

The Spring logo is a red arrow pointing to the right, with the word "Spring" written in white inside it.

Spring

Spring DI & IoC

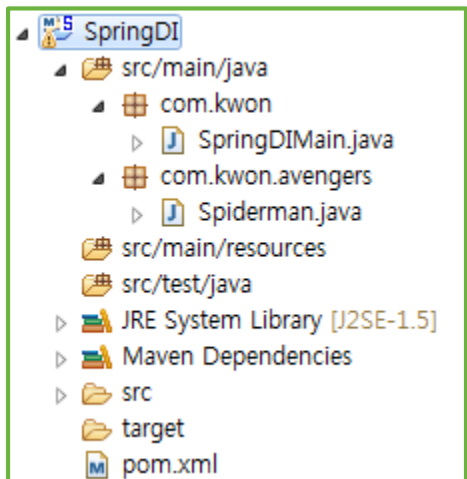
Several thin, curved lines in shades of blue and grey originate from the bottom left corner and sweep upwards and to the right, creating a decorative, organic feel.

권기웅

1.Spring DI 객체의 Scope

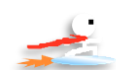
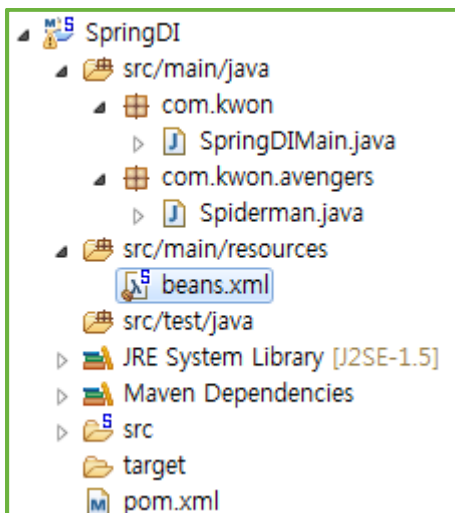
1.1 프로젝트 작성

1.1.1 클래스 작성

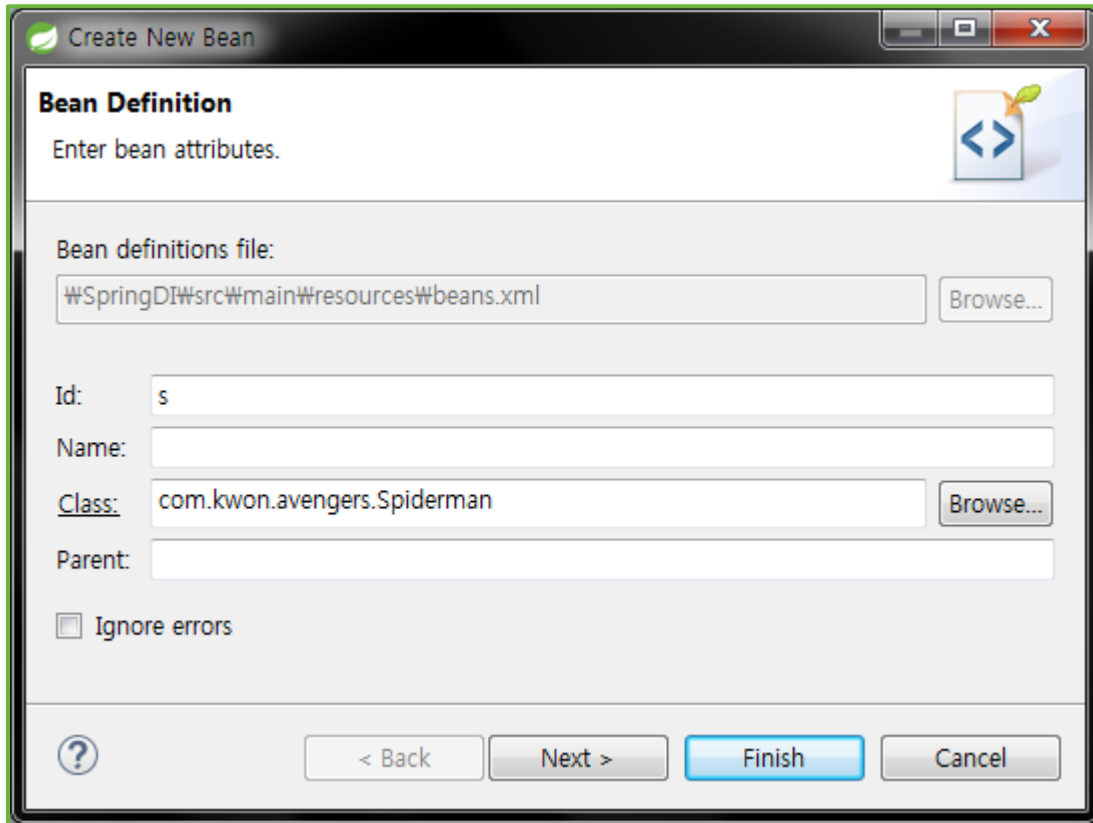


```
public class Spiderman {  
    public void webSwing(){  
        System.out.println("슌");  
    }  
}
```

1.1.2 src/main/resources 에 Spring Bean Configuration File 작성



1.1.3 bean 등록



The image shows a 'Create New Bean' dialog box with the following fields and options:

- Bean Definition**: Enter bean attributes.
- Bean definitions file**: `WSpringDI\src\main\resources\beans.xml` (with a 'Browse...' button).
- Id**: `s`
- Name**: (empty field)
- Class**: `com.kwon.avengers.Spiderman` (with a 'Browse...' button).
- Parent**: (empty field)
- ☐ **Ignore errors**
- Buttons: `< Back`, `Next >`, **Finish**, `Cancel`.

