#### Spring Boot

##### 之快速使用Spring IoC容器

问题描述

Spring Boot 2.2.5

JDK 8

在程序设计语言发展的幼年时期，人们认为能够从实例获取类就足够了。直到组件化编程的出现，接口实现对接口的依赖造成了诸多不便，于是，从类获取实例也变得十分必要。基于这种需求，出现了新的语言特性——反射。在反射特性的基础上，出现了IoC（控制反转）容器技术。

可以将Java Bean注册到Application Context（看作一张索引表）中，在Controller中通过id获取Bean实例，从而搭建起连接Model层和Controller层的桥梁。Spring框架需要，但是缺少一个通用的Application Context工具类，在这篇笔记中摘录一个比较好用的实现。

在Application的start方法中读取ctx配置文件并加载应用上下文至工具类中，即可实现IoC容器的快速使用

解决

SpringBeanFactory.java

public class SpringBeanFactory {

private static final SpringBeanFactory instance = new SpringBeanFactory();

private ApplicationContext ctx = null;

protected ApplicationContext getCtx() {

return ctx;

}

/\*\*

\* 私有构造函数

\*/

private SpringBeanFactory()

{

}

/\*\*

\* 获取单子实例

\* @return

\*/

public static SpringBeanFactory getInstance()

{

return instance;

}

/\*\*

\* SPRING是否初始化完成

\* @return

\*/

public static boolean isContextReady(){

return getInstance().getCtx()!=null;

}

/\*\*

\* 获取指定ID的Bean实例

\* @param id Spring Bean的唯一ID串

\* @return Bean实例

\*/

@SuppressWarnings("unchecked")

public static <T> T getSpringBean(String id)

{

return (T) getInstance().getBean(id);

}

/\*\*

\* 获取指定ID和类型的Bean实例列表

\* @param type Spring Bean的唯一ID串

\* @return Bean实例

\*/

public static <T> List<T> getSpringBeans(Class<T> type)

{

return getInstance().getBean(type);

}

/\*\*

\* 初始化ApplicationContext

\* @param ctx 上下文实例

\*/

public synchronized static void setApplicationContext(ApplicationContext ctx)

{

getInstance().setContext(ctx);

}

/\*\*

\* 初始化ApplicationContext

\* @param ctx 上下文实例

\*/

private void setContext(ApplicationContext ctx)

{

this.ctx = ctx;

}

/\*\*

\* 从ApplicationContext中获取Bean

\* @param id Spring Bean的唯一ID串

\* @return Bean实例

\*/

private Object getBean(String id)

{

return ctx.getBean(id);

}

/\*\*

\* 从ApplicationContext中获取Bean

\* @param type Spring Bean的唯一ID串

\* @return Bean实例

\*/

private <T> List<T> getBean(Class<T> type)

{

String[] beanNames = ctx.getBeanNamesForType(type);

if (beanNames == null)

{

return new ArrayList<T>();

}

List<T> beans = new ArrayList<T>();

for (String beanName : beanNames)

{

beans.add(ctx.getBean(beanName, type));

}

return beans;

}

}

□