知乎 首发于 springboot快速入门

Springboot2系列[5]--常见注解及实现原理简单理解

安安熊
计算机

在常见的配置中，我们这里主要选择下面三个进行说明

@SpringBootApplication

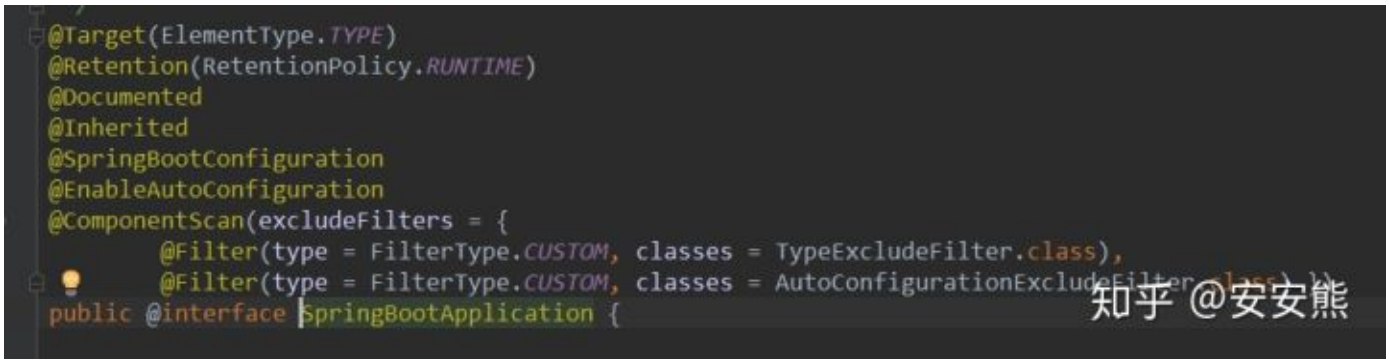
@Configuration

@EnableAutoConfiguration

@ComponentScan

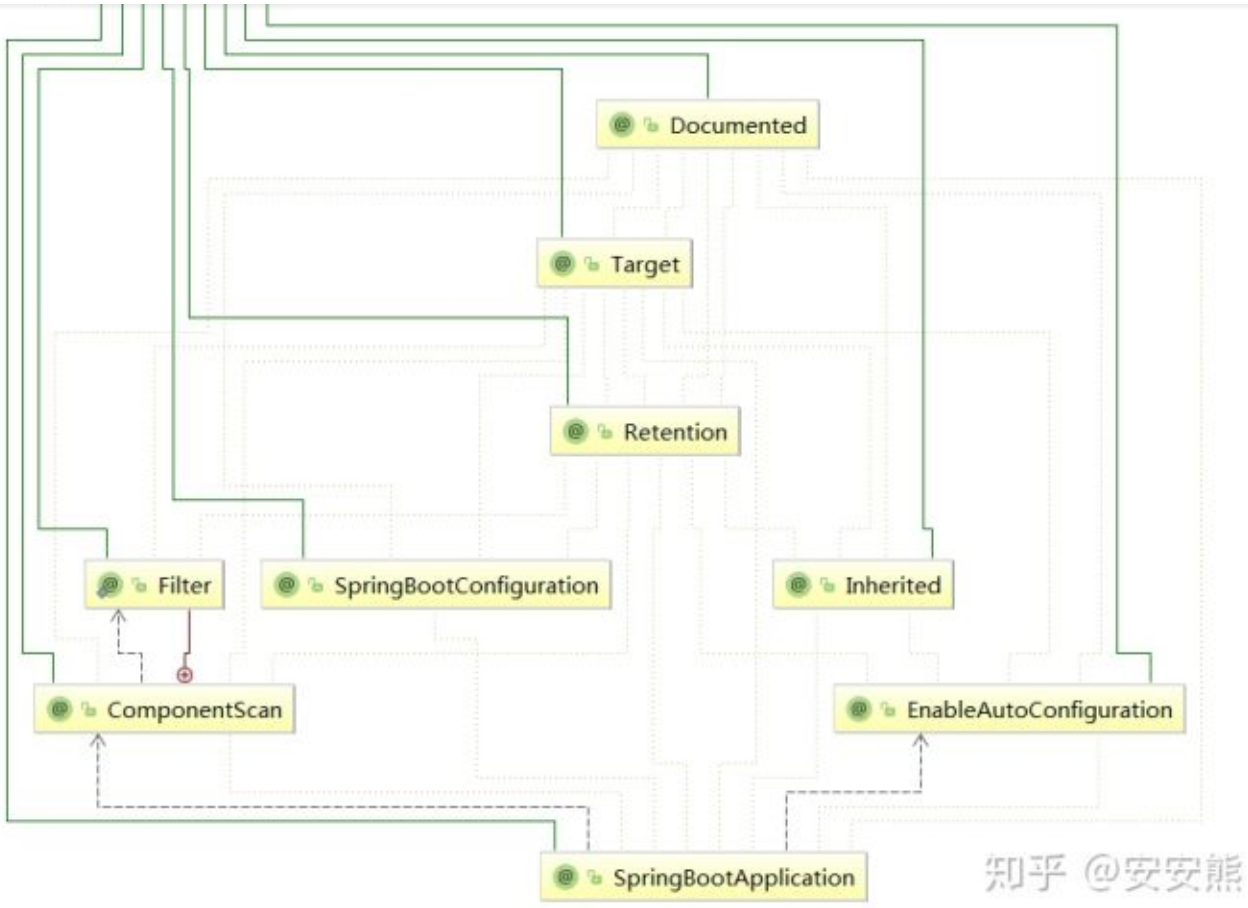
一、 @SpringBootApplication

