ApplicationEvent 分析

springboot web 启动时发布了 4 个 ApplicationEvent,通过 ApplicationListener 监听

现在编写一个类，实现 ApplicationListener 接口：

```
@Component
@Slf4j
public class ApplicationListenerDemo implements ApplicationListener {
    @Override
    public void onApplicationEvent(ApplicationEvent applicationEvent) {
        log.info("!!! ApplicationListener onApplicationEvent ! source={}
    ",applicationEvent.getClass());

    }
}
```

如下： spring boot 启动后 打印信息如下：

```
2019-03-26 17:53:39.173 INFO 29160 --- [ main] c.a.s.s.l.a.ApplicationListenerDemo : !!! ApplicationListener onApplicationEvent ! source=class org.springframework.context.event.ContextRefreshedEvent
2019-03-26 17:53:39.198 INFO 29160 --- [ main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2019-03-26 17:53:39.199 INFO 29160 --- [ main] c.a.s.s.l.a.ApplicationListenerDemo : !!! ApplicationListener onApplicationEvent ! source=class org.springframework.boot.web.servlet.context.ServletWebServerInitializedEvent
2019-03-26 17:53:39.201 INFO 29160 --- [ main] c.a.s.s.l.a.DemoApplication : Started DemoApplication in 2.389 seconds (JVM running for 3.188)
2019-03-26 17:53:39.202 INFO 29160 --- [ main] c.a.s.s.l.a.ApplicationListenerDemo : !!! ApplicationListener onApplicationEvent ! source=class org.springframework.boot.context.event.ApplicationStartedEvent
2019-03-26 17:53:39.204 INFO 29160 --- [ main] c.a.s.s.l.a.ApplicationListenerDemo : !!! ApplicationListener onApplicationEvent ! source=class org.springframework.boot.context.event.ApplicationReadyEvent
```

可见，spring boot 基于 web 的项目启动后 发布了 4 个 applcaiton event：
org.springframework.context.event.ContextRefreshedEvent
org.springframework.boot.web.servlet.context.ServletWebServerInitializedEvent

org.springframework.boot.context.event.ApplicationStartedEvent
org.springframework.boot.context.event.ApplicationReadyEvent

具体在哪里发布 event ,继续分析：
1，ContextRefreshedEvent
SpringApplication 中的 this.refreshContext(context);
 这里会调用： ServletWebServerApplicationContext.refresh()
  继续调用： AbstractApplicationContext.finishRefresh()
      finishRefresh 具体代码：

```
protected void finishRefresh() {
    this.clearResourceCaches();
    this.initLifecycleProcessor();
    this.getLifecycleProcessor().onRefresh();
    this.publishEvent((ApplicationEvent)(new ContextRefreshedEvent(this)));
    LiveBeansView.registerApplicationContext(this);
}
```

里面有一个 publishEvent ,在这里发布这个事件。

2，ServletWebServerInitializedEvent
  ConfigurableApplicationContext run()方法里面的 this.refreshContext(context);
     调用 AbstractApplicationContext 里面的 this.finishRefresh();
       上面的方法最终会调用 ServletWebServerApplicationContext 的 finishRefresh，
代码：

```
protected void finishRefresh() {
    super.finishRefresh();
    WebServer webServer = this.startWebServer();
    if (webServer != null) {
          this.publishEvent(new ServletWebServerInitializedEvent(webServer,
this));
     }

 }
```

3，ApplicationStartedEvent
    ConfigurableApplicationContext run() 会调用： listeners.started(context);

    继续调用： EventPublishingRunListener 中的 started

```
   public void started(ConfigurableApplicationContext context) {
        context.publishEvent(new ApplicationStartedEvent(this.application,
this.args, context));
```

```
    }
```

4，ApplicationReadyEvent

SpringApplication 中的 run 会调用： listeners.running(context);

继续调用： EventPublishingRunListener 中的 running 方法，具体为

```
    public void running(ConfigurableApplicationContext context) {
        context.publishEvent(new ApplicationReadyEvent(this.application,
this.args, context));
    }
```


阿伦读源码 allengent@163.com