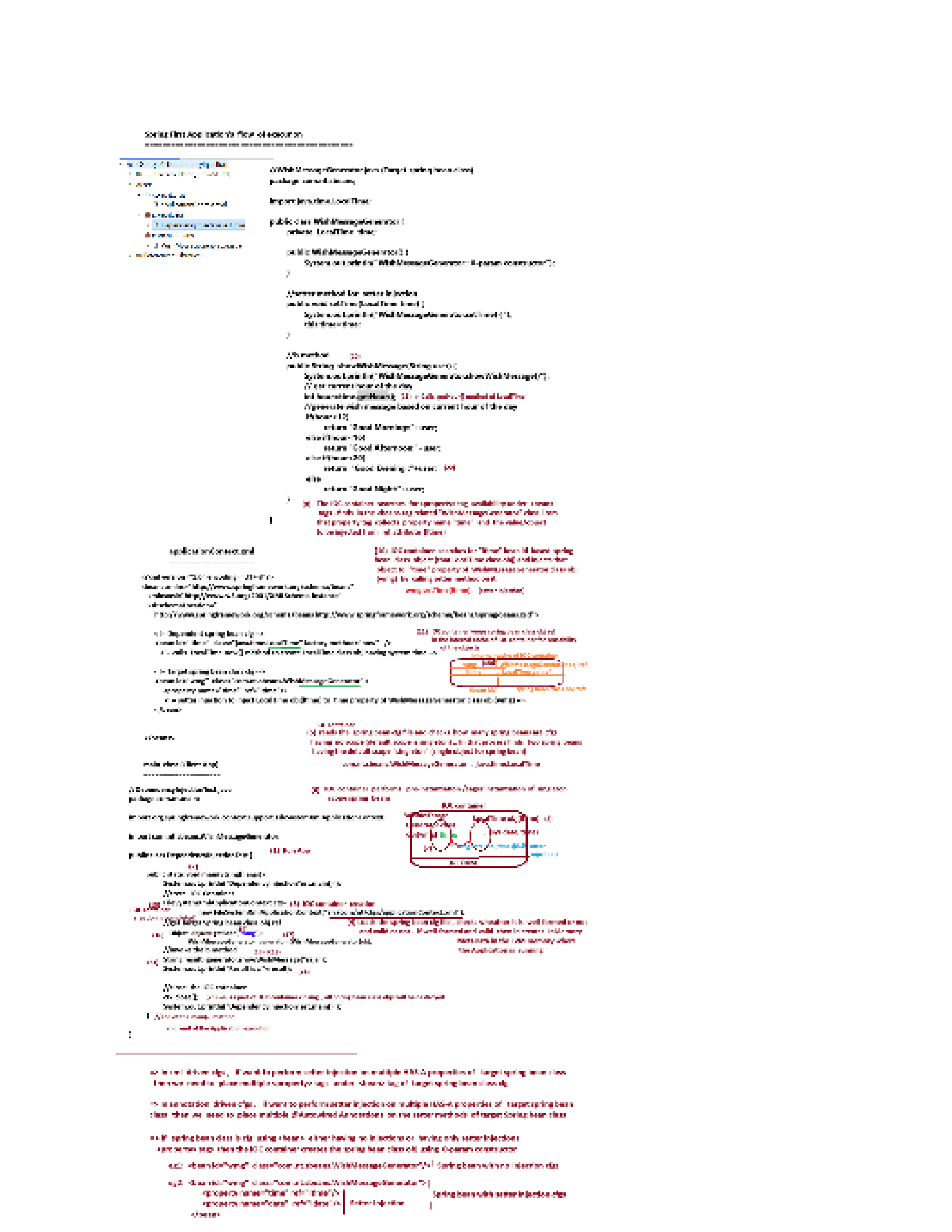

Spring First Application's flow of execution

IOCProj01-DependencyInjection

>

JRE System Library [JavaSE-21]

src

com.nt.cfgs

applicationContext.xml

com.nt.main

> Dependency Injection Test.java

✓ com.nt.sbeans

WishMessageGenerator.java

>

Referenced Libraries

//WishMessageGenerator.java (Target spring bean class) package com.nt.sbeans;

import java.time.LocalTime;

public class Wish MessageGenerator {

private LocalTime time;

public WishMessageGenerator() {

}

System.out.println("WishMessageGenerator:: 0-param constructor");

//setter method for setter Injection

public void setTime(LocalTime time) {

System.out.println("Wish MessageGenerator.setTime(-)"); this.time-time;

}

applicationContext.xml

<?xml version="1.0" encoding="UTF-8"?>

//b.method

(20)

public String showWishMessage(String user) {

}

System.out.println("WishMessageGenerator.showWishMessage()");

// get current hour of the day

int hour=time.getHour(); (21)--> Calls getHour() method of LocalTime //generate wish message based on current hour of the day if(hour<12)

return "Good Morning::" + user;

else if(hour<16)

return "Good Afternoon:" + user;

else if(hour<20)

else

return "Good Evening :"+user; (22)

return "Good Night:" + user;