1.1.4 main 클래스

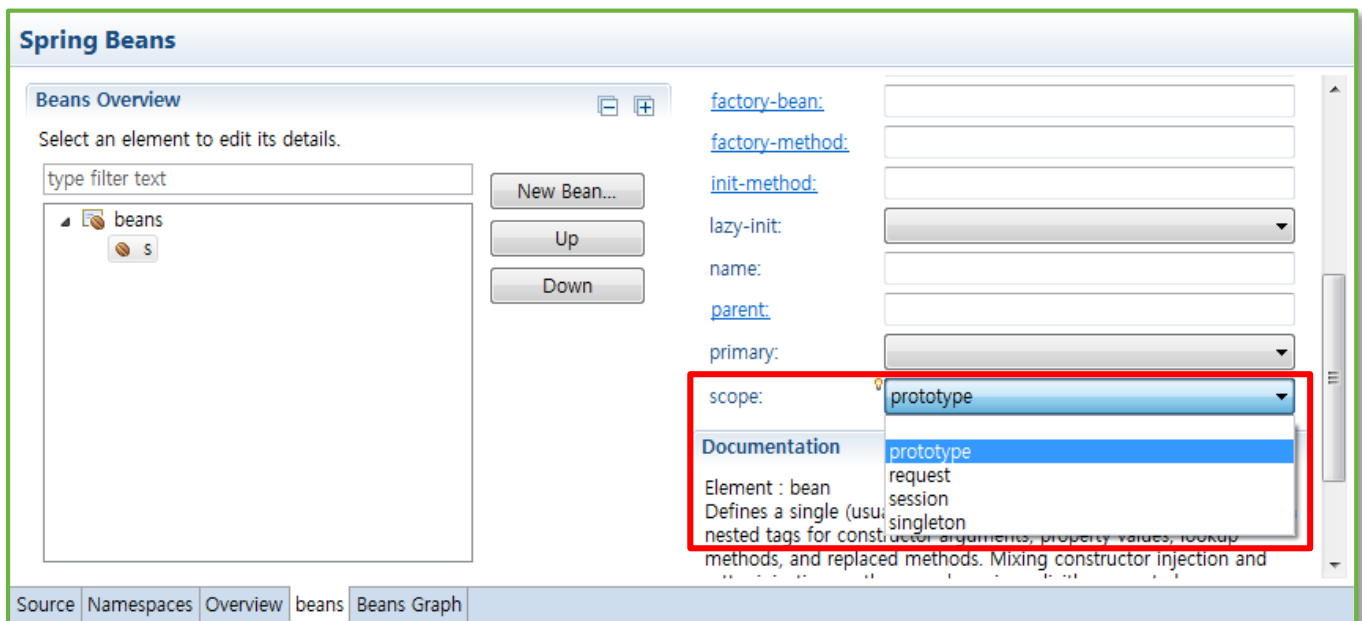
```
public class SpringDIMain {  
    public static void main(String[] args) {  
        DefaultListableBeanFactory dlbf = new DefaultListableBeanFactory();  
        XmlBeanDefinitionReader xbdr = new XmlBeanDefinitionReader(dlbf);  
        xbdr.loadBeanDefinitions(new ClassPathResource("beans.xml"));  
  
        Spiderman s = dlbf.getBean("s", Spiderman.class);  
        s.webSwing();  
  
        Spiderman s2 = dlbf.getBean("s", Spiderman.class);  
        s2.webSwing();  
  
        System.out.println(s);  
        System.out.println(s2);  
    }  
}
```



1.1.5 실행결과

```
6월 23, 2015 12:25:33 오후 org.springframework
정보: Loading XML bean definitions from class path resource
com.kwon.avengers.Spiderman@699dd00d
com.kwon.avengers.Spiderman@699dd00d
```

기본적으로 Spring 은 객체를 singleton 으로
beans.xml 에서 scope 수정으로 바꾸기 가능



The screenshot shows the 'Spring Beans' IDE window. On the left, the 'Beans Overview' tab is active, showing a tree view with 'beans' and a sub-entry 's'. In the center, there are buttons for 'New Bean...', 'Up', and 'Down'. On the right, the 'factory-bean:', 'factory-method:', 'init-method:', 'lazy-init:', 'name:', 'parent:', and 'primary:' fields are visible. The 'scope:' field is highlighted with a red box, and its dropdown menu is open, showing options: 'prototype', 'request', 'session', and 'singleton'. The 'prototype' option is selected. Below the dropdown, there is a 'Documentation' section with text: 'Element : bean', 'Defines a single (usu', 'nested tags for const', 'methods, and replaced methods. Mixing constructor injection and'.

```
6월 23, 2015 12:30:30 오후 org.springframework
정보: Loading XML bean definitions from class path resource
com.kwon.avengers.Spiderman@449d8f06
com.kwon.avengers.Spiderman@4418f61b
```



2. ApplicationContext 사용

2.1 비교

BeanFactory

getBean() 메소드가 처음 호출될 때 singleton 객체 생성

ApplicationContext

모든 singleton bean 을 context 기동 시 미리 로딩

필요할 때 즉시 사용 가능한 상태로 유지

2.2 사용

```
public class SpringDIMain {
    public static void main(String[] args) {
        // DefaultListableBeanFactory dlbfb = new DefaultListableBeanFactory();
        // XmlBeanDefinitionReader xbdr = new XmlBeanDefinitionReader(dlbfb);
        // xbdr.loadBeanDefinitions(new ClassPathResource("beans.xml"));

        AbstractApplicationContext aac
            = new ClassPathXmlApplicationContext("beans.xml");
        aac.registerShutdownHook(); //jvm이 종료될 때 ApplicationContext종료

        Spiderman s = aac.getBean("s", Spiderman.class);
        s.webSwing();

        Spiderman s2 = aac.getBean("s", Spiderman.class);
        s2.webSwing();

        System.out.println(s);
        System.out.println(s2);

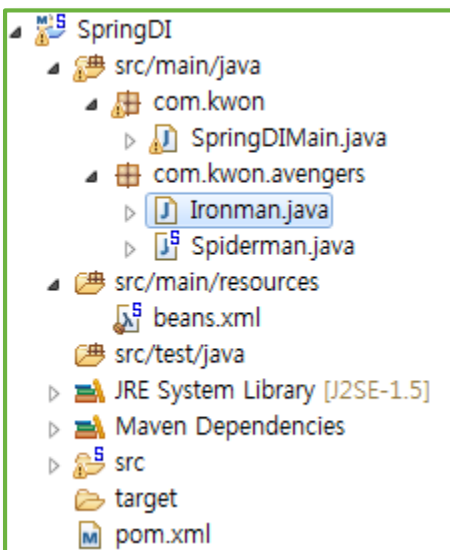
        aac.close(); // 등록된 모든 bean객체 제거
    }
}
```



