Kindly revise SpringCore
	a. Configuration[XML,JavaConfiguration]
	b. SpringBean[Rules]
	c. IOC-Container[ApplicationContainer(I) :: AdvancedContainer]

IOC-Container
	a. Creates the object
	b. Provides value to the object[Primitive,CollectionType,Reference Type]
	c. Link two objects if necessary
	d. Destroy the object

Linking 2 objects code[HAS-A relationship]
++
```
package in.pwskills.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import in.pwskills.bean.Model;
import in.pwskills.bean.Product;

@Configuration
public class AppConfig {

	static {
		System.out.println("AppConfig.class file is loading...");
	}

	@Bean
	public Product prodObj() {

		Product product = new Product();
		product.setPcost(10000.0);
		product.setPid(10);
		product.setPname("FOSSIL");

		//Linking two objects
		product.setModel(modelObj());
		return product;

	}

	@Bean
	public Model modelObj() {
		Model model = new Model();
		model.setMid(0001);
		model.setMtype("Analog");
		return model;
	}

}
```

Test.java
+++++++++
```
package in.pwskills.main;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.support.AbstractApplicationContext;
```

```
import in.pwskills.bean.Product;
import in.pwskills.config.AppConfig;

public class Test {

	public static void main(String[] args) {

		// Starting the container and informing the configuration file to scan
for spring-bean
		ApplicationContext applicationContext = new
AnnotationConfigApplicationContext(AppConfig.class);

		Product product = applicationContext.getBean(Product.class);
		System.out.println(product);

		// closing the container
		((AbstractApplicationContext) applicationContext).close();

	}
}
Output
AppConfig.class file is loading...
PRODUCT.CLASS FILE IS LOADING...
PRODUCT OBJECT CREATED BY FRAMEWORK...
Product.setPcost()
Product.setPid()
Product.setPname()

MODEL.CLASS FILE IS LOADING...
Model OBJECT CREATED BY FRAMEWORK...
Model.setMid()
Model.setMtype()
Product.setModel()
Product [pid=10, pname=FOSSIL, pcost=10000.0, model=Model [mid=1, mtype=Analog]]


++++++++++++++++++++++++++++++++
Loading data from Properties file
++++++++++++++++++++++++++++++++
1. To load the data from properties file, we need to use @PropertySource
annotation.
2. Internally SpringFramework uses Environment(I), implementation class to load the
data from Properties file

AppConfig.java
++++++++++++++
package in.pwskills.config;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.PropertySource;
import org.springframework.core.env.Environment;

import in.pwskills.bean.Student;

@Configuration
@PropertySource(value = "application.properties")//loading the properties file
```

```
public class AppConfig {

	@Autowired
	private Environment environment;//using the inbuilt object to retrieve from
properties file

	static {
		System.out.println("AppConfig.class file is loading...");
	}

	@Bean
	public Student student() {
		System.out.println("Environment impl class is :: " +
environment.getClass().getName());
		Student student = new Student();
		student.setSid(environment.getProperty("sid",Integer.class));
		student.setSage(environment.getProperty("sage", Integer.class));
		student.setSaddress(environment.getProperty("saddress"));
		student.setSname(environment.getProperty("sname"));

		return student;
	}

}

application.properties
++++++++++++++++++++++
sid = 10
sname = sachin
saddress = MI
sage = 49

Test.java
+++++++++
package in.pwskills.main;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.support.AbstractApplicationContext;

import in.pwskills.bean.Student;
import in.pwskills.config.AppConfig;

public class Test {

	public static void main(String[] args) {

		// Starting the container and informing the configuration file to scan
for SpringBean
		ApplicationContext applicationContext = new
AnnotationConfigApplicationContext(AppConfig.class);

		// use the object created by the IOC-Container
		Student std = applicationContext.getBean("student", Student.class);
		System.out.println(std);

		//closing the container
		((AbstractApplicationContext) applicationContext).close();
```

```
	}
}
```

Java Configuration
++++++++++++++++++
  => To Configure any bean in java configuration we need to write the code as shown below

```
	@Configuration
	public class ______{

		//No of objects = No of methods
		@Bean
		public <RT> methodName(){
			return RT;
		}
	  }
```

Annotation Configuration
	a. @Component(used for creating objects)
	b. @Controller(used for handling request)
	c. @Service(user for getting transaction support)
	d. @Repository(used for database operations)
	e. @RestController(used for handling request from 3rd party applications like react,angular,...)

