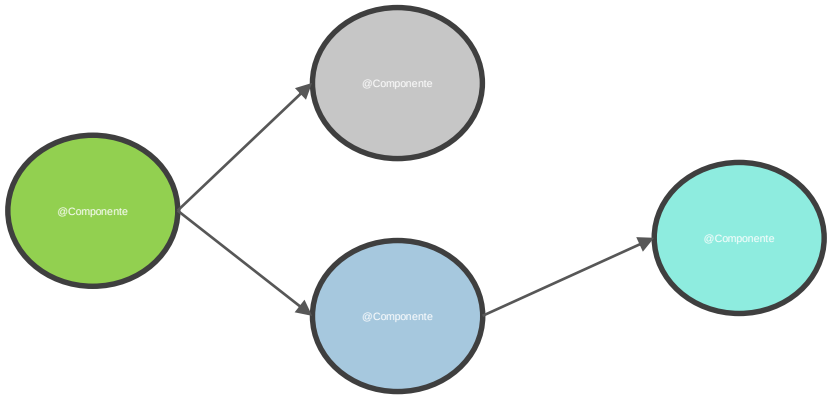
Spring Framework

El contenedor IoC

En Spring, los objetos que forman la columna vertebral de la aplicación y que son gestionados por el **contenedor IoC** de Spring se denominan **beans**.

Un **bean** es un objeto “instanciado”, “ensamblado” y “gestionado” por un **contenedor IoC**. De lo contrario, un bean es simplemente uno de los muchos objetos de la aplicación. Los beans, y las dependencias entre ellos, se reflejan en los metadatos de configuración utilizados por un contenedor.





https://www.arquitecturajava.com/spring-component-anotaciones-y-jerarquia/

https://docs.spring.io/spring-framework/reference/core/beans/introduction.html

Spring Framework

El contenedor IoC

Principio de Inversión de Control (IoC). IoC también se conoce como inyección de dependencia (DI).

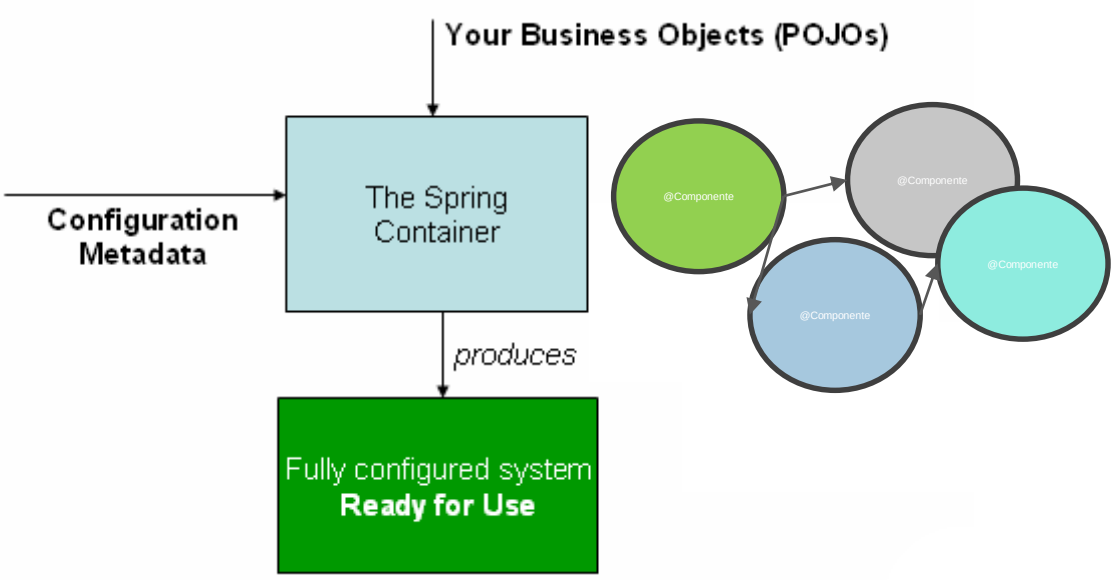


Figure 1. The Spring IoC container
https://docs.spring.io/spring-framework/reference/core/beans/basics.htmll