3.SpringDI 로 OOP

3.1 setter, 생성자 활용

3.1.1 클래스 추가

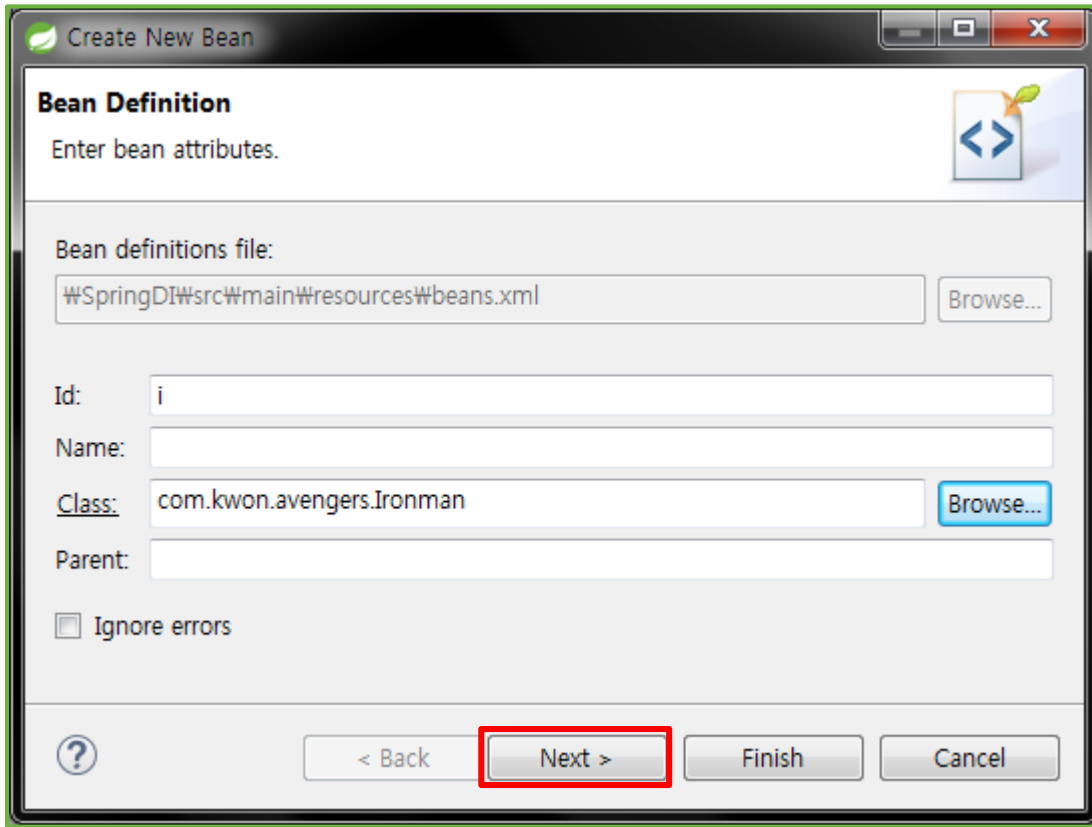


```
public class Ironman {  
    private String suitType;  
    private int money;  
  
    public Ironman() {  
        // TODO Auto-generated constructor stub  
    }  
  
    public Ironman(String suitType, int money) {  
        super();  
        this.suitType = suitType;  
        this.money = money;  
    }  
  
    public String getSuitType() {return suitType;}  
    public void setSuitType(String suitType) {this.suitType = suitType;}  
    public int getMoney() {return money;}  
    public void setMoney(int money) {this.money = money;}  
}
```



3.1.2 Properties : setter 활용

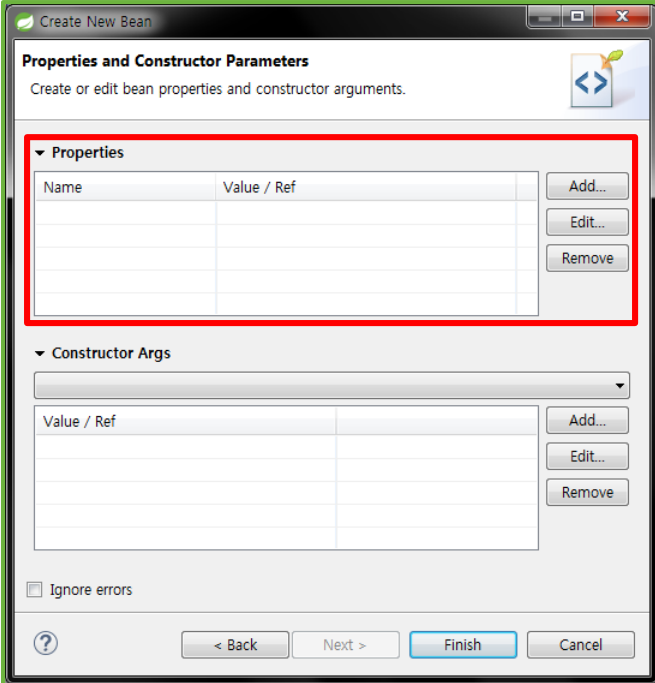
beans.xml beans 탭에서 New Bean...



