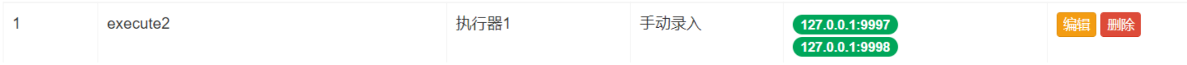
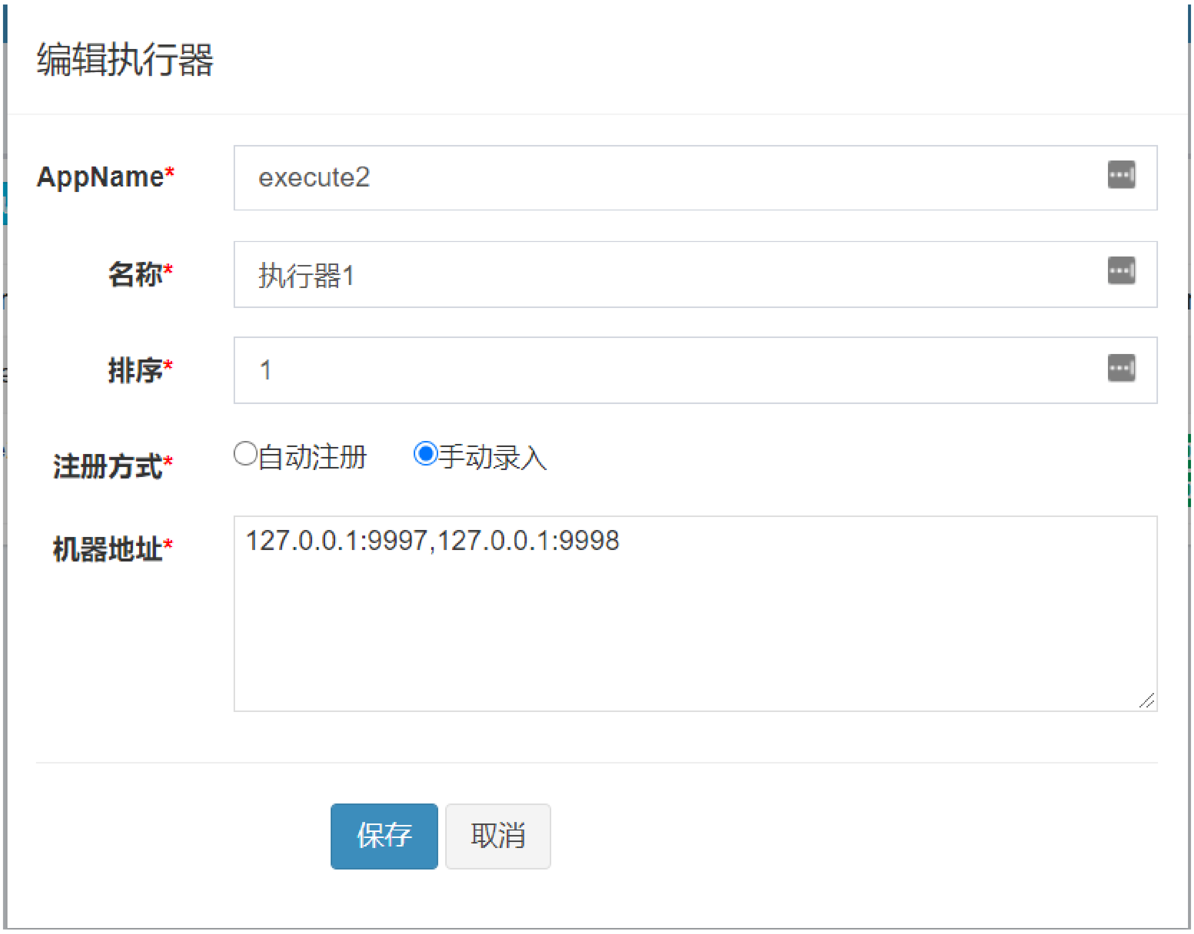
任务调度中心对接springboot项目