(9) The IOC container searches for <property> tag availability under <bean> tags..finds in the <bean> tag related "WishMessageGenerator" class.From that property tag collects property name "time" and the value/object to be injected from ref attribute (ltime)

<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="

(10) IOC container searches for "ltime" bean id based spring bean class object (that LocalTime class obj) and injects that object to "time" property of WishMessageGenerator class obj (wmg) by calling setter method on it

wmg.setTime(ltime); (setter Injection)

http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

<!-- Dependent spring bean cfg -->

<bean id="ltime" class="java.time.LocalTime" factory-method="now" />

(11) IOC container keeps spring bean class obj ref

in the internal cache of IOC container for reusability

<!-- calls LocalTime.now() method to create LocalTime class obj having system time -->

<!-- Target spring bean class cfg -->

<bean id="wmg" class="com.nt.sbeans.WishMessageGenerator"> <property name="time" ref="ltime"/>

of the objects

Internal cache of IOC container

wmg (15)? WishMessageGenerator obj ref LocalTime obj ref

ltime

(bean ids)

spring bean class obj refs

<!-- setter injection to inject LocalTime obj(ltime) to time property of Wish MessageGenerator class obj(wmg) --> </bean>

IOC container

</beans>

main class (Client App)

=========

=====

//DependencyInjection Test.java

package com.nt.main;

(5) reads the spring bean cfg file and checks how many spring beans are cfgs having no scope (default scope is singleton).. In that process finds two spring beans having the default scope "singleton" (single object for spring bean)

com.nt.sbeans.Wish MessageGenerator , java.time.LocalTime

(6) IOC container performs pre-instantiation /eager instantiation of singleton scope spring beans

import org.springframework.context.support.FileSystemXmlApplicationContext;

import com.nt.sbeans.Wish MessageGenerator;

(1) Run App

public class DependencyInjection Test {

(2)

IOC container

WishMessage Generator class obj(wmg) time:

(7)

LocalTime obj(It me) (8) ....sys date, time) wing.setTime(Itime) (#10 setter

injection)

JRE/JVM

}

public static void main(String[] args) {

System.out.println("DependencyInjection Test.main()");

//crete IOC Container

(13) FileSystemXmlApplicationContext ctx= (3) IOC container creation

(IOC container

new FileSystemXmlApplicationContext("src/com/nt/cfgs/applicationContext.xml"); get Target Spring bean class obj ref

creation is completed)

(16) Object obj=ctx.getBean("mg"); (17)

(4) Loads the spring bean cfg file, checks wheather it is well-formed or not and valid or not, if well-formed and valid then in creates InMemory MetaData in the JVM memory where

the Application is running

WishMessageGenerator generator=(Wish MessageGenerator)obj; //invoke the b.method

(18) &(19)

(23) String result-generator.showWishMessage("raja"); System.out.println("Result is ::"+result);

(24)

//close the IOC cotnainer

ctx.close(); (25) --> As part of IOC container closing, all Spring bean class objs will be destroyed System.out.println("DependencyInjection Test.main()");

} //end of the main(-) method

and end of the Application execution

=> In xml driven cfgs, if want to perform setter injection on multiple HAS-A properties of target spring bean

class then we need to place multiple <property> tags under <bean> tag of target spring bean class cfg

=> In annotation driven cfgs, if want to perform setter injection on multiple HAS-A properties of target spring bean class then we need to place multiple @Autowired Annotations on the setter methods of target Spring bean class

=> if spring bean class is cfg using <bean> either having no injections or having only setter injections <property> tags then the IOC container creates the spring bean class obj using O-param constructor

eg1: <bean id="wmg" class="com.nt.sbeans.WishMessageGenerator"/>| Spring bean with no Injection cfgs

eg2: <bean id="wmg" class="com.nt.sbeans.Wish MessageGenerator">

<property name="time" ref="ltime"/>

Spring bean with setter Injection cfgs

<property name="date" ref="ldate"/> Setter Injection </bean>

=> if spring bean class cfg having factory-method attribute in <bean> tag then IOC container peforms static factory method call based spring bean class obj creation

eg:: <bean id="ltime" class="java.time.LocalTime" factory-method="now"/>

=> calls LocalTime.now() method to create and spring bean class obj

=> if spring bean class is cfg by placing multiple <constructor-arg> tags (1 or more) under <bean> tag then the IOC container uses parameterized constructor to create spring bean class obj and also to inject Dependents to it.

(constructor Injection)

<bean id="wmg" class="com.nt.sbeans.WishMessageGenrator"> <constructor-arg name="time" ref="ltime"/>

<constructor-arg name="date" ref="ldate"/>

</bean>

if we place <constructor-arg> tags for

"n" times then the IOC uses

"n" param constructor to perform constructor Injection