初始Spring框架

一、Spring概述

含义: Spring是一个轻量级的控制反转(IOC)和面向切面(AOP)的容器框架

春天【spring的出现给软件行业带来一个春天】

网址：spring.io 我们学习它的framework模块

API：https://spring.io/projects/spring-framework

理念：使现有技术更加的实用【实质就是整合现有的框架技术】

优点:

a. 轻量级框架（不具有侵入性-使用它不用去继承或者实现某些接口，灵活性好）

b. 控制反转(IOC):Spring使用控制反转技术实现了松耦合。依赖被注入到对象，而不是创建或寻找依赖对象

c. 面向切面编程(AOP):支持面向切面编程，把应用业务逻辑与系统的服务分离开来

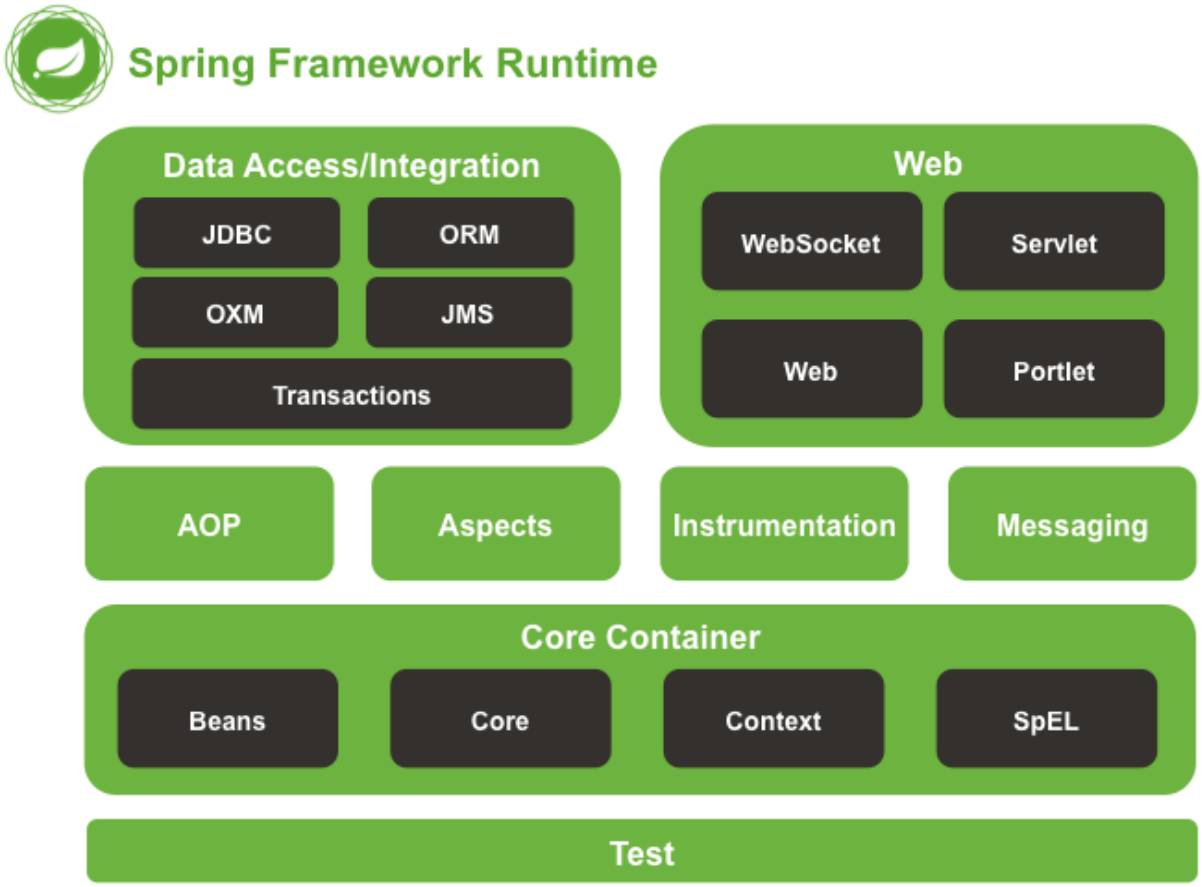
d. 容器:Spring包含并管理对象的配置及生命周期。

e.MVC框架:Spring的web框架是一个设计优良的web MVC框架，很好的取代了一些web框架。

f. 事务管理:Spring对本地业务和全局业务(JAT)提供了统一的事务管理接口。

g.异常处理:Spring提供了一个方便的API将特定技术的异常(由JDBC, Hibernate, 或JDO抛出的异常)转化为统一的Unchecked异常。

二、核心架构



三、控制反转【IOC】

IOC-inversion of control 控制反转

1、导入依赖

```
<dependencies>
 <!--spring-context依赖 -->
   <dependency>
       <groupId>org.springframework</groupId>
       <artifactId>spring-context</artifactId>
       <version>5.2.4.RELEASE</version>
   </dependency>
 </dependencies>
```

2、编写实体类

```java
public class Student {

    private String sname;

    public void setSname(String sname) {
        this.sname = sname;
    }
    public void show(){
        System.out.println("我叫:"+sname);
    }
}
```

3、编写配置文件

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        https://www.springframework.org/schema/beans/spring-beans.xsd">
       <bean id="stu" class="com.ls.entity.Student">
        <property name="sname" value="张三"/>
       </bean>
 </beans>
```

4、编写测试类

