### 23.1 Startup Failure

### 23.2 自定义 Banner

其实就是自定义启动时在控制台输出的那个字符画，可以自定义其他输出信息，也可以自己画一张【佛祖保佑，永无BUG】的字符画

### 23.3 Customizing SpringApplication

### 23.4 Fluent Builder API

### 23.5 各类application event(事件)及其Listeners

SpringApplication自定义了一些事件，并在应用生命周期的各个阶段进行触发。

有部分event的触发会早于ApplicationContext(应用上下文)创建之前，如果需要监听该类型event需要添加META-INF/spring.factories 文件，并添加类似内容：

org.springframework.context.ApplicationListener=com.example.project.MyListener

自定义的Listaner接口需要实现ApplicationListener<Event>接口，Event为所需监听的事件类型的泛型，如下列举的event按触发顺序从先到后排序：

1. ApplicationStartingEvent 当app启动时最先触发的事件；
2. ApplicationEnvironmentPreparedEvent 当所需的环境被定义但上下文被创建之前（？）
3. ApplicationPreparedEvent 发生在bean的定义和设置已经被读取后，且refresh发生前
4. ApplicationStartedEvent 上下文已经被刷新，但应用和command-line runner被调用前
5. ApplicationReadyEvent 在应用和command-line runner被调用之后发生，此时应用已经可以开始接受处理请求
6. ApplicationFailedEvent 看名字就知道了，应用启动失败时触发

### 23.6 web环境

当网络环境为spring mvc时会自动设定app上下文（ApplicationContext ）为AnnotationConfigServletWebServerApplicationContext ，也可通过setWebApplicationType(WebApplicationType)手动修改类型。

### 23.7 访问应用启动参数

可在bean的构造方法（其他方法待验证）里通过autowired注解，在方法参数注入main所获得的args，如

*@Autowired*

**public** MyBean(ApplicationArguments args) {

**boolean** debug = args.containsOption("debug");

List<String> files = args.getNonOptionArgs();

*// if run with "--debug logfile.txt" debug=true, files=["logfile.txt"]*

}

### 23.8 如何使用两个runner接口

提供了以下两个接口，在SpringApplication.run完成前调用其run方法

ApplicationRunner 以ApplicationArguments方式访问app 的args

CommandLineRunner 接口以字符串数组方式访问app 的args

实现以上接口后，可根据需要编写其他处理逻辑