@Value注解给对象的简单类型属性赋值

在c-spring-di-anno-03项目中新建一个bao02包，将bao01中的Student类复制过来，然后在其中使用@Value注解

Student类的内容如下：

```
package com.studymyself.bao02;

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


@Component("student02")
public class Student {

    public Student() {
        System.out.println("Student02无参构造方法执行！");
    }

    /**
     * @Value:简单类型属性赋值
     *       注解中的属性value：在注解中该属性是String类型的，所以需要用双引号括起来
     *    使用方法：
     *    1、使用在类中的属性上面，不调用set方法，底层使用反射机制赋值该属性，推荐使用
     *    2、使用在java类的set方法上，调用该方法进行属性的赋值
     *    下面展示使用在属性字段上
     *    该注解配置的组件扫描器跟@Component一样，所以无需多配置该注解的组件扫描器
     *    扫描@Component时也会扫描@Value
     */
    @Value("终南山")//value可省略
    private String name;
    @Value(value = "5214")
    private int age;

    public void setName(String name) {
        this.name = name;
    }

    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "Student{" +
                "name='" + name + '\'' +
                ", age=" + age +
                '}';
    }
}
```

新建测试类MyTest02.java，注意使用Student类的时候要注意导入使用那个包下的Student类，不然很容易报错

```
package com.studymyself;

import com.studymyself.bao02.Student;//注意使用的是哪个包下的Student类
import org.junit.Test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MyTest02 {

    @Test
    public void testStudent(){

        //配置文件路径
        String config = "applicationContext.xml";

        //获取容器对象
        ApplicationContext ac = new ClassPathXmlApplicationContext(config);

        //获取Student类对象
        Student student1 = (Student) ac.getBean("student02");
        System.out.println("第一次获取的student1"+student1);

    }

}
```