IDEA 插件推荐: EasyCode 一键生成所需代码

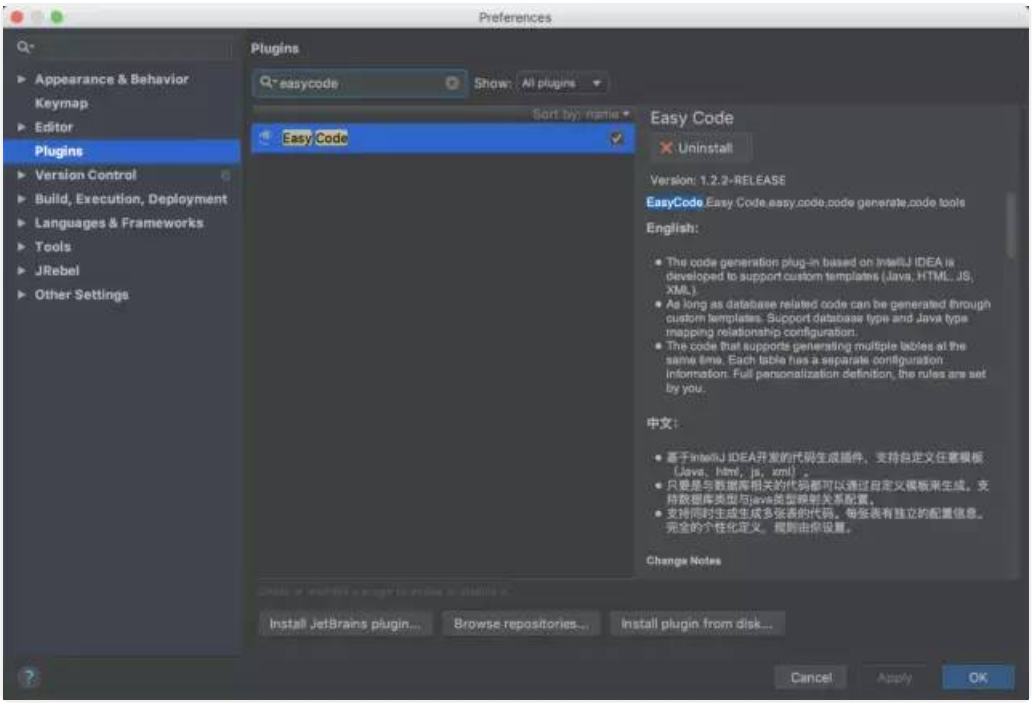
HeloWxl Java后端 2月18日

作者 | HeloWxl

来源 | http://suo.im/6v9d64

Easycode是idea的一个插件，可以直接对数据的表生成entity,controller,service,dao,mapper,无需任何编码，简单而强大。

1、安装(EasyCode)



我这里的话是已经那装好了。

- 建议大家在安装一个插件，叫做Lombok。Lombok能通过注解的方式，在编译时自动为属性生成构造器、getter/setter、equals、hashcode、toString方法。出现的神奇就是在源码中没有getter和setter方法，但是在编译生成的字节码文件中有getter和setter方法。

2、建立数据库

```
-- ----------------------------


--
Table
 structure
for
 user
```

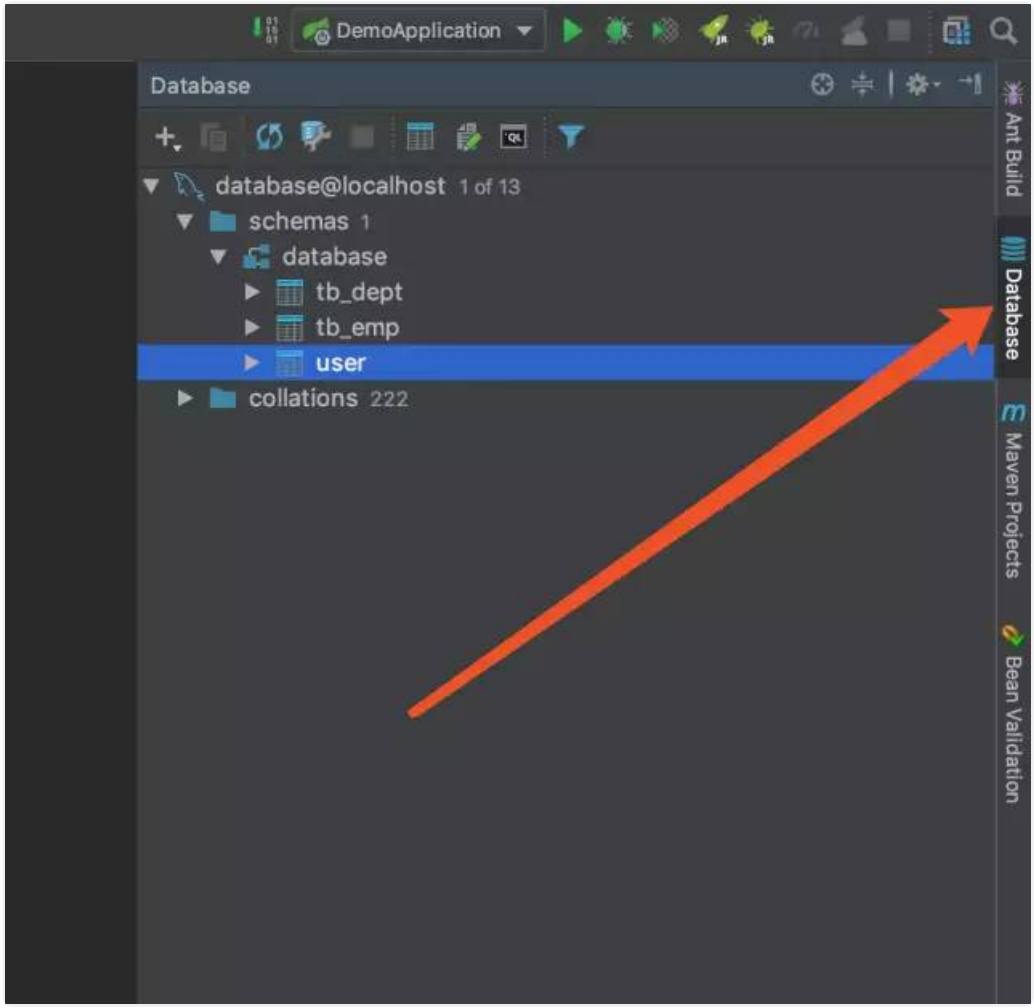
```
-- ----------------------------


DROP TABLE IF EXISTS
`user`
;


CREATE TABLE
`user`
(


`id`
int
(
11
) NOT NULL,


`username`
 varchar(
20
) DEFAULT NULL,


`sex`
 varchar(
6
) DEFAULT NULL,


`birthday`
 date DEFAULT NULL,


`address`
 varchar(
20
) DEFAULT NULL,


`password`
 varchar(
20
) DEFAULT NULL,


 PRIMARY KEY (
`id`
)


```

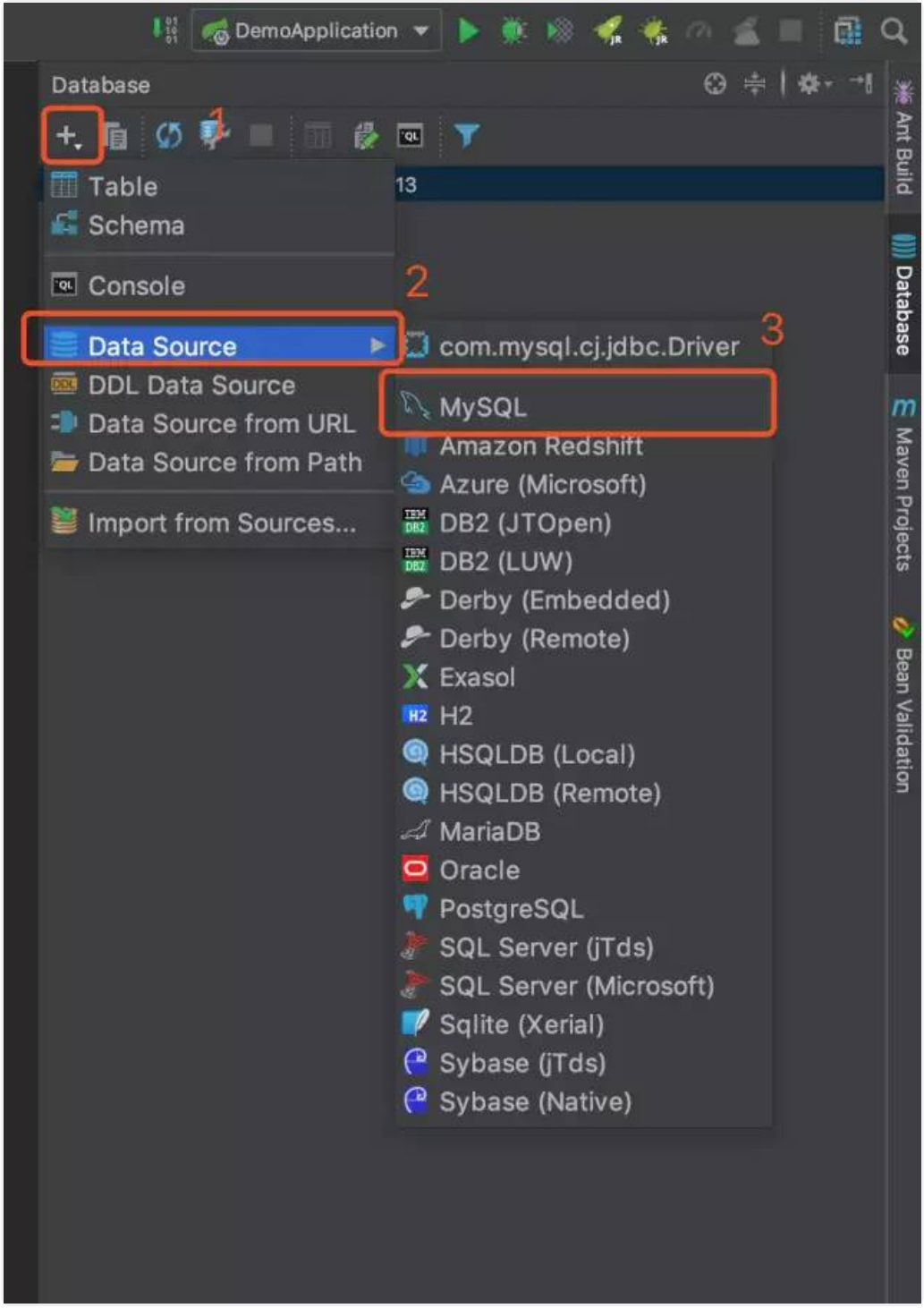
```
) ENGINE=
InnoDB
 DEFAULT CHARSET=utf8;



SET FOREIGN_KEY_CHECKS =
1
;
```

