获取Spring上下文工具—ApplicationContextHelper类

作用：获取Spring容器，以访问容器中定义的其他Bean。

注意点：

1. 在工具类上使用@Component注解让spring进行管理。
2. 需要继承ApplicationContextAware接口；
3. 重写setApplicationContext(ApplicationContext applicationContext)方法
4. Spring容器在启动的时候，会调用setApplicationContext()方法将ApplicationContext对象设置进去。

*/\*\*  
 \* 获取Spring容器，以访问容器中定义的其他Bean。  
 \*  
 \* 工具类实现ApplicationContextAware接口，  
 \* 并重写setApplicationContext(ApplicationContext applicationContext)方法，  
 \* 在工具类中使用@Component注解让spring进行管理。  
 \* spring容器在启动的时候，会调用setApplicationContext()方法将ApplicationContext对象设置进去。  
 \*/*@Component(**"applicationContextHelper"**)  
**public class** ApplicationContextHelper **implements** ApplicationContextAware {  
  
 */\*\*  
 \* Spring应用上下文环境  
 \*/* **private static** ApplicationContext *applicationContext*;  
  
 */\*\*  
 \* 实现ApplicationContextAware接口的context注入函数，将其存入静态变量。  
 \*/* @Override  
 **public void** setApplicationContext(ApplicationContext context) **throws** BeansException {  
 *applicationContext* = context;  
 }  
  
 */\*\*  
 \* 通过class获取bean  
 \*  
 \** ***@param:*** *[clazz]  
 \** ***@return:*** *T  
 \** ***@author:*** *nino  
 \** ***@date:*** *2019/5/27 15:16  
 \*/* **public static** <T> T popBean(Class<T> clazz) {  
 **if** (*applicationContext* == **null**) {  
 **return null**;  
 }  
 **return** *applicationContext*.getBean(clazz);  
 }  
  
 */\*\*  
 \* 通过name,以及Clazz返回指定的Bean  
 \*  
 \** ***@param:*** *[name, clazz]  
 \** ***@return:*** *T  
 \** ***@author:*** *nino  
 \** ***@date:*** *2019/5/27 15:20  
 \*/* **public static** <T> T popBean(String name, Class<T> clazz) {  
 **if** (*applicationContext* == **null**) {  
 **return null**;  
 }  
 **return** *applicationContext*.getBean(name, clazz);  
 }  
}

将这个类交给Spring管理：（需要首先被加载，设置懒加载的属性为false）



测试：

SysAclModuleMapper moduleMapper = ApplicationContextHelper.*popBean*(SysAclModuleMapper.**class**);  
SysAclModule module = moduleMapper.selectByPrimaryKey(12);  
***log***.info(JsonMapper.*obj2String*(module));