The image shows a "Create New Bean" dialog box with the following fields and controls:

- Bean Definition** section with the instruction "Enter bean attributes."
- Bean definitions file:** A text field containing `₩SpringDI₩src₩main₩resources₩beans.xml` and a "Browse..." button.
- Id:** A text field containing the value `i`.
- Name:** An empty text field.
- Class:** A text field containing `com.kwon.avengers.Ironman` and a "Browse..." button.
- Parent:** An empty text field.
- An ☐ **Ignore errors** checkbox.
- Navigation buttons at the bottom: a help icon (?), "< Back", "Next >" (highlighted with a red rectangle), "Finish", and "Cancel".





Create New Bean

Properties and Constructor Parameters
Create or edit bean properties and constructor arguments.

Properties

Name	Value / Ref

Add...
Edit...
Remove

Constructor Args

Value / Ref

Add...
Edit...
Remove

☐ Ignore errors

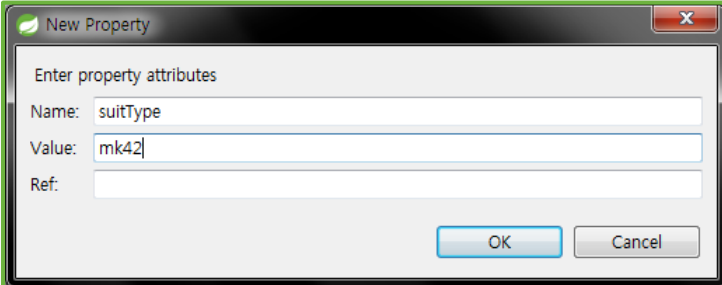
? < Back Next > Finish Cancel

Add.. 눌러서

Name : 멤버 변수명

Value : 값

Ref : 다른 객체 참조할 때



New Property

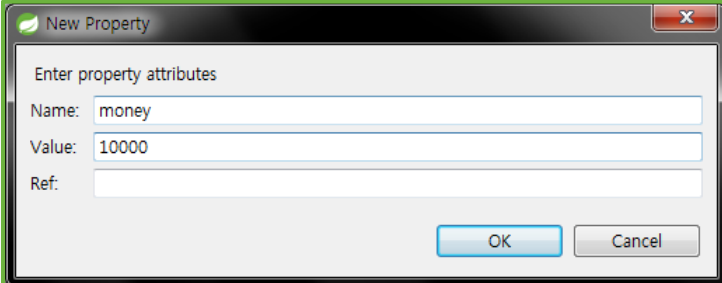
Enter property attributes

Name: suitType

Value: mk42

Ref:

OK Cancel



New Property

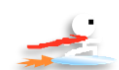
Enter property attributes

Name: money

Value: 10000

Ref:

OK Cancel



SpringDIMain.java

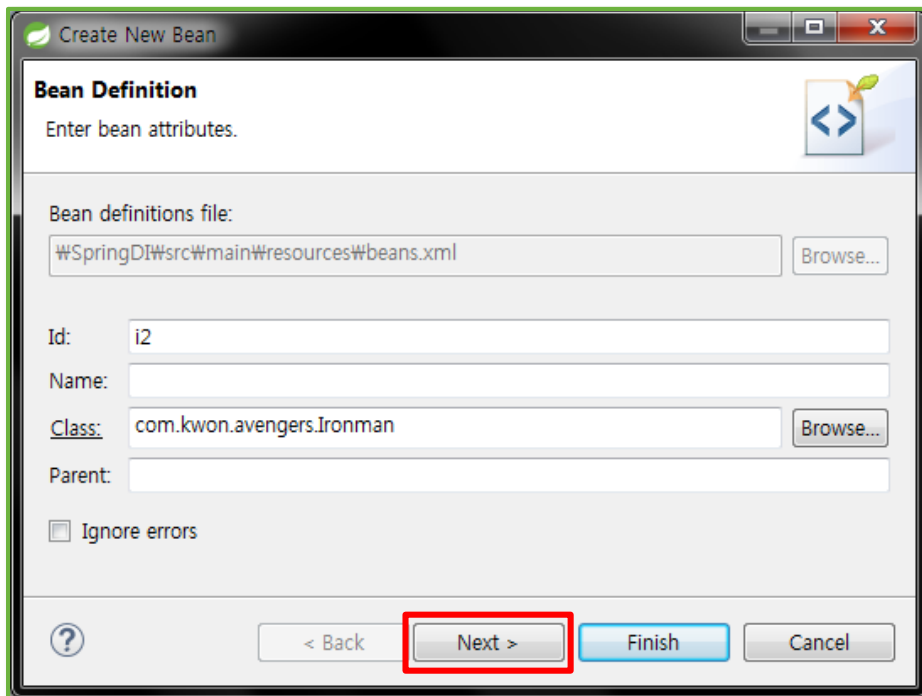
```
public class SpringDIMain {  
    public static void main(String[] args) {  
        DefaultListableBeanFactory dlbf = new DefaultListableBeanFactory();  
        XmlBeanDefinitionReader xbdr = new XmlBeanDefinitionReader(dlbf);  
        xbdr.loadBeanDefinitions(new ClassPathResource("beans.xml"));  
  
        Ironman i = dlbf.getBean("i", Ironman.class);  
        System.out.println("suitType : " + i.getSuitType());  
        System.out.println("money : " + i.getMoney());  
    }  
}
```

실행결과

```
6월 22, 2015 11:5  
정보: Loading XML  
suitType : mk42  
money : 10000
```

3.1.3 Constructor Args : 생성자 활용

beans.xml beans 탭에서 New Bean...



The image shows a 'Create New Bean' dialog box in an IDE. The 'Bean Definition' tab is active, with the instruction 'Enter bean attributes.' and a code icon. The 'Bean definitions file' field contains the path '#SpringDI#src#main#resources#beans.xml' and has a 'Browse...' button. The 'Id' field is filled with 'i2'. The 'Name' field is empty. The 'Class' field is filled with 'com.kwon.avengers.Ironman' and has a 'Browse...' button. The 'Parent' field is empty. There is an unchecked checkbox for 'Ignore errors'. At the bottom, there are four buttons: a help icon, '< Back', 'Next >' (which is highlighted with a red rectangle), 'Finish', and 'Cancel'.



