# Mybean API文档

Mybean是一个仿Spring的轻量级框架，有IOC（控制反转）和DI（依赖注入）两个功能，目前支持properties配置文件配置开发和注解开发两种方式。

**1、Mybean启动入口**

Mybean的启动入口的一个名为Application的接口，真正实现功能的是它的实现类PropertiesApplication。

* 1. **PropertiesApplication**

PropertiesApplication有两个构造参数：

|  |
| --- |
| PropertiesApplication() |
| PropertiesApplication(String fileName) |

一个为无参构造，那么Mybean框架会自动加载默认的配置文件名，Mybean默认的配置文件名为：application.properties。

另一为有参构造，传入一个文件名字，那么就会加载自定义的配置文件名字（仅局限于properties后缀的文件）。

* 1. **GetBean**

GetBean的Application中的一个方法，在PropertiesApplication中有具体显现，这个方法同样有两个重载方法。

|  |
| --- |
| getBean(String naem) |
| getBean(String name,Class<T> objClass) |

一个参数的方法传入Bean的id名，就可以从Mybean的核心容器中获取与该id名对应的实例。返回一个Object类型的对象。

两个参数的方法与单参数方法类似，不过多了一个字节码对象参数，传入对应字节码对象，就可以返回对应的类型实例。

【注意】单参数构造方法虽然返回的是Object类型的对象，但是如果使用对应的类去接口该实例，那么Mybean会自动帮你转换类型，无须手动强转。

**2、application.properties**

application.properties的Mybean的默认配置文件，配置方式为可key-value形式。

**2.1、bean管理**

bean管理是IOC功能的核心，只有将Java类交给Mybean管理，才能够实现统一的控制反转。

将bena交给Mybean管理的方法有两种，其中一种为配置文件方式，在配置文件中配置key和value。其中key可以自己拟定，value为需要被Mybean管理的类的全限定类名。

例如：

|  |
| --- |
| userDao=com.hrp.dao.UserDao |

以上配置的意思是：将com.hrp.dao下的UserDao类交给Mybean管理，并给予唯一id为userDao。

【注意】mybean框架在1.1.0版本后，开始支持面向接口编程的bean管理。

**2.2、包扫描**

除此之外，application.properties中有一个特殊的键值对，包扫描。

|  |
| --- |
| package.scan=com.hrp |

以上配置的意思是说，递归扫描com.hrp下的所有包，并将这些包下的被注解的类，并将这些类放入核心容器中。

【注意】这个键值对中key是固定的值，value可以由开发者自己指定。

**3、注解**

Mybean中目前有两个注解，分别是@MyBean和@GetBean。

**3.1、@MyBean**

@MyBean用于注解在类上，凡是被@MyBean注解的类，在包扫描的时候，就会被管理到Mybean核心容器中。

@MyBean有一个属性为value，用于指定被管理的类的id。

* 当value属性为空是，那么代表该类的id默认为该类类名首字母小写。
* 当value被指定是，那么代码该属性就是该类的id。

例如：

当value属性为空的时候：

|  |
| --- |
| package com.hrp.pojo;  import com.mybean.annotation.MyBean;  @MyBean  public class Student(){  } |

这个注解方法等同于在配置文件中配置：

|  |
| --- |
| student=com.hrp.pojo.Student |

当value属性有值的时候：

|  |
| --- |
| package com.hrp.pojo;  import com.mybean.annotation.MyBean;  @MyBean(“stu”)  public class Student(){  } |

这个注解方式等同于在配置文件中配置：

|  |
| --- |
| stu=com.hrp.pojo.Student |

**3.2、@GetBean**

@GetBean用于注解在类成员属性上，凡是被@GetBean注解的属性，如果在核心容器中有对应的类型，那么就会自动注入该属性的实例。

【注意】

* @GetBean注解的属性所在的那个类应该被Mybean管理到，否则该注解不生效。
* @GetBean注解的属性的类型，应该被Mybean管理到，否则无法注入。

@GetBean有一个属性value，用于指定获取的实例的id。

* 当value属性为空时，那么@GetBean注解会自动按类型注入实例。
* 当value属性有值时，那么@GetBean注解会按照id来注入实例。

**3.3、@PackageScan**

@PackageScan注解作用于类上，主要是配合AnnotationApplication启动使用，

在AnnotationApplication中传入被@PackageScan注解的类的字节码对象，这样就可以达到全注解开发的作用。

@PackageScan有一个属性value，用于指定包扫描的路径。

* 当value属性为空是，那么就默认扫描该注解所在的类所在的包。
* 当value属性有值的时候，那么就扫描指定的包。

**4、Web开发（功能未完善）**

web开发，新版的mybean框架新增了mvc模块，可以和前端进行无缝兼容。

**4.1、MyBeanControllerFilter**

MyBeanContextFilter是mybean框架web开发的核心过滤器，所有web开发的功能都是基于这个过滤器的，该过滤器有三大核心功能：

1、负责监听Web服务器的初始化和销毁，当Web服务器被初始化的时候，那么过滤器会自动帮我们启动框架，并初始化Application核心容器。

参数：ConfigLocation 用于指定配置文件的路径和名称。

参数：PackageScan 用于指定包扫描的路径。

例如：

|  |
| --- |
| <filter>  <filter-name>mybean</filter-name>  <filter-class>com.mybean.web.filter.MyBeanControllerFilter</filter-class>  <init-param>  <param-name>PackageScan</param-name>  <param-value>com.hrp</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>mybean</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping> |

2、前端控制器，负责前端请求的分发功能，只要前端出现请求，就会将请求进行分发到具体的类和具体的方法上。

其中该过滤器会直接放行在static包下的所有资源，所以将静态资源全部放在static包下，那么我们就可以不需要处理和静态资源相关的问题。

【注意】被前端请求访问的方法，必须是被public修饰，参数必须为HttpServletRequest和HttpServletResponse两个参数。

3、视图解析器，负责解析所有执行方法返回的字符串，会自动请求转发到与该字符串对应的static包下的视图中。