## JetBrains全家桶

### 全家桶激活过程

适用的版本为百度网盘/软件/JetBrains全家桶。

激活过程记录如下所示：

1、下载软件安装包

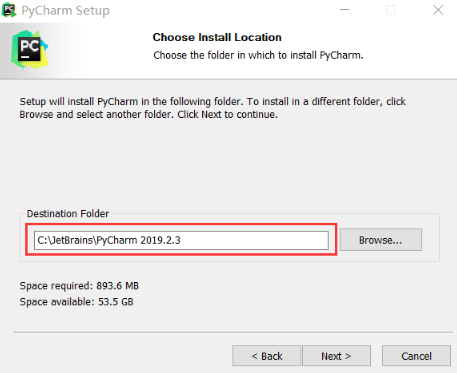
2、下载破解补丁

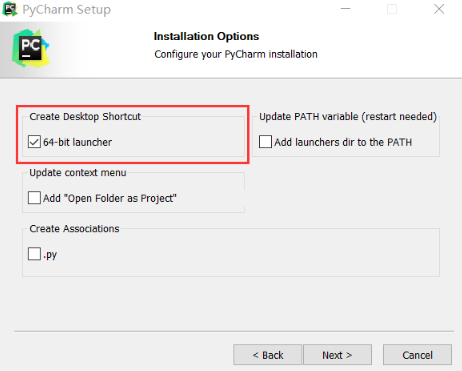
3、修改 hosts 文件，添加如下两行代码：

|  |
| --- |
| # JetBrains 配置  0.0.0.0 account.jetbrains.com  0.0.0.0 www.jetbrains.com |

4、安装软件包

比如我把 PyCharm 安装到 C:\JetBrains\PyCharm 2019.2.3 目录下：





5、激活

1. 先把 jetbrains-agent.jar 复制到 C:\JetBrains 目录下，方便全家桶所有软件共同引用。
2. 然后打开 C:\JetBrains\软件\bin 目录下的 xxx.exe.vmoptions 和 xxx64.exe.vmoptions 文件，添加如下代码：

|  |
| --- |
| -javaagent:C:\JetBrains\jetbrains-agent.jar |

不论是全家桶的哪款软件，都是 -javaagent 后面跟上破解的 jar 包的路径。

1. 激活方法：下面是两种永久激活的方式：
2. 、启动软件 → Help → Register → License Server → http://jetbrains-license-server → Active 即可，该方式授权后，没有显示到期日期。
3. 选择 Activation code 方式， https://zhile.io/custom-license.html https://zhile.io/custom/license 需要登录 GitHub 账号并授权给 zhile 即可，该方式授权后，过期日期为 2089 年。
4. 查看软件激活状态和过期时间：→ Help → About。

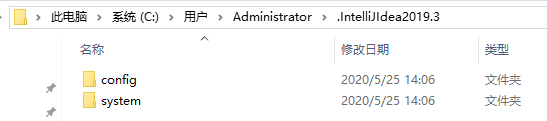
激活参考：

https://www.cnblogs.com/ccav1/p/11713114.html

<https://zhile.io/2018/08/17/jetbrains-license-server-crack.html>

### 修改 C 盘中默认配置文件位置

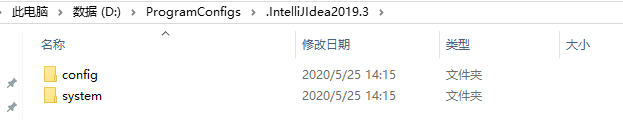
默认配置位置为：C:\Users\Administrator\.IntelliJIdea2019.3，如下图所示：



打开 C:\JetBrains\IntelliJ\bin\idea.properties 修改如下两个配置：

|  |
| --- |
| # 修改如下两个参数  idea.config.path=D:/ProgramConfigs/.IntelliJIdea2019.3/config  idea.system.path=D:/ProgramConfigs/.IntelliJIdea2019.3/system |

并把环境配置的文件夹复制到对应的位置，如下图所示：



然后重启 IDE 即可。

### 连接MySQL数据库时出现关时区问题

出现的问题如下描述：

Server returns invalid timezone. Go to ‘Advanced’ tab and set ‘serverTimezone’ property manually.