Los metadatos puede ser basados: en XML o Anotaciones

```
<bean id="componente1"
class="org.springframework.Componente1">
   <property name="accountRepo" ref="accountRepo"/>
<!-- additional collaborators and configuration for this
bean go here -->
</bean>
```

```
@Component
public class Componente1{
   AccountRepo accountRepo;
}
```

Dependency Injection

La inyección de dependencias (DI)

Normalmente un objeto de negocio necesita la colaboración de otros objetos de negocio.

DI existe en dos variantes principales:

- Inyección de dependencia basada en constructor (**Constructor-based Dependency Injection**)
- Inyección de dependencia basada en Setter.(**Setter-based Dependency Injection**)

“Confiamos” en que “alguien” instancie y nos pase el objeto que necesitamos para trabajar

En nuestro caso ese “alguien” es Spring Core

Dependency Injection

La inyección de dependencias (DI)

Constructor-based Dependency Injection

```
public class SimpleMovieLister {

    // the SimpleMovieLister has a dependency on a MovieFinder
    private final MovieFinder movieFinder;

    // a constructor so that the Spring container can inject a MovieFinder
    public SimpleMovieLister(MovieFinder movieFinder) {
        this.movieFinder = movieFinder;
    }

    // business logic that actually uses the injected MovieFinder is omitted...
}
```

Tenga en cuenta que no hay nada especial en esta clase. Es un POJO que no tiene dependencias en interfaces, clases base o anotaciones específicas del contenedor.

Dependency Injection

La inyección de dependencias (DI)

Properties and Setter-based Dependency Injection

```
public class SimpleMovieLister {

    // the SimpleMovieLister has a dependency on the MovieFinder
    private MovieFinder movieFinder;

    // a setter method so that the Spring container can inject a MovieFinder
    public void setMovieFinder(MovieFinder movieFinder) {
        this.movieFinder = movieFinder;
    }

    // business logic that actually uses the injected MovieFinder is omitted...
}
```

Tenga en cuenta que no hay nada especial en esta clase. Es un POJO que no tiene dependencias en interfaces, clases base o anotaciones específicas del contenedor.

Definir los *beans*

Configuración de contenedor basada en anotaciones

Using @Autowired

```
public class MovieRecommender {

private final CustomerPreferenceDao customerPreferenceDao;

@Autowired
public MovieRecommender(CustomerPreferenceDao
customerPreferenceDao) {
   this.customerPreferenceDao = customerPreferenceDao;
}
// ...
}
```

```
public class SimpleMovieLister {

   private MovieFinder movieFinder;

   @Autowired
  public void setMovieFinder(MovieFinder movieFinder)
{
    this.movieFinder = movieFinder;
}
 // ...
}
```

```
public class SimpleMovieLister {

 @Autowired
  private MovieFinder movieFinder;


    public void setMovieFinder(MovieFinder movieFinder)
{
        this.movieFinder = movieFinder;
  }
 // ...
}
```

https://docs.spring.io/spring-framework/reference/core/beans/annotation-config/autowired.html

Definir los *beans*

Hay opciones para la configuración

Un **bean** en Spring es un componente cuyo ciclo de vida está “gestionado” por Spring

XML: la clásica. independiente del código fuente.
Anotaciones: más “sencilla” de usar

Definición de beans

```
<?xml version="1.0"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
<bean id=“usuarioService"
                class=“org.spring.negocio.UsuarioService">
</bean
</beans>
```

```
package org.spring.negocio;
@Service(“usuarioService”)
public class UsuarioService {

    UsuarioRepository usuarioRepository;

   public Usuario login(String login, String password) {
...   //validar
     usuarioRepository.buscaUsuario(login,password)
}
}
```

Definir los *beans*

Spring y Jerarquías.