@Component => Used to create an Object
@Value   => Used to inject the value to the Object.

Employee.java
+++++++++++++
```
package in.pwskills.bean;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class Employee {

	@Value("7")
	private Integer empId;

	@Value("dhoni")
	private String empName;

	@Value("CSK")
	private String empAdress;

	@Value("3500.0")
	private Double empSalary;

	static {
		System.out.println("EMPLOYEE.CLASS FILE IS LOADING...");
	}

	public Employee() {
		System.out.println("EMPLOYEE OBJECT CREATED BY FRAMEWORK...");
```

```
	}

	@Override
	public String toString() {
		return "Employee [empId=" + empId + ", empName=" + empName + ",
empAdress=" + empAdress + ", empSalary="
				+ empSalary + "]";
	}

	public Integer getEmpId() {
		return empId;
	}

	public void setEmpId(Integer empId) {
		this.empId = empId;
	}

	public String getEmpName() {
		return empName;
	}

	public void setEmpName(String empName) {
		this.empName = empName;
	}

	public String getEmpAdress() {
		return empAdress;
	}

	public void setEmpAdress(String empAdress) {
		this.empAdress = empAdress;
	}

	public Double getEmpSalary() {
		return empSalary;
	}

	public void setEmpSalary(Double empSalary) {
		this.empSalary = empSalary;
	}

}

AppConfig.java
++++++++++++++
package in.pwskills.config;

import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan(basePackages = "in.pwskills.bean")
public class AppConfig {

	static {
		System.out.println("AppConfig.class file is loading...");
	}
```

```
}

TestApp.java
++++++++++++
package in.pwskills.main;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.support.AbstractApplicationContext;

import in.pwskills.bean.Employee;
import in.pwskills.config.AppConfig;

public class Test {

      public static void main(String[] args) {

            // Starting the container and informing the configuration file to scan
for SpringBean
            ApplicationContext applicationContext = new
AnnotationConfigApplicationContext(AppConfig.class);

            Employee employee = applicationContext.getBean(Employee.class);
            System.out.println(employee);

            // closing the container
            ((AbstractApplicationContext) applicationContext).close();

      }
}

Output
AppConfig.class file is loading...
EMPLOYEE.CLASS FILE IS LOADING...
EMPLOYEE OBJECT CREATED BY FRAMEWORK...
Employee [empId=7, empName=dhoni, empAdress=CSK, empSalary=3500.0]


@Autowired
++++++++++
 1. Injection is compulsory
      a. One object found and the Same object injected using byType.
            Employee(C)<------Address(C)
      b. Object not found :: NoSuchBeanDefnitionException
            solution : Mark @Autowired(required = false)
      c. more than one :: NoUniqueBeanDefnitionException
            solution 1:: Give the refName same as that of any objName.
                              Courier bdart; [Out of 3 object :: bdart,dtdc,fedex]
bdart will be injected.
            solution 2:: Use @Qualifer Annoation and bind the required bean
                              @Qualifier("bdart")
                              Courier courier; [Out of 3 object ::
bdart,dtdc,fedex] bdart will be injected.

                        refer:: SpringAutowiredAnnotationApp-02

Employee.java
package in.pwskills.bean;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class Employee {

	@Value("7")
	private Integer empId;

	@Value("dhoni")
	private String empName;

	@Value("3500.0")
	private Double empSalary;

	@Autowired //Linking 2 objects using HAS-A relationship
	private Address address;

	static {
		System.out.println("EMPLOYEE.CLASS FILE IS LOADING...");
	}

	public Employee() {
		System.out.println("EMPLOYEE OBJECT CREATED BY FRAMEWORK...");
	}


	@Override
	public String toString() {
		return "Employee [empId=" + empId + ", empName=" + empName + ",
empSalary=" + empSalary + ", address=" + address
				+ "]";
	}

}
```

Address.java

```
++++++++++++
package in.pwskills.bean;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class Address {

	@Value("IND")
	private String country;

	@Value("MAHARASHTRA")
	private String state;

	@Value("560035")
	private Integer pinCode;

	@Override
	public String toString() {
		return "Address [country=" + country + ", state=" + state + ",
```

```
pinCode=" + pinCode + "]";
      }

}
```