3、在IDEA配置连接数据库

在这个之前，新建一个Springboot项目，这个应该是比较简单的。

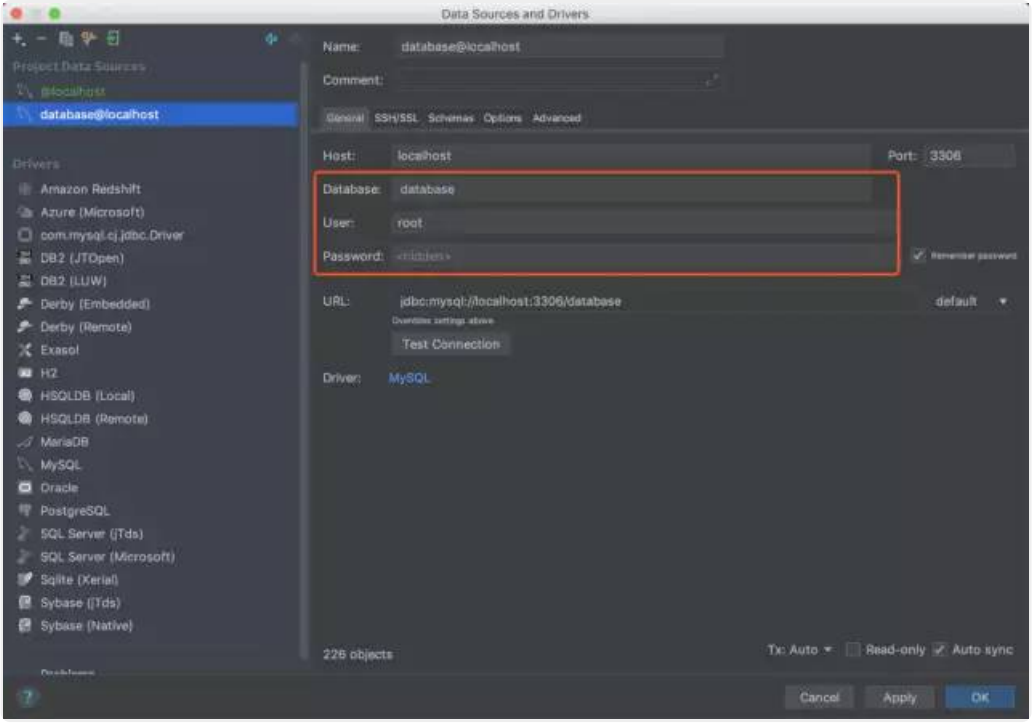
建好SpringBoot项目之后，如下图所示，找到这个Database





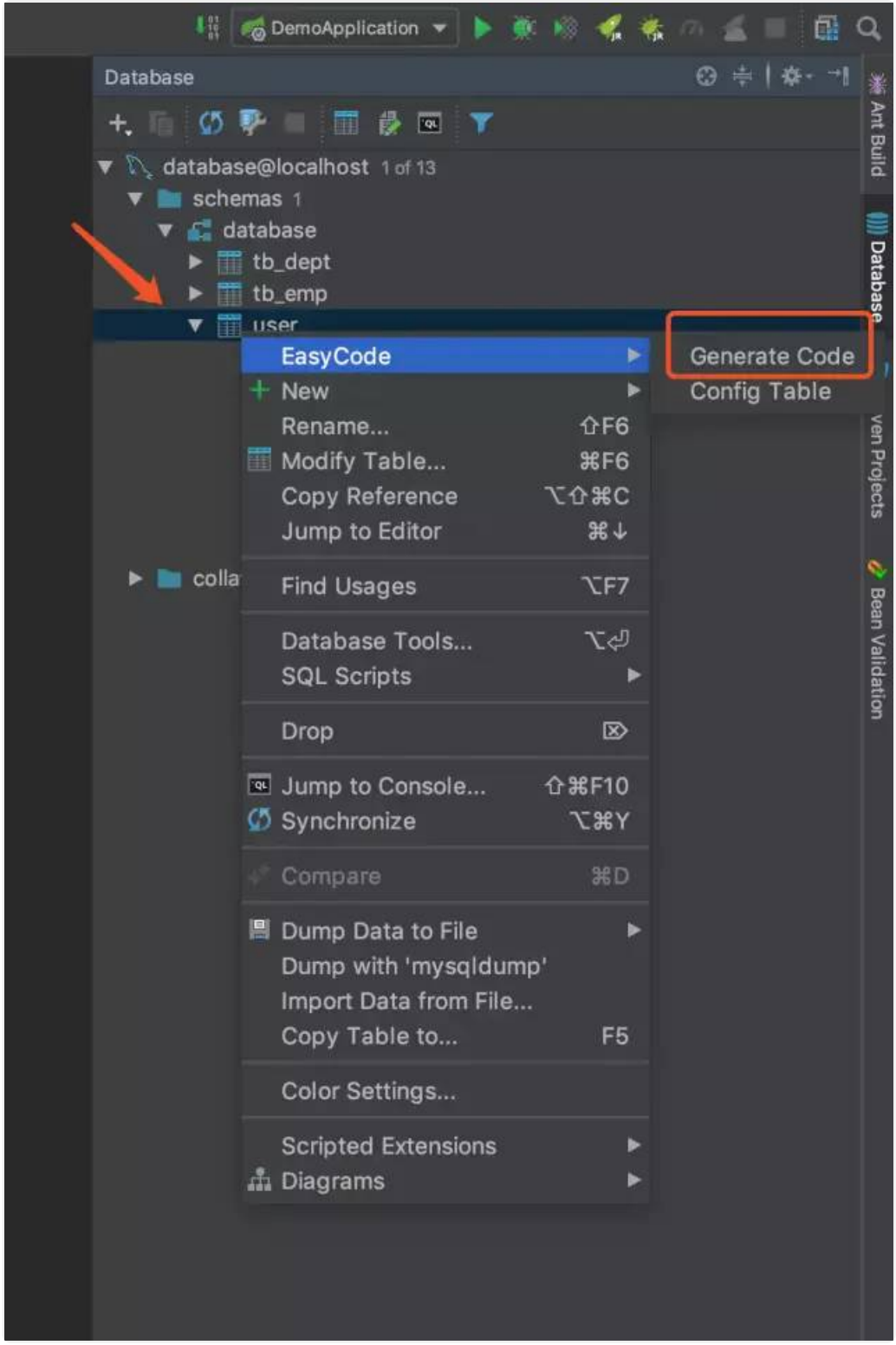
按照如下图所示进行操作：

然后填写数据库名字，用户名，密码。点击OK即可。这样的话，IDEA连接数据库就完事了。

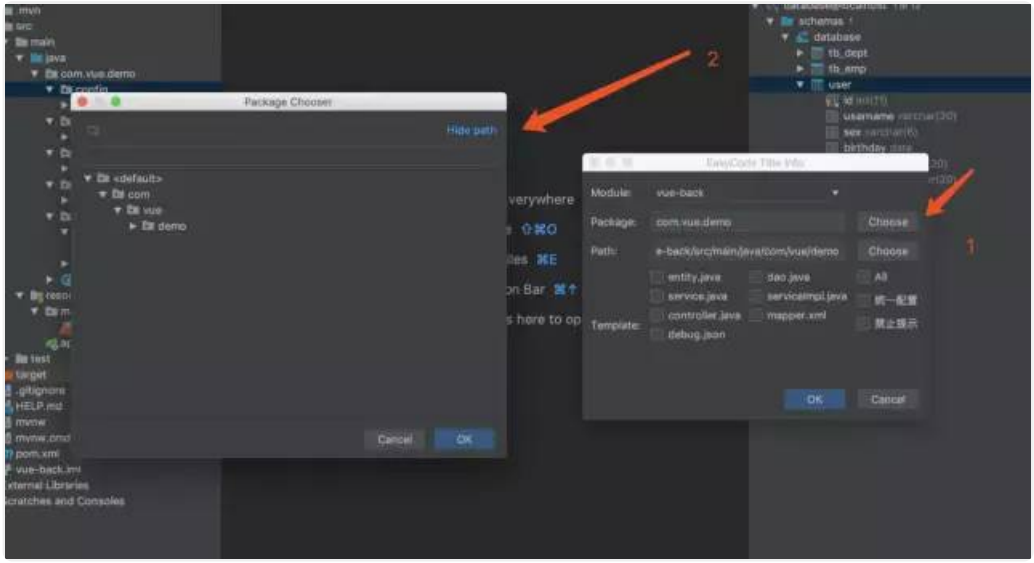


4、开始生成代码

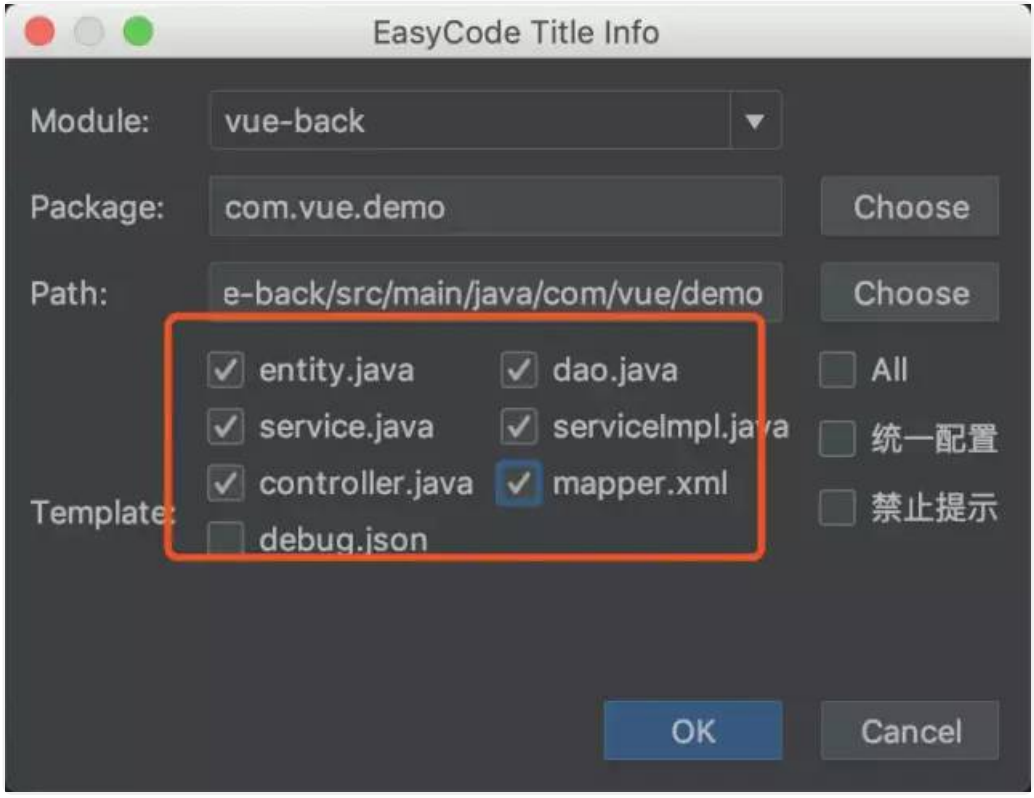
在这个里面找到你想生成的表，然后右键，就会出现如下所示的截面。



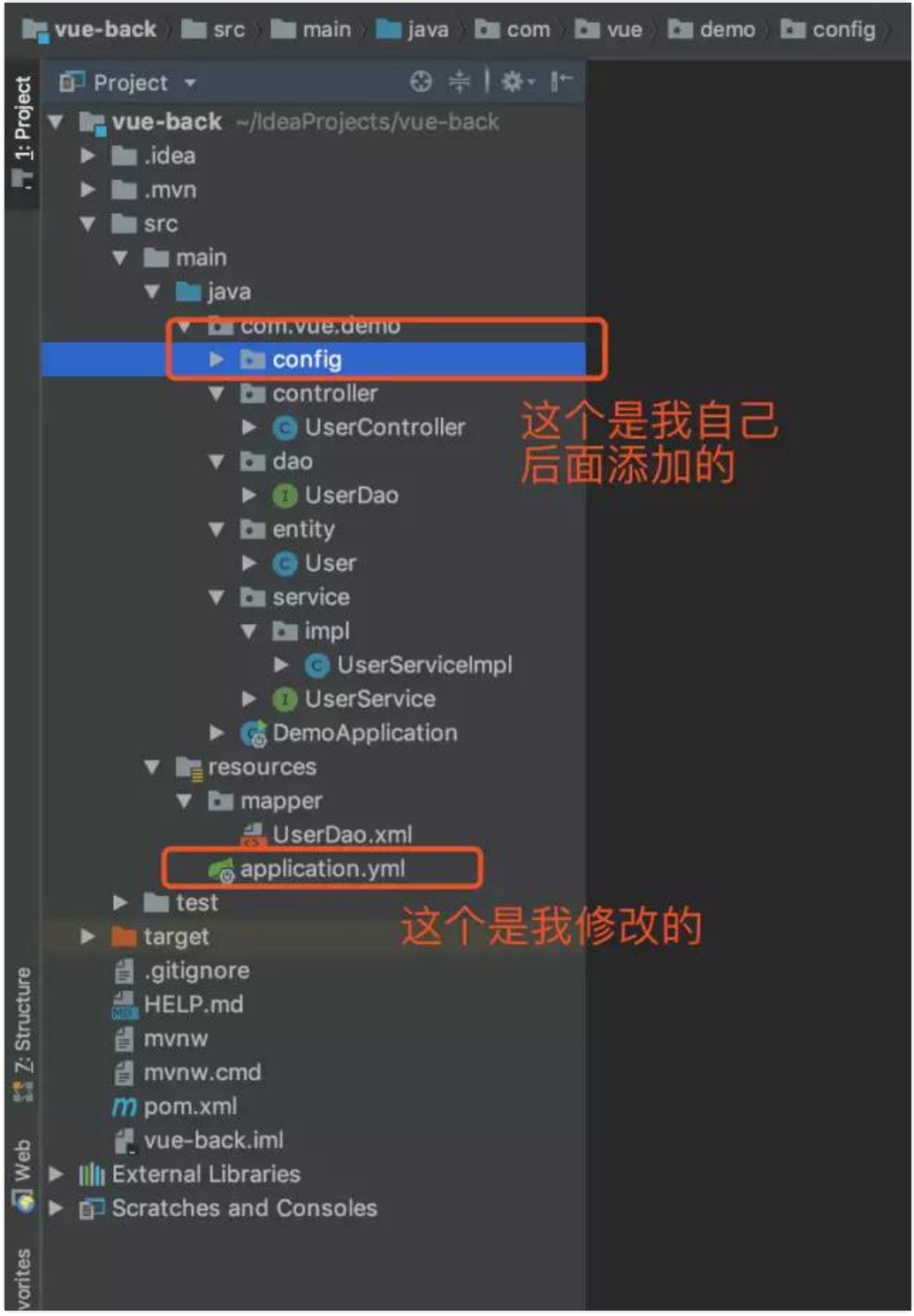
点击1所示的位置，选择你要将生成的代码放入哪个文件夹中，选择完以后点击OK即可。



勾选你需要生成的代码，点击OK。



这样的话就完成了代码的生成了，生成的代码如下图所示：



5、pom.xml

```
<dependency>


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


<artifactId>
spring-boot-starter
</artifactId>
```

```
</dependency>



<dependency>



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



<artifactId>
spring-boot-starter-web
</artifactId>



</dependency>



<dependency>



<groupId>
org.projectlombok
</groupId>



<artifactId>
lombok
</artifactId>



<optional>
true
</optional>



</dependency>



<!--热部署-->
```

```
<dependency>


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


<artifactId>
spring-boot-devtools
</artifactId>


<optional>
true
</optional>
<!-- 这个需要为 true 热部署才有效 -->


</dependency>


<!--mybatis-->


<dependency>


<groupId>
org.mybatis.spring.boot
</groupId>


<artifactId>
mybatis-spring-boot-starter
</artifactId>


<version>
1.3.2
</version>


</dependency>
```

```
<!-- mysql -->

<dependency>

<groupId>
mysql
</groupId>

<artifactId>
mysql-connector-java
</artifactId>

<version>
5.1.47
</version>

</dependency>

<!--阿里巴巴连接池-->

<dependency>

<groupId>
com.alibaba
</groupId>

<artifactId>
druid
</artifactId>

<version>
1.0.9
```

```
</version>


</dependency>
```

6、Application.yml

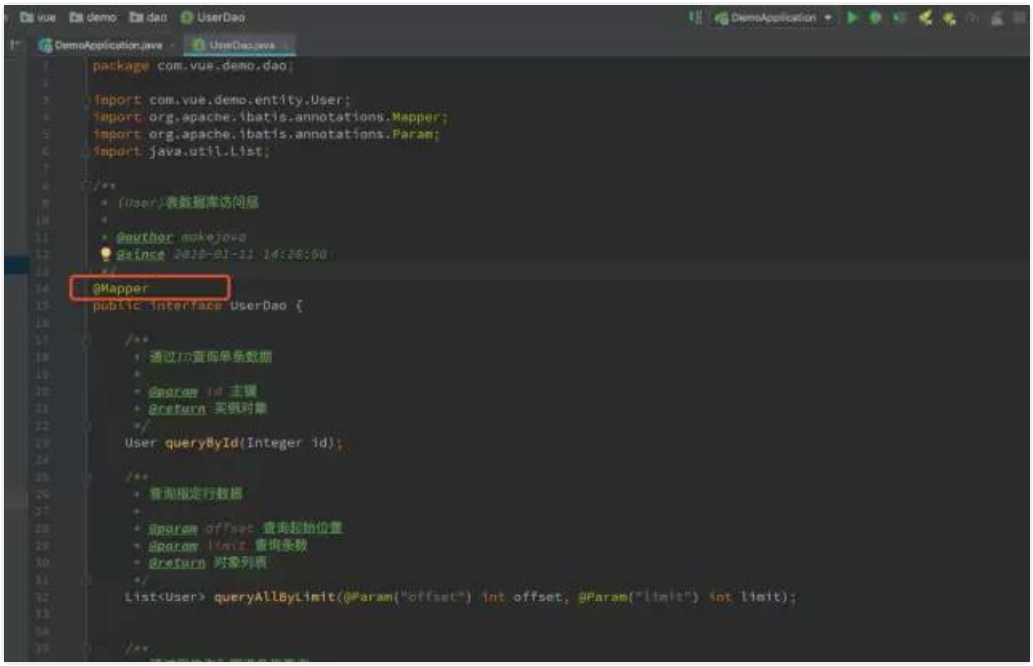
```
server:


  port:
8089


spring:


  datasource:


    url: jdbc:mysql:
//127.0.0.1:3306/database?useUnicode=true&characterEncoding=UTF-8


    username: root


    password:
123456


    type: com.alibaba.druid.pool.
DruidDataSource


    driver-
class
-name: com.mysql.jdbc.
Driver
```

```
mybatis:



  mapper-locations: classpath:
/mapper/
*
Dao
.xml



  typeAliasesPackage: com.vue.demo.entity
```

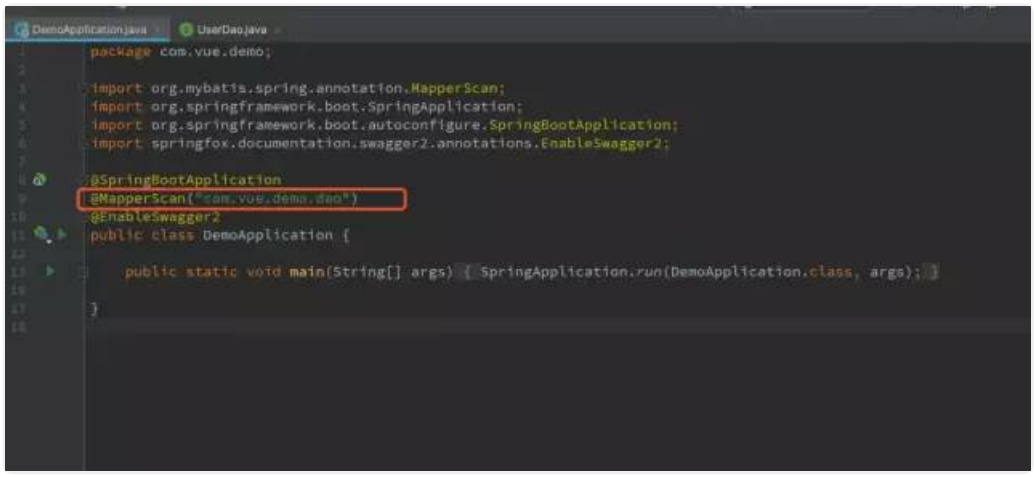
7、启动项目

在启动项目之前，我们需要先修改两个地方。

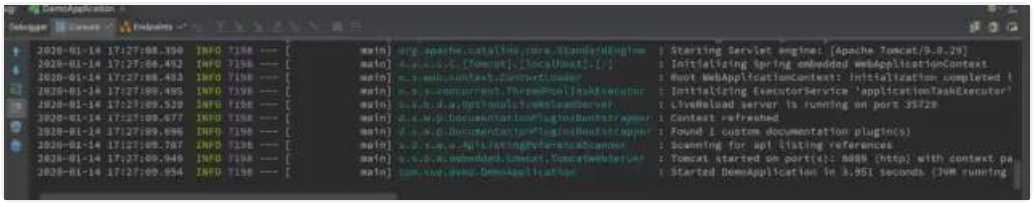
在dao层加上@mapper注解



在启动类里面加上@MapperScan("com.vue.demo.dao")注解。



启动项目

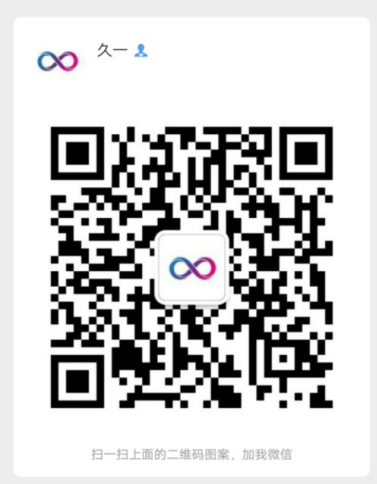


测试一下

-END-

如果看到这里，说明你喜欢这篇文章，请**转发、点赞**。微信搜索「web_resource」，欢迎添加小编微信「focusoncode」，每日朋友圈更新一篇高质量技术博文（无广告）。

↓扫描二维码添加小编↓



推荐阅读

1. Dubbo 爆出严重漏洞!
2. Spring Boot+Redis 分布式锁：模拟抢单
3. 安利一款 IDEA 中强大的代码生成利器
4. 如何获取靠谱的新型冠状病毒疫情

Intellij IDEA 高级用法之：集成 JIRA、SSH、FTP、Database管理、UML类图插件

菩提树下的杨过 Java后端 2019-11-18

作者 | 菩提树下的杨过

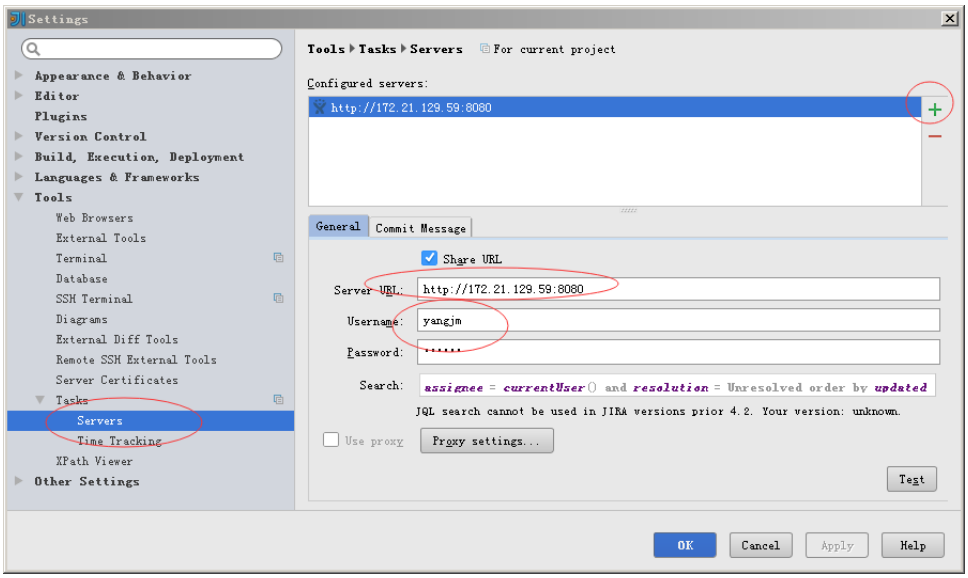
来源 | www.cnblogs.com/yjmyzz

上篇 | 除了《颈椎康复指南》，还有这 9 本书

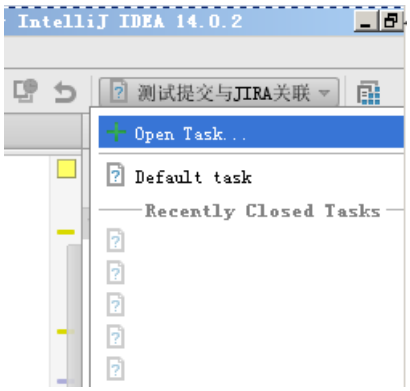
一、与JIRA集成

jira是一个广泛使用的项目与事务跟踪工具，被广泛应用于缺陷跟踪、客户服务、需求收集、流程审批、任务跟踪、项目跟踪和敏捷管理等工作领域。idea可以很好的跟它集成，参考下图：

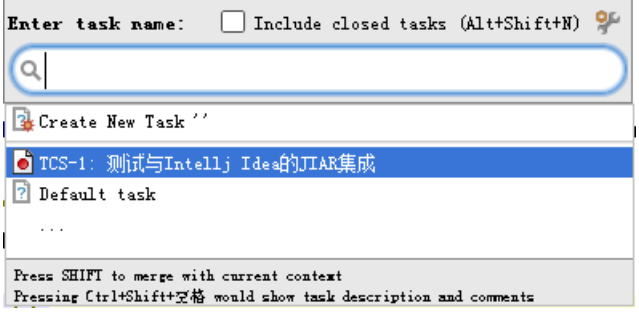
File -> Settings ->Task -> Servers 点击右侧上面的+号，选择JIRA，然后输入JIRA的Server地址，用户名、密码即可