这个注解为springboot项目的启动注解类，用在springboot项目main方法中，为入口类的注解。



可以看到这个注解引入的主要注解有：

SpringBootConfiguration，EnableAutoConfiguration 、ComponentScan 主要依赖关系如下：



1） SpringBootConfiguration

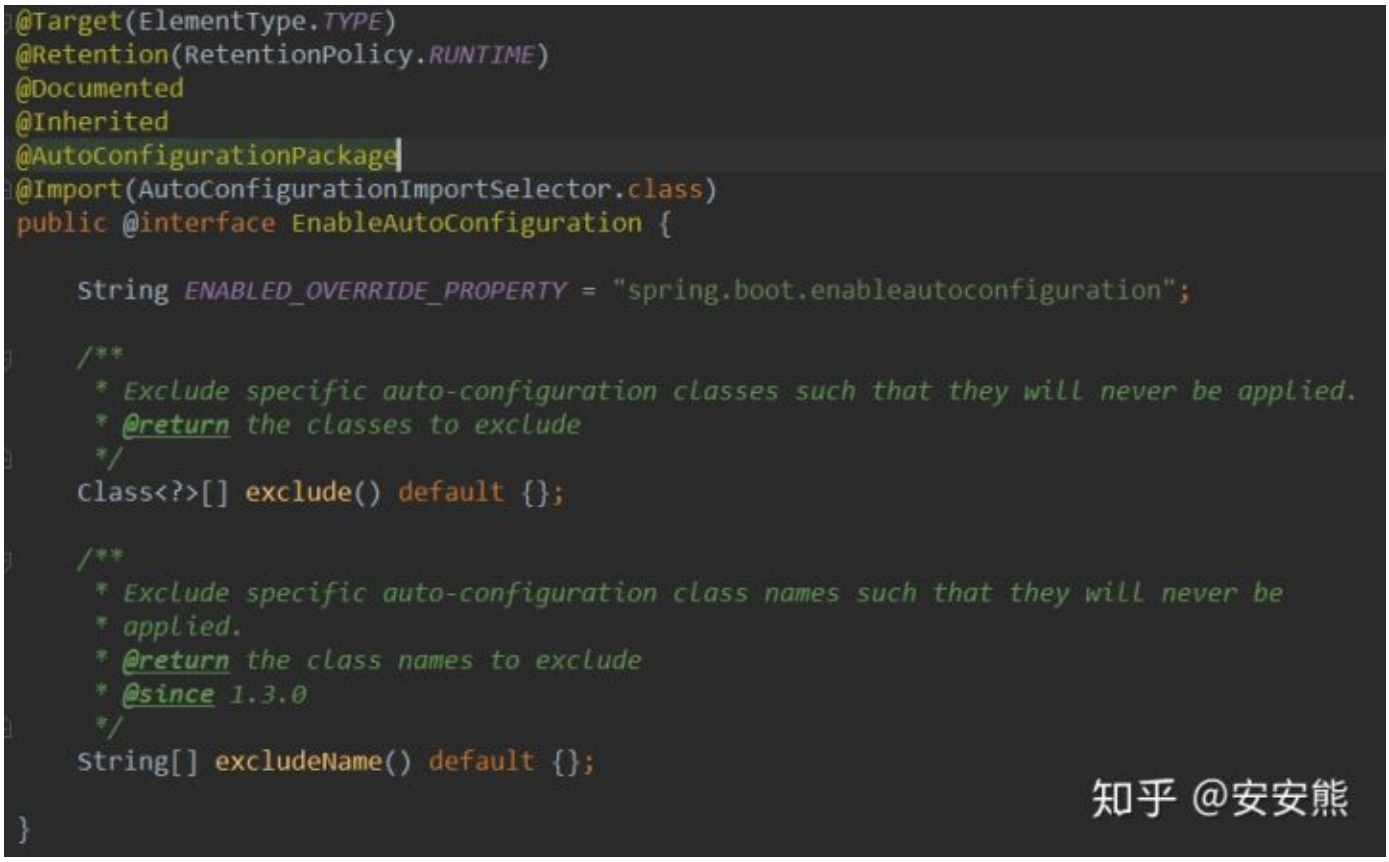
注解如下： 引入的为Configuration注解，指定需要初始化的bean，可参考：