```java
public class TestIOC {
    public static void main(String[] args) {
        //解析xml并创建应用程序容器对象
        ApplicationContext ac=new ClassPathXmlApplicationContext("spring-config.xml");
        //Student stu=(Student)ac.getBean(Student.class);
        Student stu=(Student)ac.getBean("stu");
        stu.show();
    }
}
```

5、思考问题

Student对象是由谁创建的，其属性是由谁设置的？均由Spring容器来完成的，

这个过程就是IOC-也叫 控制反转

6、真正理解控制反转

控制:由谁来控制对象的创建

以前对象的创建是由程序本身或者说是我们自己创建，使用Spring后由Spring容器来创建对象

反转:指程序本身不创建对象，而变为被动接收对象

结论:IOC的实质就是spring帮我们创建、管理、维护对象

7、IOC创建对象的方式

IOC常见创建对象方式 A.无参构造 B.有参构造

```
public Student(String sname) {//有参构造--在实体类中
    super();
    this.sname = sname;
}

public Student() {//无参构造
    super();
}
```

```
<!--核心配置文件中-->
<bean id="stu1" class="com.ls.entity.Student">
    <constructor-arg name="sname" value="燕子"/>
</bean>
```

```
Student stu1=(Student)ac.getBean("stu1");//构造器创建对象--测试类中
stu1.show();
```

四、依赖注入[DI]

1、概念理解

控制反转也叫依赖注入(DI- Dependency Injection)

依赖：指bean对象创建依赖于容器

注入：指bean对象依赖的资源由容器来设置和装配

2、Spring注入方式

A、构造器注入(见IOC创建对象)

B、Setter注入

要求被注入的属性必须有set方法。

set方法的方法名由set+属性首字母大写格式，如果属性是boolean类型

则方法名由is+属性首字母大写格式

b0、实体类Student和User

```
public class Student {

```

```
    private String sname;
    private Date dt;//出生日期
    private String[] likes; //爱好
    private List<String> friends; //朋友
    private Set<String> games; //游戏
    private Map<String,String> cards;//会员卡
    private String work; //工作
    private Properties baseinfo; //基本信息

    public void setSname(String sname) {
        this.sname = sname;
    }

    //封装打印
    public void pt(Object obj){
        System.out.println(obj);
    }

    public void show(){ //自我介绍
        pt("我叫:"+sname);
        pt("出生日期:"+new SimpleDateFormat("yyyy-MM-dd").format(dt));
        pt("爱好:"+likes);
        pt("朋友:"+friends);
        pt("游戏:"+games);
        pt("会员卡:"+cards);
        pt("工作:"+work);
        pt("信息:"+baseinfo);
    }

    public Student(String sname) {//有参构造
        super();
        this.sname = sname;
    }

    public Student() {//无参构造
        super();
    }

    public void setDt(Date dt) {
        this.dt = dt;
    }
    public void setLikes(String[] likes) {
        this.likes = likes;
    }
    public void setFriends(List<String> friends) {
        this.friends = friends;
    }
    public void setGames(Set<String> games) {
        this.games = games;
    }
    public void setCards(Map<String, String> cards) {
        this.cards = cards;
    }
```

```
    public void setWork(String work) {
        this.work = work;
    }
    public void setBaseinfo(Properties baseinfo) {
        this.baseinfo = baseinfo;
    }
}
```

```
public class User {

    private String name;
    private String sex;
    private int age;


    public void show(){ //自我介绍方法
        System.out.println(name+":"+sex+":"+age);
    }

    public void setName(String name) {
        this.name = name;
    }
    public void setSex(String sex) {
        this.sex = sex;
    }
    public void setAge(int age) {
        this.age = age;
    }

    public User(String name, String sex, int age) {
        super();
        this.name = name;
        this.sex = sex;
        this.age = age;
    }
    public User() {
        super();
    }

}
```

b1、常量注入

```
<bean id="stu" class="com.ls.entity.Student">
  <property name="sname" value="张三"/>
</bean>
```

b2、Bean注入

```
<!--Bean注入 -->
<property name="dt" ref="dt"/>
 <!--日期时间对象 -->
<bean id="dt" class="java.util.Date"/>
```

b3、数组注入

```
<!--数组注入 -->
<property name="likes">
    <array>
        <value>桌球</value>
        <value>蓝球</value>
        <value>足球</value>
    </array>
</property>
```

b4、List注入

```
<!--list注入 -->
<property name="friends">
    <list>
        <value>张三</value>
        <value>李四</value>
        <value>王五</value>
    </list>
</property>
```

b5、Set注入

```
<!--Set注入-->
<property name="games">
    <set>
        <value>王者毒药</value>
        <value>cs2</value>
    </set>
</property>
```

b6、Map注入

```
<!--Map注入 -->
<property name="cards">
    <map>
        <entry key="京东" value="888"/>
        <entry key="天猫" value="999"/>
    </map>
</property>
```

b7、Null注入

```
<!--null注入 -->
<property name="work" value="null"/>
```

b8、Properties注入

```
<!--Properties -->
<property name="baseinfo">
    <props>
        <prop key="籍贯">北京</prop>
        <prop key="性别">男</prop>
    </props>
</property>
```

b9、P命名空间注入

```
<!--加上p命名空间xmlns:p="http://www.springframework.org/schema/p"-->
<!--p命令空间注入-属性设置set方法 -->
<bean id="u1" class="com.ls.entity.User" p:name="小李"/>
```

b10、C命名空间注入

```
<!--加上c命名空间xmlns:c="http://www.springframework.org/schema/c"-->
<!--c命令空间注入-要求有对应参数的构造方法 -->
<bean id="u2" class="com.ls.entity.User" c:name="小雨" c:sex="女" c:age="18"/>
```

b11、测试Setter注入

```
User u1=(User)ac.getBean("u1");
u1.show();
User u2=(User)ac.getBean("u2");
u2.show();
```

五、更多视频扩展资源

今日头条：https://www.toutiao.com/c/user/1736832959129566/ 进去后-选择视频-在线点击观看

哔哩哔哩：https://space.bilibili.com/425233867