Create New Bean

Properties and Constructor Parameters
Create or edit bean properties and constructor arguments.

▼ Properties

Name	Value / Ref

Add...
Edit...
Remove

▼ Constructor Args

Value / Ref

Add...
Edit...
Remove

☐ Ignore errors

? < Back Next > Finish Cancel

Create New Bean

Properties and Constructor Parameters
Create or edit bean properties and constructor arguments.

▼ Properties

Name	Value / Ref

Add...
Edit...
Remove

▼ Constructor Args

ironman(String, int)

Value / Ref
<EMPTY>
<EMPTY>

Add...
Edit...
Remove

☐ Ignore errors

? < Back Next > Finish Cancel

value : 값

ref : 다른 객체 읽을 때



Edit Constructor Argument

Enter constructor argument attributes

value:

ref:

Create New Bean

Properties and Constructor Parameters

Create or edit bean properties and constructor arguments.

▼ Properties

Name	Value / Ref

▼ Constructor Args

Ironman(String, int)

Value / Ref
필크버스터
50000

☐ Ignore errors



SpringDIMain.java

```
public class SpringDIMain {  
    public static void main(String[] args) {  
        DefaultListableBeanFactory dlbf = new DefaultListableBeanFactory();  
        XmlBeanDefinitionReader xbdr = new XmlBeanDefinitionReader(dlbf);  
        xbdr.loadBeanDefinitions(new ClassPathResource("beans.xml"));  
  
        Ironman i = dlbf.getBean("i", Ironman.class);  
        System.out.println("suitType : " + i.getSuitType());  
        System.out.println("money : " + i.getMoney());  
  
        Ironman i2 = dlbf.getBean("i2", Ironman.class);  
        System.out.println("suitType : " + i2.getSuitType());  
        System.out.println("money : " + i2.getMoney());  
    }  
}
```

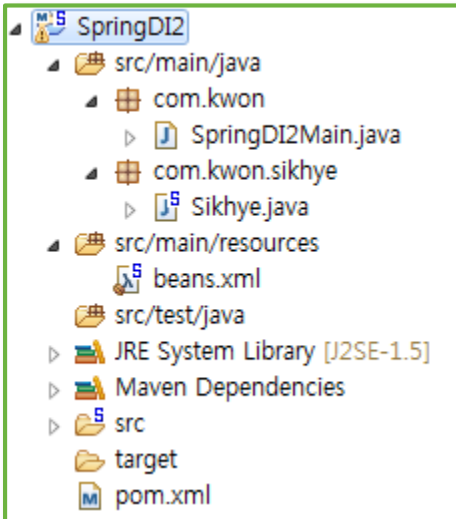
실행결과

```
6월 22, 2015 12:04:2  
정보: Loading XML be  
suitType : mk42  
money : 10000  
suitType : 헐크버스터  
money : 50000
```



3.2 컬렉션 사용

3.2.1 프로젝트 작성



```
public class Sikhye {  
    private String[] info;  
    private HashMap<String, Double> nutrient;  
  
    public String[] getInfo() {  
        return info;  
    }  
  
    public void setInfo(String[] info) {  
        this.info = info;  
    }  
  
    public HashMap<String, Double> getNutrient() {  
        return nutrient;  
    }  
  
    public void setNutrient(HashMap<String, Double> nutrient) {  
        this.nutrient = nutrient;  
    }  
}
```



```
public class SpringDI2Main {
    public static void main(String[] args) {
        DefaultListableBeanFactory dlbf = new DefaultListableBeanFactory();
        XmlBeanDefinitionReader xbdr = new XmlBeanDefinitionReader(dlbf);
        xbdr.loadBeanDefinitions(new ClassPathResource("beans.xml"));

        Sikhye s = dlbf.getBean("s", Sikhye.class);

        String[] info = s.getInfo();
        for (String i : info) {
            System.out.println(i);
        }

        HashMap<String, Double> nutrient = s.getNutrient();
        System.out.println("탄수화물 : " + nutrient.get("탄수화물"));
        System.out.println("지방 : " + nutrient.get("지방"));
        System.out.println("단백질 : " + nutrient.get("단백질"));
    }
}
```

3.2.2 beans.xml

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

    <bean id="s" class="com.kwon.sikhye.Sikhye">
        <property name="info">
            <list value-type="java.lang.String">
                <value>설탕</value>
                <value>밥알</value>
                <value>물</value>
            </list>
        </property>
        <property name="nutrient">
            <map>
                <entry key="탄수화물" value="0.7"></entry>
                <entry key="지방" value="0.2"></entry>
                <entry key="단백질" value="0.1"></entry>
            </map>
        </property>
    </bean>