1.创建执行器



1	execute2	执行器1	手动录入	127.0.0.1:9997 127.0.0.1:9998	编辑 删除



编辑执行器

AppName*: execute2

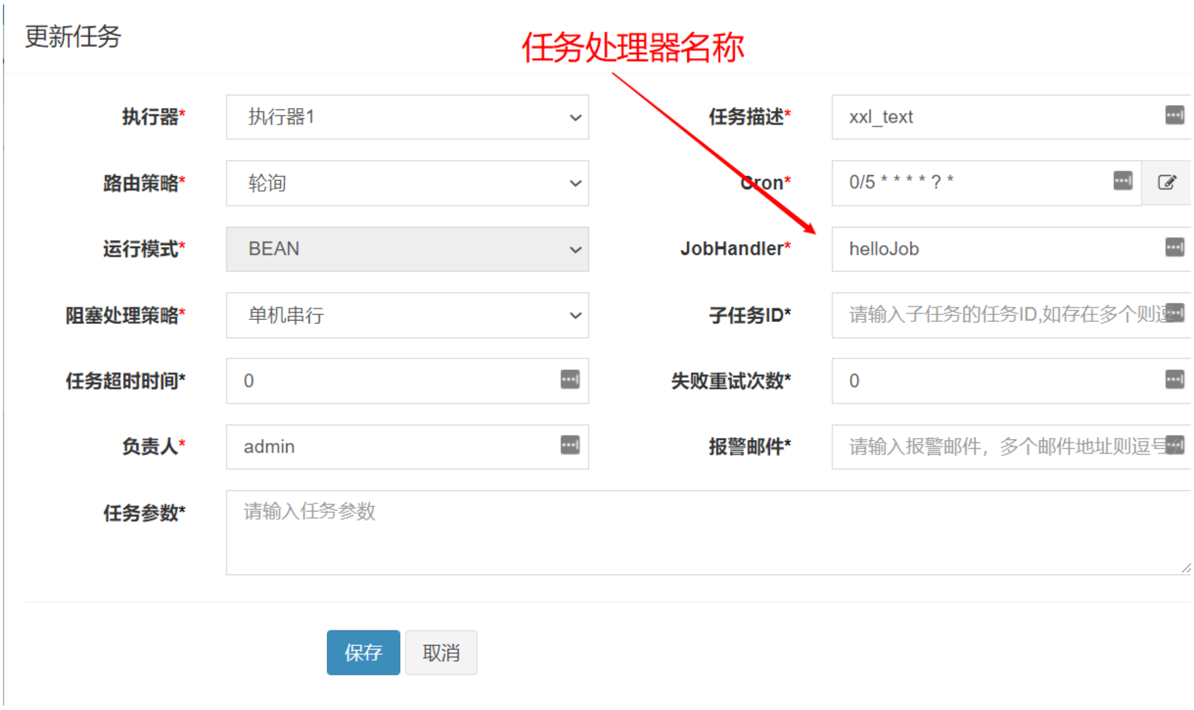
名称*: 执行器1

排序*: 1

注册方式*: ○自动注册 ●手动录入

机器地址*: 127.0.0.1:9997,127.0.0.1:9998

2.创建任务

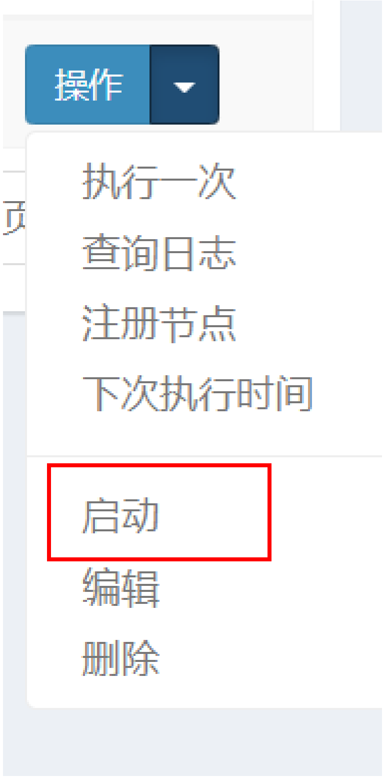


更新任务

任务处理器名称

执行器*	执行器1	任务描述*	xxl_text
路由策略*	轮询	Cron*	0/5 * * * * ? *
运行模式*	BEAN	JobHandler*	helloJob
阻塞处理策略*	单机串行	子任务ID*	请输入子任务的任务ID,如存在多个则逗
任务超时时间*	0	失败重试次数*	0
负责人*	admin	报警邮件*	请输入报警邮件，多个邮件地址则逗号
任务参数*	请输入任务参数		