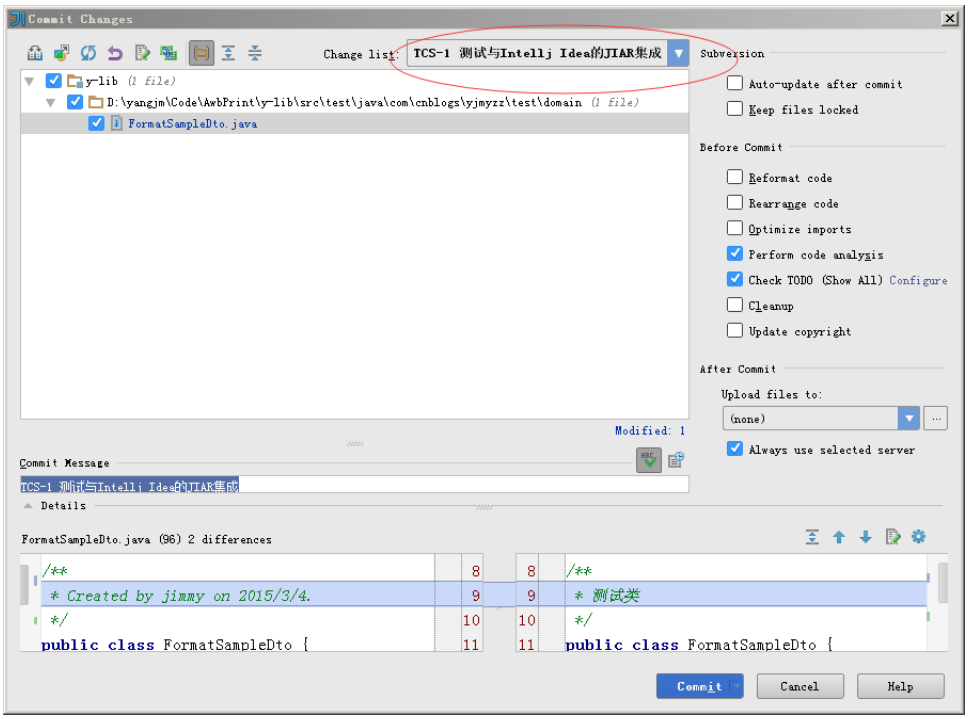
然后打开Open Task界面



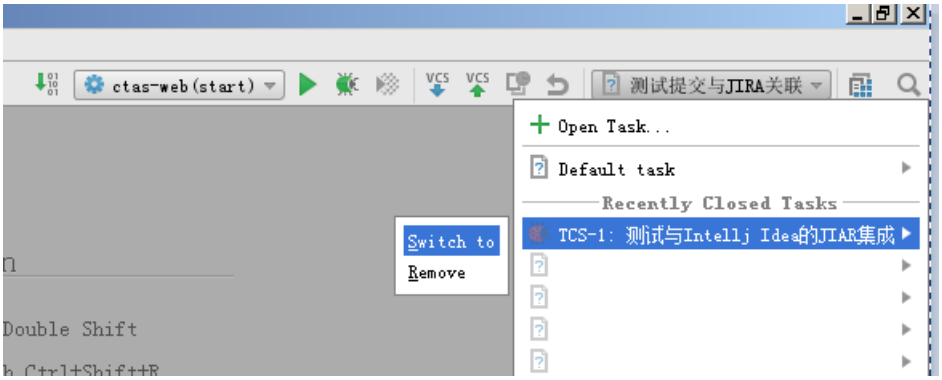
如果JIRA中有分配给你的Task，idea能自动列出来



代码修改后，向svn提交时，会自动与该任务关联



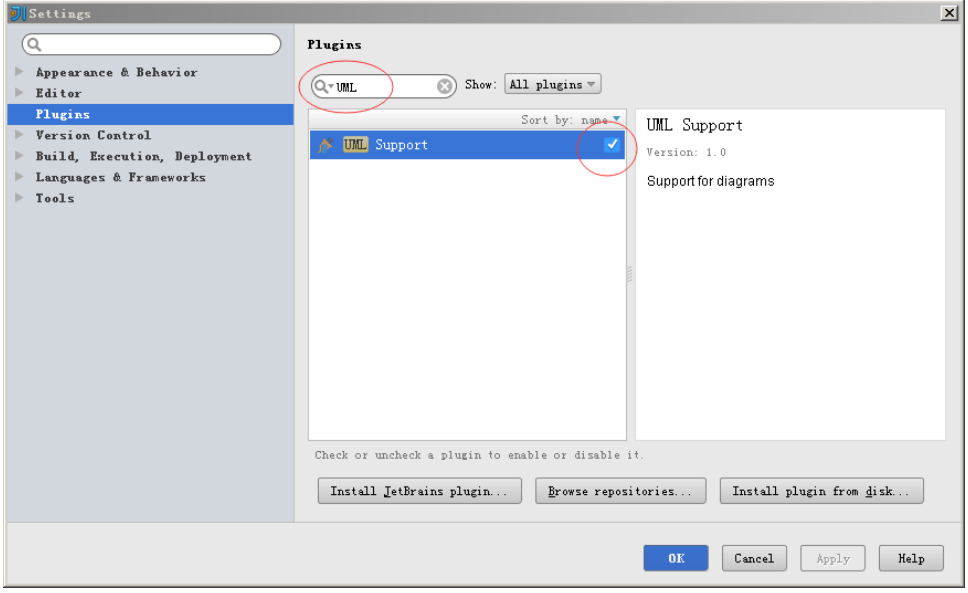
将每次提交的代码修改与JIRA上的TASK关联后，有什么好处呢？我们每天可能要写很多代码，修复若干bug，日子久了以后，谁也不记得当初为了修复某个bug做了哪些修改，不要紧张，只要你按上面的操作正确提交，idea都会帮你记着这些细节



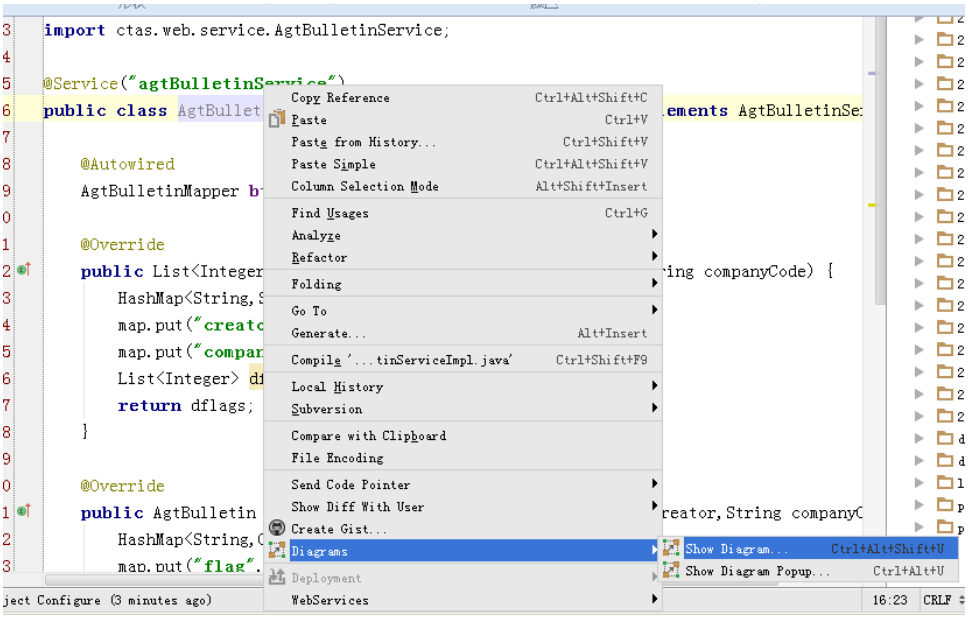
如上图，选择最近提交的TASK列表，选择Switch to，idea就会自动打开该TASK关联的源代码，并定位到修改过的代码行。 当然如果该TASK已经Close了，也可以选择Remove将其清空。

二、UML类图插件

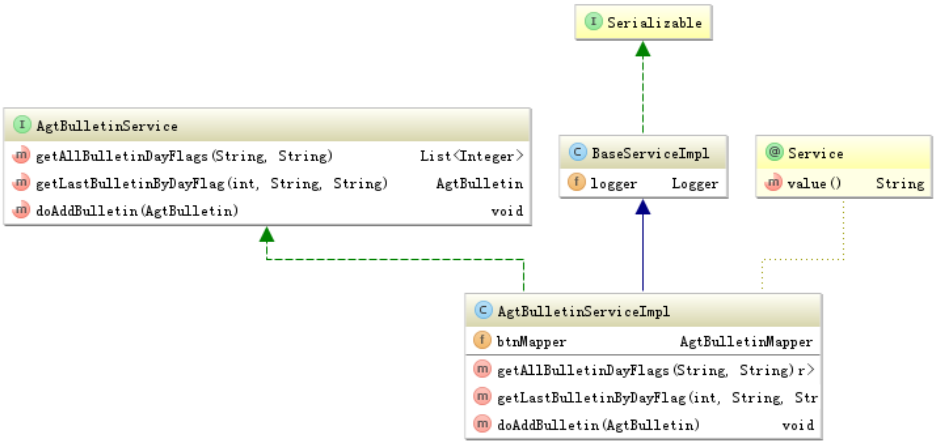
idea已经集成了该功能，只是默认没打开，仍然打开Settings界面，定位到Plugins，输入UML，参考下图：



确认UML 这个勾已经勾上了，然后点击Apply，重启idea，然后仍然找一个java类文件，右击Diagram

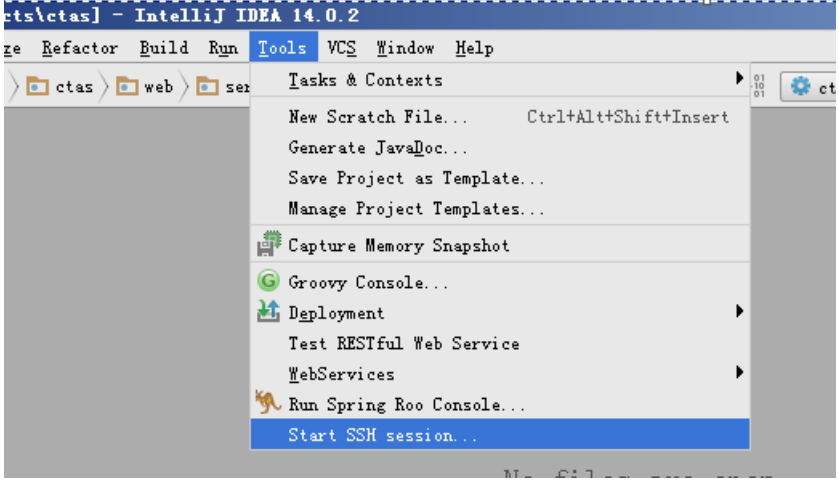


然后，就自个儿爽去吧

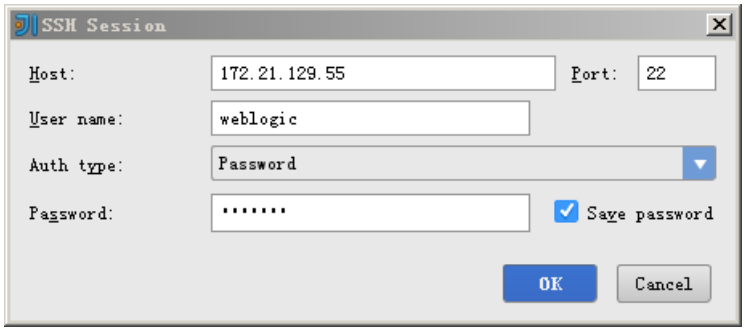


三、SSH集成

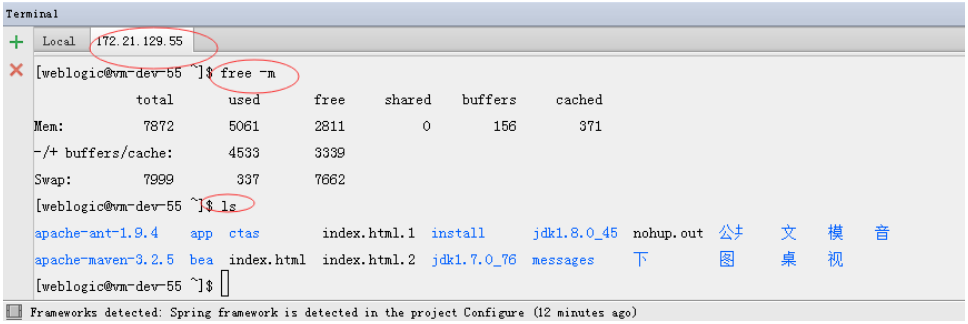
java项目经常会在linux上部署，每次要切换到SecureCRT这类终端工具未免太麻烦，idea也想到了这一点:



然后填入IP、用户名、密码啥的

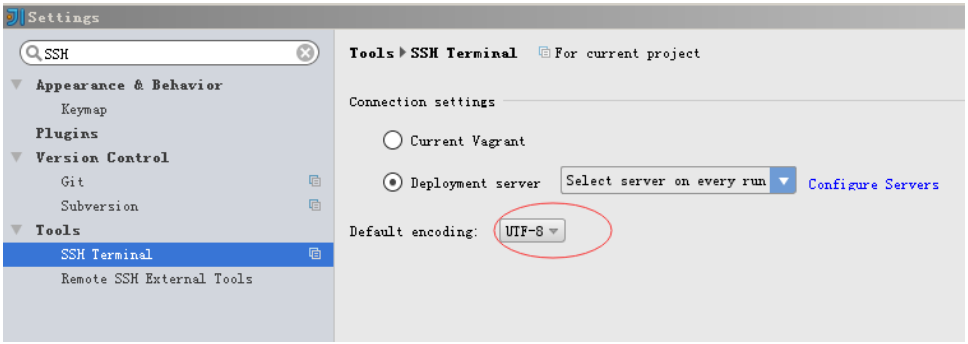


点击OK，就能连接上linux了

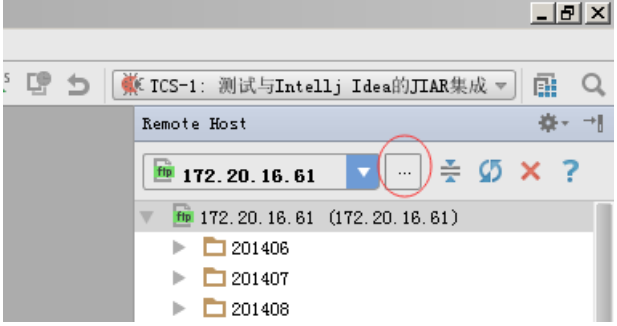


注：如果有中文乱码问题，可以在Settings里调整编码为utf-8

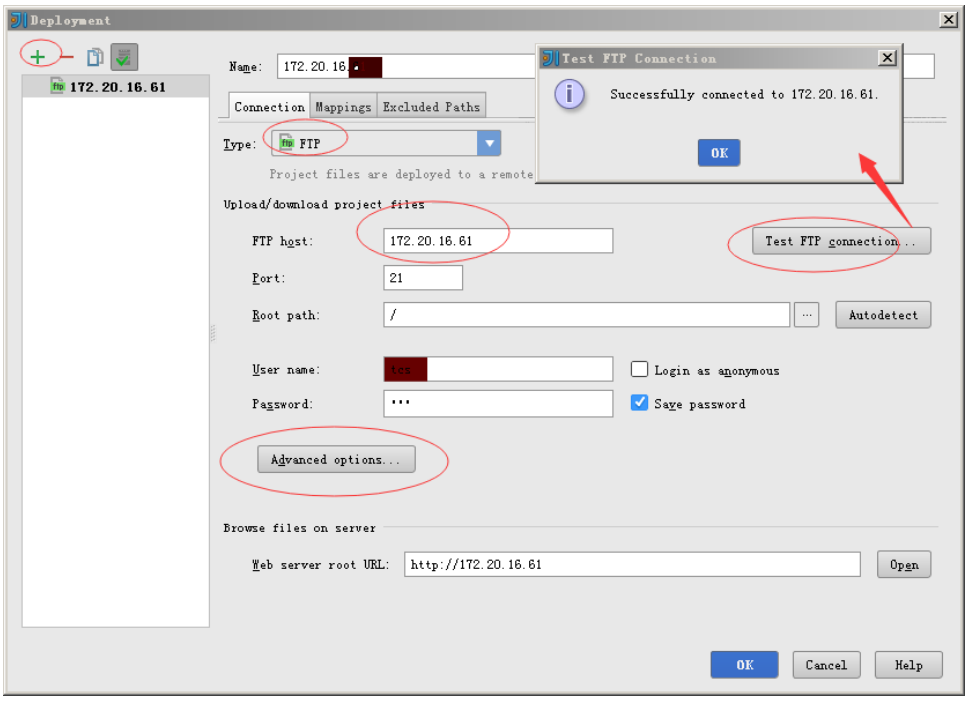
Tips：欢迎关注微信公众号：Java后端，每日技术博文推送。



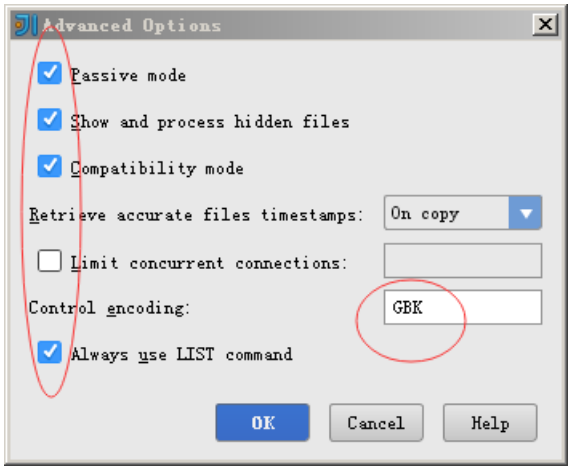
四、集成FTP



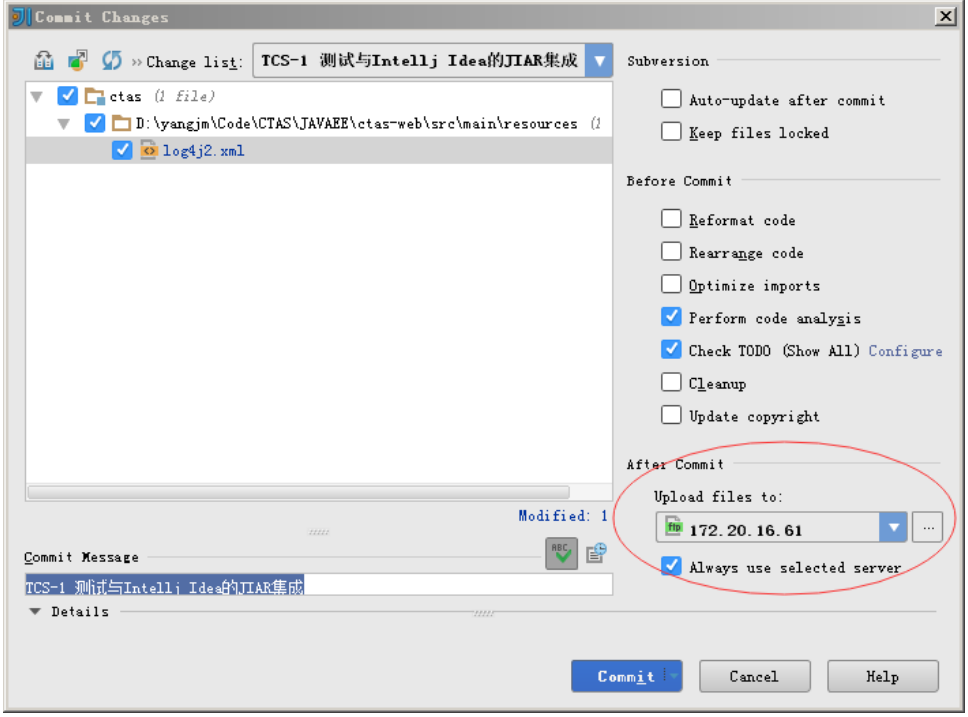
点击上图中的...，添加一个Remote Host



填写ftp的IP、用户名、密码，根路径啥的，然后点击Test FTP Connection，正常的话，应该能连接，如果连接不通，点击Advanced Options，参考下图调整下连接选项

