https://www.cnblogs.com/qingruihappy/p/7861623.html

[**@postconstruct初始化的操作**](https://www.cnblogs.com/qingruihappy/p/7861623.html)

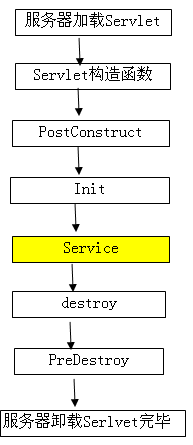
从Java EE 5规范开始，Servlet中增加了两个影响Servlet生命周期的注解（Annotion）；@PostConstruct和@PreDestroy。这两个注解被用来修饰一个非静态的void()方法 。写法有如下两种方式：

@PostConstruct

Public void someMethod() {}  
                                                                                      
或者

public @PostConstruct void someMethod(){}

    被@PostConstruct修饰的方法会在服务器加载Servle的时候运行，并且只会被服务器执行一次。PostConstruct在构造函数之后执行,init()方法之前执行。PreDestroy（）方法在destroy()方法执行执行之后执行



被注解的Servlet生命周期

    需要注意的是，注解会多多少少地影响到服务器的启动速度。服务器在启动时候会遍历Web 应用的WEB-INF/classes下的所有class文件与WEB-INF/lib下的所有jar文件,以检查哪些类使用了注解。如果应用程序中没有 使用任何注解，可以在Web.xml中设置的metadata-complete属性为true.(支持@PostConstruct和 @PreDestroy的服务器需要支持Servlet2.5规范。Tomcat5.x仅支持Servlet2.4规范。)

我现在要说的是用实例说明它有什么作用。

比如说我有一种情况，在我的servlet初始化加载之前我想处理一些东西，像加载缓存等等。

怎么做。@PostConstruct就派上用场了。那为什么这玩意用的不多呢，这是因为如果初始化之前我们要加载或处理某些玩意完全可以在构造器初始化时就处理了，但这种方法需要自己重写构造器。好吧。直接上代码看看具体用它的时候怎么做的。

[复制代码](javascript:void(0);)

1 ackage com.whaty.products.whatysns.web.info;

2

3 import javax.annotation.PostConstruct;

4 import javax.annotation.Resource;

5

6 import org.springframework.stereotype.Service;

7 import org.springframework.util.Assert;

8

9 import com.whaty.framework.cache.core.model.Cache;

10 import com.whaty.framework.cache.core.service.CacheService;

11 import com.whaty.framework.cache.entitycache.service.EntityCacheHelper;

12 import com.whaty.framework.cache.entitycache.service.IEntityDaoAdapter;

13

14 /\*\*

15 \* @author bc\_qi

16 \* @param <KEY>

17 \* @param <ENTITY>

18 \*/

19 @Service("AjaxCacheableService")

20 public class AjaxCacheableService{

21

22 @Resource(name="cacheService")

23 protected CacheService cacheService;

24

25 protected boolean useReadWriteEntityDao = false;

26 protected boolean useCache = true;

27 protected int entityCacheMaxSize = 1000;

28 protected int entityCacheMaxLiveSeconds = 3600;

29 protected Cache entityCache;

30

31

32 /\*\*

33 \* 构造方法执行后，初始化，

34 \*/

35 @PostConstruct

36 public void init() {

37 Assert.notNull(cacheService, "cacheService must be set!");

38 getCache();

39 }

40

41 /\*\*

42 \* 获取cache

43 \* @return

44 \*/

45 protected Cache getCache() {

46 if (entityCache == null) {

47 entityCache = cacheService.addCache(this.getClass().getName(),entityCacheMaxLiveSeconds);

48 }

49 return entityCache;

50 }

51

52 /\*\*

53 \* @param id

54 \* @param useCache 是否使用Cache

55 \* @return

56 \*/

57 public Object getCache(String id) {

58 String strid = String.valueOf(id);

59 Object o = entityCache.get(strid);

60 return o;

61 }

62

63 public Object putCache(int tTLSeconds,String cacheId,Object value) {

64 String strid = String.valueOf(cacheId);

65 Object o = entityCache.get(strid);

66 if (o != null) {

67 return o;

68 } else {

69 entityCache.put(strid, value, tTLSeconds);

70 return value;

71 }

72 }

73

74 }

[复制代码](javascript:void(0);)