1 问题来源

- Spring 一个分层的 JavaSE/EE 一栈式轻量级开源框架
- Springboot 设计目的是用来简化新 Spring 应用的初始搭建以及开发过程
- 熟悉 `Springboot` 整体的启动流程，可以快速定位项目出错的原因，并能够更好的对项目做一些扩展

2 源码分析

- Springboot 2.1.1.RELEASE 源码
- springboot 项目的入口是 main 函数

```
//这个args主要可以是传一些配置参数，如：server.port，spring.active.profile等
//一般默认为 null 即可
public static void main(String[] args) {
    SpringApplication.run(SpringBootDependsApplication.class, args);
}
//程序走到这，首先要初始化 SpringApplication 类，之后调用此实例的 run() 方法
public static ConfigurableApplicationContext run(Class<?>[] primarySources, String[]
args) {
    return (new SpringApplication(primarySources)).run(args);
}
```

2.1 初始化 SpringApplication 实例时，设置的一些属性

- 初始化 SpringApplication 实例时，会设置一些默认的属性
- 获取并创建 ApplicationContextInitializer
- 获取并创建 ApplicationListener

```
public SpringApplication(ResourceLoader resourceLoader, Class... primarySources) {
    this.resourceLoader = resourceLoader;//这里 resourceLoader 为 null
    this.primarySources = new LinkedHashSet<>
(Arrays.asList(primarySources));//primarySources一般就是main对应的类
    this.webApplicationType = WebApplicationType.deduceFromClasspath();//获取应用的类型，
一般就是SERVLET类型
    //从项目中的所有的spring.factories文件中，获取并创建 ApplicationContextInitializer
    setInitializers((Collection)
getSpringFactoriesInstances(ApplicationContextInitializer.class));
    //从项目中的所有的spring.factories文件中，获取并创建 ApplicationListener
    setListeners((Collection) getSpringFactoriesInstances(ApplicationListener.class));
    this.mainApplicationClass = deduceMainApplicationClass();//main对应的类
}
```

2.2 调用 SpringApplication#run 方法，正式启动

- 重要：查找并设置配置文件信息（包括解析 `commandLines` ）----------
- 打印 `banner`
- 创建 `Spring容器`
- 调用所有初始化类的 `initialize` 方法，准备上下文资源
- 重要：解析配置文件，加载业务 `bean` ，启动 `tomcat` 等----------
- 调用 `ApplicationRunner` 和 `CommandLineRunner` 的运行方法

```
public ConfigurableApplicationContext run(String... args) {

    ConfigurableApplicationContext context = null;
    Collection<SpringBootExceptionReporter> exceptionReporters = new ArrayList<>();
    configureHeadlessProperty();//设置使用Headless，对于只有远程登录使用的服务器来说这样性能要好
一些
    //从项目中的所有的spring.factories文件中，获取并创建 SpringApplicationRunListener
    SpringApplicationRunListeners listeners = getRunListeners(args);
    listeners.starting();//通知所有监听器启动
    try {
        ApplicationArguments applicationArguments = new DefaultApplicationArguments(
                args);
        //1 重要：查找并设置配置文件信息（包括解析commandLines）----------
        ConfigurableEnvironment environment = prepareEnvironment(listeners,
                applicationArguments);
        //打印banner，可以自定义banner.txt
        Banner printedBanner = printBanner(environment);
        //根据应用的类型，创建Spring容器
        context = createApplicationContext();
        //从Spring.factories中查找这个类，用于异常路径的报错
        exceptionReporters = getSpringFactoriesInstances(
                SpringBootExceptionReporter.class,
                new Class[] { ConfigurableApplicationContext.class }, context);
        //主要是调用所有初始化类的initialize方法，准备上下文资源
        prepareContext(context, environment, listeners, applicationArguments,
                printedBanner);
        //2 重要：解析配置文件，加载业务bean，启动tomcat等----------
        refreshContext(context);
```

```
            afterRefresh(context, applicationArguments);// 空实现

            stopWatch.stop();
            if (this.logStartupInfo) {
                new StartupInfoLogger(this.mainApplicationClass)
                        .logStarted(getApplicationLog(), stopWatch);
            }
            listeners.started(context);//通知监听者 Spring 容器启动完成
            //调用 ApplicationRunner 和 CommandLineRunner 的运行方法
            callRunners(context, applicationArguments);
        }
        catch (Throwable ex) {
            handleRunFailure(context, ex, exceptionReporters, listeners);
            throw new IllegalStateException(ex);
        }

        try {
            listeners.running(context);//通知监听者，应用在运行
        }
        catch (Throwable ex) {
            handleRunFailure(context, ex, exceptionReporters, null);
            throw new IllegalStateException(ex);
        }
        return context;
    }
```

3 相关链接

Springboot 源码分析—— prepareEnvironment() 解析 Springboot 源码分析—— refreshContext() 解析
Springboot 源码分析—— 原始笔记(总)