Springboot2系列[4]-- Bean的注入

```
*/
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Configuration
public @interface SpringBootConfiguration {

}
```


2）EnableAutoConfiguration

知乎 首发于 springboot快速入门



AutoConfigurationPackage：

这个注解继承的为： @Import(AutoConfigurationPackages.Registrar.class)

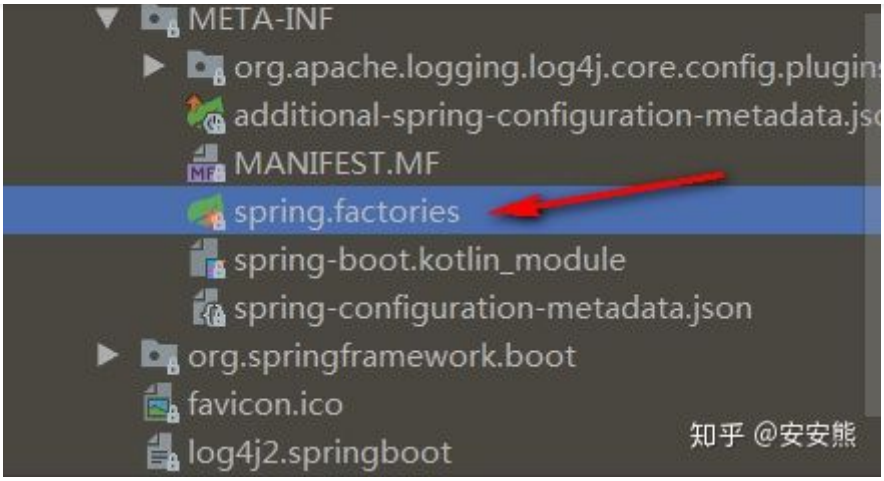
类包数据的加载

Import：

这个注解为： @Import(AutoConfigurationImportSelector.class)

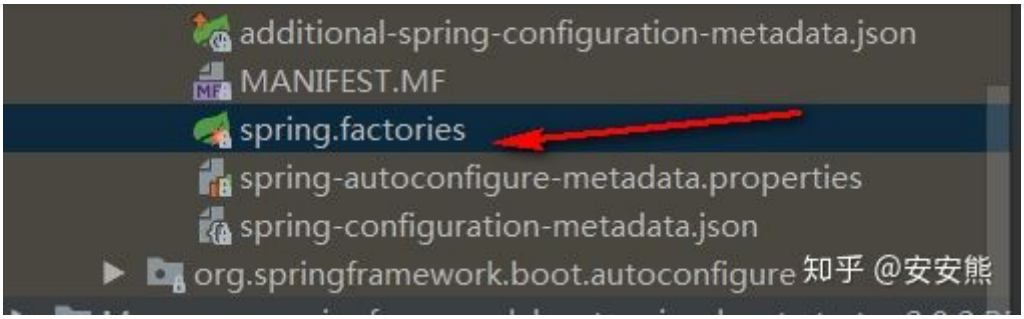
这个类核心部分为将所有符合条件的@Configuration配置都加载到当前SpringBoot创建并使用的IoC容器. 大致流程为，找到jar包下面***META-INF/spring.factories 实例化符合条件的bean，***

