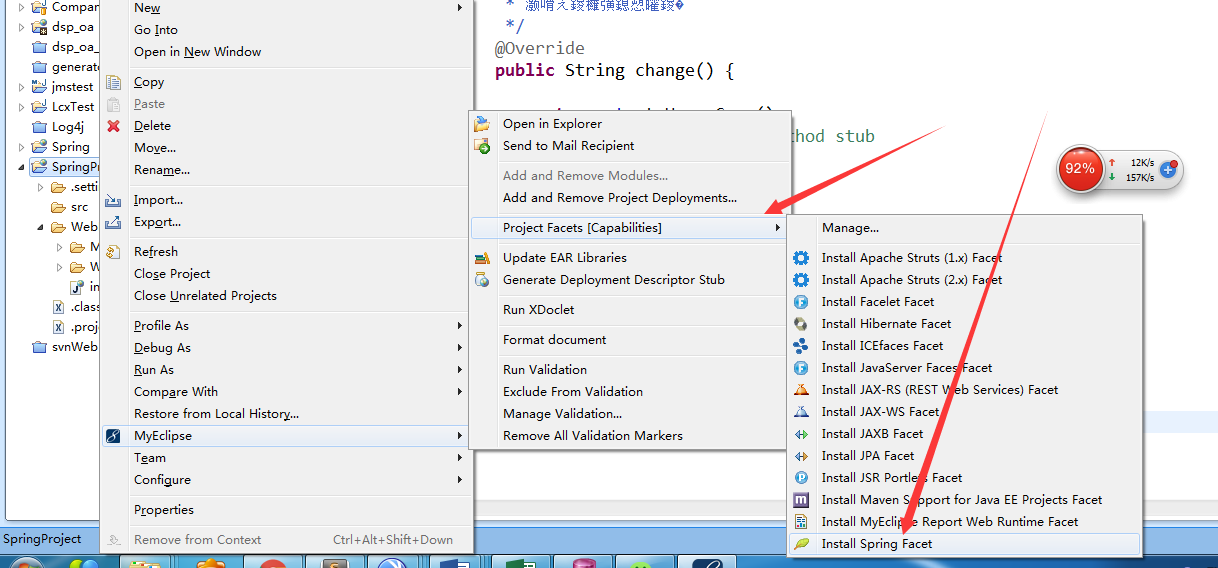
# 1、spring项目的建立

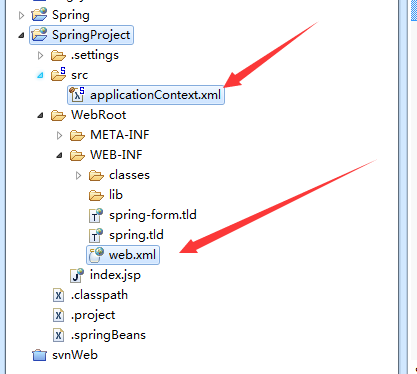
## 1、先建立一个简单的web项目

## 2、选中项目鼠标右键搭载spring环境



## 3、spring搭载完成主要信息如下

### 1、项目结构如下图



### 2、web.xml中

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns=*"http://java.sun.com/xml/ns/javaee"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"* version=*"3.0"*>  <display-name>SpringProject</display-name>  <listener>  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  </listener>  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:applicationContext.xml</param-value>  </context-param>  </web-app> |

### 3、spring配置文件

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <beans  xmlns=*"http://www.springframework.org/schema/beans"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xmlns:p=*"http://www.springframework.org/schema/p"*  xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.1.xsd"*>  </beans> |

### 4、测试spring项目建立成功

#### 1、建立service

|  |
| --- |
| **package** com.hlj.springTestSuccess;  **public** **class** TestService {  **private** String name;  **public** String getName() {  **return** name;  }  **public** **void** setName(String name) {  **this**.name = name;  }    **public** **void** testSpringBean(){  System.*out*.println("bye"+name);  }  } |