Spring ofrece varios estereotipos

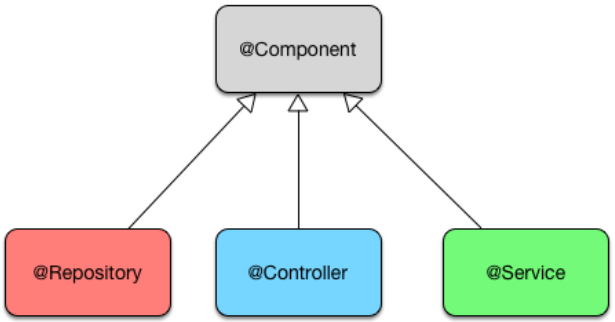
@Controller : realiza las tareas de controlador y gestión de la comunicación entre el usuario y la interfaz grafica(UI). Para ello se apoya habitualmente en algún motor de plantillas o librería de etiquetas que facilitan la creación de páginas.

@ RestController : es una especialización de @Controller que registrar los controladores que son los encargados de enlazarnos y registrar las diferentes URLS a las que el componente responde a través de los métodos.

@Service: identifica un componente de negocio.

@Repository: identifica una “parte” de nuestros objetos con interacción a u

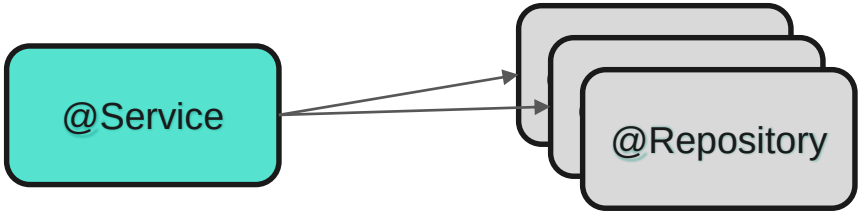
@Component: de propósito general



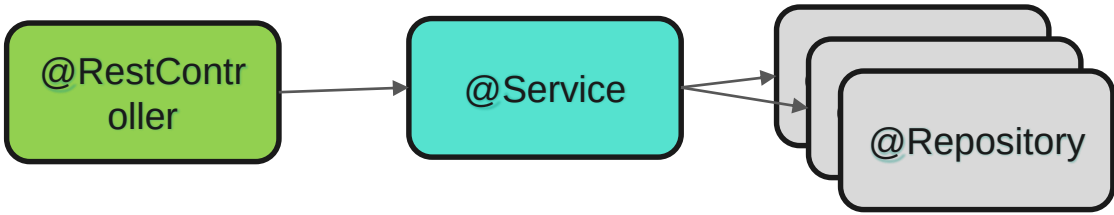
“Enlazando” los *beans*

```
@Repository
class UsuarioRepository {
}
```