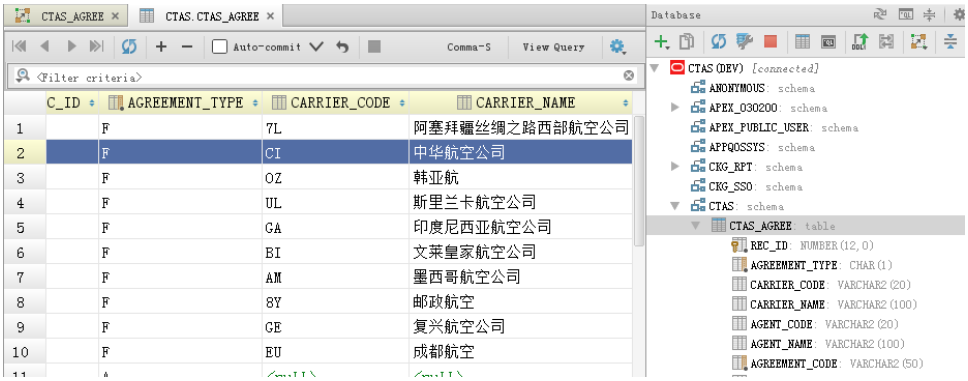


配置了FTP连接后，在提交代码时，可以选择提交完成后将代码自动上传到ftp服务器

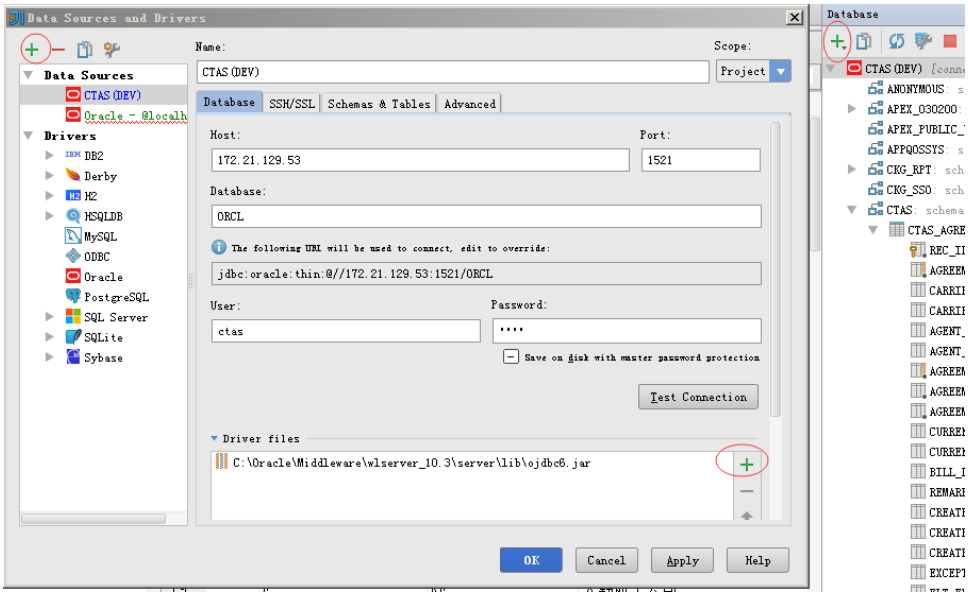


五、Database管理工具

先看效果吧：



有了这个，再也不羡慕vs.net的db管理功能了。配置也很简单，就是点击+号，增加一个Data Source即可

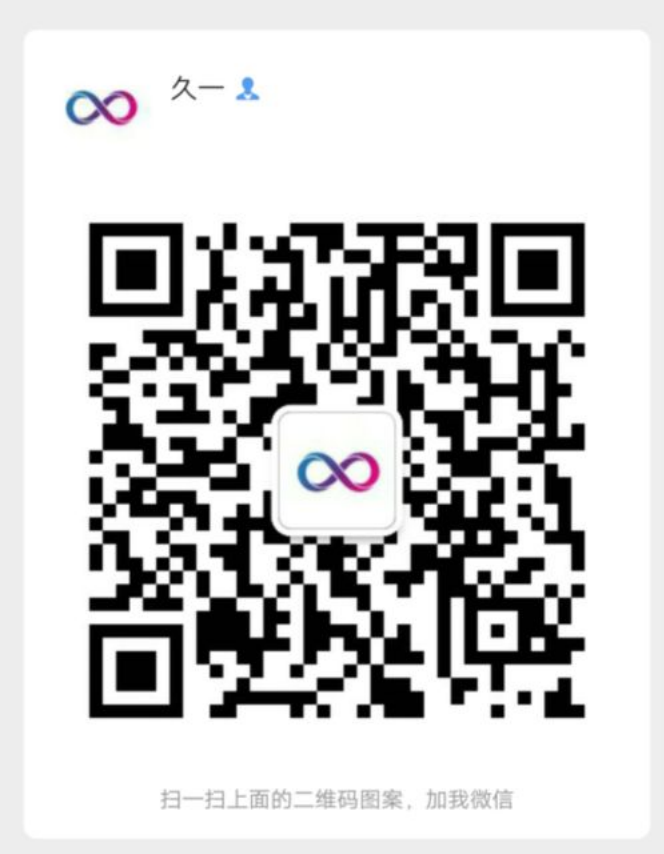


唯一要注意的是，intellij idea不带数据库驱动，所以在上图中，要手动指定db driver的jar包路径。

-END-

如果看到这里，说明你喜欢这篇文章，请**转发、点赞**。微信搜索「web_resource」，关注后回复「进群」或者扫描下方二维码即可进入无广告交流群。

↓扫描二维码进群↓



推荐阅读

1. 你们心心念念的 GitHub 客户端终于来了！
2. Redis 实现「附近的人」这个功能
3. 一个秒杀系统的设计思考
4. 零基础认识 Spring Boot
5. 团队开发中 Git 最佳实践

喜欢文章，点个在看

阅读原文

MyBatis大揭秘：Plugin 插件设计原理

Java后端 2019-12-02

作者 | 祖大俊

链接 | my.oschina.net/zudajun/blog/738973

大多数框架，都支持插件，用户可通过编写插件来自行扩展功能，Mybatis也不例外。

我们从插件配置、插件编写、插件运行原理、插件注册与执行拦截的时机、初始化插件、分页插件的原理等六个方面展开阐述。

1. 插件配置

Mybatis的插件配置在configuration内部，初始化时，会读取这些插件，保存于Configuration对象的InterceptorChain中。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN" "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
  <plugins>
    <plugin interceptor="com.mybatis3.interceptor.MyBatisInterceptor">
      <property name="value" value="100" />
    </plugin>
  </plugins>
</configuration>
```

```
public class Configuration {
  protected final InterceptorChain interceptorChain = new InterceptorChain();
}
```

org.apache.ibatis.plugin.InterceptorChain.java源码。

```
public class InterceptorChain {

 private final List<Interceptor> interceptors = new ArrayList<Interceptor>();

 public Object pluginAll(Object target) {
  for (Interceptor interceptor : interceptors) {
   target = interceptor.plugin(target);
  }
  return target;
 }

 public void addInterceptor(Interceptor interceptor) {
  interceptors.add(interceptor);
 }

 public List<Interceptor> getInterceptors() {
  return Collections.unmodifiableList(interceptors);
 }

}
```

上面的for循环代表了只要是插件，都会以责任链的方式逐一执行（别指望它能跳过某个节点），所谓插件，其实就类似于拦截

器。

2. 如何编写一个插件

插件必须实现org.apache.ibatis.plugin.Interceptor接口。

```
public interface Interceptor {

 Object intercept(Invocation invocation) throws Throwable;

 Object plugin(Object target);

 void setProperties(Properties properties);

}
```

- **intercept()方法：**执行拦截内容的地方，比如想收点保护费。由plugin()方法触发，interceptor.plugin(target)足以证明。
- **plugin()方法：**决定是否触发intercept()方法。
- **setProperties()方法：**给自定义的拦截器传递xml配置的属性参数。

下面自定义一个拦截器：

```
@Intercepts({
    @Signature(type = Executor.class, method = "query", args = { MappedStatement.class, Object.class,
        RowBounds.class, ResultHandler.class }),
    @Signature(type = Executor.class, method = "close", args = { boolean.class }) })
public class MyBatisInterceptor implements Interceptor {

  private Integer value;

  @Override
  public Object intercept(Invocation invocation) throws Throwable {
    return invocation.proceed();
  }

  @Override
  public Object plugin(Object target) {
    System.out.println(value);
    // Plugin类是插件的核心类，用于给target创建一个JDK的动态代理对象，触发intercept()方法
    return Plugin.wrap(target, this);
  }

  @Override
  public void setProperties(Properties properties) {
    value = Integer.valueOf((String) properties.get("value"));
  }

}
```

面对上面的代码，我们需要解决两个疑问：

1. 为什么要写Annotation注解？注解都是什么含义？

答：Mybatis规定插件必须编写Annotation注解，是必须，而不是可选。

@Intercepts注解：装载一个@Signature列表，一个@Signature其实就是一个需要拦截的方法封装。那么，一个拦截器要拦截

多个方法，自然就是一个@Signature列表。

`type = Executor.class, method = "query", args = { MappedStatement.class, Object.class, RowBounds.class, ResultHandler.class }`

解释：要拦截Executor接口内的query()方法，参数类型为args列表。

2. Plugin.wrap(target, this)是干什么的?

答：使用JDK的动态代理，给target对象创建一个delegate代理对象，以此来实现方法拦截和增强功能，它会回调intercept()方法。

org.apache.ibatis.plugin.Plugin.java源码：

```
public class Plugin implements InvocationHandler {

 private Object target;
 private Interceptor interceptor;
 private Map<Class<?>, Set<Method>> signatureMap;

 private Plugin(Object target, Interceptor interceptor, Map<Class<?>, Set<Method>> signatureMap) {
  this.target = target;
  this.interceptor = interceptor;
  this.signatureMap = signatureMap;
 }

 public static Object wrap(Object target, Interceptor interceptor) {
  Map<Class<?>, Set<Method>> signatureMap = getSignatureMap(interceptor);
  Class<?> type = target.getClass();
  Class<?>[] interfaces = getAllInterfaces(type, signatureMap);
  if (interfaces.length > 0) {
   // 创建JDK动态代理对象
   return Proxy.newProxyInstance(
     type.getClassLoader(),
     interfaces,
     new Plugin(target, interceptor, signatureMap));
  }
  return target;
 }

 @Override
 public Object invoke(Object proxy, Method method, Object[] args) throws Throwable {
  try {
   Set<Method> methods = signatureMap.get(method.getDeclaringClass());
   // 判断是否是需要拦截的方法(很重要)
   if (methods != null && methods.contains(method)) {
    // 回调intercept()方法
    return interceptor.intercept(new Invocation(target, method, args));
   }
   return method.invoke(target, args);
  } catch (Exception e) {
   throw ExceptionUtil.unwrapThrowable(e);
  }
 }
//...
}
```

`Map<Class<?>, Set<Method>> signatureMap`：缓存需拦截对象的反射结果，避免多次反射，即target的反射结果。

所以，我们不要动不动就说反射性能很差，那是因为你没有像Mybatis一样去缓存一个对象的反射结果。

判断是否是需要拦截的方法，这句注释很重要，一旦忽略了，都不知道Mybatis是怎么判断是否执行拦截内容的，要记住。

3. Mybatis可以拦截哪些接口对象?

```
public class Configuration {
//...
public ParameterHandler newParameterHandler(MappedStatement mappedStatement, Object parameterObject, BoundSql boundSql) {
  ParameterHandler parameterHandler = mappedStatement.getLang().createParameterHandler(mappedStatement, parameterObject, bo
  parameterHandler = (ParameterHandler) interceptorChain.pluginAll(parameterHandler); // 1
  return parameterHandler;
 }

 public ResultSetHandler newResultSetHandler(Executor executor, MappedStatement mappedStatement, RowBounds rowBounds, Para
    ResultHandler resultHandler, BoundSql boundSql) {
  ResultSetHandler resultSetHandler = new DefaultResultSetHandler(executor, mappedStatement, parameterHandler, resultHandler, bou
  resultSetHandler = (ResultSetHandler) interceptorChain.pluginAll(resultSetHandler); // 2
  return resultSetHandler;
 }

 public StatementHandler newStatementHandler(Executor executor, MappedStatement mappedStatement, Object parameterObject, Ro
  StatementHandler statementHandler = new RoutingStatementHandler(executor, mappedStatement, parameterObject, rowBounds, resu
  statementHandler = (StatementHandler) interceptorChain.pluginAll(statementHandler); // 3
  return statementHandler;
 }

 public Executor newExecutor(Transaction transaction) {
  return newExecutor(transaction, defaultExecutorType);
 }

 public Executor newExecutor(Transaction transaction, ExecutorType executorType) {
  executorType = executorType == null ? defaultExecutorType : executorType;
  executorType = executorType == null ? ExecutorType.SIMPLE : executorType;
  Executor executor;
  if (ExecutorType.BATCH == executorType) {
   executor = new BatchExecutor(this, transaction);
  } else if (ExecutorType.REUSE == executorType) {
   executor = new ReuseExecutor(this, transaction);
  } else {
   executor = new SimpleExecutor(this, transaction);
  }
  if (cacheEnabled) {
   executor = new CachingExecutor(executor);
  }
  executor = (Executor) interceptorChain.pluginAll(executor); // 4
  return executor;
 }
//...
}
```

Mybatis只能拦截ParameterHandler、ResultSetHandler、StatementHandler、Executor共4个接口对象内的方法。

重新审视interceptorChain.pluginAll()方法：该方法在创建上述4个接口对象时调用，其含义为给这些接口对象注册拦截器功能，注意是注册，而不是执行拦截。

拦截器执行时机：plugin()方法注册拦截器后，那么，在执行上述4个接口对象内的具体方法时，就会自动触发拦截器的执行，也就是插件的执行。

所以，一定要分清，何时注册，何时执行。切不可认为pluginAll()或plugin()就是执行，它只是注册。

4. Invocation

```
public class Invocation {
 private Object target;
 private Method method;
 private Object[] args;
}
```

`intercept(Invocation invocation)` 方法的参数Invocation ，我相信你一定可以看得懂，不解释。

5. 初始化插件源码解析

`org.apache.ibatis.builder.xml.XMLConfigBuilder.parseConfiguration(XNode)` 方法部分源码。

```
pluginElement(root.evalNode("plugins"));

 private void pluginElement(XNode parent) throws Exception {
   if (parent != null) {
    for (XNode child : parent.getChildren()) {
     String interceptor = child.getStringAttribute("interceptor");
     Properties properties = child.getChildrenAsProperties();
     Interceptor interceptorInstance = (Interceptor) resolveClass(interceptor).newInstance();
     //这里展示了setProperties()方法的调用时机
     interceptorInstance.setProperties(properties);
     configuration.addInterceptor(interceptorInstance);
    }
   }
 }
```

对于Mybatis，它并不区分是何种拦截器接口，所有的插件都是Interceptor，Mybatis完全依靠Annotation去标识对谁进行拦截，所以，具备接口一致性。

6. 分页插件原理

由于Mybatis采用的是逻辑分页，而非物理分页，那么，市场上就出现了可以实现物理分页的Mybatis的分页插件。

要实现物理分页，就需要对String sql进行拦截并增强，Mybatis通过BoundSql对象存储String sql，而BoundSql则由StatementHandler对象获取。

```
public interface StatementHandler {
   <E> List<E> query(Statement statement, ResultHandler resultHandler) throws SQLException;

