JAVA SPRING DEPENDANCY INJECTION

Steps :

1. **Create a Interface**

Package - org.studyeasy.interfaces

Interface - Car

Code :

```
package org.studyeasy.interfaces;

public interface Car {
void specs();
}
```

2. Create a Class

Package - org.studyeasy.cars

Class - Swift

Swift Class implements Car interface

Code :

```
package org.studyeasy.cars;

import org.studyeasy.interfaces.Car;

public class Swift implements Car{

    @Override
    public void specs() {
    // TODO Auto-generated method stub
    }
}
```

3. Create a Class

Package - org.studyeasy.cars

Class - Corolla

Corolla Class implements Car interface

Code :

```
package org.studyeasy.interfaces;

public class Corolla implements Car {
    @Override
    public void specs() {
    }
}
```

4. Create a Class

Package - org.studyeasy

Class - App

Code :

```
package org.studyeasy;

public class App {
    public static void main(String[] args) {
    }
}
```

5. Create a Class

Package - org.studyeasy

Class - AppConfig -

Telling Spring that it is configuration class and which particular package to scan for components

Code :

```
package org.studyeasy;

// Telling Spring that it is configuration class and which particular package
// to scan for components

public class AppConfig {
}
}
```

Final Code

App.java

```
package org.studyeasy;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.studyeasy.interfaces.Car;

public class App_Modified_SolutionForAmbiguityOfClasses {

	public static void main(String[] args) {

		/*
		 * Car o1 = new Swift(); o1.specs();
		 *
		 * Car o2 = new Swift(); o2.specs();
		 */

		//---------------------------------------------------------------------

		//Outsourcing Object Creation process
		AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext(AppConfig.class);

		//Outsource the work to Spring.
		Car myCar = context.getBean("swift" , Car.class);
		myCar.specs();

		//Closing the context also
		context.close();
	}
}
```

Corolla.java

```
package org.studyeasy.cars;

import org.springframework.stereotype.Component;
import org.studyeasy.interfaces.Car;

@Component("corolla")
public class Corolla implements Car {

	@Override
	public void specs() {
		System.out.println("Sedan from Toyota");
	}

}
```

Swift.java

```
package org.studyeasy.cars;

import org.springframework.stereotype.Component;
import org.studyeasy.interfaces.Car;

@Component("swift")
public class Swift implements Car{

	@Override
	public void specs() {
		System.out.println("Hatchback from Suzuki");
	}
}
```