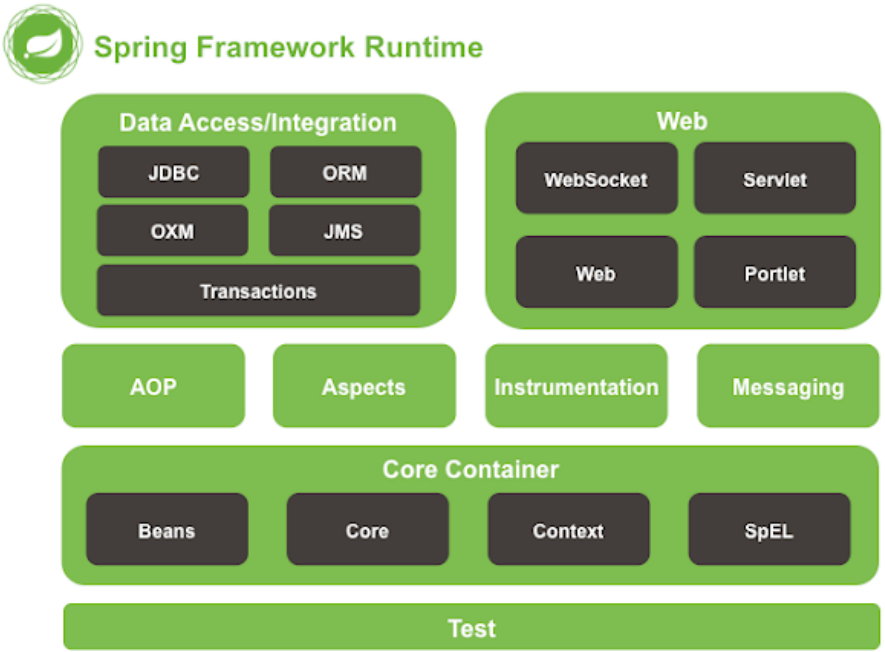

```
@Service
public class UsuarioService {
@Autowired
UsuarioRepository usuarioRepository;
}
```




```
@RestController
public class UsuarioREST {
@Autowired
UsuarioService usuarioService;
@RequestMapping(value = "/usuarios", ….)
public List < Usuario > lista() {
    return usuarioService.listar();
}
}
```



Descripción general de los módulos Spring Framework



Spring Core

Es un contenedor que gestiona el ciclo de vida de los objetos de nuestra aplicación

- En realidad no es más que un conjunto de librerías que se puede ejecutar en cualquier servidor web java
- Ofrece servicios a nuestros objetos, como inyección de dependencias

Juntándolo con otros módulos, más servicios

- (+ AOP): Transaccionalidad declarativa
- (+ Spring Security): Seguridad declarativa

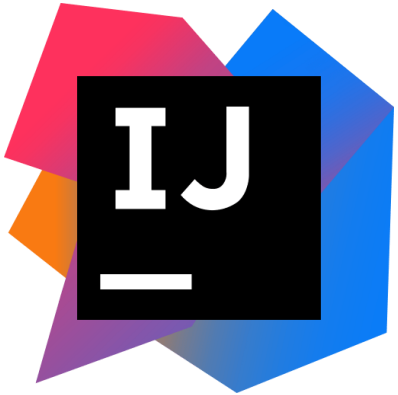
UNRN

Entorno de desarrollo

LIA LABORATORIO DE INFORMÁTICA APLICADA

Las herramientas a utilizar son:

- Maven
- IDE
- Conocer el lenguaje Java



UNRN

Spring Boot

LIA LABORATORIO DE INFORMÁTICA APLICADA

Spring Boot es básicamente una extensión del Framework Spring que eliminó las configuraciones repetitivas necesarias para configurar una aplicación Spring.

Spring Boot es un framework que ayuda a los desarrolladores a crear aplicaciones basadas en Spring de forma rápida y sencilla. El objetivo principal de Spring Boot es crear rápidamente aplicaciones basadas en Spring sin necesidad de que los desarrolladores escriban una y otra vez la misma configuración repetitiva.

UNRN

Spring Boot

LIA LABORATORIO DE INFORMÁTICA APLICADA

Objetivos principales

Proporcionar una experiencia de puesta en marcha radicalmente más rápida y ampliamente accesible para todo el desarrollo de Spring

Ser de fácil uso, pero salir del paso rápidamente cuando los requisitos empiecen a alejarse de los estándares fijados por defecto.

Proporcionar una serie de características no funcionales que son comunes a grandes tipos de proyectos (como servidores integrados, seguridad, métricas, comprobaciones de estado y configuraciones externas).

No generar código en absoluto y no requerir configuración XML..

Muchas Gracias!

UNRN Universidad Nacional de Río Negro

LIA LABORATORIO DE INFORMÁTICA APLICADA

Ing. Mauro Cambarieri
Universidad Nacional de Río Negro

maurocambarieri

UNRN Universidad Nacional de Río Negro

LIA LABORATORIO DE INFORMÁTICA APLICADA