   BoundSql getBoundSql();
}
```

```
public class BoundSql {
  public String getSql() {
   return sql;
  }
}
```

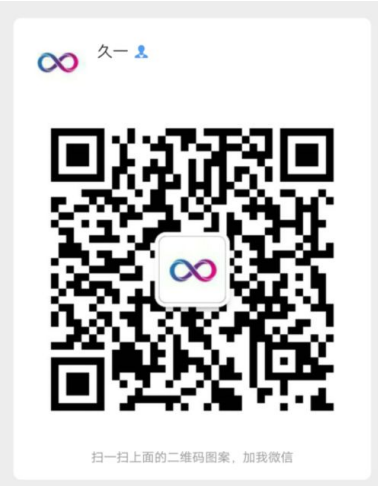
因此，就需要编写一个针对StatementHandler的query方法拦截器，然后获取到sql，对sql进行重写增强。

任它天高海阔，任它变化无穷，我们只要懂得原理，再多插件，我们都可以对其投送王之蔑视。

【END】

如果看到这里，说明你喜欢这篇文章，请**转发、点赞**。微信搜索「web_resource」，关注后回复「进群」或者扫描下方二维码即可进入无广告交流群。

↓扫描二维码进群↓



推荐阅读

1. 你在公司项目里面看过哪些操蛋的代码?
2. 你知道 Spring Batch 吗?
3. 面试题：Lucene、Solr、ElasticSearch
4. 3 分钟带你彻底搞懂 Java 泛型背后的秘密
5. 团队开发中 Git 最佳实践

喜欢文章，点个在看

如何使用牛逼的插件帮你规范代码

Java后端 2019-09-05

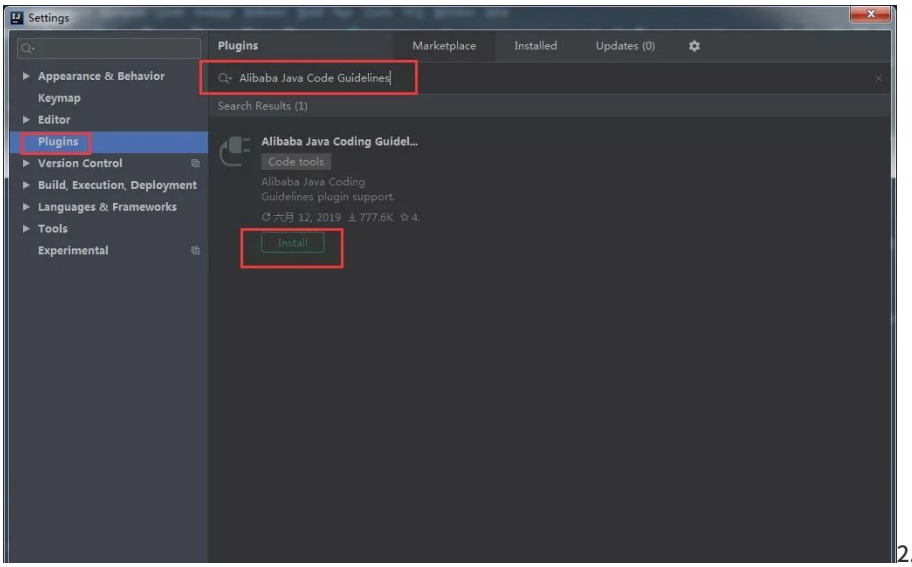
以下文章来源于阿拉奇学Java ，作者阿拉奇学Java

前言

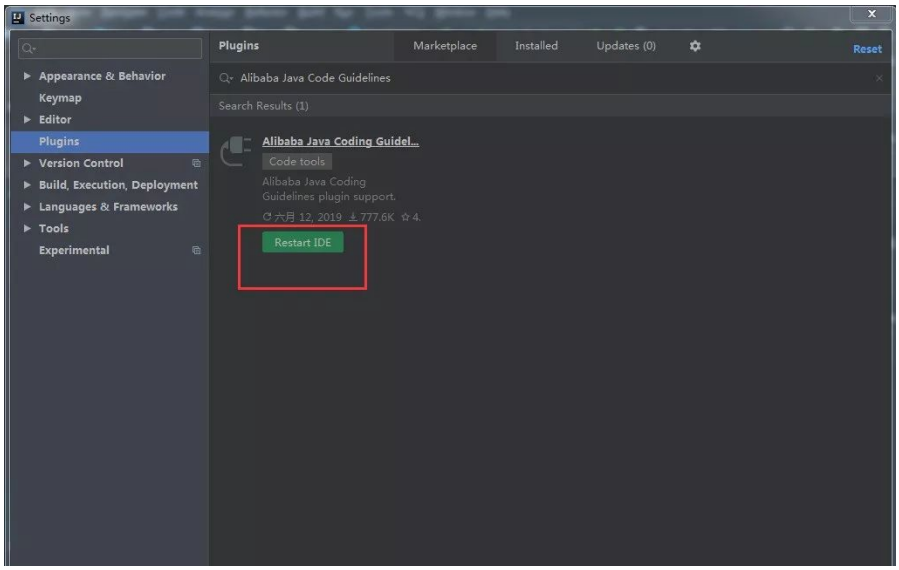
阿里巴巴将《阿里巴巴Java开发手册》文档进行了升级，在2017年10月14日杭州云栖大会，Java代码规约扫描插件全球首发仪式正式启动，规范正式以插件形式公开走向业界，引领Java语言的规范之路。使用该插件进行扫描工程，可以扫描出Blocker/Critical/Major三个等级的隐患代码，在Snoar中对代码规则有五个级别，这是前三个，翻译下就是：崩溃/严重/重要，也就是说前两级别是必须要处理掉的。同时还会给出修改意见。可以说不但规范了代码，也带你避免掉了一些潜在的bug。

一、IDEA如何安装阿里巴巴代码规范插件

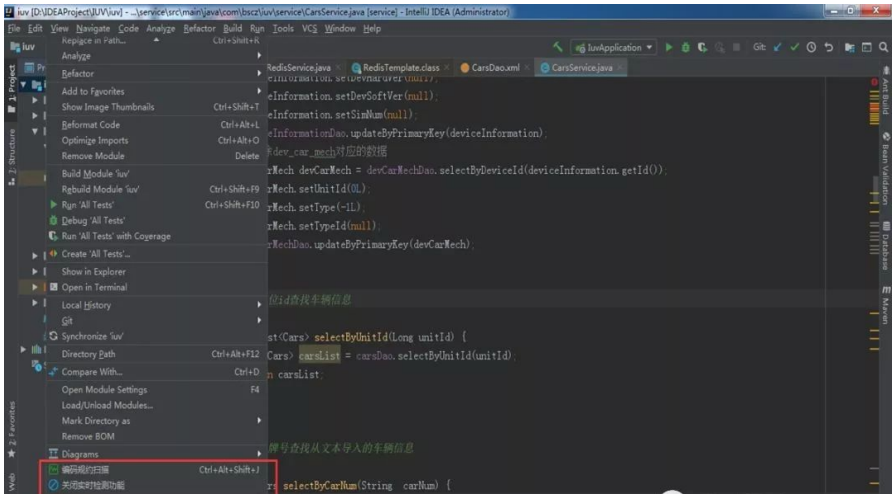
1. 启动IDEA >> File >> Settings >> Plugins，搜索Alibaba Java Code Guidelines(阿里巴巴Java代码指南)插件，点击Install进行安装。



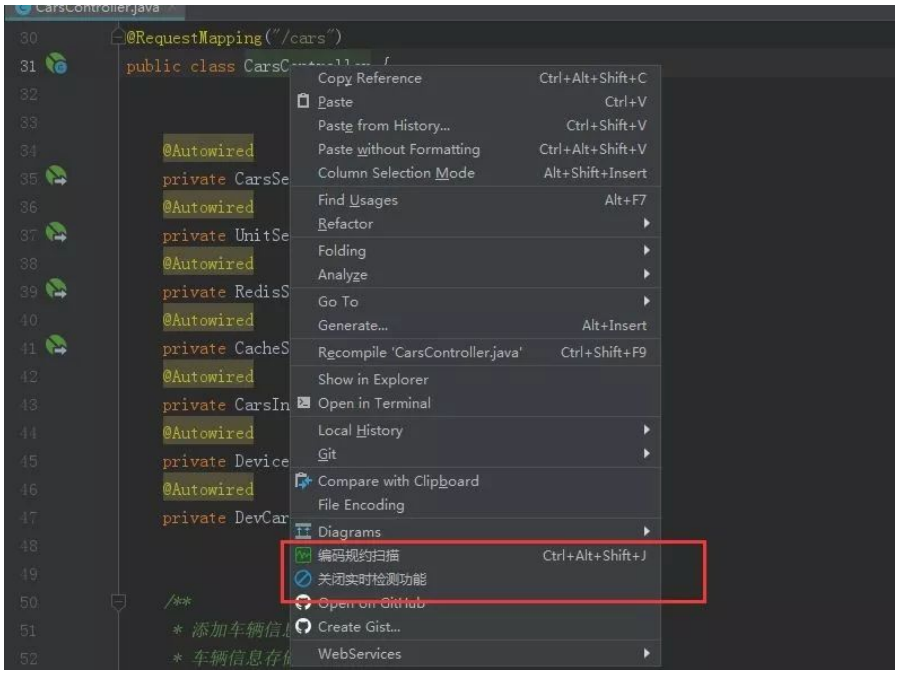
2. 安装好重启IDEA，之后生效。需要注意的是插件基于JDK1.7打包，如果IDEA启动时使用的JDK版本是1.6的话就会报Unsupported.major.minor version 51.0异常，所以建议大家升级一下。+



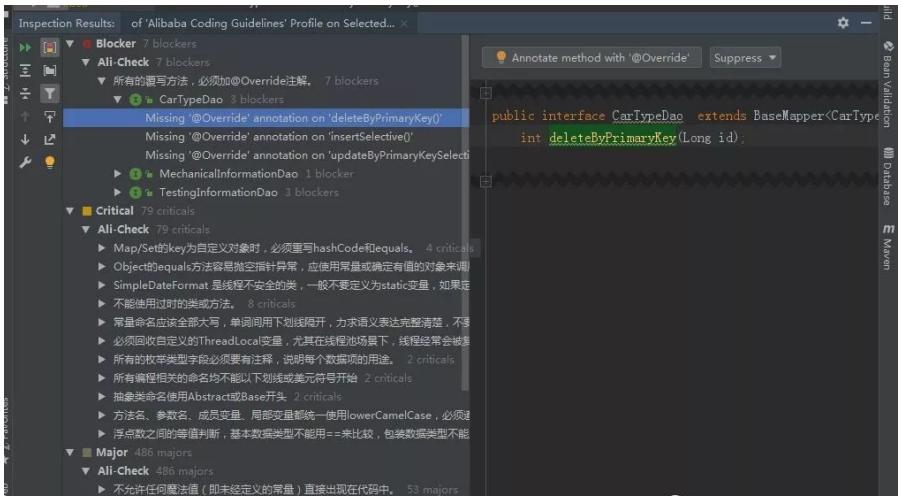
3. IDEA重启之后点击工程右键，或者使用默认快捷键Ctrl+Shift+Alt+J来扫描你的工程吧。看看你的代码有多少不规范的地方呢。



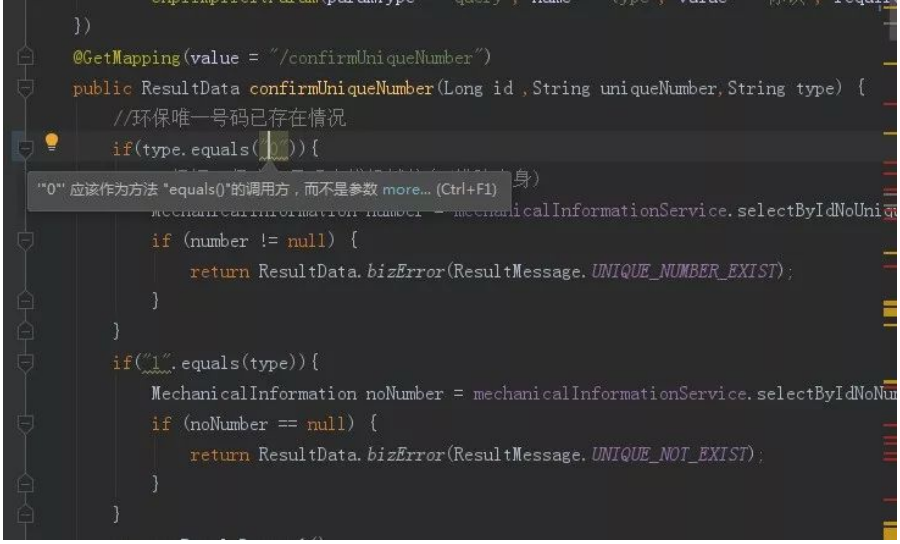
4. 也可以点击具体的某个类右键进行扫描。



5. 扫描整个工程，将不符合规约的代码按 **Blocker**、**Critical**、**Major** 三个等级显示，右侧窗口还有针对代码的批量修复功能。

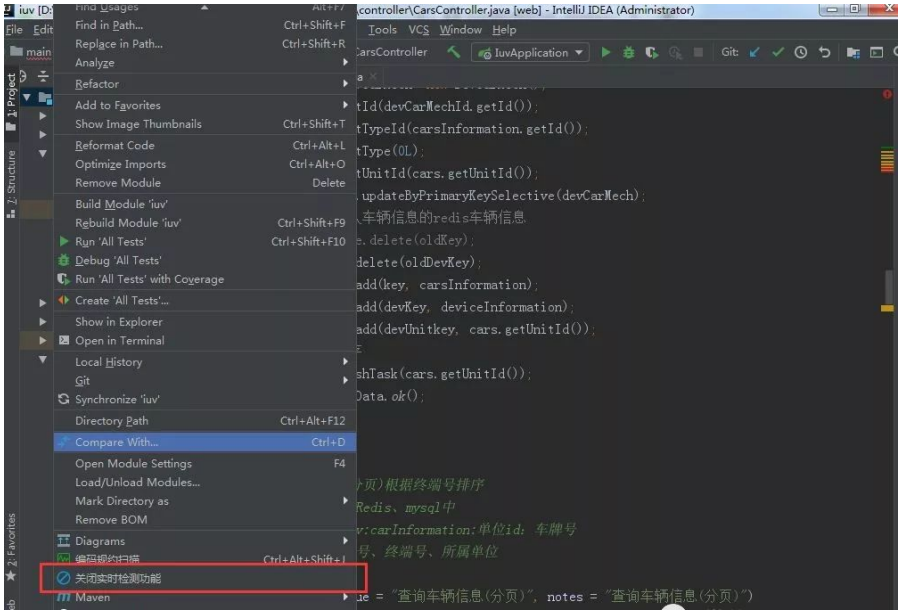


6. 实时检测功能，在开发时，对当前文件实时进行检测，并高亮显示出来，同时也给出修改提示。可以说是非常好用了。



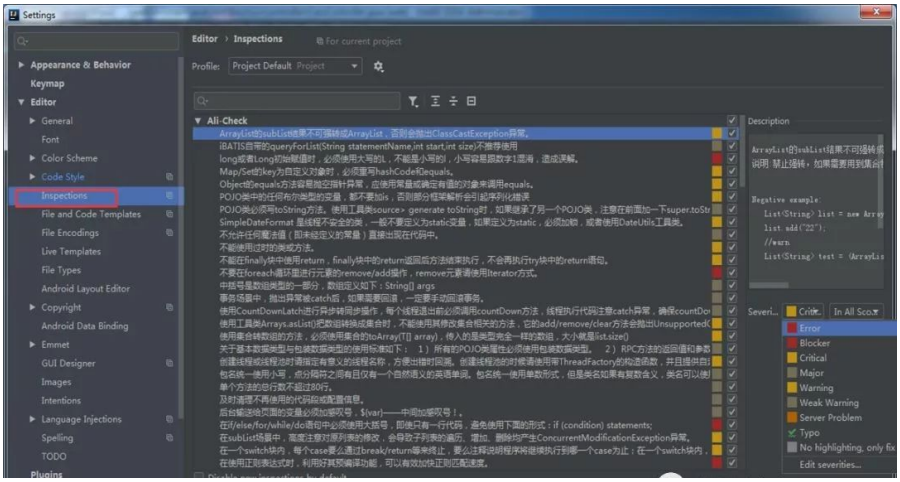
7. 关闭实时检测功能

如果你正在阅读一些代码，恰巧代码又没有按照阿里代码规范写，此时屏幕上都是一些红色、黄色的警告，严重影响阅读。这个时候可以右击工程点击关闭实时检测的功能。



8. 或许阿里条约有些并不适合自己团队的开发要求，这也不需要担心，因为也可以单独的关闭某条规则。或者是修改其提示的级别。是不是很人性化呀。

File>>Settings >>Editor >> Inspections



二、修改不规范代码

扫描完整个工程之后发现有很多不符合规范的代码，小编就挑一些跟大家一起修改吧。

1. 不允许出现任何魔法值（即未经定义的常量）直接出现在代码中。所谓的魔法值就是，未经定义的常量字面量，所有在代码中使用的常量必须预先经过定义。

反例：

```
if (key.equals("zhangSan")) {
//...
      }
```

建议改为：

```
String KEY_PRE = "zhangSan"
;
if(KEY_PRE.equals(key)){
//...
         }
```

2.Object的equals方法容易抛空指针异常，应使用常量或确定有值的对象来调用equals。(是不是有的小伙伴没有注意过这个问题？)

反例：

```
public void f(String str) {
        String inner = "hi"
;
if (str.equals(inner)) {
            System.out.println("hello world")
;
        }
    }
```

建议改为：

```
public void f(String str) {
        String inner = "hi"
;
        if (inner.equals(str))
{
            System.out.println("hello world")
;
        }
    }
```

3.所有的覆写方法，必须加@Override注解。

其实Override注解的本身并没有什么作用，但是它可以告诉代码的读者，这个是覆盖父类的方法。如果方法名、参数、异常定义错误，导致不能正确覆盖父类的方法，编译器会提示错误。比如getObject()与get0bject()的问题。一个是字母的O，一个是数字的0，加@Override可以准确判断是否覆盖成功。所以在意细节还是很重要的。

4. 事务场景中，抛出异常被catch后，如果需要回滚，一定要手动回滚事务。

反例 (注解【Transactional】需要设置rollbackFor属性。)：

```
@Service
@Transactional
public class MechanicalInformationService {

}
```

建议改为：

例子一：

```
@Service
@Transactional(rollbackFor = Exception.class)
public class UserServiceImpl implements UserService {
@Override
public void save(User user) {

    }
}
```

例子二：

```
@Service
public class UserServiceImpl implements UserService {
@Override
@Transactional(rollbackFor = Exception.class)
public void save(User user) {

    }
}
```

例子三：

```
@Service
public class UserServiceImpl implements UserService {
@Autowired
private DataSourceTransactionManager transactionManager;

@Override
```

```
@Transactional
public void save(User user) {
        DefaultTransactionDefinition def = new DefaultTransactionDefinition();
        def.setName("SomeTxName")
;
        def.setPropagationBehavior(TransactionDefinition.PROPAGATION_REQUIRED);
        TransactionStatus status = transactionManager.getTransaction(def);
try {
        } catch (Exception ex) {

            transactionManager.rollback(status);
throw ex;
        }
    }
}
```

5. 循环体内，字符串的连接方式，使用StringBuilder的append方法进行扩展。

说明：反编译出的字节码文件显示每次循环都会new出一个StringBuilder对象，然后进行append操作，最后通过toString方法返回String对象，造成内存资源浪费。(这一点是需要十分注意的，看完之后不要再用 + 连接字符串了哦)

反例：

```
String result;
for(String string:tagNameList){
        result=result+string
;
}
```

建议改为：

```
StringBuilder stringBuilder = new StringBuilder();
for(String string:tagNameList){
        stringBuilder.append(string)
;
    }
String result=stringBuilder.toString();
```

6. 除常用方法（如getXxx/isXxx）等外，不要在条件判断中执行复杂的语句，将复杂逻辑判断的结果赋值给一个有意义的布尔变量，以提高可读性。

说明：很多if语句内的逻辑相当复杂，阅读者需要分析条件表达式的最终结果，才能明确什么样的条件执行什么样的语句，那么，如果阅读者分析逻辑表达式错误呢?

反例：

```
if((file.open(fileName,"w")!=null)&&(...)||(...)){
// ...
```

```
}
```

建议改为：

```
boolean existed=(file.open(fileName,"w")!=null)&&(...)||(...);
if(existed){
//...
}
```

7. 集合初始化时，指定集合初始值大小。

说明：HashMap使用如下构造方法进行初始化，如果暂时无法确定集合大小，那么指定默认值（16）即可。

反例：

```
Map<String, String> map = new HashMap<String, String>();
```

建议改为：

```
Map<String, String> map = new HashMap<String, String>(16)
;
```

大家可以将《阿里巴巴Java开发手册》下下来认真阅读。这都是阿里技术精英经过实战不断完善的经验总结，可以很好的帮我们规范Java编码，提高Java开发质量和效率、同时大大降低了代码的维护成本。让程序员码出更优质的代码。将不符合规范的代码修改，逐步养成良好的编码习惯！

本文中由一枚 Java 程序猿投稿，喜欢的读者可以扫描下方二维码关注小姐姐：

常用 Maven 插件介绍（收藏大全）

种菜得瓜 Java后端 2019-11-08

作者 | 种菜得瓜

链接 | cnblogs.com/crazy-fox

我们都知道Maven本质上是一个插件框架，它的核心并不执行任何具体的构建任务，所有这些任务都交给插件来完成，例如编译源代码是由maven- compiler-plugin完成的。进一步说，每个任务对应了一个插件目标（goal），每个插件会有一个或者多个目标，例如maven- compiler-plugin的compile目标用来编译位于src/main/java/目录下的主源码，testCompile目标用来编译位于src/test/java/目录下的测试源码。

用户可以通过两种方式调用Maven插件目标。第一种方式是将插件目标与生命周期阶段（lifecycle phase）绑定，这样用户在命令行只是输入生命周期阶段而已，例如Maven默认将maven-compiler-plugin的compile目标与 compile生命周期阶段绑定，因此命令mvn compile实际上是先定位到compile这一生命周期阶段，然后再根据绑定关系调用maven-compiler-plugin的compile目标。第二种方式是直接在命令行指定要执行的插件目标，例如mvn archetype:generate 就表示调用maven-archetype-plugin的generate目标，这种带冒号的调用方式与生命周期无关。

认识上述Maven插件的基本概念能帮助你理解Maven的工作机制，不过要想更高效率地使用Maven，了解一些常用的插件还是很有必要的，这可 以帮助你避免一不小心重新发明轮子。多年来Maven社区积累了大量的经验，并随之形成了一个成熟的插件生态圈。Maven官方有两个插件列表，第一个列 表的GroupId为org.apache.maven.plugins，这里的插件最为成熟，具体地址为：http://maven.apache.org/plugins/index.html。第二个列表的GroupId为org.codehaus.mojo，这里的插件没有那么核心，但也有不少十分有用，其地址为：http://mojo.codehaus.org/plugins.html。

接下来笔者根据自己的经验介绍一些最常用的Maven插件，在不同的环境下它们各自都有其出色的表现，熟练地使用它们能让你的日常构建工作事半功倍。

maven-antrun-plugin

http://maven.apache.org/plugins/maven-antrun-plugin/

maven-antrun-plugin能让用户在Maven项目中运行Ant任务。用户可以直接在该插件的配置以Ant的方式编写Target， 然后交给该插件的run目标去执行。在一些由Ant往Maven迁移的项目中，该插件尤其有用。此外当你发现需要编写一些自定义程度很高的任务，同时又觉 得Maven不够灵活时，也可以以Ant的方式实现之。maven-antrun-plugin的run目标通常与生命周期绑定运行。

maven-archetype-plugin

http://maven.apache.org/archetype/maven-archetype-plugin/

Archtype指项目的骨架，Maven初学者最开始执行的Maven命令可能就是mvn archetype:generate，这实际上就是让maven-archetype-plugin生成一个很简单的项目骨架，帮助开发者快速上手。可能也有人看到一些文档写了mvn archetype:create，但实际上create目标已经被弃用了，取而代之的是generate目标，该目标使用交互式的方式提示用户输入必要的信息以创建项目，体验更好。maven-archetype-plugin还有一些其他目标帮助用户自己定义项目原型，例如你由一个产品需要交付给很多客户进行二次开发，你就可以为 他们提供一个Archtype，帮助他们快速上手。

maven-assembly-plugin

http://maven.apache.org/plugins/maven-assembly-plugin/

maven-assembly-plugin的用途是制作项目分发包，该分发包可能包含了项目的可执行文件、源代码、readme、平台脚本等等。maven-assembly-plugin支持各种主流的格式如zip、tar.gz、jar和war等，具体打包哪些文件是高度可控的，例如用户可以 按文件级别的粒度、文件集级别的粒度、模块级别的粒度、以及依赖级别的粒度控制打包，此外，包含和排除配置也是支持的。maven-assembly- plugin要求用户使用一个名为assembly.xml的元数据文件来表述打包，它的single目标可以直接在命令行调用，也可以被绑定至生命周期。

maven-dependency-plugin

http://maven.apache.org/plugins/maven-dependency-plugin/

maven-dependency-plugin最大的用途是帮助分析项目依赖，dependency:list能够列出项目最终解析到的依赖列表，dependency:tree能进一步的描绘项目依赖树，dependency:analyze可以告诉你项目依赖潜在的问题，如果你有直接使用到的却未声明的依赖，该目标就会发出警告。maven-dependency-plugin还有很多目标帮助你操作依赖文件，例如dependency:copy-dependencies能将项目依赖从本地Maven仓库复制到某个特定的文件夹下面。

maven-enforcer-plugin

http://maven.apache.org/plugins/maven-enforcer-plugin/

在一个稍大一点的组织或团队中，你无法保证所有成员都熟悉Maven，那他们做一些比较愚蠢的事情就会变得很正常，例如给项目引入了外部的 SNAPSHOT依赖而导致构建不稳定，使用了一个与大家不一致的Maven版本而经常抱怨构建出现诡异问题。maven-enforcer- plugin能够帮助你避免之类问题，它允许你创建一系列规则强制大家遵守，包括设定Java版本、设定Maven版本、禁止某些依赖、禁止 SNAPSHOT依赖。只要在一个父POM配置规则，然后让大家继承，当规则遭到破坏的时候，Maven就会报错。除了标准的规则之外，你还可以扩展该插 件，编写自己的规则。maven-enforcer-plugin的enforce目标负责检查规则，它默认绑定到生命周期的validate阶段。

maven-help-plugin

http://maven.apache.org/plugins/maven-help-plugin/

maven-help-plugin是一个小巧的辅助工具，最简单的help:system可以打印所有可用的环境变量和Java系统属性。help:effective-pom和help:effective-settings最 为有用，它们分别打印项目的有效POM和有效settings，有效POM是指合并了所有父POM（包括Super POM）后的XML，当你不确定POM的某些信息从何而来时，就可以查看有效POM。有效settings同理，特别是当你发现自己配置的 settings.xml没有生效时，就可以用help:effective-settings来验证。此外，maven-help-plugin的describe目标可以帮助你描述任何一个Maven插件的信息，还有all-profiles目标和active-profiles目标帮助查看项目的Profile。

Tips：关注微信公众号：Java后端，获取每日技术博文推送。

maven-release-plugin

http://maven.apache.org/plugins/maven-release-plugin/

maven-release-plugin的用途是帮助自动化项目版本发布，它依赖于POM中的SCM信息。release:prepare用来准备版本发布，具体的工作包括检查是否有未提交代码、检查是否有SNAPSHOT依赖、升级项目的SNAPSHOT版本至RELEASE版本、为项目打标签等等。release:perform则 是签出标签中的RELEASE源码，构建并发布。版本发布是非常琐碎的工作，它涉及了各种检查，而且由于该工作仅仅是偶尔需要，因此手动操作很容易遗漏一 些细节，maven-release-plugin让该工作变得非常快速简便，不易出错。maven-release-plugin的各种目标通常直接在 命令行调用，因为版本发布显然不是日常构建生命周期的一部分。

maven-resources-plugin

http://maven.apache.org/plugins/maven-resources-plugin/

为了使项目结构更为清晰，Maven区别对待Java代码文件和资源文件，maven-compiler-plugin用来编译Java代码，maven-resources-plugin则用来处理资源文件。默认的主资源文件目录是src/main/resources，很多用户会需要添加额外的资源文件目录，这个时候就可以通过配置maven-resources-plugin来实现。此外，资源文件过滤也是Maven的一大特性，你可以在资源文件中使用${propertyName}形式的Maven属性，然后配置maven-resources-plugin开启对资源文件的过滤，之后就可以针对不同环境通过命令行或者Profile传入属性的值，以实现更为灵活的构建。

maven-surefire-plugin

http://maven.apache.org/plugins/maven-surefire-plugin/

可能是由于历史的原因，Maven 2/3中用于执行测试的插件不是maven-test-plugin，而是maven-surefire-plugin。其实大部分时间内，只要你的测试 类遵循通用的命令约定（以Test结尾、以TestCase结尾、或者以Test开头），就几乎不用知晓该插件的存在。然而在当你想要跳过测试、排除某些 测试类、或者使用一些TestNG特性的时候，了解maven-surefire-plugin的一些配置选项就很有用了。例如 mvn test -Dtest=FooTest 这样一条命令的效果是仅运行FooTest测试类，这是通过控制maven-surefire-plugin的test参数实现的。

build-helper-maven-plugin

http://mojo.codehaus.org/build-helper-maven-plugin/

Maven默认只允许指定一个主Java代码目录和一个测试Java代码目录，虽然这其实是个应当尽量遵守的约定，但偶尔你还是会希望能够指定多个 源码目录（例如为了应对遗留项目），build-helper-maven-plugin的add-source目标就是服务于这个目的，通常它被绑定到 默认生命周期的generate-sources阶段以添加额外的源码目录。需要强调的是，这种做法还是不推荐的，因为它破坏了 Maven的约定，而且可能会遇到其他严格遵守约定的插件工具无法正确识别额外的源码目录。

build-helper-maven-plugin的另一个非常有用的目标是attach-artifact，使用该目标你可以以classifier的形式选取部分项目文件生成附属构件，并同时install到本地仓库，也可以deploy到远程仓库。

exec-maven-plugin

http://mojo.codehaus.org/exec-maven-plugin/

exec-maven-plugin很好理解，顾名思义，它能让你运行任何本地的系统程序，在某些特定情况下，运行一个Maven外部的程序可能就是最简单的问题解决方案，这就是exec:exec的 用途，当然，该插件还允许你配置相关的程序运行参数。除了exec目标之外，exec-maven-plugin还提供了一个java目标，该目标要求你 提供一个mainClass参数，然后它能够利用当前项目的依赖作为classpath，在同一个JVM中运行该mainClass。有时候，为了简单的 演示一个命令行Java程序，你可以在POM中配置好exec-maven-plugin的相关运行参数，然后直接在命令运行 mvn exec:java 以查看运行效果。

jetty-maven-plugin

http://wiki.eclipse.org/Jetty/Feature/Jetty_Maven_Plugin

在进行Web开发的时候，打开浏览器对应用进行手动的测试几乎是无法避免的，这种测试方法通常就是将项目打包成war文件，然后部署到Web容器 中，再启动容器进行验证，这显然十分耗时。为了帮助开发者节省时间，jetty-maven-plugin应运而生，它完全兼容 Maven项目的目录结构，能够周期性地检查源文件，一旦发现变更后自动更新到内置的Jetty Web容器中。做一些基本配置后（例如Web应用的contextPath和自动扫描变更的时间间隔），你只要执行 mvn jetty:run ，然后在IDE中修改代码，代码经IDE自动编译后产生变更，再由jetty-maven-plugin侦测到后更新至Jetty容器，这时你就可以直接 测试Web页面了。需要注意的是，jetty-maven-plugin并不是宿主于Apache或Codehaus的官方插件，因此使用的时候需要额外 的配置settings.xml的pluginGroups元素，将org.mortbay.jetty这个pluginGroup加入。

versions-maven-plugin

http://mojo.codehaus.org/versions-maven-plugin/

很多Maven用户遇到过这样一个问题，当项目包含大量模块的时候，为他们集体更新版本就变成一件烦人的事情，到底有没有自动化工具能帮助完成这件 事情呢？（当然你可以使用sed之类的文本操作工具，不过不在本文讨论范围）答案是肯定的，versions-maven- plugin提供了很多目标帮助你管理Maven项目的各种版本信息。例如最常用的，命令 mvn versions:set -DnewVersion=1.1-SNAPSHOT 就能帮助你把所有模块的版本更新到1.1-SNAPSHOT。该插件还提供了其他一些很有用的目标，display-dependency- updates能告诉你项目依赖有哪些可用的更新；类似的display-plugin-updates能告诉你可用的插件

更新；然后use- latest-versions能自动帮你将所有依赖升级到最新版本。最后，如果你对所做的更改满意，则可以使用 mvn versions:commit 提交，不满意的话也可以使用 mvn versions:revert 进行撤销。

小结

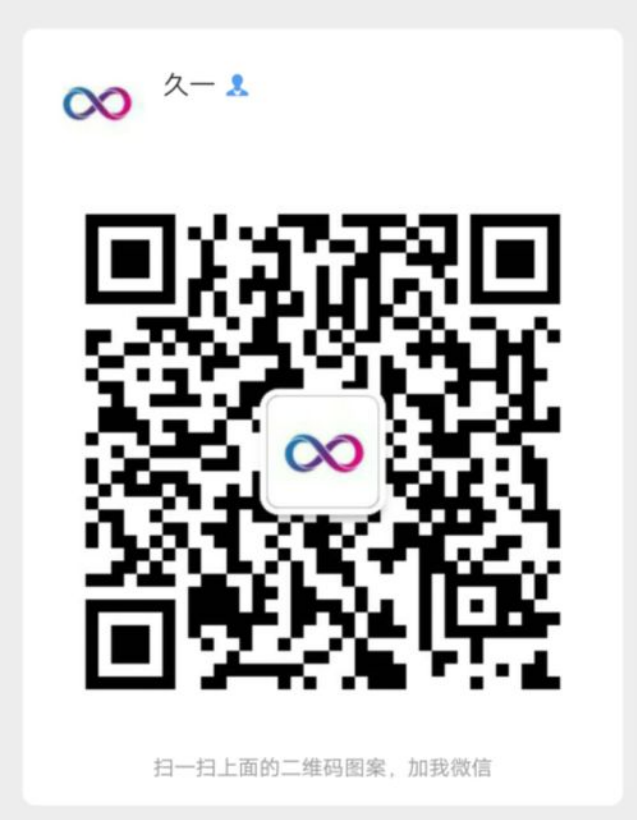
本文介绍了一些最常用的Maven插件，这里指的“常用”是指经常需要进行配置的插件，事实上我们用Maven的时候很多其它插件也是必须的，例如 默认的编译插件maven-compiler-plugin和默认的打包插件maven-jar-plugin，但因为很少需要对它们进行配置，因此不在 本文讨论范围。了解常用的Maven插件能帮助你事倍功半地完成项目构建任务，反之你就可能会因为经常遇到一些难以解决的问题而感到沮丧。本文介绍的插件 基本能覆盖大部分Maven用户的日常使用需要，如果你真有非常特殊的需求，自行编写一个Maven插件也不是难事，更何况还有这么多开放源代码的插件供 你参考。

本文的这个插件列表并不是一个完整列表，读者有兴趣的话也可以去仔细浏览一下Apache和Codehaus Mojo的Maven插件列表，以的到一个更为全面的认识。最后，在线的Maven仓库搜索引擎如http://search.maven.org/也能帮助你快速找到自己感兴趣的Maven插件。

-END-

如果看到这里，说明你喜欢这篇文章，请**转发、点赞**。微信搜索「web_resource」，关注后回复「进群」或者扫描下方二维码即可进入无广告交流群。

↓扫描二维码进群↓



推荐阅读

1. Java 代码是如何一步步输出结果的?
2. IntelliJ IDEA 详细图解最常用的配置
3. Maven 实战问题和最佳实践
4. 12306 的架构到底有多牛逼?
5. 团队开发中 Git 最佳实践

学Java，请关注公众号：Java后端

喜欢文章，点个在看

效率 Max！ IDEA 会飞？只因我装了这 12 个插件

Java后端 2019-10-04

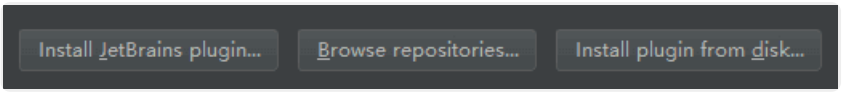
上一篇：彻底理解 Cookie，Session，Token

今天介绍一下IDEA的一些炫酷的插件，IDEA强大的插件库不仅能给我们带来一些开发的便捷，还能体现我们的与众不同。有了插件的辅佐，开发效率会大大提升，这就是为什么我的的 IDEA 会飞！

1. 插件的安装

打开setting文件选择Plugins选项

- Ctrl + Alt + S
- File -> Setting

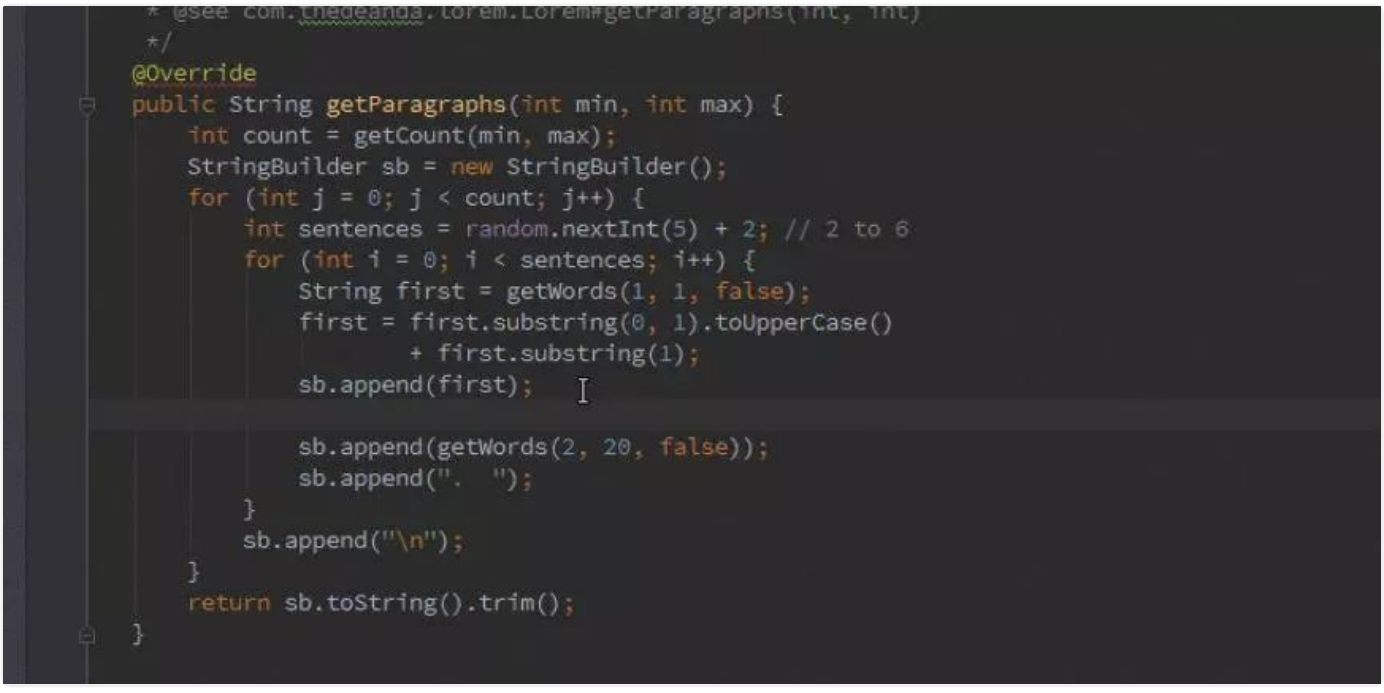


分别是安装JetBrains插件，第三方插件，本地已下载的插件包。详情见往期关于settings的文章。

2.各种插件

1. activate-power-mode 和 Power mode II

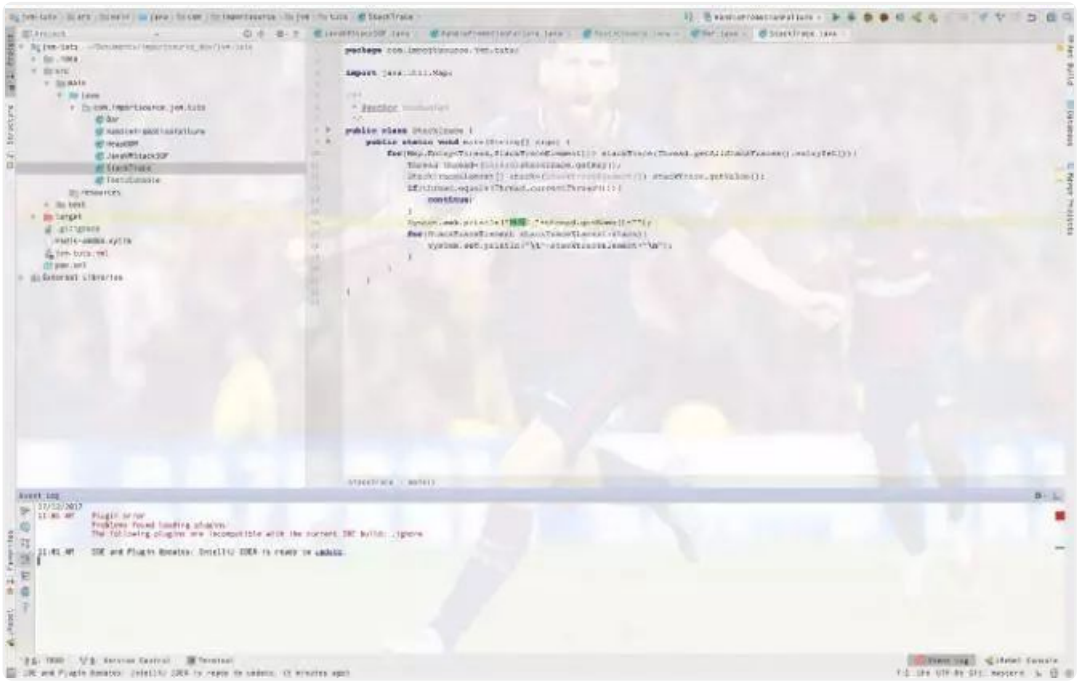
根据Atom的插件activate-power-mode的效果移植到IDEA上



写代码是整个屏幕都在抖动，activate-power-mode是白的的，Power mode II色彩更酷炫点。欢迎关注公众号 Java后端 获取更多推送。

2.Background Image Plus

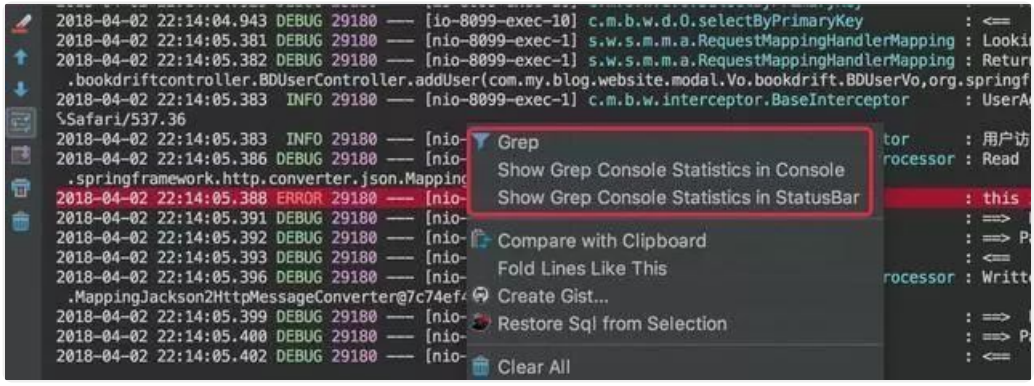
idea背景修改插件，让你的idea与众不同，可以设置自己喜欢的图片作为code背景。



安装成功之后重启，菜单栏的VIew标签>点击Set Background Image(没安装插件是没有这个标签的)，在弹框中路由选择到本地图片，点击OK即可。

3.Grep console

自定义日志颜色，idea控制台可以彩色显示各种级别的log，安装完成后，在console中右键就能打开。



并且可以设置不同的日志级别的显示样式。

可以直接根据关键字搜索你想要的，搜索条件是支持正则表达式的。

4.Free Mybatis plugin

mybatis 插件，让你的mybatis.xml像java代码一样编辑。我们开发中使用mybatis时时长需要通过mapper接口查找对应的xml中的sql语句，该插件方便了我们的操作。欢迎关注公众号 Java后端 获取更多推送。

安装完成重启IDEA之后，我们会看到code左侧或多出一列绿色的箭头，点击箭头我们就可以直接定位到xml相应文件的位置。

mapper

```
public interface LoanOrderMapper {

    int insert(LoanOrder record);

    LoanOrder selectByPrimaryKey(Long id);

    LoanOrder selectByTrdOrderId(@Param("trdLoanOrderId") Long trdLoanOrderId);
```

xml

```
    <update id="updateLoanOrderStatusByTrdLoanOrderId">
        update
        cashman_loan_order
        set
        status = #{status, jdbcType=VARCHAR}
        where
        trd_loan_order_id = #{trdLoanOrderId, jdbcType=BIGINT}
    </update>
```

5.MyBatis Log Plugin

Mybatis现在是java中操作数据库的首选，在开发的时候，我们都会把Mybatis的脚本直接输出在console中，但是默认的情况下，输出的脚本不是一个可以直接执行的。

```
c.m.b.w.d.O.selectByPrimaryKey           : ==>  Preparing: select name, value, description from t_options where name = ?
c.m.b.w.d.O.selectByPrimaryKey           : ==> Parameters: site_record(String)
c.m.b.w.d.O.selectByPrimaryKey           : <==      Total: 1
```

如果我们想直接执行，还需要在手动转化一下。欢迎关注公众号 Java后端 获取更多推送。

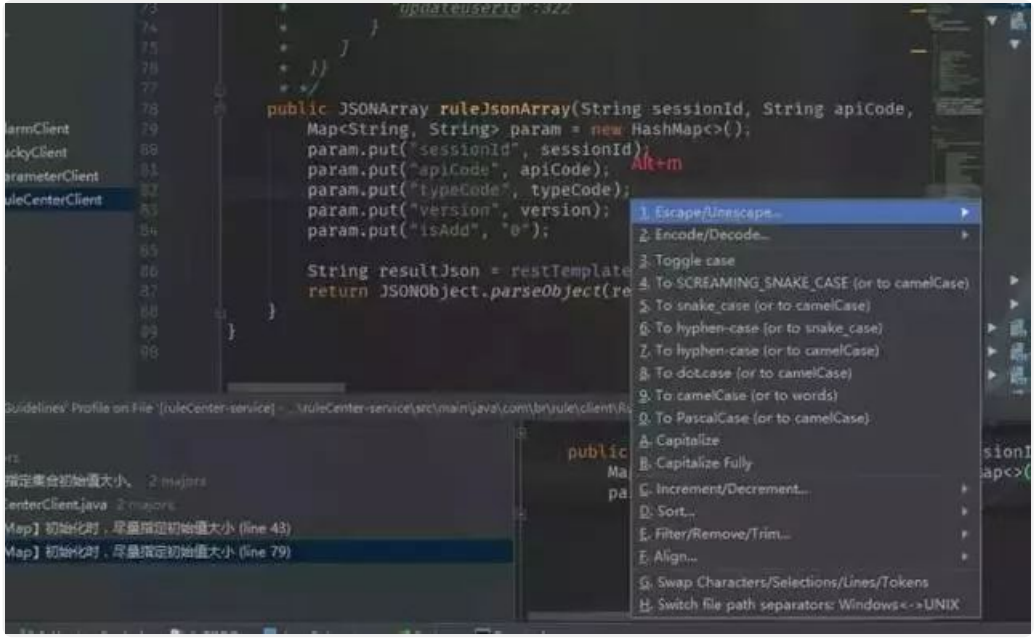
MyBatis Log Plugin 这款插件是直接将Mybatis执行的sql脚本显示出来，无需处理，可以直接复制出来执行的，如图：

```
9  2018-04-02 22:24:37.879 DEBUG 29180 --- [nio-8099-exec-7] c.m.b.w.d.O.selectByPrimaryKey          : ==>
select name, value, description
 FROM t_options
 WHERE name = 'site_record';
10  2018-04-02 22:24:38.342 DEBUG 29180 --- [nio-8099-exec-8] c.m.b.w.d.B.getCountforLogin            : ==>
SELECT count(*)
 FROM bd_user
 WHERE username = 'test';
```

执行程序后，我们可以很清晰的看到我们执行了哪些sql脚本，而且脚本可以执行拿出来运行。

6.String Manipulation

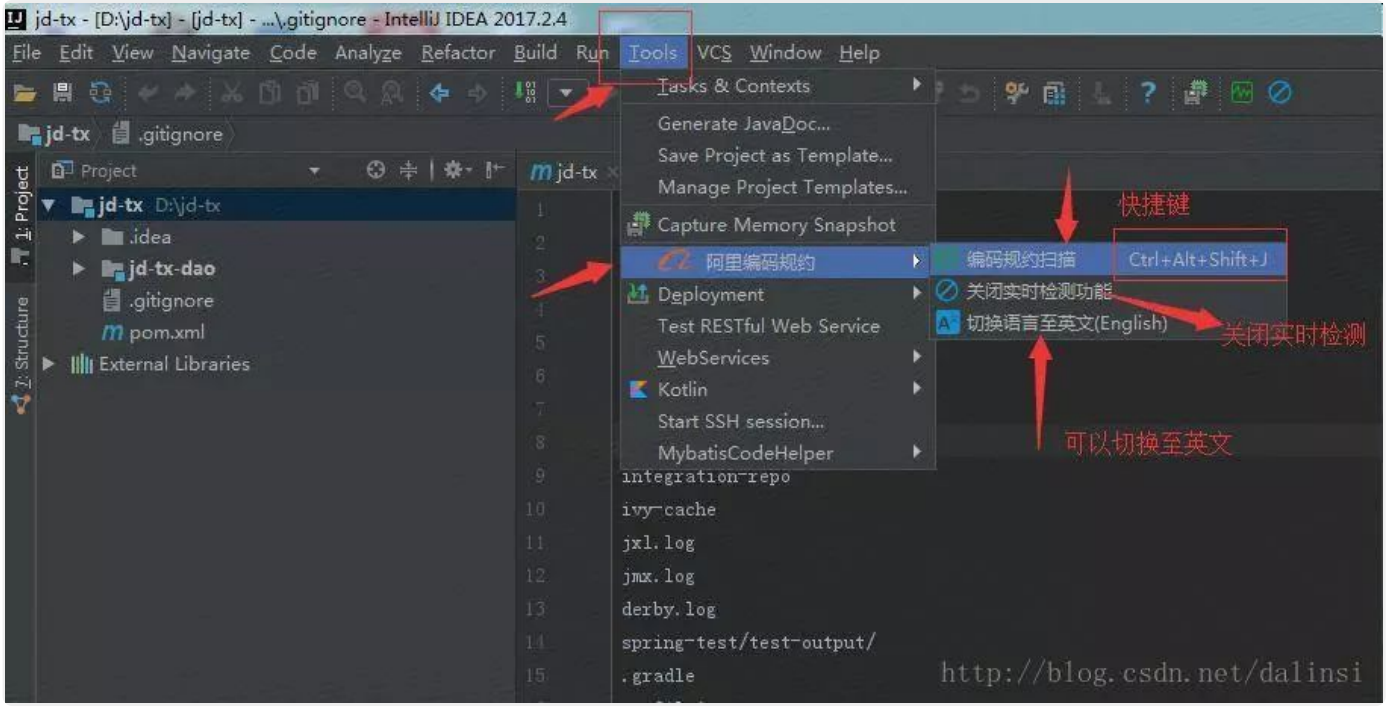
强大的字符串转换工具。使用快捷键，Alt+m。



- 切换样式(camelCase，hyphen-lowercase，HYPHEN-UPPERCASE，snakecase，SCREAMINGSNAKE_CASE，dot.case，words lowercase, Words Capitalized, PascalCase)
- 转换为SCREAMINGSNAKECASE (或转换为camelCase)
- 转换为 snake_case (或转换为camelCase)
- 转换为dot.case (或转换为camelCase)
- 转换为hyphen-case (或转换为camelCase)
- 转换为hyphen-case (或转换为snake_case)
- 转换为camelCase (或转换为Words)
- 转换为camelCase (或转换为lowercase words)
- 转换为PascalCase (或转换为camelCase)
- 选定文本大写
- 样式反转

7.Alibaba Java Coding Guidelines

阿里巴巴代码规范检查插件，当然规范可以参考《阿里巴巴Java开发手册》。



8.Lombok

Java语言，每次写实体类的时候都需要写一大堆的setter，getter，如果bean中的属性一旦有修改、删除或增加时，需要重新生成或删除get/set等方法，给代码维护增加负担，这也是Java被诟病的一种原因。Lombok则为我们解决了这些问题，使用了lombok的注解(@Setter,@Getter,@ToString,@@RequiredArgsConstructor,@EqualsAndHashCode或@Data)之后，就不需要编写或生成get/set等方法，很大程度上减少了代码量，而且减少了代码维护的负担。欢迎关注公众号 Java后端 获取更多推送。

安装完成之后，在应用Lombok的时候注意别忘了需要添加依，maven为例：

```
<dependency>
  <groupId>org.projectlombok</groupId
>
  <artifactId>lombok</artifactId
>
</dependency>
```

```
@Setter
@Getter
@ToString
@EqualsAndHashCode
public class People {
    private String name;
    private int age;
    private String male;
}
```

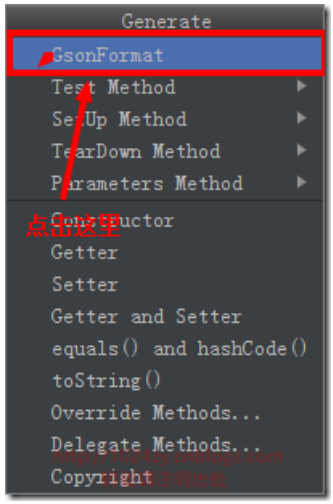
9.Key promoter

Key promoter 是IntelliJ IDEA的快捷键提示插件，会统计你鼠标点击某个功能的次数，提示你应该用什么快捷键，帮助记忆快捷键，等熟悉了之后可以关闭掉这个插件。

10.Gsonformat

可根据json数据快速生成java实体类。

自定义个javaBean(无任何内容，就一个空的类)，复制你要解析的Json，然后alt+insert弹出如下界面或者使用快捷键 Alt+S，在里面粘贴刚刚复制的Json，点击OK即可。

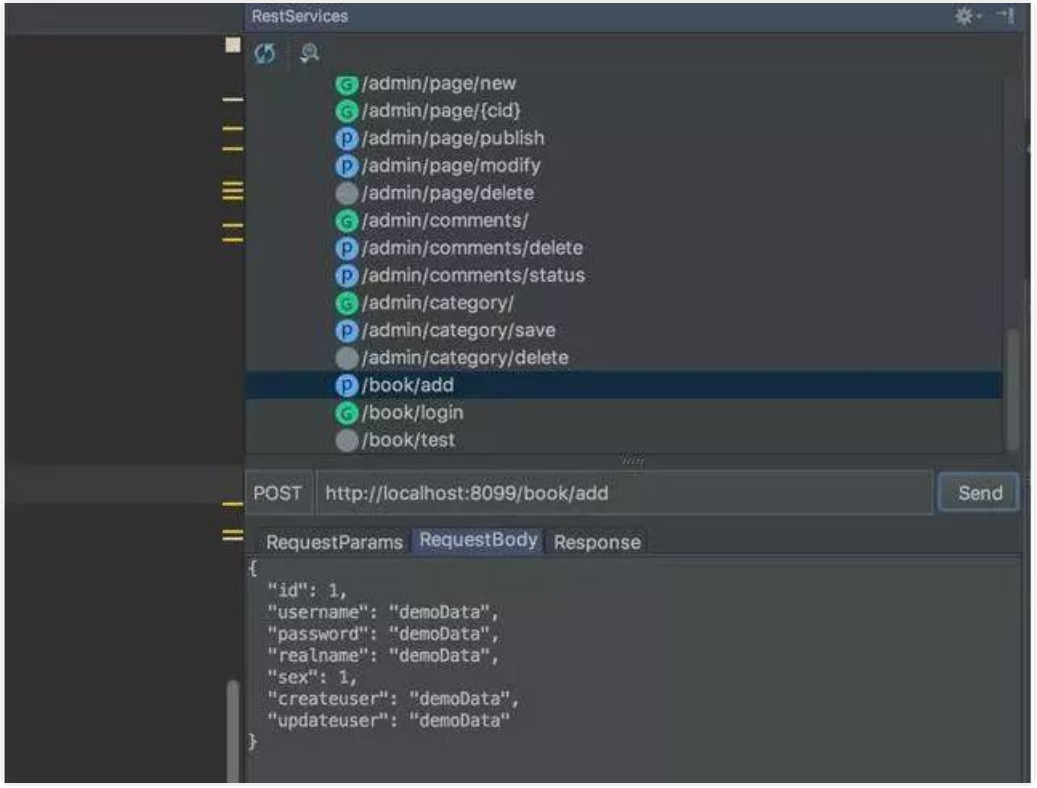


11.Restfultookit

Spring MVC网页开发的时候，我们都是通过requestmapping的方式来定义页面的URL地址的，为了找到这个地址我们一般都是cmd+shift+F的方式进行查找，大家都知道，我们URL的命名一个是类requestmapping+方法requestmapping，查找的时候还是有那么一点不方便的，restfultookit就能很方便的帮忙进行查找。欢迎关注公众号 Java后端 获取更多推送。

例如：我要找到/user/add 对应的controller,那么只要Ctrl+斜杠，(图片来自于网络)

就能直接定位到我们想要的controller。这个也是真心方便，当然restfultookit还为我们提供的其他的功能。根据我们的controller帮我们生成默认的测试数据，还能直接调用测试，这个可以是解决了我们每次postman调试数据时，自己傻傻的组装数据的的操作，这个更加清晰，比在console找数据包要方便多了。欢迎关注公众号 Java后端 获取更多推送。



12.JRebel

JRebel是一种热部署生产力工具，修改代码后不用重新启动程序，所有的更改便可以生效。它跳过了Java开发中常见的重建、重新启动和重新部署周期。

更多插件推荐

插件名称	插件介绍	官网地址
Gitee	开源中国的码云插件	https://plugins.jetbrains.com/plugin/8383-gitee
Alibaba Java Coding Guidelines	阿里巴巴出的代码规范检查插件	https://plugins.jetbrains.com/plugin/10046-alibaba-java-coding-guidelines
IDE Features Trainer	IntelliJ IDEA 官方出的学习辅助插件	https://plugins.jetbrains.com/plugin/8554?pr=idea
Key promoter	快捷键提示	https://plugins.jetbrains.com/plugin/4455?pr=idea
Grep Console	自定义设置控制台输出颜色	https://plugins.jetbrains.com/idea/plugin/7125-grep-console
String Manipulation	驼峰式命名和下划线命名交替变化	https://plugins.jetbrains.com/plugin/2162?pr=idea
CheckStyle-IDEA	代码规范检查	https://plugins.jetbrains.com/plugin/1065?pr=idea
FindBugs-IDEA	潜在 Bug 检查	https://plugins.jetbrains.com/plugin/3847?pr=idea
MetricsReloaded	代码复杂度检查	https://plugins.jetbrains.com/plugin/93?pr=idea
Statistic	代码统计	https://plugins.jetbrains.com/plugin/4509?pr=idea

FindBugs-IDEA	潜在 Bug 检查	https://plugins.jetbrains.com/plugin/3847?pr=idea
MetricsReloaded	代码复杂度检查	https://plugins.jetbrains.com/plugin/93?pr=idea
Statistic	代码统计	https://plugins.jetbrains.com/plugin/4509?pr=idea
JRebel Plugin	热部署	https://plugins.jetbrains.com/plugin/?id=4441
CodeGlance	在编辑代码最右侧，显示一块代码小地图	https://plugins.jetbrains.com/plugin/7275?pr=idea
GsonFormat	把 JSON 字符串直接实例化成类	https://plugins.jetbrains.com/plugin/7654?pr=idea
Markdown Navigator	书写 Markdown 文章	https://plugins.jetbrains.com/plugin/7896?pr=idea
Eclipse Code Formatter	使用 Eclipse 的代码格式化风格，在一个团队中如果公司有规定格式化风格，这个可以使用。	https://plugins.jetbrains.com/plugin/6546?pr=idea
Jindent-Source Code Formatter	自定义类、方法、doc、变量注释模板	http://plugins.jetbrains.com/plugin/2170?pr=idea
Translation	翻译插件	https://github.com/YiiGuxing/TranslationPlugin
Maven Helper	Maven 辅助插件	https://plugins.jetbrains.com/plugin/7179-maven-helper
Properties to YAML Converter	把 Properties 的配置格式改为 YAML 格式	https://plugins.jetbrains.com/plugin/8000-properties-to-yaml-converter
Git Flow Integration	Git Flow 的图形界面操作	https://plugins.jetbrains.com/plugin/7315-git-flow-integration
Rainbow Brackets	对各个对称括号进行着色，方便查看	https://github.com/izhangzhihao/intellij-rainbow-brackets
MybatisX	mybatis 框架辅助（免费）	https://plugins.jetbrains.com/plugin/10119-mybatisx
Lombok Plugin	Lombok 功能辅助插件	https://plugins.jetbrains.com/plugin/6317-lombok-plugin
.ignore	各类版本控制忽略文件生成工具	https://plugins.jetbrains.com/plugin/7495--ignore
mongo4idea	mongo客户端	https://github.com/dboissier/mongo4idea
iedis	redis客户端	https://plugins.jetbrains.com/plugin/9228-iedis
GenerateAllSetter	new POJO类的快速生成 set 方法	https://plugins.jetbrains.com/plugin/9360-generateallsetter

作者 | jajian

来源 | www.cnblogs.com/jajian/p/8081658.html

-END-

推荐阅读

1. Java后端优质文章整理
2. 彻底理解 Cookie，Session，Token
3. 这 26 条，你赞同几个？
4. 7 个开源的 Spring Boot 前后端分离项目
5. 如何设计 API 接口，实现统一格式返回？

Java后端

长按识别二维码，关注我的公众号

喜欢文章，点个在看

阅读原文

还没抢到票吗？实测两款 GitHub 开源抢票插件，所有坑我们都帮你踩过了

关注前沿科技 Java后端 1月1日

晓查 郭一璞 发自 凹非寺

量子位 报道 | 公众号 QbitAI

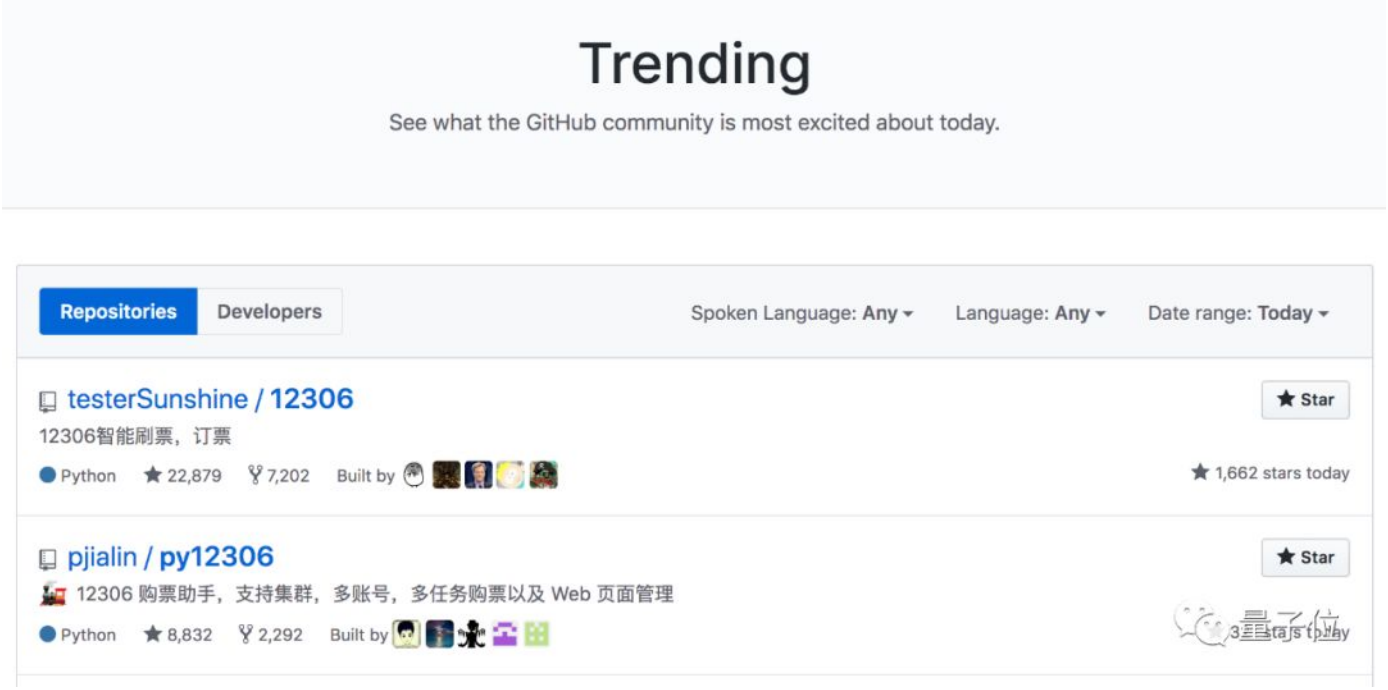
今天，已经可以抢过完年回来的火车票了。

如果你对自己手速和市面上的各种“加速包”都没什么信心的话，不妨试试用程序员的手段抢票？

况且，12306官方宣布屏蔽了一大批付费抢票软件，这也意味着你即使给这些软件付了会员费，也依旧抢不到票。

所以只能回到最初的手动抢票?No!No!No!

GitHub上有两个“年经”项目，每到春运前一段时间，就会定时定点的登顶热榜，助力一代又一代程序员和姿势丰富的非程序员们抢票回家。



那么，这两个项目究竟怎么用?好不好用?

和手动抢票、第三方软件抢票比起来优势是否明显?

安装使用过程中，会不会遇到不可名状的bug？

今天为大家一一横评。

霸占热榜的两个项目

排在GitHub热榜第1的项目名字简单粗暴，就直接叫**12306**，已经有21300星，来自ID为testerSunshine的作者。

testerSunshine的12306项目可以自动登录用户账户，能卡点抢票，也能捡漏候补，抢到票后会通过邮件进行通知。

排在热榜第2位的，叫做**py12306**，目前已有8300星，今年初就已经非常火了。

我该选哪个抢票软件

那么整体来看，这两款开源软件，哪个更好用呢？

py12306从去年开始就已经被大家广泛使用，其优点是设置比较简洁，甚至还有图形界面。但是它在验证码登录过程中容易出现下载失败的情况，如果把打码接口从free换成“若快”平台，还需要充值。

如果你对运行代码一无所知，建议使用py12306.

testerSunshine的12306插件设置比较复杂，自动获取cookie经常失败，因此需要手动填写，这一点比不上py12306；但是它自带了开发者的12306图片识别模型，我们可以在本地完成图片识别码的登录过程。

从实际使用效果来看，我们更推荐testerSunshine的12306，目前它在GitHub上的高人气，也反映的用户们的选择。

测评细节&使用教程

两个抢票插件的关键都在配置文件的修改：testerSunshine/12306的关键在于TickerConfig.py，而py12306的关键在于env.py。

热榜第一：12306

我们先开始测试现在趋势榜第一的12306，先将项目复制到本地：