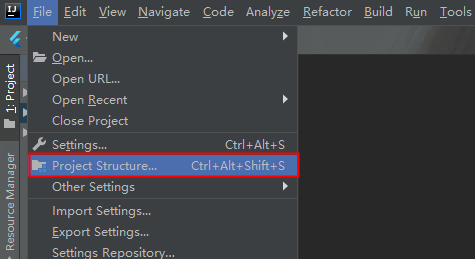
解决办法如下：

找到 Advanced 选项卡 → 找到名字为 serverTimezone 的一栏 → 输入 UTC → 点击 Apply 保存即可。

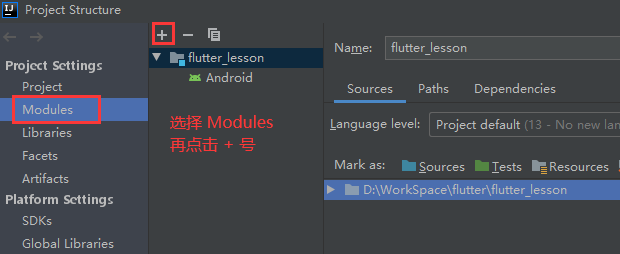
### 在一个 project 界面同时打开多个项目

这里，我们以Flutter项目为例说明：

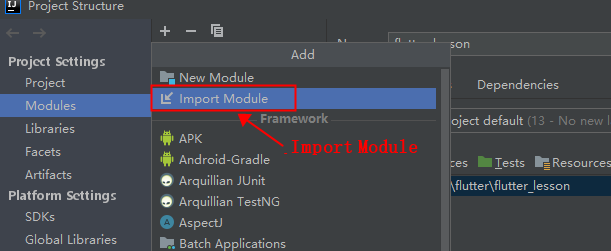
打开 IDE 之后，选择 File → Project Structure，如下图所示：



