<https://ch1hub.cognizant.com/sites/SC48/ALA3/School%20of%20Learning/Forms/AllItems.aspx?RootFolder=%2Fsites%2FSC48%2FALA3%2FSchool%20of%20Learning%2FTechnical%2FJava%20Technologies%2FWeb%20Frameworks%2FSpring%203%20MVC>

A a new A();

A b new A();

A c new A();

Everytime we see ‘new’ keyword, new object is created and memory is allocated, but not used.

Spring supports Singleton DispatcherServlet. Singleton – one instance per class

@Autowired

* A a; – JDK 1.6 and above supports annotations It means initiate class A

2 xmls:

* Web.xml, xxxxx-servlet.xml where xxxx = servlet name

Ways to create objects:

* new keyword --- (dis adv. : new object is created with all properties empty/null )
* Reflection – class.forName() – No need to know the type of object. Used to load JDBC drivers

In Spring, we have autowired – creates single instance of the class and does not use new keyword. It means we are wiring or injecting one object ( as is) in the next object. It is the beauty of spring is: it thinks everything as "objects". So, here DB connection= object, hibernate connection = object, From bean = object, Transaction api = object.

Spring is solely based on xml. spring registers all objects once and for all in the xml. It does not create an object. We will have application-context.xml, every class is an object and is given an Id in xml. <bean id="abc" class= com.cts.A>. spring 2.5 onwards does not use xml, uses the concept of annotatuion as well

spirng is a servlet running inside a servlet container.

DS is the parent/central servlet/controlling servlet which controls the lifecycle of spring running inside a servlet container.(IOC container). One instance of DS at a time to will handle to each request.

For one request - one dispatcher servlet instance. Container will recognized the java classes using web.xml.

Component --@Controller

ModelAndView – Spring class which used for the output

@RequestMapping – can be used at method and class level. it is auto-handled by the annotation

We cannot 2 controllers for the same request

It will accept the request from the browser and is redirected to proper handler. Each request is unique.

Lifecycle of the spring bean:init-service-destroy

If the object is prefilled with valuesm, then the same object will be injected to class B.

**View resolver:**

**It is an interface**

it is only if the fisrt page contains the event

1st page is a welcome page...web context

web context does not have a view resolver

spriong context uses the view resolver...hence u dont mentiobn any path anywhere

u just regioster the resolver in the xml

InternalResourceViewResolver is the parent view resolver. It can resolve jdp,jsf,xml etc

Multiple resolvers are possible.

Keep InternalResourceViewResolver at last.

ModelAndView is a Spring class. Captures the process o/p(object) to be displayed and displays to the view layer defined. ModelAndView---is a bucket of Model + presentation jsp name

At req scope the value provided for fields are @RequestParam – to denote req params that are passed to a method

@ModelAttribute – It binds the form bean

enity bean (any ordinary bean match every object property with the DB ). It is used to match the object with the presentation

Form bean - match every object property with the JSP or view

Model – it

Transactuion = group of sql statements executed as a bulk

each transaction has a unique transaction id

set of operation performed at a time, if one get failed all get revert

rolled back to the savepoint

trransaction1 -- create table employye,update employee set name=.. where id=. And commit

transaction 2= create table address, update address set pin =..where empid=.. And commit

hibernate iis an ORM

transaction manager ---will manage a set of transdactions...transaction can be managed by Spring in coordination with hibernate orm in a particular session.  
the transaction Manager will only be functional if there is a session with a DB connection.

In our xml,

Requests are based url

DB is session based DB connection

AnnonationSessionFactoryBean – Spring bean which works with hibernate

Hibernate needs dialect(language used to communicate to Hsql/DB)

dtatsource---DB connection

session factory--- execute queries with hibernate

Session facotry...hibernate

annotationsessionfacotry---hibernate session facotry  
transaction manager--- to manage the transaction or data gets locked.

<context:property-placeholder location="classpath:jdbc.properties" />

everything under "src" is compiled, "classes" fiolder get genrated and the properties file is put here

When we stop the server, session gets killed

interresource vuiew resolver is a spearate class and it is used to resolve views.

Factory desih=gn pattern - maintain pool of objects. Lly, seesionfactory maintains in one session and has three properties.

MVC flow :Controller----Service interface---service impls---DAO interface---DAO impls....presentation layer ( jsp)

@Repository - Spring annotation---- @Repository---DAO

Whenever Sp contr see @Re, it gets to knw that the class is DAO.

@Component – Spring ann --- @Component –POJOs/helper class/yutility classes. Spr contr will scan and gets to knw.

@Service = this is the class that will call the DAO ...and here there will be transaction manager will take effect

@Transaction – Spr annotation. Class level – to declare the class which will have a transaction and TM will be initiated. It follws ACId – so each tran will be unique, isolated

readONly = true – cannot modify sql queries in a transaction

readONly = false – for create/update

Propogation

Create table employee; - savepoint 1

Update employee set; savepoint 2

Select \* from employee;

Commit;

Entire thing is one transaction. If poo-true, then transaction propagates to the subsequent queries. If update fails, then rollback will happen for update. By default, prop is true.

route===handler - Handler

incerpte---detect – Interceptor, it will find the request before and after it is handled