#### 2、spring中注入service

|  |
| --- |
| <bean id=*"testService"* class=*"com.hlj.springTestSuccess.TestService"*>  <!-- 这里就体现出注入的概念. -->  <property name=*"name"* value=*"张宇晋"* />  </bean> |

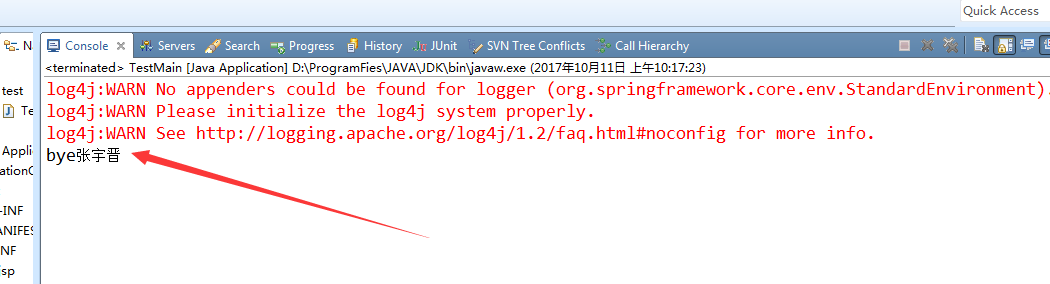
#### 3、建立单例模式获取spring配置文件

|  |
| --- |
| **final** **public** **class** ApplicaionContextUtil {  **private** **static** ApplicationContext *ac*=**null**;    **private** ApplicaionContextUtil(){    }      **privice** **static** **synchronized** **void** installAc(){  **if**(*ac*==**null**){  *ac* = **new** ClassPathXmlApplicationContext("applicationContext.xml");  }  }    **public** **static** ApplicationContext getApplicationContext(){  **if**(*ac*==**null**){  *installAc*();  }  **return** *ac*;  }    } |

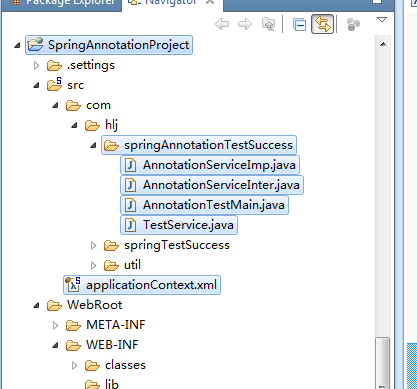
#### 4、开始测试

|  |
| --- |
| /\*\*  \* **@author** 作者 HealerJean  \* **@version** 创建时间：2017年10月11日 上午10:05:22  \* 类说明  \*/  **public** **class** TestMain {  **public** **static** **void** main(String[] args) {    ApplicationContext applicationContext = ApplicaionContextUtil.*getApplicationContext*();    TestService testService = (TestService) applicationContext.getBean("testService");  testService.testSpringBean();  }  } |

#### 5、控制台打印



# 2、spring项目使用注解开发SpringAnnotationProject



## 1、上面利用eclipse建立的spring 配置文件applicationContext 中缺少信息，所以不能使用注解扫描配置，需要替换为下面的更全面（中科软复制过来的）

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <beans xmlns=*"http://www.springframework.org/schema/beans"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xmlns:context=*"http://www.springframework.org/schema/context"*  xmlns:tx=*"http://www.springframework.org/schema/tx"*  xmlns:aop=*"http://www.springframework.org/schema/aop"*  xsi:schemaLocation=*"*  *http://www.springframework.org/schema/beans*  *http://www.springframework.org/schema/beans/spring-beans-3.1.xsd*  *http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd*  *http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.0.xsd*  *http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.0.xsd"*>  <context:annotation-config /> |

## 2、建立接口AnnotationServiceInter

|  |
| --- |
| **public** **interface** AnnotationServiceInter {  **public** **void** testSpringAnnotationBean();  } |

