* **依赖注入的概念**
* 依赖注入，即大对象由小对象组装起来的构建过程。
* **依赖注入的3种方式**

下表列出了按时间顺序出现的各种依赖注入设置技术

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **方式** | **特征** | **其它** |
| ① | Xml配置 |  | 需要指定这些xml的路径，如在web.xml文件中写上  <!-- 设置Spring容器加载所有的配置文件的路径 -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath\*:config/spring-\*.xml</param-value>  </context-param>  于spring1阶段使用。 |
| ② | 注解配置：主要是自动注入  @AutoWired 标记成员  @Service 标记组件类  @Resource 标记组件类 | 自动注入在某种程度上使得原先需要在xml中设置的东西很多转移到了写在注解中。  此后业界就倾向于在xml文件中只写应用的基本配置，比如：数据源、资源文件路径等。而应用的业务逻辑需要的注入关系以及其它很多方面的都尽量往注解中写。 | 一般需要配置文件中指定了包扫描路径，以便容器能够正确找到  于spring2阶段出现（java1.5带来了注解） |
| ③ | Java代码配置 | 即用纯粹的java类来配置，依靠@Configuration标记指定配置类；  将依赖注入的指定写在真正的java代码中；  原先写在xml中的东西可以完全写在java代码中了。可取代xml。  比xml更简洁一些，java程序员阅读起来也不麻烦。 | 于spring3阶段出现。  于spring4阶段成熟。 |

* 关于property文件

共性：它们都可以访问property文件。

* 由于Spring4出现了③这种方式，springboot项目多采用Spring4推荐的方式了，即如下：

|  |
| --- |
| 1、尽量使用②+③方式，而不使用①了。  2、可从property文件读取数据，但增开了yml文件，利用注解@value和@PropertySource来配合。 |

* 历史上Spring3阶段的配置方式：

|  |
| --- |
| 历史上对用xml还是用注解进行IOC等配置有过争论，后形成共识  1.使用①+②的方式：  应用的基本配置用xml，比如：数据源、资源文件等；  业务开发的类之间的关系用注解表示，比如：Service中注入bean等。  2、同样可从property文件读取数据 |

* **SpringBoot**