

[首页](#) [资讯](#) [精华](#) [论坛](#) [问答](#) [博客](#) [专栏](#) [群组](#) [更多 ▼](#)

[您还未登录！](#) [登录](#) [注册](#)

Fish Where The Fish Are

- [博客](#)
- [微博](#)
- [相册](#)
- [收藏](#)
- [留言](#)
- [关于我](#)


Spring Boot 入门 - 进阶篇 (6) - 启动加载(CommandLineRunner)

博客分类：


- [SpringBoot](#)

启动成功后可以通过以下方法运行自己的初始代码：

- @PostConstruct注解
- ApplicationReadyEvent事件
- CommandLineRunner/ApplicationRunner接口

Java代码 

```
1. @Component
2. public class StartUpInit {
3.
4.     @Autowired
5.     private SomeService service;
6.
7.     @PostConstruct
8.     public void init(){
9.         // ...
10.    }
11.
12. }
```

Java代码 

```
1. @Component
2. public class GeneralEventHandler {
3.
4.     @EventListener
5.     public void handleApplicationReady(ApplicationReadyEvent event) {
6.         log.info("The application is ready to service requests..");
7.     }
8.
9. }
```

Spring Boot提供了两个接口：CommandLineRunner、ApplicationRunner，用于启动应用时做特殊处理，这些代

码会在SpringApplication的run()方法运行完成之前被执行。

通常用于应用启动前的特殊代码执行、特殊数据加载、垃圾数据清理、微服务的服务发现注册、系统启动成功后的通知等。相当于Spring的ApplicationListener、Servlet的ServletContextListener。

CommandLineRunner 和 ApplicationRunner 的区别是run()方法的参数不同。

(1) CommandLineRunner : 参数是字符串数组

Java代码 ✨ ☆

```
1. @Component
2. public class CommandLineAppStartupRunner implements CommandLineRunner {
3.     private static final Logger logger = LoggerFactory.getLogger(CommandLineAppStartupRunner.class);
4.
5.     @Override
6.     public void run(String... args) throws Exception {
7.         logger.info("Application started with command-
           line arguments: {} . \n To kill this application, press Ctrl + C.", Arrays.toString(args));
8.     }
9. }
```

(2) ApplicationRunner : 参数被放入ApplicationArguments

通过getOptionNames()、getOptionValues()、getSourceArgs()获取参数

Java代码 ✨ ☆

```
1. @Component
2. public class AppStartupRunner implements ApplicationRunner {
3.     private static final Logger logger = LoggerFactory.getLogger(AppStartupRunner.class);
4.
5.     @Override
6.     public void run(ApplicationArguments args) throws Exception {
7.         logger.info("Your application started with option names : {}", args.getOptionNames());
8.     }
9. }
```

也可以两个接口同时实现，但是没有必要。

Java代码 ✨ ☆

```
1. @Component
2. public class StartupRunner implements CommandLineRunner, ApplicationRunner {
3.     private static final Logger logger = LoggerFactory.getLogger(CommandLineAppStartupRunner.class);
4.
5.     @Override
6.     public void run(String... args) throws Exception {
7.         logger.info("Application started with command-
           line arguments: {} . \n To kill this application, press Ctrl + C.", Arrays.toString(args));
8.     }
9.
10.    @Override
11.    public void run(ApplicationArguments args) throws Exception {
12.        logger.info("Your application started with option names : {}", args.getOptionNames());
13.    }
14. }
```

也可以通过@Bean定义

Java代码 ✨ ☆

```
1. @Configuration
2. public class RunnerConfig {
3.
4.     @Bean
5.     public CommandLineRunner runner(){
6.         return new CommandLineRunner() {
7.             public void run(String... args){
8.                 System.out.println("CommandLineRunner run()");
9.             }
10.        };
11.    }
12.
13. }
```

(3) 通过@Order设置执行顺序

Java代码 ✨ ☆

```
1. @Component
2. @Order(3)
3. public class Runner1 implements CommandLineRunner {
4.     @Override
5.     public void run(String... args) throws Exception {
6.         System.out.println("Runner1 run()");
7.     }
8. }
9.
10. @Component
11. @Order(2)
12. public class Runner2 implements CommandLineRunner {
13.     @Override
14.     public void run(String... args) throws Exception {
15.         System.out.println("Runner2 run()");
16.     }
17. }
18.
19. @Component
20. @Order(1)
21. public class Runner3 implements CommandLineRunner {
22.     @Override
23.     public void run(String... args) throws Exception {
24.         System.out.println("Runner3 run()");
25.     }
26. }
```



(4) 注入Bean

CommandLineRunner在被执行时，Spring内部已经启动完成，可以注入Spring的Bean。

Java代码 ✨ ☆

```
1. @Component
2. public class StartupRunner implements CommandLineRunner {
3.
4.     @Autowired
```

```
5. private SampleService sampleService;
6.
7. @Override
8. public void run(String... args) throws Exception {
9.     sampleService.executeSample();
10. }
11.
12. }
```

分享到:  

[Spring Boot 入门 - 进阶篇 \(7\) - 自动配 ...](#) | [Spring Boot 入门 - 进阶篇 \(5\) - 数据缓 ...](#)

- 2017-03-15 15:04
- 浏览 710
- [评论\(0\)](#)
- 分类: [开源软件](#)
- [查看更多](#)

相关资源推荐

Spring Boot 启动加载数据 CommandLineRunner	Spring Boot 入门 - 进阶篇 (7) - 自动配置(AutoCo...
【Android进阶篇】Fragment的两种加载方式	Windows Azure使用入门 第六课：运行开源软件与...
Spring boot 热加载 springloaded-1.2.4.RELEASE J...	SymmetricDS 数据库双向同步开源软件入门
springloaded spring-boot 热加载	spring-boot笔记-日志记录、启动加载、定时任务 (...
Spring Boot (教程九：启动加载)	Protege新手入门-进阶篇
Linux-HA开源软件Heartbeat (配置篇)	Launchy-超级好用的管理管理启动程序-开源软件
金山卫士开源软件之旅(八) netmon下netmon工程的...	金山卫士开源软件之旅(八) netmon下netmon工程的...
Protege新手入门(进阶篇)(1).doc	Protege新手入门教程-进阶篇
Protege新手入门-进阶篇配套工程Animal3.1.1.7	Linux-HA开源软件Heartbeat (概念篇)
Linux-HA开源软件Heartbeat (测试篇)	Linux-HA开源软件Heartbeat (安装篇)

参考知识库



[Android知识库](#) 39005 关注 / 3162 收录



[React知识库](#) 3950 关注 / 393 收录



[人工智能基础知识库](#) 18251 关注 / 212 收录



[Java 知识库](#) 38149 关注 / 3748 收录

评论

发表评论



[您还没有登录,请您登录后再发表评论](#)

rensanning的博客