</beans>
```

[], ArrayList 는 사용법 같음

set 의 경우는 <set><value></value></set>...으로



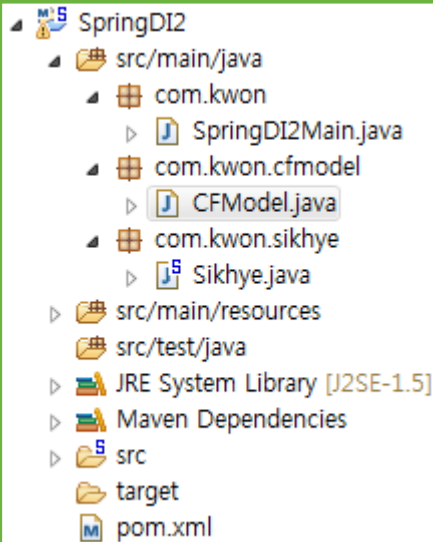
3.2.3 실행결과

6월 22, 2015 12:4
정보: Loading XML
설탕
밥알
물
탄수화물 : 0.7
지방 : 0.2
단백질 : 0.1



3.3 Has A 관계

3.3.1 클래스 추가/수정



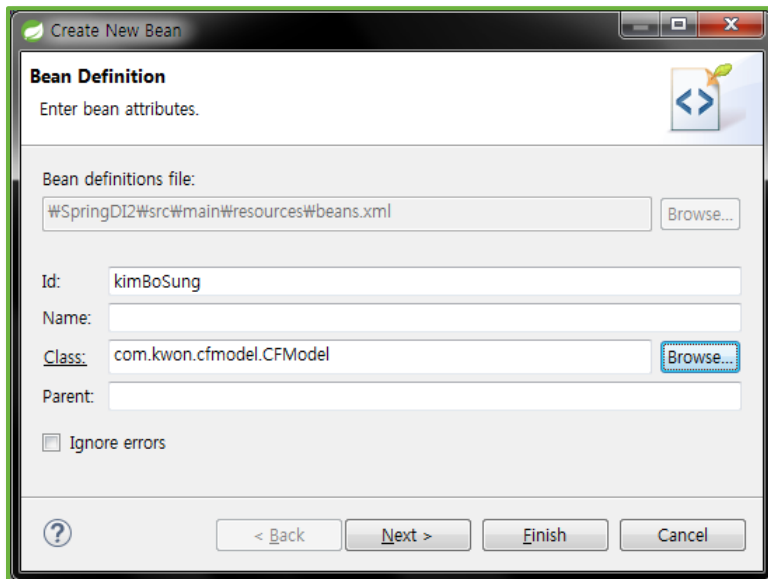
```
public class CFModel {  
    private String comment;  
  
    public CFModel(String comment) {  
        super();  
        this.comment = comment;  
    }  
  
    public void say() {  
        System.out.println(comment);  
    }  
}
```

```
public class Sikhye {  
    private String[] info;  
    private HashMap<String, Double> nutrient;  
    private CFModel kimBoSung;  
  
    public CFModel getKimBoSung() {  
        return kimBoSung;  
    }  
  
    public void setKimBoSung(CFModel kimBoSung) {  
        this.kimBoSung = kimBoSung;  
    }  
  
    public String[] getInfo() {return info;}  
    public void setInfo(String[] info) {this.info = info;}  
    public HashMap<String, Double> getNutrient() {return nutrient;}  
    public void setNutrient(HashMap<String, Double> nutrient) {this.nutrient = nutrient;}  
}
```



3.3.2 beans.xml

김보성 생성



Create New Bean

Bean Definition
Enter bean attributes.

Bean definitions file:
W:\SpringDI2\src\main\resources\beans.xml Browse...

Id: kimBoSung

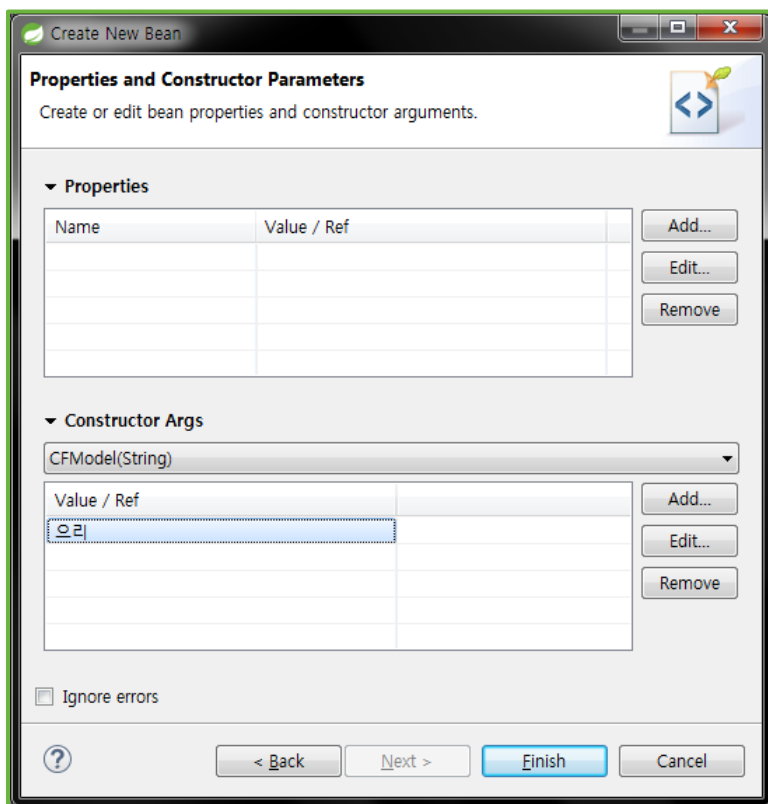
Name:

Class: com.kwon.cfmodel.CFModel Browse...

Parent:

☐ Ignore errors

? < Back Next > Finish Cancel



Create New Bean

Properties and Constructor Parameters
Create or edit bean properties and constructor arguments.

▼ Properties

Name	Value / Ref

Add...
Edit...
Remove

▼ Constructor Args

CFModel(String)

Value / Ref
오리

Add...
Edit...
Remove

☐ Ignore errors

? < Back Next > Finish Cancel



식해 s 에 property 추가

Spring Beans

Beans Overview

Select an element to edit its details.

New Bean...
Up
Down

- Insert <constructor-arg> element
- Insert <description> element
- Insert <lookup-method> element
- Insert <meta> element
- Insert <property> element
- Insert <qualifier> element
- Insert <replaced-method> element
- Delete <bean> element

Element Details

Set the properties of the selected element. Required fields are denoted by "*".

id:

abstract:

autowire:

autowire-candidate:

class:

depends-on:

destroy-method:

factory-bean:

factory-method:

init-method:

lazy-init:

Source | Namespaces | Overview | beans | Beans Graph

Spring Beans

Beans Overview

Select an element to edit its details.

New Bean...
Up
Down

- beans
 - s
 - info
 - nutrient
 - kimBoSung
 - kimBoSung

Element Details

Set the properties of the selected element. Required fields are denoted by "*".

name*:

ref:

value:

Documentation

Element : property
Bean definitions can have zero or more properties. Property elements correspond to JavaBean setter methods exposed by the bean classes. Spring supports primitives, references to other beans in the same or related factories, lists, maps and properties.

Content Model : (description?, (meta | bean | ref | idref | value | null | array | list | set | map | props | namespace:uri="##other")?)

[Click for additional documentation at springsource.org](#)

Source | Namespaces | Overview | beans | Beans Graph



3.3.3 main 클래스

