#### Spring脱坑记

##### 之Spring IoC：Introduction

问题描述

Spring IoC

解决方案

**什么是IoC(控制反转)**？

控制反转是面向对象编程中的一种设计原则，可以用来降低计算机代码之间的耦合度。控制反转容器可以帮助避免接口和实现的耦合，即接口的实现不能硬编码，避免耦合。编码的时候只关注接口，接口的具体实现依赖容器完成。

常用的IoC容器有：Spring Beans / Context、EJB、Pico Container、Avalon、JBoss、HiveMind等。

**Spring IoC容器有哪些特点**？

有ApplicationContext并能按照类型获取Bean

被@Component标记的类会被自动加载到容器中

被@Autowired标记的属性会自动注入

Spring IoC采用两级级联容器：DispatcherServlet创建的IoC容器的父容器就是ContextLoaderListener创建的IoC容器。父容器管理的Bean可以动态注入到子容器中；同时，保证子容器中的Bean相互隔离。这种设计思想曾用于Ali WebX框架（Deprecated）

**项目中是如何使用Spring IoC容器的**？

private void createApplicationContext() {  
 File[] applicationContextFiles = PmFileUtils.*getFiles*(new File(PmFileUtils.*getAppHome*() + "/conf/xml"), Pattern.*compile*("appctx.\*\\.xml"));  
 //转换为路径字符串  
 List<String> applicationContextFileLocations = new ArrayList<String>();  
 if (applicationContextFiles != null) {  
 for (File acf : applicationContextFiles) {  
 if (acf == null) {  
 continue;  
 }  
 //为解决unix系统上spring认为以"/"开头的路径是相对路径，所以在文件路径前再加一个"file:".  
 applicationContextFileLocations.add("file:" + acf.getAbsolutePath());  
 }  
 }  
 //使用FileSystemXmlApplicationContext建立上下文对象  
 ApplicationContext ctx = new FileSystemXmlApplicationContext(applicationContextFileLocations.toArray(new String[0]));  
 //添加到SpringBeanFactory中，使得句柄被一直持有  
 SpringBeanFactory.*setApplicationContext*(ctx);  
}

项目采用单一IoC容器，一次读出并加载加载所有Java Bean。小型项目避免使用Spring默认配置（级联容器）的成功实践。

□