知乎 首发于 springboot快速入门

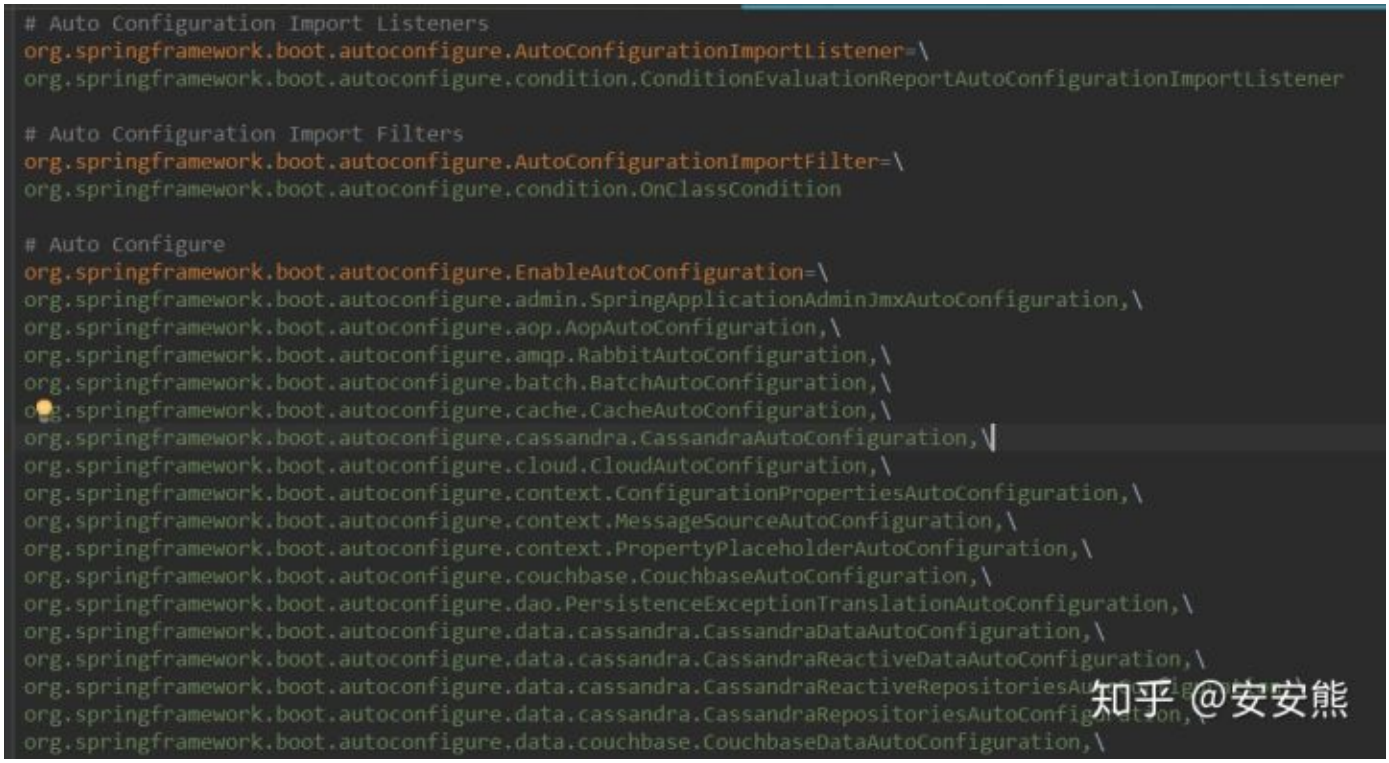


文件内容：

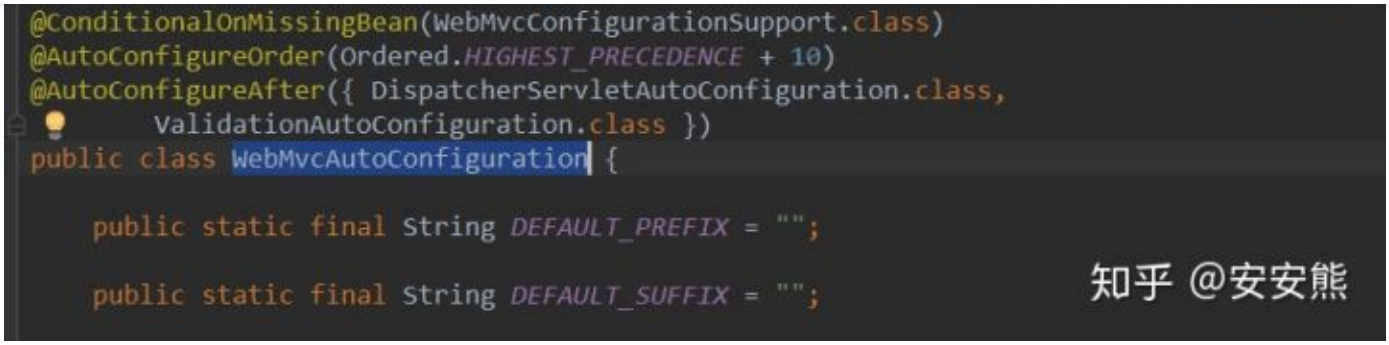
```
org.springframework.boot.SpringApplicationRunListener=\
org.springframework.boot.context.event.EventPublishingRunListener

# Error Reporters
org.springframework.boot.SpringBootExceptionReporter=\
org.springframework.boot.diagnostics.FailureAnalyzers

# Application Context Initializers
org.springframework.context.ApplicationContextInitializer=\
org.springframework.boot.context.ConfigurationWarningsApplicationContextInitialize
org.springframework.boot.context.ContextIdApplicationContextInitializer,\
org.springframework.boot.context.config.DelegatingApplicationContextInitializer,\
org.springframework.boot.web.context.ServerPortInfoApplicationContextInitializer

# Application Listeners
org.springframework.context.ApplicationListener=\
org.springframework.boot.ClearCachesApplicationListener,\
org.springframework.boot.builder.ParentContextCloserApplicationListener,\
org.springframework.boot.context.FileEncodingApplicationListener,\
org.springframework.boot.context.config.AnsiOutputApplicationListener,\
org.springframework.boot.context.config.ConfigFileApplicationListener,\
org.springframework.boot.context.config.DelegatingApplicationListener,\
org.springframework.boot.context.logging.ClasspathLoggingApplicationListener,\
org.springframework.boot.context.logging.LoggingApplicationListener,\
org.springframework.boot.liquibase.LiquibaseServiceLocatorApplicationListener

# Environment Post Processors
```