```
public class SpringDI2Main {  
    public static void main(String[] args) {  
        DefaultListableBeanFactory dlbf = new DefaultListableBeanFactory();  
        XmlBeanDefinitionReader xbdr = new XmlBeanDefinitionReader(dlbf);  
        xbdr.loadBeanDefinitions(new ClassPathResource("beans.xml"));  
  
        Sikhye s = dlbf.getBean("s", Sikhye.class);  
  
        String[] info = s.getInfo();  
        for (String i : info) {  
            System.out.println(i);  
        }  
  
        HashMap<String, Double> nutrient = s.getNutrient();  
        System.out.println("탄수화물 : " + nutrient.get("탄수화물"));  
        System.out.println("지방 : " + nutrient.get("지방"));  
        System.out.println("단백질 : " + nutrient.get("단백질"));  
  
        s.getKimBoSung().say();  
    }  
}
```

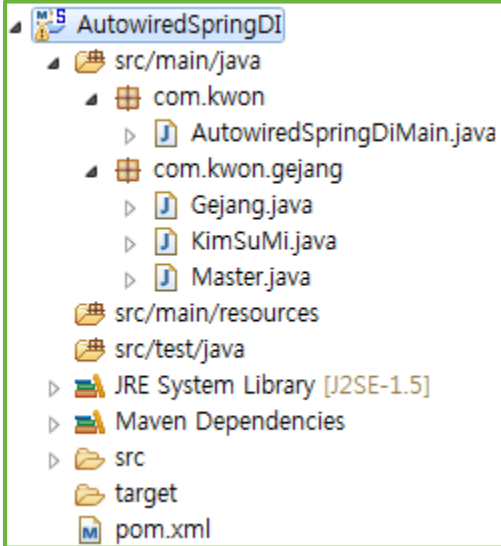
3.3.4 실행결과

```
6월 23, 2015 11:54:00 AM  
정보: Loading XML  
설탕  
밥알  
물  
탄수화물 : 0.7  
지방 : 0.2  
단백질 : 0.1  
으리
```



3.4 Autowired 활용 Has A

3.4.1 프로젝트 작성



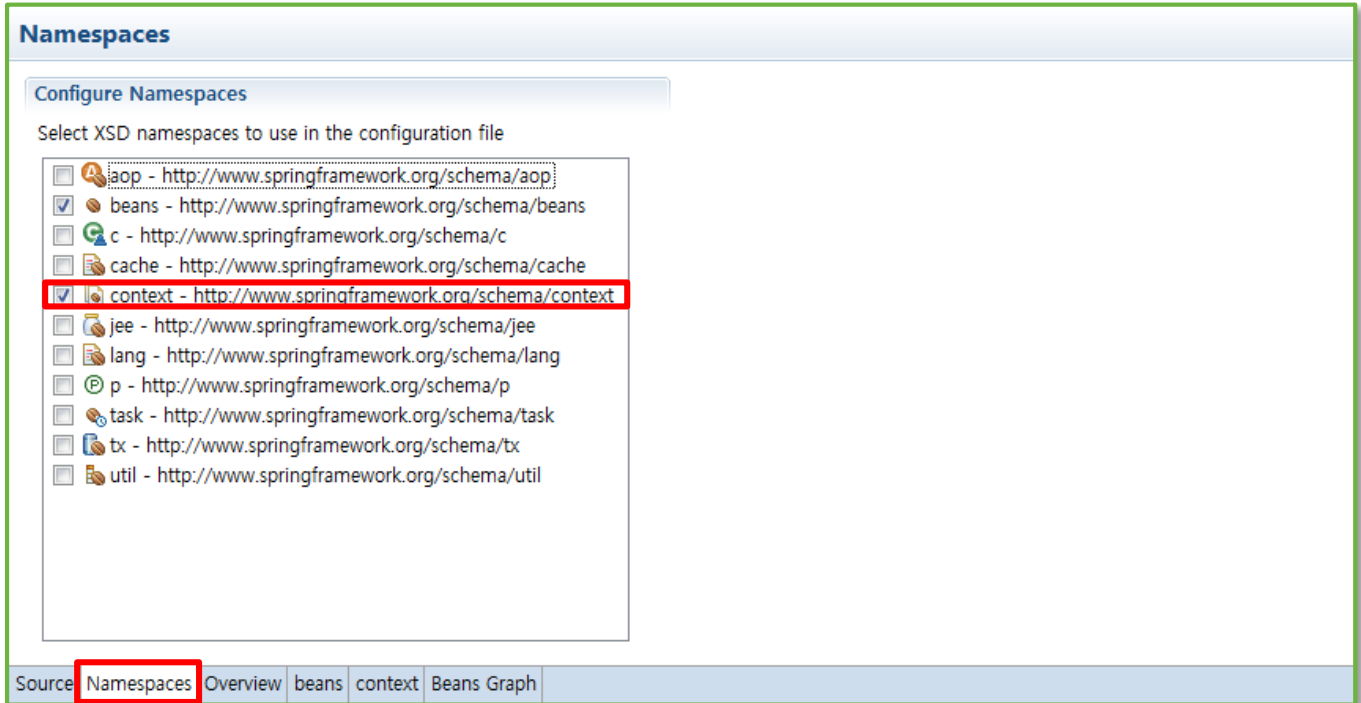
```
public interface Master {  
    public abstract void bargain();  
}
```

