## Shiro

令牌，登录，登录管理

Subject，SecurityManager，Realms。

导包：shiro-core，slf4j-simple

加载配置文件：Factory<SecurityManager> fac =

new IniSecurityManagerFactory(“classpath:shiro.ini”);

获取安全管理：SecurityManager sm = fac.getInstance();

设置安全管理：SecurityUtils.setSecurityManager(sm);

获取项目操作：Subject sub = SecurityUtils.getSubject();

获取用户令牌：UsernamePasswordToken token = new UsernamePasswordToken();

通过sub进行用户管理，方法：getSession(); isAuthenticated(); isPermited(); hasRole(); login()/logout();

## Springboot整合Shiro

核心类：ShiroFilterFactory，SecurityManager，Realm

过滤器拦截，进行登录、权限控制；

1. pom.xml中添加Shiro依赖；导包shiro-spring  
   (b) 配置类，注入Shiro Factory、SecurityManager、AuthorizingRealm实现类、凭证管理器(HashedCredentialsMatcher)；开启shiro aop注解支持；注入ShiroFactoryBean时，设置过滤器、登录url、登出url等参数。  
   (c) 实现AuthorizingRealm  
   (d) 在controller中添加@RequiresPermissions("userInfo:view")，实现权限管理。

## Quartz

Scheduler，Trigger，JobDetail/Job；反射

导包：quartz，quartz-jobs

编写Job实现类MyJob。实现Job接口，excute()方法。

编写调度类。获取scheduler，trigger，jobDetail；而后将jobDetail，trigger加入scheduler，启动。

获取scheduler：new StdSchedulerFactory().getScheduler();

获取jobDetail：

JobBuilder.newJob(MyJob.class).withDescription().withIdentity().build();

其他方法：requestRecovery(); storeDurably(); setJobData(); usingJobData();

获取calendar：SimpleSchedulerBuilder.repeatMinutelyForever();

SimpleSchedulerBuilder，CronSchedulerBuilder

获取trigger： TriggerBuilder.newTrigger().withDescription().withIdentity()

startAt().withSchedule(calendar).build();

注册任务：scheduler.scheduleJob(jobDetail,trigger);

启动/停止: scheduler.start()/shutdown();

传值给Job实例：JobDataMap，usingJobData()。

一个Job可以对应多个Trigger， 但一个Trigger只能对应一个Job。

配置文件为quartz.properties，放在classpath下。

## Springboot quartz整合

1，配置quartz.properties；配置scheduler、threadPool、jobStore、plugin等

2， 配置类，自动装配schedulerFactoryBean、properties、quartzInitializerListener、scheduler。

3，编写Job实现类。

4，controller中获取trigger，实现调度。

问题：

1，报错：The import org.springframework.scheduling.quartz cannot be resolved

导包spring-context-support。

2，找不到配置的Bean

SpringBoot默认扫描Application类所在包及其子包。

## H2内存数据库

导包h2。

<**dependency**>

<**groupId**>com.h2database</**groupId**>

<**artifactId**>h2</**artifactId**>

<**scope**>runtime</**scope**>

</**dependency**>

修改配置文件

url: jdbc:h2://D:/testDB

username: sa

password: sa

driver-class-name: org.h2.Driver

控制台：打开bin--h2.dat，选择server连接；输入地址等信息即可。

## 上传项目到GitHub

Git init

Git config --global user.name “yun6713”

Git config --global user.email “986156499@qq.com”

Git remote add origin https://github.com/yun6713/MyLearning.git

Git status

Git add -A

Git commit -m “commit to local” //提交到本地

Git push -u origin master //第一次提交 添加命令参数 -u 确保关联本地库和远程库

Git push origin master //非第一次提交使用此命令即可

Git clone url -b branchName

## maven打包、插件、pom文件标签

springboot maven打包：配置maven插件，

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<configuration>

<jvmArguments>-Dfile.encoding=UTF-8</jvmArguments>

<fork>true</fork>

</configuration>

</plugin>

maven clean package；打包好的文件位于target目录下；java -jar MyWork.jar，运行打包文件。