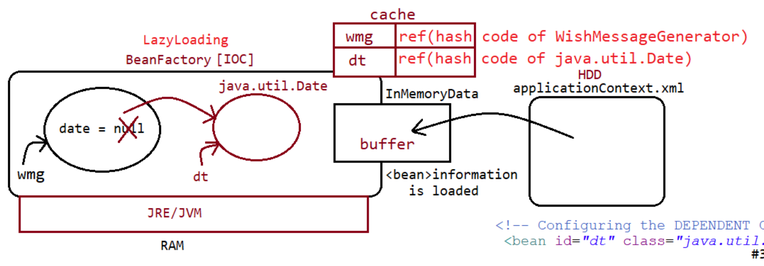



```
<!-- Configuring the DEPENDENT Component -->
 <bean id="dt" class="java.util.Date"/>
                                  #3
 <!-- Configuring the Target Component -->      #1
<bean id="wmg" class="in.ineuron.comp.WishMessageGenerator">
     <property name="date" ref="dt"/>
                           #2
</bean>
```

Behind the scenes
==========
1. Object obj = factory.getBean("wmg");
      Container will check the internal cache for the bean with the id "wmg".

   Class c =Class.forName("in.ineuron.comp.WishMessageGenerator");
   WishMessageGenerator wmg = (WishMessageGenerator) c.newInstance();

 **Container will keep the ref(wmg) in the cache and the object reference as the value in the cache.

   Container will check in the internal cache for the bean with the id "dt".

      Class c1 =Class.forName("java.util.Date");
      Date dt =(Date)c1.newInstance();

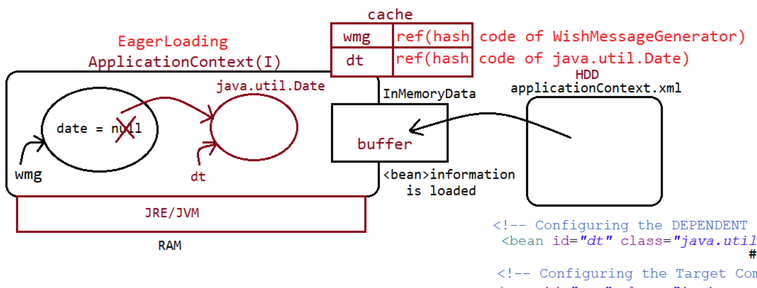
** Container will keep the ref(dt) in the cache and the object reference as the value in the cache.

   Because of <property name="date" ref="dt"/>
      container will setDate(dt) on the wmg object as shown below
             wmg.setDate(dt);

Note: Default scope of all the beans in IOC-container is "Singleton".

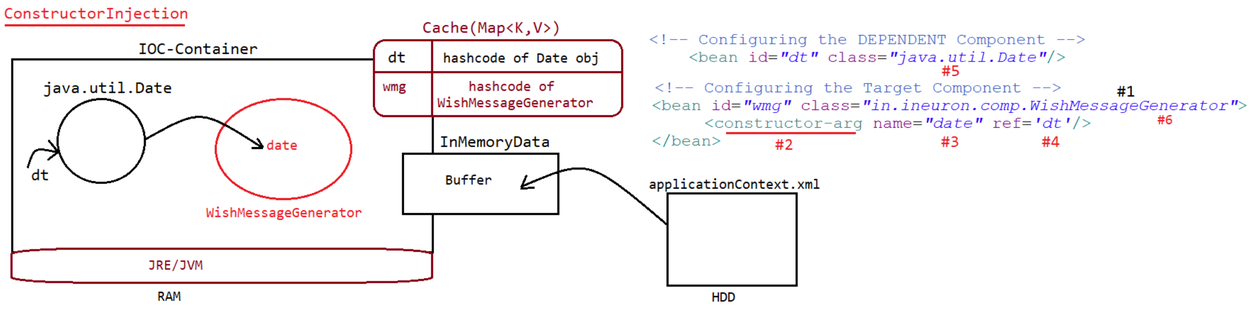
2. Since the programm in running with the help of main(), when the main() finishes the execution
      => Automatically container will be closed,due to which the internal cache, objects and
         InMemory data will be flushed from JVM area.
      => Finally JVM also will be cleaned from JRE region.

ApplicationContext(I)




```
<!-- Configuring the DEPENDENT Component -->
 <bean id="dt" class="java.util.Date"/>
                                  #3
 <!-- Configuring the Target Component -->      #1
<bean id="wmg" class="in.ineuron.comp.WishMessageGenerator">
     <property name="date" ref="dt"/>
                           #2
</bean>
```

ConstructorInjection




```
<!-- Configuring the DEPENDENT Component -->
    <bean id="dt" class="java.util.Date"/>
                                  #5
 <!-- Configuring the Target Component -->      #1
<bean id="wmg" class="in.ineuron.comp.WishMessageGenerator">
     <constructor-arg name="date" ref='dt'/>          #6
</bean>          #2          #3        #4
```

Behind the scenes
----
1. Object obj = factory.getBean("wmg");
      Container will check the internal cache for the bean with the id "wmg".
      since it is not available,container will also see how the bean has to be created.
      if it is through <constructor-arg> then container will check what is the bean id of the
      referenced object.
          <constructor-arg name='date' ref='dt'/>

2. Since the injection is of <constructor-arg> container will create an object of depedent class for
   the bean id "dt".
       <bean id ="dt" class="java.util.Date"/>

```
Date dt =(Date)Class.forName("java.util.Date").newInstance();
```

   The "dt" reference will be kept in the internal cache of IOC-cotnainer for future usage.

