# Gradle项目规范

## Gradle项目目录

|  |  |
| --- | --- |
| 目录 | 说明 |
| src/main/java | java源文件 |
| src/main/resources | 资源文件 |
| src/main/java | java测试用例源文件 |
| src/main/resources | 测试用例资源文件 |
| src/main/webapp | web资源文件 |
| src/main/resources/db/migration | 数据库脚本文件 |

项目编码格式：utf-8

## Gradle安装

下载地址：https://downloads.gradle.org/distributions/gradle-3.1-all.zip

网盘地址：X:\devtool\gradle-3.1-all.zip

安装略

## Eclipse上安装Gradle插件

在线安装gradle插件地址，如下

* https://builds.gradle.org/repository/download/Tooling\_Master\_IntegrationTests\_Linux\_Eclipse46Build/.lastSuccessful/update-site (latest 4.6 development snapshot)
* https://builds.gradle.org/repository/download/Tooling\_Master\_IntegrationTests\_Linux\_Eclipse45Build/.lastSuccessful/update-site (latest 4.5 development snapshot)
* https://builds.gradle.org/repository/download/Tooling\_Master\_IntegrationTests\_Linux\_Eclipse44Build/.lastSuccessful/update-site (latest 4.4 development snapshot)
* https://builds.gradle.org/repository/download/Tooling\_Master\_IntegrationTests\_Linux\_Eclipse43Build/.lastSuccessful/update-site (latest 4.3 development snapshot)
* https://builds.gradle.org/repository/download/Tooling\_Master\_IntegrationTests\_Linux\_Eclipse42Build/.lastSuccessful/update-site (latest 4.2 development snapshot)
* https://builds.gradle.org/repository/download/Tooling\_Master\_IntegrationTests\_Linux\_Eclipse37Build/.lastSuccessful/update-site (latest 3.7 development snapshot)
* https://builds.gradle.org/repository/download/Tooling\_Master\_IntegrationTests\_Linux\_Eclipse36Build/.lastSuccessful/update-site (latest 3.6 development snapshot)

## Gradle配置文件

### 模版工程

Gradle模版项目地址http://10.163.37.9/root/template-project，可参考模版工程配置。（针对内部员工通过你的电脑开机用户和密码即可登录，外部用户则需向项目经理统一报备，然后由项目经理统一发邮件申请。发送给liliangang@lifeisgreat.com.cn 和 sundan@lifeisgreat.com.cn）

### 设置私服仓库

为了减少访问外网的带宽，需设置内网maven私服库。

repositories {

maven{

url "http://10.163.66.81/nexus/content/groups/public/"

}

}

### 查找依赖包

查找依赖包网址

<http://mvnrepository.com/>

<http://search.maven.org/>

### 为项目添加插件

java项目需要添加java插件，javaweb项目还需添加war插件。

eclipse-wtp插件可以让当前项目适应eclipse项目规范。

添加flyway插件去管理项目数据库脚本升级。

buildscript {

repositories {

maven {

url "http://10.163.66.81/nexus/content/groups/public/"

}

}

dependencies {

classpath 'mysql:mysql-connector-java:5.1.40'

classpath "org.flywaydb:flyway-gradle-plugin:4.0.3"

classpath 'org.hidetake:gradle-ssh-plugin:2.7.0'

}

}

apply plugin: 'java'

apply plugin: 'war'

apply plugin: 'eclipse-wtp'

apply plugin: "org.flywaydb.flyway"

apply plugin: "org.hidetake.ssh"

### 添加依赖包

在dependecies的compile节点下添加项目编译需要的jar包。

在dependecies的test节点下添加项目单元测试需要的jar包。

dependencies {

compile (

'org.springframework:spring-webmvc:4.3.3.RELEASE',

'org.springframework:spring-web:4.3.3.RELEASE',

'org.springframework:spring-tx:4.3.3.RELEASE',

'org.springframework:spring-jdbc:4.3.3.RELEASE',

'org.springframework:spring-context-support:4.3.3.RELEASE',

'org.mybatis:mybatis:3.4.1',

'org.mybatis:mybatis-spring:1.3.0',

'org.apache.shiro:shiro-all:1.3.2',

'com.alibaba:fastjson:1.2.20',

'com.fasterxml.jackson.core:jackson-databind:2.8.4',

'org.apache.logging.log4j:log4j-core:2.7',

'org.apache.logging.log4j:log4j-api:2.7',

'org.apache.logging.log4j:log4j-web:2.7',

'org.apache.logging.log4j:log4j-slf4j-impl:2.7',

'mysql:mysql-connector-java:5.1.40',

'org.aspectj:aspectjweaver:1.8.9',

'org.apache.tomcat:tomcat-jdbc:8.5.6'

)

testCompile(

'junit:junit:4.12'

)

}

## Gradle常用命令

Gradle命令主要格式是gradle 任务名 任务名，可以多个任务名一起执行。比如gradle clean build，该命令的意思是先清理项目编译目录再编译。以下是常用gradle 命令使用方法。

### tasks

查看项目下可用的gradle任务task

$ gradle tasks

### build

编译项目

$ gradle build

### clean

清理编译目录的文件

$ gradle clean

### jar

打jar包

$ gradle jar

### war

打war包命令

$ gradle war

### test

单元测试命令

$ gradle test

# 数据库脚本规范

## 数据库脚本规范

### Sql脚本所在目录

sql脚本存放目录：src/main/resources/db/migration

sql脚本文件编码格式：utf-8

### Sql脚本文件命名

Sql脚本文件命名规范：**V{大版本号}\_{小版本号}\_{描述action}\_{表}.sql**

**Sql脚本文件版本号不能重复，并且需要连续的。如遇到多人同时提交相同版本号，需修改文件版本号后再提交。**

例如 V1\_1\_create\_user.sql

V1\_2\_insert\_user.sql

V1\_3\_alter\_user.sql

### Sql脚本内容

对于每个Sql脚本，flyway是按照一个脚本文件一个事务。并且脚本内容需要满足如下条件。

* Sql脚本内容需要遵循标准的Sql语法规范。如果有多个Sql，则每条sql后必须添加分号。
* 提交脚本时必须确保脚本内容能正确执行不报错。
* 由于Sql脚本是自动化执行，故而Sql脚本中不能出现需要用户确认输入消息等内容。
* 由于大部分数据库不支持DDL语言事务，故而尽量将DDL语言和DML语言分开两个脚本文件编写，除非你的脚本文件完全正确。因为当DMLsql失败时，同一个文件的DDLsql语句将可能需要手工回滚。

## Gradle下flyway命令使用

Gradle的flyway插件提供了六个常用命令：baseline、migrate、repair、info、validate、clean。以下是六个命令的使用方法。

### baseline

以当前数据库作为基线。如果当前数据库不为空（空指的是没有创建过任何数据库对象，比如表、视图、存储过程等），则必须需要建立一个基线。从已经存在的数据库对象为出发点，开始使用flyway脚本管理。

$ gradle flywayBaseline

### migrate

迁移升级数据库

$ gradle flywayMigrate

### repair

修复数据库，可能会有如下几个影响：

* 将会删除数据库版本表的失败升级记录。
* 修复数据库脚本文件的校验码checksum

$ gradle flywayRepair

### info

查看所有脚本文件信息

$ gradle flywayInfo

### validate

校验数据库结构和与本地迁移脚本是否一致

$ gradle flywayValidate

### clean

清空数据库，将删除所有数据库对象（表、视图、存储过程、定时器等），慎用

$ gradle flywayClean