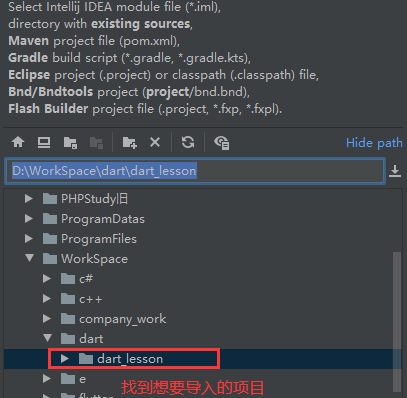
然后，做如下操作：

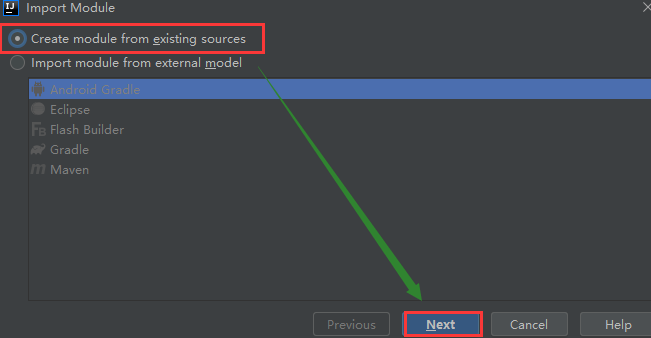


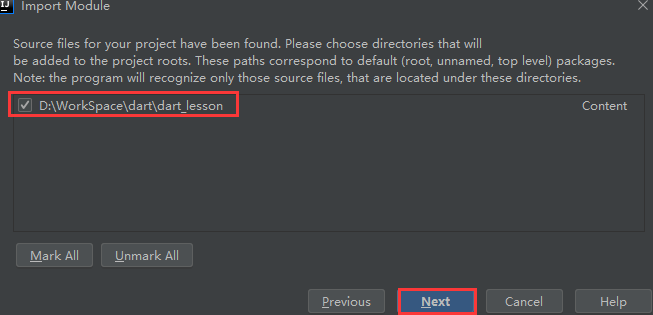
然后选择 Import Module，如下图所示：

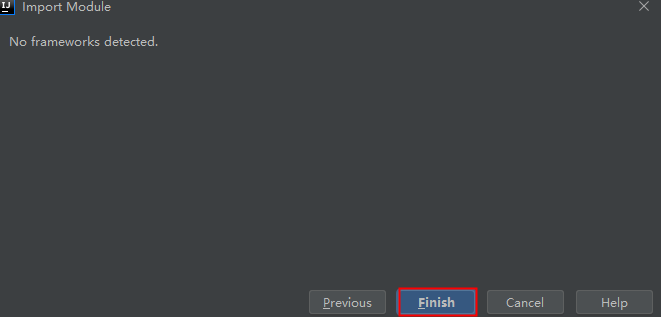


找到想要导入的项目，如下图：

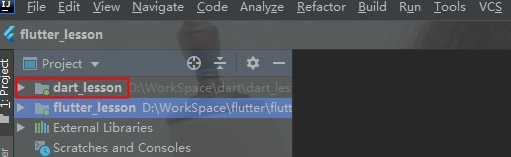




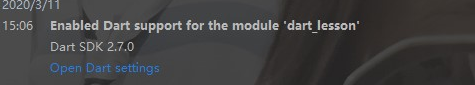




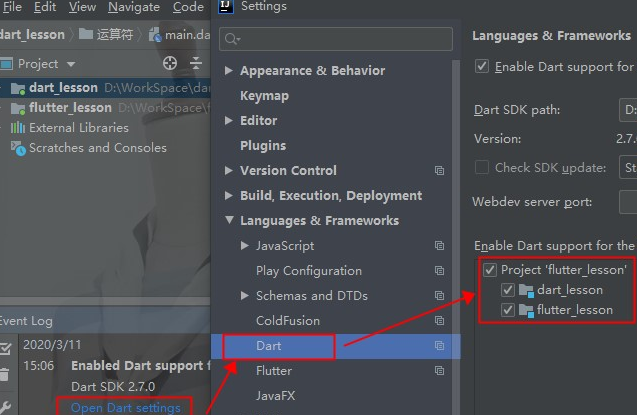
完成导入之后，发现多了一个我们导入的项目了，如下图所示：



我们打开新导入的项目的程序文件，这时候会弹出提示，大意如下图所示：



意思是给 dart\_lesson 模块激活 Dart 支持，查看是否激活：



激活支持之后，才能正常运行 Dart 程序，否则右键点击时，是没有运行功能的。

## IntelliJ IDEA

### 通用配置

#### 设置背景图

File → Settings → Appearance & Behavior → Appearance → UI Options → BACKGROUND IMAGE…… → Image 选择图片即可。

#### 设置波浪线

Settings → Editor → Color Scheme → General → Scheme：Default →

* Errors and Warnings
  + Weak Warning：Effects 去掉√
  + Typo：Effects 去掉√
* Hyperlinks
  + Followed：Effects 去掉√
  + Inactive：Effects 去掉√
  + Reference：Effects 去掉√
  + Unfollowed：Effects 去掉√

#### 使用主题

1、Settings → Appearance & Behavior → Appearance → Theme → Material Oceanic

2、Settings → Editor → Color Scheme → General → Scheme → Material Oceanic

3、Settings → Appearance & Behavior → Material Theme

#### 设置主题

Settings → Appearance & Behavior → Appearance → Theme → IntelliJ

#### 设置主题中HTML注释文字的颜色

Settings → Editor → Color Scheme → HTML → Comment → Inherit values from √ 去掉 → Foreground → 选择需要的颜色。

#### 设置文字的字体和大小

Settings → Editor → Font → Font ：Courier New | Size：18

#### 设置文件编码

Settings → Editor → File Encoding ：Global Encoding：UTF-8

Project Encoding：UTF-8

Default encoding for properties files：UTF-8

### 配置Python开发环境

#### 安装Python插件

|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| Python | JetBrains |
|  |  |
|  |  |
|  |  |

#### 导入Python项目

File → Project Structure → Project Settings → Modules

#### 配置Python项目虚拟环境

右键单击项目 → Open Module Settings → Project Settings → Modules → 选择项目 → Dependencies → Module SDK

#### 配置Python文件模板

File → Settings → Editor → File and Code Templates → