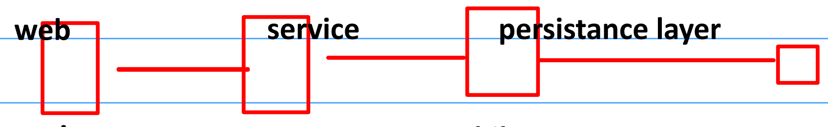
Day 14:
✓ spring di in details
✓ Spring AOP
✓ Spring Hibernate, tx mgt
✓ Spring MVC

task 1: run hello world of spring di example

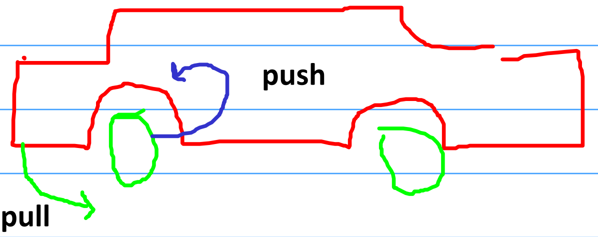


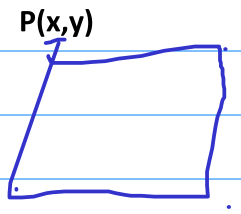
spring mvc

spring+hib
aop

hibernate

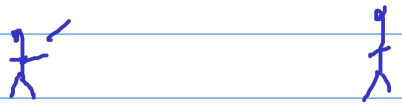
Spring





BeanFactory vs ApplicationContext

Bean factory is light wt spring container
that dont support some concept i18n, prop file



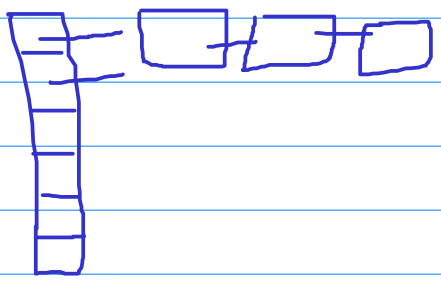
review of spring di
xml collection mapping in details

spring bean life cycle

Map<K, V> ✓ EntrySet is used to iterate the map

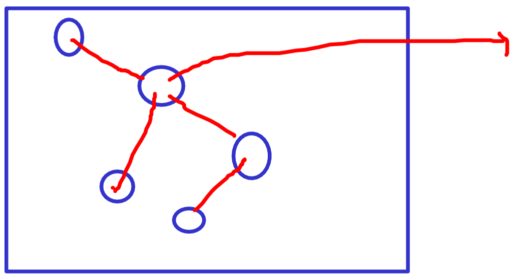
Map<String, List<String>>

KeySet



spring bean life cycle

Spring act as a container and mange life cycle of spring bean



life cycle of spring bean

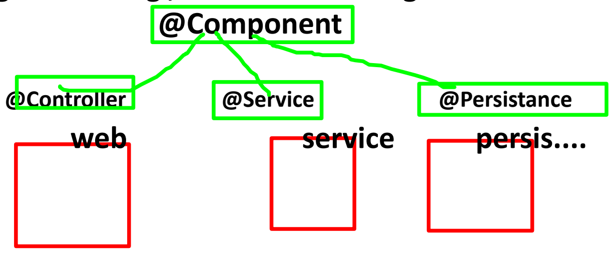
BeanPostProcessor

ctr of foo is called
setter of foo is called
postProcessBeforeInitialization is called..
@PostConstruct wala method is called
postProcessAfterInitialization is called..
foo value !
foo value !
@PreDestroy wala method is called

BeanFactoryPostProcessor

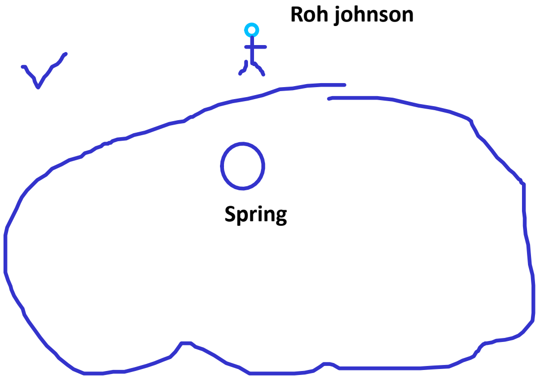
it is going to run just
once before any bean can be register in
the container...

Spring bean life cycle
Spring autowiring , annotation configuration



webconfig.xml ... ...

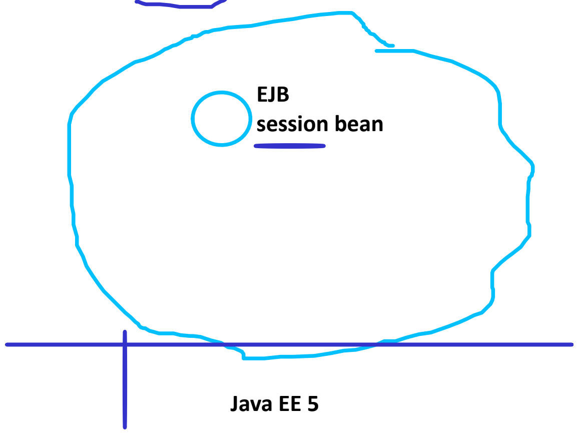
JSR 250



2003

J2ee

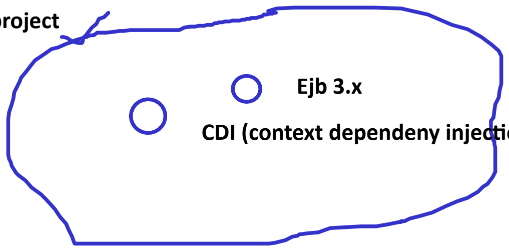
:(



Java EE 5

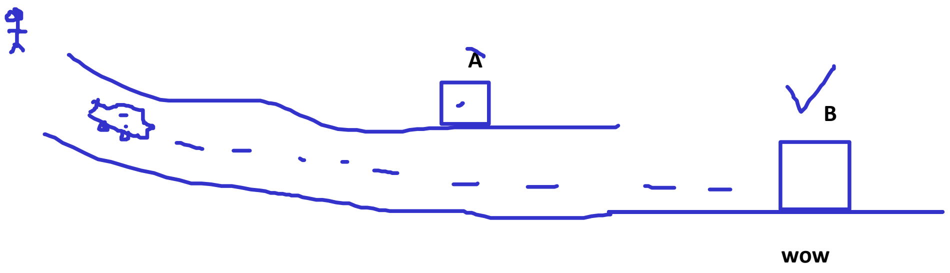
99%

1% project



CDI (context dependeny injection) JSR 250

"one most imp quality of champions : they bounce back"



1. xml
2. annotation
3. java config

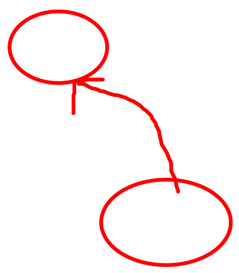
Bean inheritance :

```
<bean id="pshape" class="com.demo5.beaninheritance.Shape" abstract="true">
          <property name="p1" ref="p1"/>
    </bean>

    <bean id="shape" class="com.demo5.beaninheritance.Shape" parent="pshape">
          <property name="p2" ref="p2"/>
          <property name="p3" ref="p3"/>
    </bean>
    <bean id="p1" class="com.demo5.beaninheritance.Point">
          <property name="x" value="2" />
          <property name="y" value="12" />
    </bean>

    <bean id="p2" class="com.demo5.beaninheritance.Point">
          <property name="x" value="2" />
          <property name="y" value="-2" />
    </bean>


    <bean id="p3" class="com.demo5.beaninheritance.Point">
          <property name="x" value="92" />
          <property name="y" value="42" />
    </bean>
```



SpEL examples:

```
class Book {
     private int id;
     private String name;
     private double price;
}
```

```
class BookCollection {
       public List<Book> bookList;

       public Book getFirstBook() {
              return bookList.get(0);

       }
       public void setBookList(List<Book> bookList) {
              this.bookList = bookList;
       }
}
```

```
public class BookLib {
       private List<Book> allBooks;
       private Book firstBook;

       public void setAllBooks(List<Book> allBooks) {
              this.allBooks = allBooks;
       }

       public void setFirstBook(Book firstBook) {
              this.firstBook = firstBook;
       }
       public void printBookLib() {
              System.out.println("---first book-----");
              System.out.println(firstBook);

              System.out.println("-----alll books-----");
              for(Book temp: allBooks) {
                     System.out.println(temp);
              }
       }
}
```

```
<bean id="bookCollection" class="com.demo.BookCollection">
      <property name="bookList">
            <list>
                  <ref bean="book1" />
                  <ref bean="book2" />
            </list>
      </property>
  </bean>
  <!-- we want to map BookLib with book collection using SPEL -->
  <bean id="bl" class="com.demo.BookLib">
      <property name="allBooks" value="#{bookCollection.bookList}"/>
      <property name="firstBook" value="#{bookCollection.getFirstBook()}"/>
  </bean>

  <bean id="book1" class="com.demo.Book">
      <property name="id" value="121" />
      <property name="name" value="java basics" />
      <property name="price" value="300" />
  </bean>

  <bean id="book2" class="com.demo.Book">
      <property name="id" value="128" />
      <property name="name" value="spring basics" />
      <property name="price" value="600" />
  </bean>
```

```
<bean id="rect" class="com.demo.Rectangle">
          <property name="lengh" value="22"/>
          <property name="bredth" value="2"/>
      </bean>

      <!-- 2*(l+b) -->
      <bean id="peri" class="com.demo.Paremeter">
          <property name="perimeter" value="#{2*(rect.lengh+rect.bredth)}"/>
      </bean>
```

Spring provide 3 funda
✓1. DI

✓2. AOP

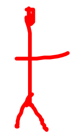
✓3. Reduction of boilerplat code(last recording)
spring jdbc
Spring hibernate

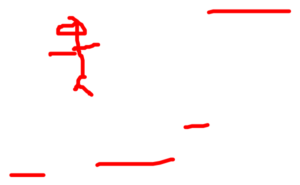
Aspect oriented programming? vs OO
in fact aop is helping oo to do better programing

Aspect = advice + point cut

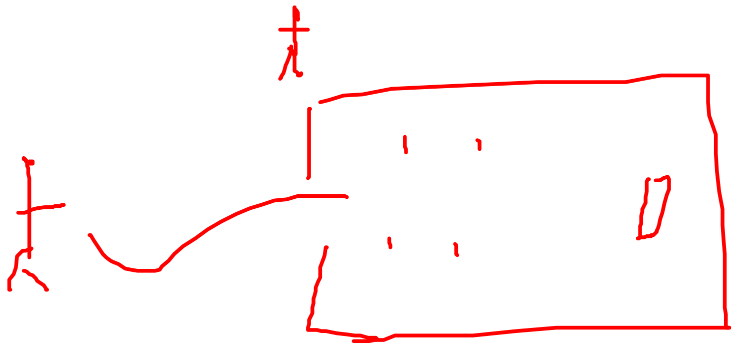
jointput
pointcut

doMagic(){
}





clapping(){}



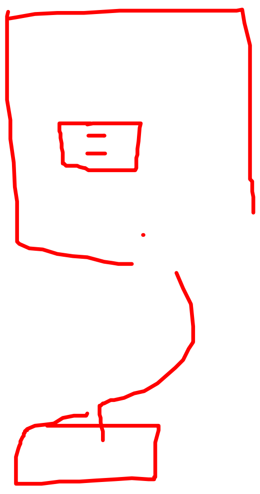
bringing infra
is a ccc for me

project
whiteboard

proxy dp GOF

Atm card

Aspect J + Spring



ccc
it can
logging
caching
tx
sec

$\boxed{\text{Aspect= advice + pointcut}}$

when to
apply

jp > pc

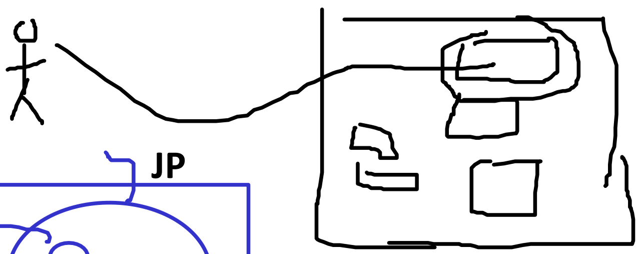
pointcut

joinput

ccc

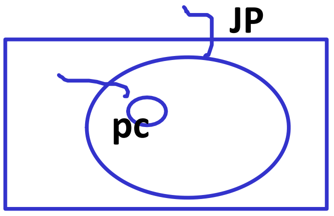
target

Advice : head first



all the items in the menu
is aka jointpoint

pointcut: somthing that ordered



What is JP?
it is point of execution in a programm in which behaviour can be alter by AOP

in spring jp is always method execution...

pc? pointcut

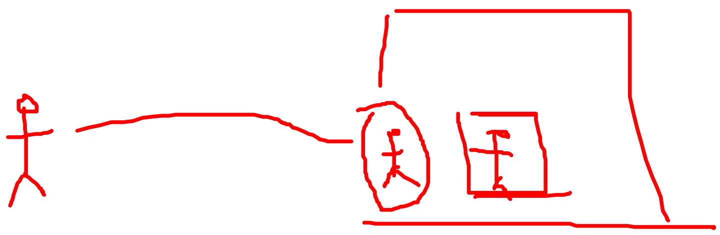
it is a predicate (condition) used to match amoung JP

additional code , called advice is executed in all part of the progam where it match pointcut

spring use AspectJ point cut expression language by default

Around advice: as can act as filter
spring security, method level sec, tx

@Before
@After
@Around
@AfterReturn
@AftherThrows



Spring MVC

u will revision spring jdbc
spring hibernate (at least)

Annotation ? custom annotation?
Java reflection and annotation processing?
Log4j : logging vs sysout
review spring jdbc
review spring hib

What is annotation:aka meta data ie data about data
Java 5

replaced xml for
meta data

inbuild annotation

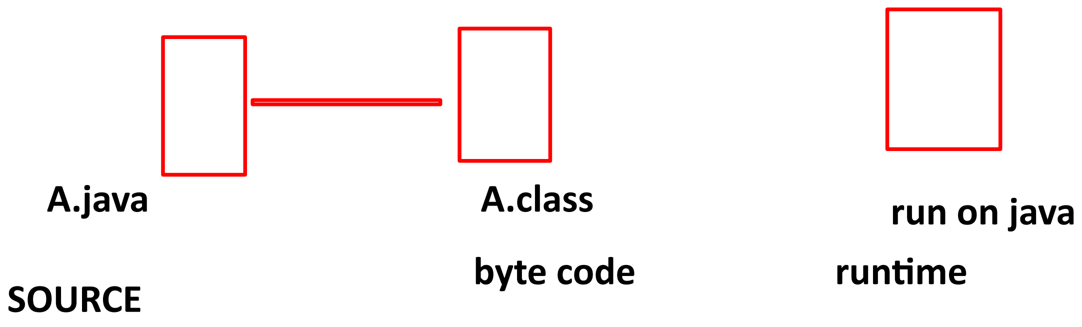
custom annotation

@Entity @Autowire....

where u can apply the anntation

```
@Target(ElementType.METHOD)
@Retention(RetentionPolicy.SOURCE)
public @interface Override {
}
```

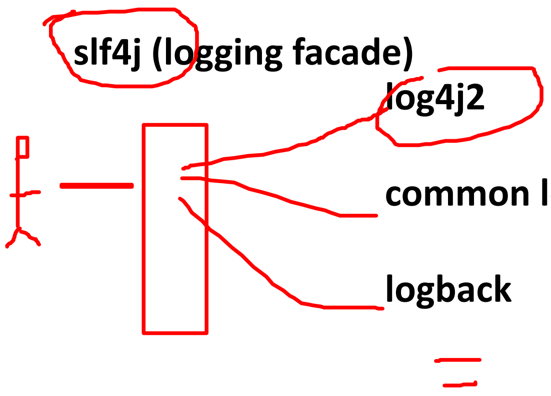
upto where annotation
reach?

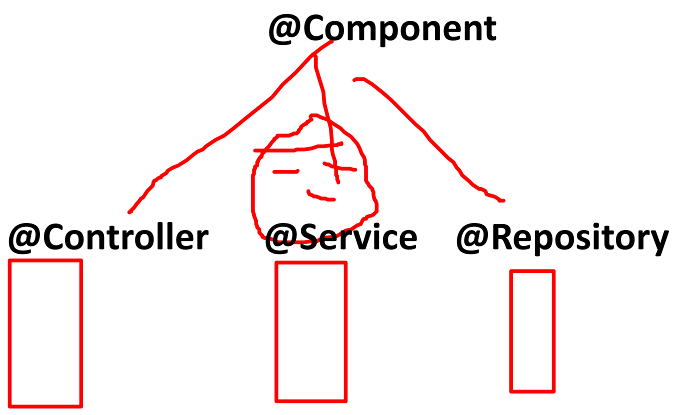


If u write ur own annotation u have to write code to process it
and it is done by an core java api ie callled "java reflection"

java reflection : core java api that is used to know information about
a class. method , paramter etc at run time...

log4j:it is a logging framework



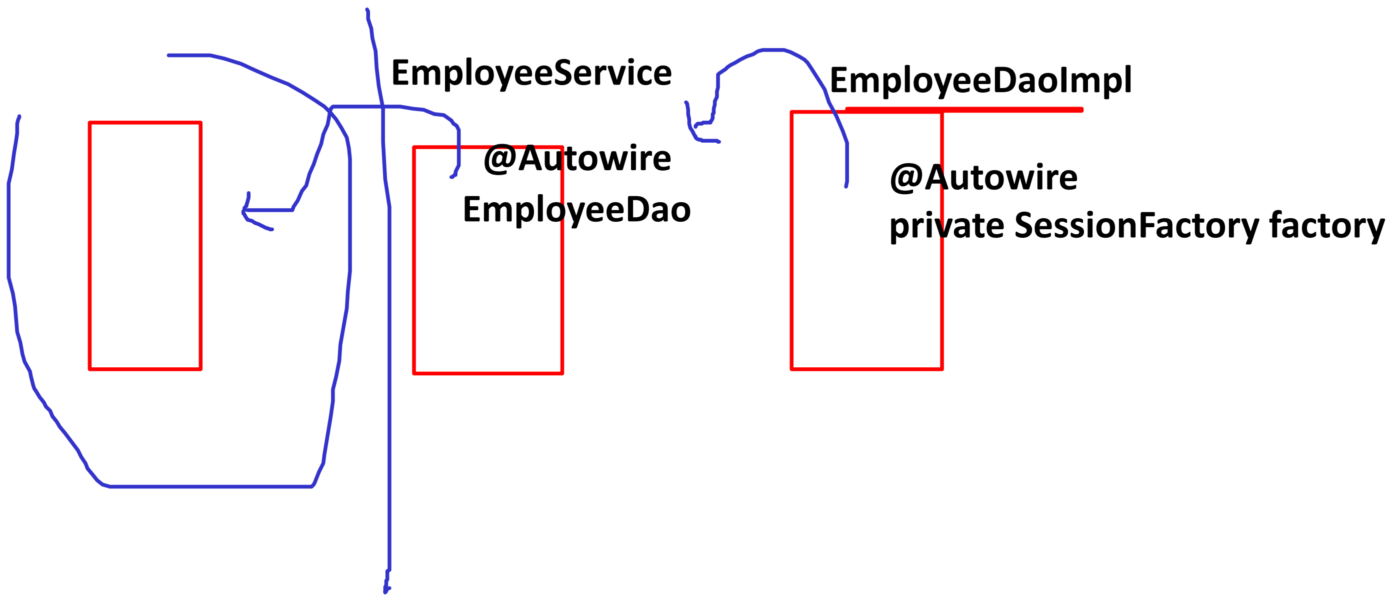


SL= BL + ccc

(logging sec, tx)

Spring + hibernate integration

Spring mvc + hibernate integation



```
<bean class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
        <property name="location" value="classpath:db.properties"/>
    </bean>
    <bean id="ds" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="username" value="${jdbc.username}"/>
        <property name="password" value="${jdbc.password}"/>
        <property name="url" value="${jdbc.url}"/>
        <property name="driverClassName" value="${jdbc.driverName}"/>
    </bean>
    <bean id="sf" class="org.springframework.orm.hibernate5.LocalSessionFactoryBean">
        <property name="dataSource" ref="ds"/>
        <property name="packagesToScan">
            <list>
                <value>com.customerapp.dao</value>
            </list>
        </property>

        <property name="hibernateProperties">
            <props>
                <prop key="hibernate.hbm2ddl.auto">update</prop>
                <prop key="hibernate.show_sql">true</prop>
                <prop key="hibernate.dialect">org.hibernate.dialect.MySQL57Dialect</prop>
                <prop key="hibernate.format_sql">update</prop>
            </props>
        </property>

    </bean>

    <bean id="transactionManager" class="org.springframework.orm.hibernate5.HibernateTransactionManager">
        <property name="sessionFactory" ref="sf"/>
    </bean>
    <tx:annotation-driven transaction-manager="transactionManager"/>

    <context:annotation-config/>
    <context:component-scan base-package="com.customerapp"/>
    <aop:aspectj-autoproxy/>
```

To do:

1:30-2:30 Lunch

2:30-4PM:
run the project ...15 min chnage un password driver name , jar if req
then
make similer app:
bank application:

dao dto service ...

```
class Account{
    int id;
    String name;
    double balance;
    String phone;
    String address;
}
```

```
class Account{
        int id;
        String name;
        double balance;
        String phone;
        String address;
}
```

```
interface AccountDao{
        public List<Account> getAllAccounts();
       public Account  getAccountById();
        public void addAccount(Account account);
        public void deleteAccount(int id);
        public void updateAccount(int id, Account account);
}
```

```
class AccountDaoImpl implements AccountDao{

        @Autowrire
        private SessionFactory factory;

        ///


}
```

```
inteface AccountService(


        public List<Account> getAllAccounts();
       public Account  getAccountById();
        public void addAccount(Account account);
        public void deleteAccount(int id);
        public void updateAccount(int id, Account account);
       public void deposit(int id, double amount);
       public void withdraw(int id, double amount);
       pubic void transfer(int fromAccId, int toAccId, double balance);
     public void updateAddress(int id,String address);
    public void updatePhone(int id,String phoneNumber);
}

class AccountServiceImpl imp...... {



}
```

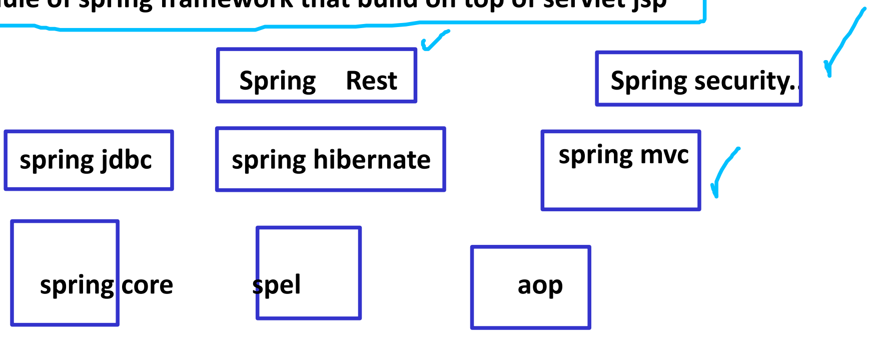
main
----

test

Spring MVC + hibernate integration

Spring MVC + hibernate integration

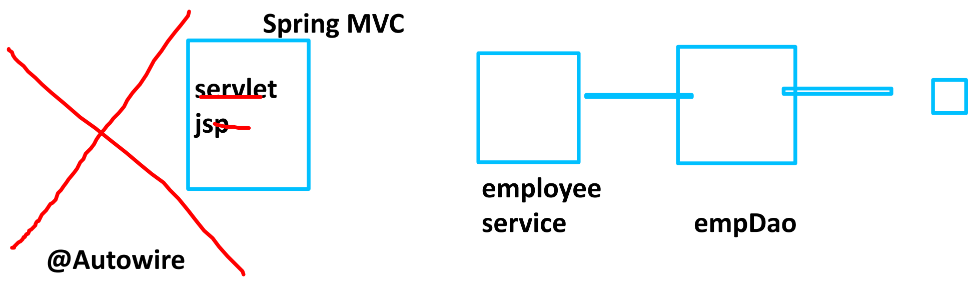
Spring MVC: is a module of spring framework that build on top of servlet jsp



Why i should use spring mvc

servlet jsp: junit ~~mockito~~

back end application :spring DI, AOP, integration with other framework



Spring MVC provide support for DI and AOP out of box
it is a pojo based model and it easy to test(unit and integration)

Spring MVC: server side validation :)
data buffering *
data conversion auto *

how to write hello world spring mvc application?

1. maven project with proper dependencies

2. Configure DS

```
<servlet>
        <servlet-name>springDispatcherServlet</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/web-config.xml</param-value>
        </init-param>

        <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>springDispatcherServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
```

```
    <context:component-scan base-package="com.demo"/>

    <mvc:annotation-driven/>      <!-- i18n, rest, validation data processing etx  -->

    <!-- view resolver? to find the location of jsp where result must be rendered -->
    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/views/"/>
        <property name="suffix" value=".jsp"/>
    </bean>
```

http://localhost:8080/mvc1/hellourl

/WEB-INF/views/hello.jsp

Back controller

```
@Controller
public class HelloController {
    @GetMapping(value = "hellourl")
    public ModelAndView hello() {
        ModelAndView mv=new ModelAndView();
        mv.setViewName("hello");
        mv.addObject("data", "spring mvc hello world");
        return mv;
    }
}
```

```
</head>
<body>
${data}
</body>
```

hashmap

/hellourl

handler mapping

Request