## 3、实现接口类AnnotationService

|  |
| --- |
| **public** **class** *AnnotationServiceImp* **implements** AnnotationServiceInter {  **public** **void** testSpringAnnotationBean(){  System.*out*.println("AnnotationTestService");  }  } |

## 4、spring中注册这个实现接口

|  |
| --- |
| <context:annotation-config />  <bean id=*"idannotationService"* class=*"com.hlj.springAnnotationTestSuccess.AnnotationServiceImp"*></bean> |

## 5、为了使用Resource，要再建立一个service，对象的名字是随意起的，就可以成功。

|  |
| --- |
| **public** **class** TestService {    @Resource  **public** AnnotationServiceInter nameService;    **public** **void** test (){  nameService.testSpringAnnotationBean();  }  } |

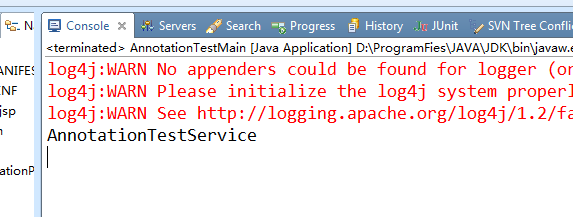
## 6、注册这个TestService

|  |
| --- |
| <context:annotation-config />  <bean id=*"idannotationService"* class=*"com.hlj.springAnnotationTestSuccess.AnnotationServiceImp"*></bean>  <bean id=*"idtestService"* class=*"com.hlj.springAnnotationTestSuccess.TestService"*></bean> |

## 7、开始测试

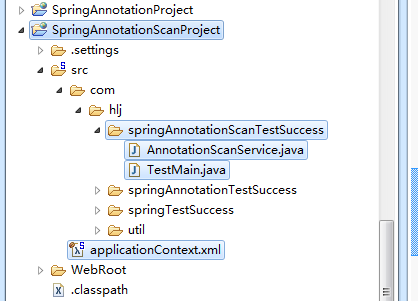
|  |
| --- |
| **public** **class** AnnotationTestMain {    **public** **static** **void** main(String[] args) {    ApplicationContext applicationContext = ApplicaionContextUtil.*getApplicationContext*();    TestService testService = (TestService) applicationContext.getBean("idtestService");  //下面这个是错误的，因为new的话，不能够将它里面的resource 注入进去  // TestService testService = new TestService();  testService.test();  }    } |

## 8、控制台打印



# 3、spring 使用扫描包 <context:component-scan

## 这种方式了具有<context:annotation-config>的功能之外，<context:component-scan>还可以在指定的package下扫描以及注册javabean），也就是说有了这个可以不使用 <context:annotation-config />[也可以对@service生效，但是不能对applicationContext.xml中注册的的bean](mailto:也可以对@service生效，但是不能对applicationContext.xml中注册的的bean)生效



## 1、建立service使用注解@Service

|  |
| --- |
| @Service  **public** **class** AnnotationScanService {  **public** **void** testSpringAnnotationScanBean(){  System.*out*.println("AnnotationScanService");  }    } |

## 2、applicatonContext.xml添加自动扫描包路径，只写com也能生效，这样就是扫描com下所有的类

|  |
| --- |
| <context:component-scan base-package=*"com"* />  <context:component-scan base-package=*"com.hlj.springAnnotationTestSuccess"* /> |

## 3、测试，不在applicationContext.xml中注册bean也可以获取

|  |
| --- |
| **public** **class** TestMain {  **public** **static** **void** main(String[] args) {    ApplicationContext applicationContext = ApplicaionContextUtil.*getApplicationContext*();    AnnotationScanService annotationScanService = (AnnotationScanService) applicationContext.getBean(AnnotationScanService.**class**);    annotationScanService.testSpringAnnotationScanBean();  }  } |

# 4、