可以看到autoconfigure的包，spring.factories 内容如下:



其中配置的key为注解名称，值为对应的configruation配置类，如：WebMvcAutoConfiguration



这样借助于 @EnableAutoConfiguration 和 SpringFactoriesLoader 可以将我们需要的Bean初始化，并加载到IOC容器中，从而完成上下文的初始化工作。

二、 @Configuration

Springboot2系列[4]-- Bean的注入

三、 @ComponentScan

@ComponentScan的功能其实就是自动扫描并加载符合条件的组件（比如@Component和@Repository等）或者bean定义，最终将这些bean定义加载到IoC容器中。

我们可以通过basePackages等属性来细粒度的定制@ComponentScan自动扫描的范围，如果不指定，则默认Spring框架实现会从声明@ComponentScan所在类的package进行扫描。

我们可以自定义扫描路径：

1） 指定包名进行扫描

```
@ComponentScan(basePackages = {"org.sp.test"})
```

2）可以自定义过滤，过滤掉不需要加载的类

怎么做？分两步

步骤一： 自定义过滤器类

```
 * @author:xiongyongjie
 * @date:2018年3月7日
 * @description:
 */
public class MyTypeFilter implements TypeFilter{

        @Override
        public boolean match(MetadataReader metadataReader, MetadataReaderFacto
                        throws IOException {
                if(metadataReader.getResource().getFilename().contentEquals("Co
                        return true;
                }
                return false;
        }
}
```

步骤二：配置顾虑器到 ComponentScan 注解中

```
@ComponentScan(excludeFilters={@Filter(type=FilterType.CUSTOM,value=MyTypeFilte
//@EnableAspectJAutoProxy
public class SpringbootaopApplication { 。。。}
```

上述例子过滤掉了类名为：ConditionConfigBean 的类，此类不会被初始化。

四、重点

- **常见注解的使用**
- **简单了解springboot加载启动原理（后续会有专题讲解）**

编辑于 2018-06-13

Spring Boot Spring Cloud Spring MVC

知乎 首发于 springboot快速入门

springboot快速入门
全面介绍springboot相关内容

进入专栏

推荐阅读

10分钟详解Spring全家桶7大知识点

拉勾·开悟训练营

浅析SpringIOC容器

韩数

5 分

程序

还没有评论

写下你的评论...

▲ 赞同 1 ▼ 添加评论 分享 收藏 ...