3.开启当前创建的任务



4.构建springboot 项目

1.依赖

```
<dependency>
    <groupId>com.xuxueli</groupId>
    <artifactId>xxl-job-core</artifactId>
    <version>2.2.0-SNAPSHOT</version>
</dependency>
```

2.核心配置文件

```
# web port  微服务端口号
server.port=${port:8801}
# no web
#spring.main.web-environment=false
# log config  日志
logging.config=classpath:logback.xml
### xxl-job admin 服务端地址    xxl-job-admin-2.2.0-SNAPSHOT.jar
xxl.job.admin.addresses=http://localhost:8888/xxl-job-admin
### xxl-job, access token
xxl.job.accessToken=
### xxl-job executor appname  添加的执行器的名称
xxl.job.executor.appname=execute2
### xxl-job executor registry-address: default use address to registry ,
otherwise use ip:port if address is null  默认localhost
xxl.job.executor.address=
### xxl-job executor server-info  默认localhost
xxl.job.executor.ip=
xxl.job.executor.port=${executor.port:9997}
### xxl-job executor log-path
xxl.job.executor.logpath=/data/applogs/xxl-job/jobhandler
### xxl-job executor log-retention-days
xxl.job.executor.logretentiondays=30
```

3.添加config 配置类 XxlJobConfig

```
@Configuration
public class XxlJobConfig {
    private Logger logger = LoggerFactory.getLogger(XxlJobConfig.class);

    @Value("${xxl.job.admin.addresses}")
    private String adminAddresses;

    @Value("${xxl.job.accessToken}")
    private String accessToken;

    @Value("${xxl.job.executor.appname}")
    private String appName;

    @Value("${xxl.job.executor.address}")
    private String address;

    @Value("${xxl.job.executor.ip}")
    private String ip;

    @Value("${xxl.job.executor.port}")
    private int port;

    @Value("${xxl.job.executor.logpath}")
    private String logPath;

    @Value("${xxl.job.executor.logretentiondays}")
    private int logRetentionDays;


    @Bean
    public XxlJobSpringExecutor xxlJobExecutor() {
        logger.info(">>>>>>>>>>> xxl-job config init.");
        XxlJobSpringExecutor xxlJobSpringExecutor = new XxlJobSpringExecutor();
        xxlJobSpringExecutor.setAdminAddresses(adminAddresses);
        xxlJobSpringExecutor.setAppName(appName);
        xxlJobSpringExecutor.setAddress(address);
        xxlJobSpringExecutor.setIp(ip);
        xxlJobSpringExecutor.setPort(port);
        xxlJobSpringExecutor.setAccessToken(accessToken);
        xxlJobSpringExecutor.setLogPath(logPath);
        xxlJobSpringExecutor.setLogRetentionDays(logRetentionDays);

        return xxlJobSpringExecutor;
    }

    /**
     * 针对多网卡、容器内部署等情况，可借助 "spring-cloud-commons" 提供的 "InetUtils" 组件灵活定制注册IP；
     *
     *      1、引入依赖：
     *          <dependency>
     *             <groupId>org.springframework.cloud</groupId>
     *             <artifactId>spring-cloud-commons</artifactId>
     *             <version>${version}</version>
     *         </dependency>
     *
     *      2、配置文件，或者容器启动变量
     *          spring.cloud.inetutils.preferred-networks: 'xxx.xxx.xxx.'
     *
     *      3、获取IP
     *          String ip_ = inetUtils.findFirstNonLoopbackHostInfo().getIpAddress();
     */


}
```

4.编写定时任务

```
@Component
public class HelloJob {

    @Value("${server.port}")
    private String appPort;
    // helloJob  == 任务中心配置的任务名称
    @XxlJob("helloJob")
    public ReturnT<String> hello(String param) throws Exception {
        System.out.println("helloJob: "+ LocalDateTime.now()+",端口号"+appPort);
        return ReturnT.SUCCESS;
    }
}
```

5启动当前springboot 项目查看当前方法是都定时执行