```
git clone https://github.com/testerSunshine/12306
cd 12306
```

再安装依赖项，为了避免多python环境产生问题，建议以root用户方式安装：

```
pip3 install -i https://pypi.tuna.tsinghua.edu.cn/simple -r requirements.txt
```

下面到了整个过程**最关键**的环节，配置TickerConfig.py文件，这一步将决定你的成败！非常重要！

看到配置文件里一长串字符，你是不是觉得很头疼呢？我们经过测试，找到了其中的几个关键点：

1、如果你没有抢到票，寄希望于其他人退票后捡漏，令TICKET_TYPE = 2，否则设置为1；

```
# 刷票模式：1=刷票 2=候补+刷票
TICKET_TYPE = 1
```

2、**STATION_TRAINS**可以填入一串你想要抢的车次，比如北京到合肥方向，你想购买G267、G29次列车，就设置为STATION_TRAINS = [“G267”, “G29”]，中间用逗号隔开，不填写等于抢当日全部车次；

```
# 出发日期(list) "2018-01-06", "2018-01-07"
STATION_DATES = [
   "2020-01-30"
]

# 填入需要购买的车次(list)，"G1353"
# 修改车次填入规则，注：(以前设置的车次逻辑不变)，如果车次填入为空，那么就是当日乘车所有车次都纳入筛选返回
# 不填车次是整个list为空才算，如果不是为空，依然会判断车次的，这种是错误的写法 [""], 正确的写法 []
STATION_TRAINS = ["G267","G29"]
```

3、**STATION_DATES**填入你出发的日期，这一步不难；

4、出发站FROM_STATION和到达站TO_STATION不必精确到具体的站，只需填入城市，除非你想从特定站点出发；

```
# 出发城市，比如深圳北，就填深圳就搜得到
FROM_STATION = "北京"

# 到达城市 比如深圳北，就填深圳就搜得到
TO_STATION = "合肥"
```

5、乘车人填入你12306账号中常用联系人的姓名，比如TICKET_PEOPLES = [“张三”]，如果没有，需要登录12306手动添加；

```
# 乘车人(list) 多个乘车人ex:
# "张三",
# "李四"
TICKET_PEOPLES = ["李雷","韩梅梅"]
```

6、填入你的12306账户名和密码；

```
# 12306登录账号
USER = "Lilei"
PWD = "123456"
```

7、如果不需要邮箱和Server酱提醒，请把EMAIL_CONF和SERVER_CHAN_CONF的第一项都设置为False；

8、开放抢票时间根据自己车次填入，可以提前几秒，比如下午一点开发抢票，则填入OPEN_TIME = “12:29:57”；

```
# 预售放票时间, 如果是捡漏模式，可以忽略此操作
OPEN_TIME = "12:29:57"
```

9、cookie设置这一步尤为关键，经过我们实测，COOKIE_TYPE设置为1或2都有些问题，建议设置为3；

```
# 1=使用selenium获取devicesID
# 2=使用网页端/otn/HttpZF/logdevice获取devicesId，这个接口的算法目前可能有点问题，如果登录一直302的请改为配置1
# 3=自己打开浏览器在headers-Cookies中抓取RAIL_DEVICEID和RAIL_EXPIRATION，这个就不用配置selenium
COOKIE_TYPE = 3
```

10、上一步设置为3以后，还需要手动设置后面的两个参数**RAIL_EXPIRATION**和**RAIL_DEVICEID**。

```
# 如果COOKIE_TYPE=3, 则需配置RAIL_EXPIRATION、RAIL_DEVICEID的值
RAIL_EXPIRATION = "1577327361278"
RAIL_DEVICEID = "jFAKbAeAInPnm0pAdqdCNG0ID_dU6SW6L8gaX7zDEewcWVJC5w7nTSw63oMK9sd9c6FcdhyDMsuVUV4aknfXw
# RAIL_EXPIRATION = "1577034103293"
# RAIL_DEVICEID = "CDno29Erc_Pf3FSXb4dzq-Op64EhWrsi5yUZKVIKR1MAfYo2qFlCeXD8VkexY7_1qg-ClV-fE8j9jgVlPZxRh3wVc2iq
```

如何找到这两个参数?先用Chrome浏览器打开12306.cn，点击网站地址左边的“锁形”图标：再点击下发的Cookie

接着会出现一组Cookie，选择来自12306.cn的Cookie：

看到**RAIL_EXPIRATION**和**RAIL_DEVICEID**两个参数，点击它，将内容里的一串字符复制到配置文件中。

需要注意的是RAIL_DEVICEID参数很长，可能显示不全，请完整复制。

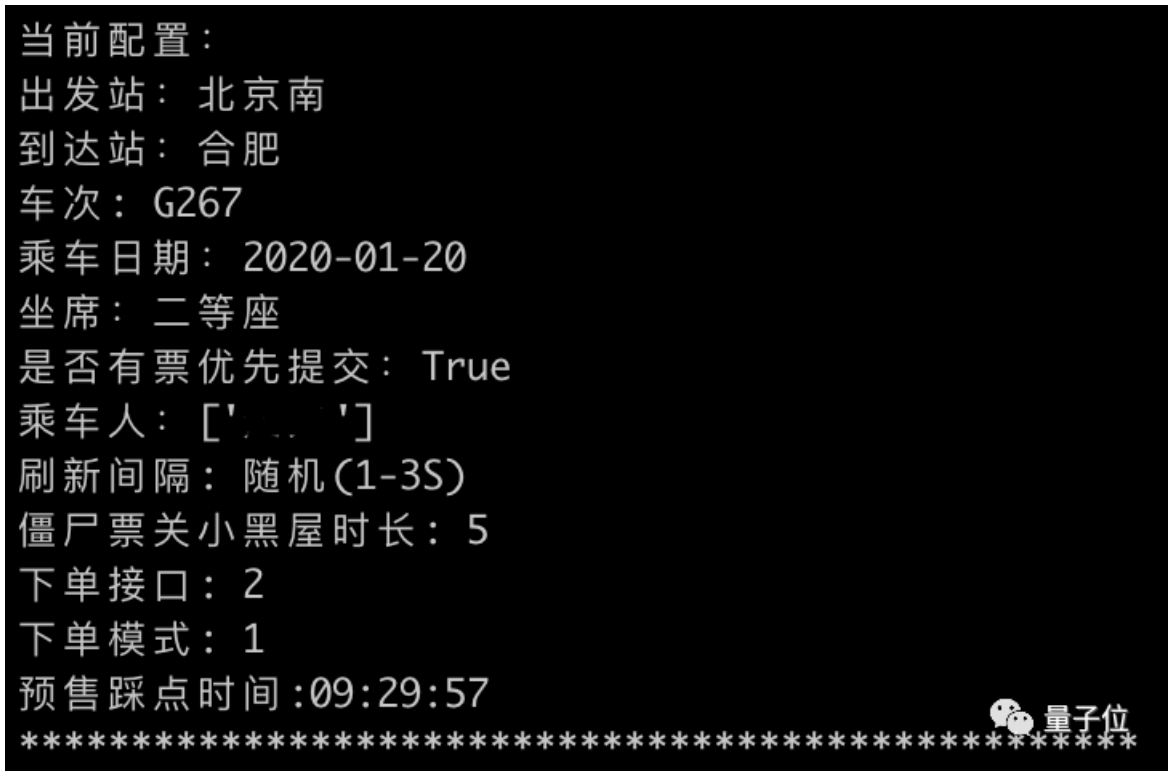
至此，准备工作已全部完成，启动前请先筛选cdn，这点很重要！

```
python3 run.py c
```

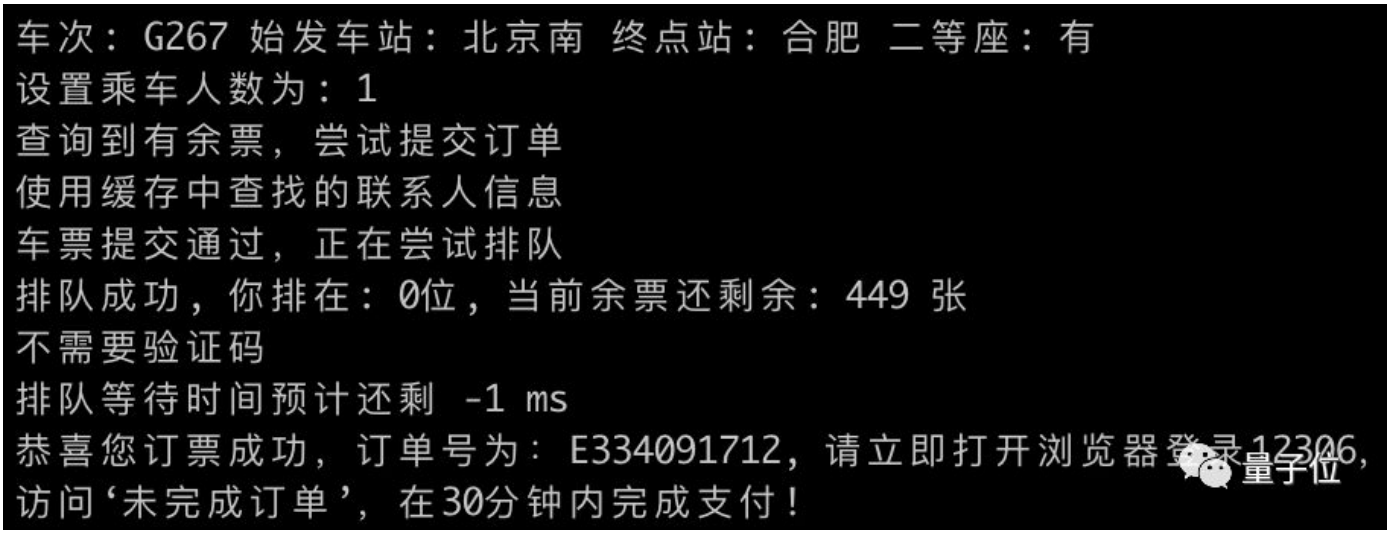
接着启动服务：

```
python3 run.py r
```

接着它会提出登录成功，并列出当日所有相关车次信息，然后开始抢票：



抢票成功后，这个软件不能帮你完成支付，你还需要在手机或者电脑上登录12306网站，在30分钟内完成支付动作，否则你辛辛苦苦抢的票就没了。



热榜第二：py12306

py12306的安装方式与前者类似：

```
git clone https://github.com/pjialin/py12306
cd py12306
pip3 install -r requirements.txt
```

然后修改配置文件，现更改后缀名

```
cp env.py.example env.py
```

默认配置文件中需要修改的选项有：

1、你的12306账户和密码；

```
# 12306 账号
USER_ACCOUNTS = [
  # 目前已支持仅查询，不下单，屏蔽掉下面的账号即可
  {
    'key': 'lilei', # 如使用多个账号 key 不能重复
    'user_name': 'Lilei',
    'password': '123456'
  },
```

2、查询任务QUERY_JOBS中的各项参数，包括出发日期left_dates、出发站left和到达站arrive、乘客姓名members、坐席seats、车次train_numbers；

```
# 查询任务
QUERY_JOBS = [
  {
    # 'job_name': 'bj -> yl', # 任务名称，不填默认会以车站名命名，不可重复
    'account_key': 0, # 将会使用指定账号下单
    'left_dates': [ # 出发日期 :Array
      "2020-01-22",
    ],
    'stations': { # 车站 支持多个车站同时查询 :Dict or :List
      'left': '北京',
      'arrive': '合肥',
    },
    'members': [ # 乘客姓名，会根据当前账号自动识别乘客类型 购买儿童票 设置两个相同的姓名即可，程序会自动识别 如 ['张三', '张三
      "李雷",
      "韩梅梅"#在姓名前加*表示学生购买成人票
      # 7, # 支持通过序号确定唯一乘客，序号查看可通过 python main.py -t 登录成功之后在 runtime/user/ 下找到对应的 用户名_pass
    ],
    'allow_less_member': 0, # 是否允许余票不足时提交部分乘客
    'seats': [ # 筛选座位 有先后顺序 :Array
      # 可用值: 特等座, 商务座, 一等座, 二等座, 软卧, 硬卧, 动卧, 软座, 硬座, 无座
      '二等座'
    ],
    'train_numbers': [ # 筛选车次 可以为空，为空则所有车次都可以提交 如 [] 注意大小写需要保持一致
      "G267",
    ],
    'except_train_numbers': [ # 筛选车次，排除车次 train_numbers 和 except_train_numbers 不可同时存在
    ],
    'period': { # 筛选时间
      'from': '00:00',
      'to': '24:00'
    }

  },
```

接着运行程序：

```
python3 main.py
```

如果你不善于使用命令行模式，还可以使用图形界面观察抢票任务。

将配置中的WEB_ENABLE打开，启动程序后访问当前主机地址+端口号 (默认 8008) 即可，然后在浏览器中输入 http://127.0.0.1:8008。

现在，就可以让程序帮你抢票啦。

不过，抢票软件并非万能，巧coder难为无票之炊，除了技术，你可能还需要一点点运气。

无论采取哪种交通方式，祝大家都能开开心心过年回家，平平安安回来搬砖~

2020已经开始，量子位也祝大家新年快乐^_^

传送门

testerSunshine/12306
https://github.com/testerSunshine/12306

py12306
https://github.com/pjialin/py12306

- END -

推荐阅读

1. 实战:SpringBoot & Restful API 构建示例
2. 关于 CPU 的一些基本知识总结
3. 发布没有答案的面试题，都是要流氓
4. 什么是一致性 Hash 算法?
5. 团队开发中 Git 最佳实践

喜欢文章，点个在看