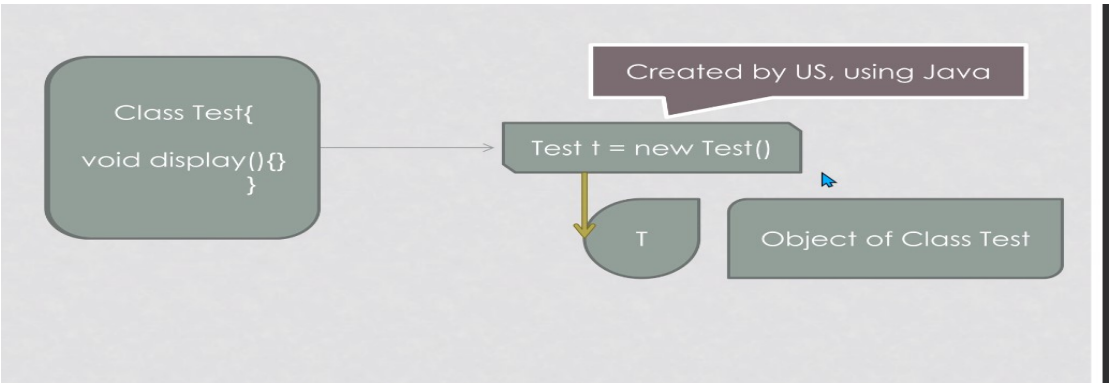
Car.java -> Interface

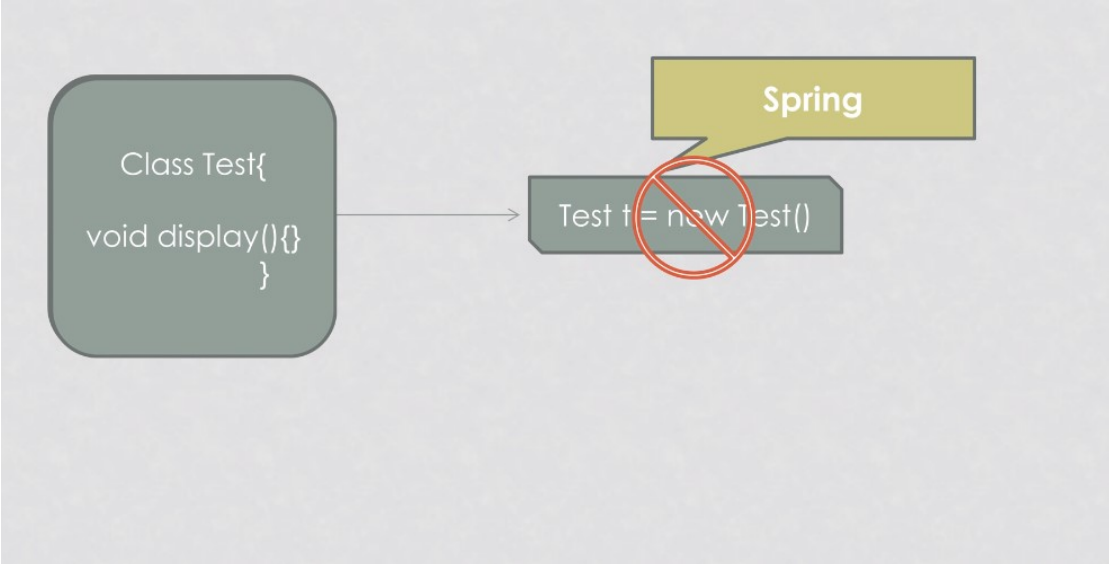
```
package org.studyeasy.interfaces;

public interface Car {
void specs();
}
```

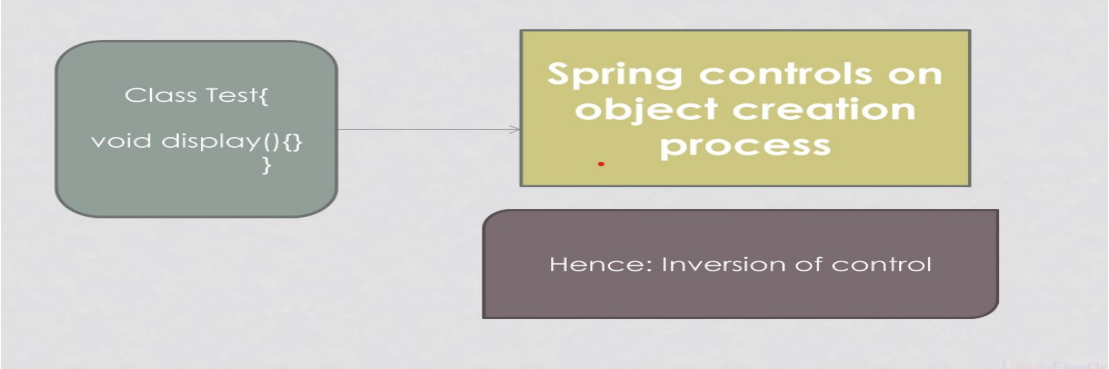
AppConfig.java

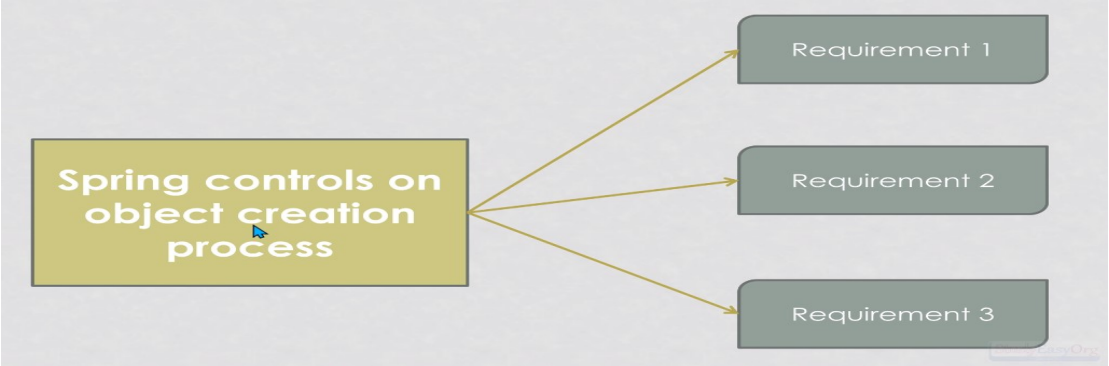
```
package org.studyeasy;

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

// Telling Spring that it is configuration class and which particular package to scan for components

@Configuration
@ComponentScan("org.studyeasy")
        public class AppConfig {
        }
}
```

INVERSION OF CONTROL










```
AnnotationConfigApplicationContext context =
    new AnnotationConfigApplicationContext(AppConfig.class);

Car myCar = context.getBean("myCorolla",Car.class);
System.out.println(myCar.specs());
context.close();
```

We are just creating a context.
And then we are assigning or we are making use of a bean for creation of a object and that is it.