3. Now the container will create an object of Target class by using Dependant object through constructor injection.

```
  <bean id="wmg" class="in.ineuron.comp.WishMessageGenerator">
        <constructor-arg name="date" ref="dt"/>
  </bean>
```

ReflectionAPI

```
Class c = Class.forName("in.ineuron.comp.WishMessageGenerator");
    Constructor const[] = c.getDeclaredConstructors();

WishMessageGenerator wmg = (WishMessageGenerator) const[0].newInstance(dt);
```

Spring jars can be downloaded from the following link
  => https://repo.spring.io/ui/native/release/org/springframework/spring/

Scroll down and search for link with Spring5.X version(5.3.17)

To add the namespace(xmlns),while writing the configuration file use the folowing location
      D:\jars\spring-5.3.17-dist\spring-framework-5.3.17\docs\reference\html\core.pdf

Setter Injection

=> Here IOC container calls setter method of Target class to assign Dependant class Object to target class Object.
=> java.util.Date(C)
                 |=> year(int),month(int),date(int)
                 |=> hours(int),minutes(int),seconds(int)
=> Setter injection will happen
            a. Creating an object of Target component using zero param constructor.
            b. Creating an object of Dependent component using Zero argument constructor/parameterized constructor.
            c. Calling setter method on Target component to perform dependency injection.

=> BeanFactory container performs LazyLoading for all the beans present in spring cfg file.

applicationContext.xml

```
<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
https://www.springframework.org/schema/beans/spring-beans.xsd">

  <!-- Dependent spring bean cfg -->
  <bean id="dt"  class="java.util.Date"/>

  <!-- Target spring bean cfg -->
  <bean id="wmg"  class="in.ineuron.comp.WishMessageGenerator">
      <property name="date"  ref="dt"/> ===> Instruction to IOCcontainer to Perform Setter Injection.(wmg.setDate(dt))
  </bean>

</beans>
```

Note:: In the <property> tag,"name" value is not a bean property name, it is the word of setXXXX(,) method.
        If the injection is of reference type, then we need to use <property name="" ref=''/>.
        If the inejction is of value type,then we need to use <property name='' value=''/>

SetterInjection.java

```
FileSystemResource resource = new FileSystemResource("src/in/ineuron/cfg/applicationContext.xml");
System.out.println("***********BeanFactory container starting***********");
XmlBeanFactory factory = new XmlBeanFactory(resource);
```

```
System.out.println("***********BeanFactory container started*************");

WishMessageGenerator wmg = (WishMessageGenerator) factory.getBean("wmg",
WishMessageGenerator.class);
System.out.println(wmg);
String result = wmg.generateMessage("sachin");
System.out.println(result);

System.out.println();

System.out.println("*******Container is closing**********");


output
------
***********BeanFactory container starting************
***********BeanFactory container started*************

WishMessageGenerator.class file is loading...
WishMessageGenerator object is instantiated...
Setter method is called to perform Setter injection...
WishMessageGenerator [date=Tue Mar 21 20:35:30 IST 2023]
Good Night :: sachin

*******Container is closing**********

What is the difference b/w FileSystemResource and ClassPathResource?
=> Both are implementation classes of Resource(I)
=> FileSystemResource(C)
       It makes the BeanFactory container to locate the given Springbean cfg file
from the Specified of path File System.
       We can give either relative path(recomended) or absolute path of the spring
bean configuration file.
            eg:: FileSystemResource res=new
FileSystemResource("src/in/ineuron/cfgs/applicationContext.xml");
=> ClassPathResource(C)
       It makes the BeanFactory container to locate the given Springbean cfg file
from the directories or jar files that are added to
      classpath or buildpath.
            eg: ClassPathResource res=new
ClassPathResource("in/ineuron/cfgs/applicationContext.xml");


Constructor Injection
---------------------
Here IOC container uses parameterized constructor to create target class object.
In this process it assigns/injectes dependant object to the Target class Object.
In Setter injection first target class object is created,next Dependant object will
be created.
In case of Constructor injection,First Dependant class object should be
created,next target class object will be created and this dependant object  will be
passed as the argument to the  constructor.
      syntax:
                  <bean id ='' class=''>
                        <constructor-arg name='' ref=''/>
                  </bean>
if we place<constructor-arg> tag for 'n' time under bean tag, then IOC container
uses "n-param" constructor to create the Spring bean class Object.
```

Note: <ref> attribute to cfg bean id is based on Spring bean class object injection to target bean class properites.
<value> to cfg is to inject "simple values" to Spring bean class properties.
In the <constructor-arg> tag, "name" value is not the bean property name, it is the parameter name of the parameterized constructor.

Note:
What happens if we enable both setter injection and construcor injection to bean property?
Tell me which id will be injected as final values?
Ans> Since setter method is called after the constructor execution, we say setter injection overrides the value injected by the
constructor injection. Values/Object injected by the Setter injection will become final values.

Note: In case of Setter injection, IOC container uses Zero parameter constructor to create the Object.
In case of Constructor injection,IOC container use Parameterized constructor to create the Object.

Note:
Bean id should be unique w.r.t IOC container
We can configure 2 spring beans having same class names but we should take different bean id.