```
public class KimSuMi implements Master{  
    @Override  
    public void bargain() {  
        // TODO Auto-generated method stub  
        System.out.println("멤뵁");  
    }  
}
```

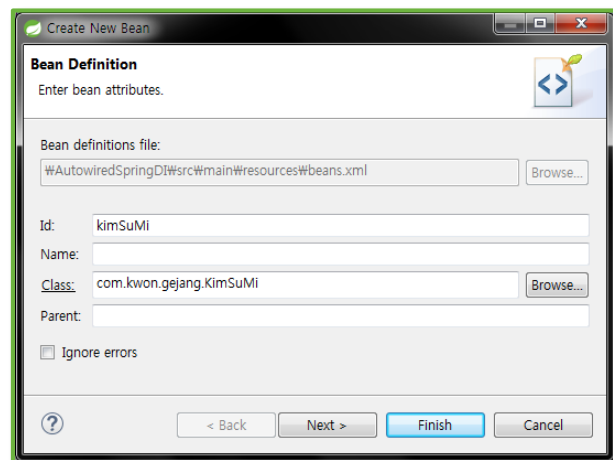
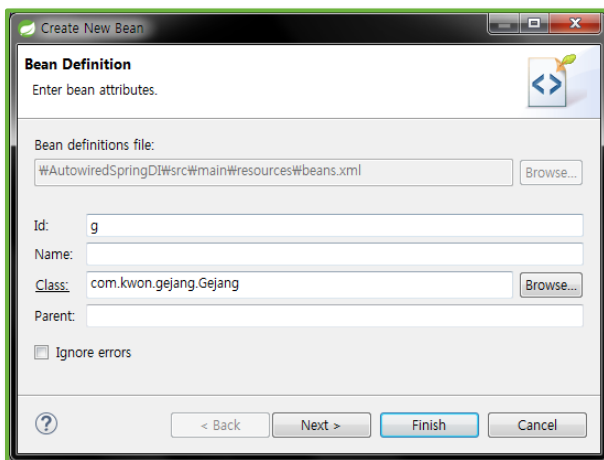
```
public class Gejang {  
    @Autowired  
    private Master master;  
  
    public void bargain() {  
        master.bargain();  
    }  
}
```



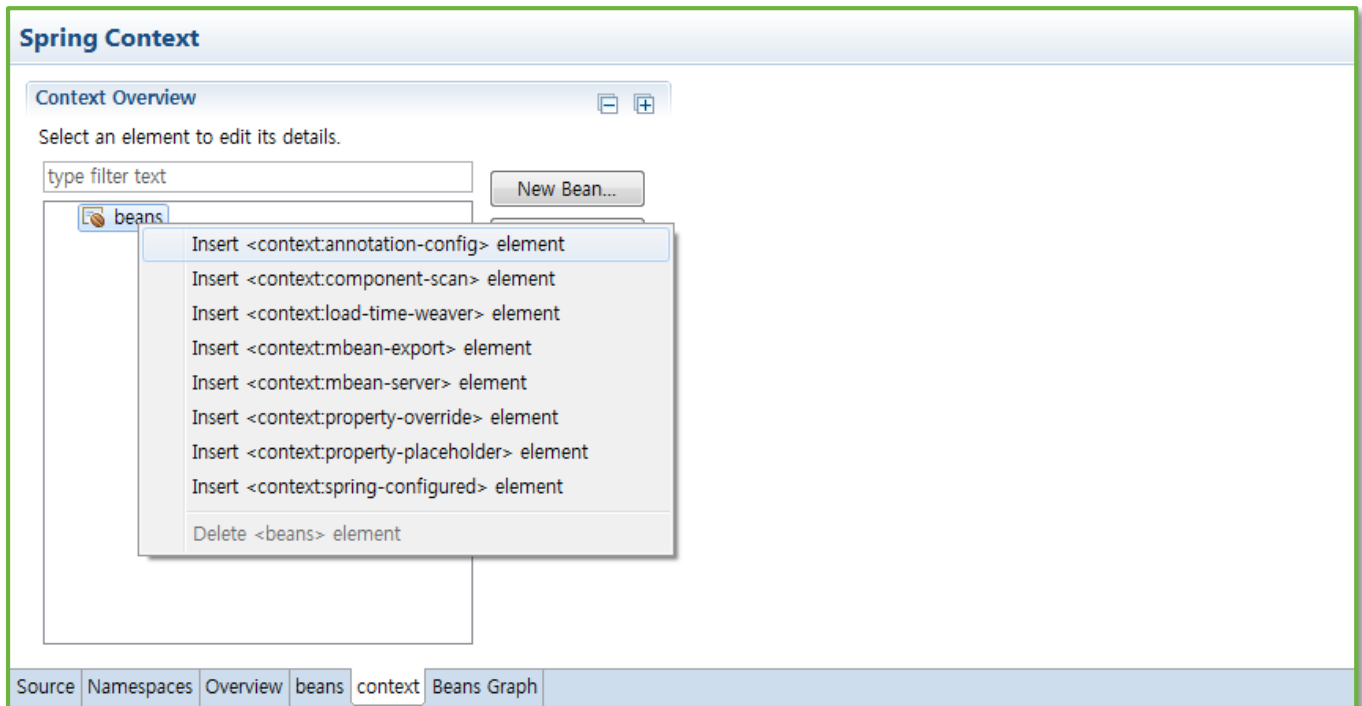
3.4.2 beans.xml



두 객체 모두 등록



context 탭에서 추가



3.4.3 main 클래스

```
public class AutowiredSpringDiMain {  
  
    public static void main(String[] args) {  
  
        AbstractApplicationContext aac  
            = new ClassPathXmlApplicationContext("beans.xml");  
        aac.registerShutdownHook();  
  
        Gejang g = aac.getBean("g", Gejang.class);  
        g.bargain();  
  
        aac.close();  
    }  
}
```



3.4.4 실행결과

```
6월 23, 2015 1
정보: Refreshi
6월 23, 2015 1
정보: Loading
6월 23, 2015 1
정보: Pre-inst
메 병
6월 23, 2015 1
정보: Closing
6월 23, 2015 1
정보: Destroyi
```

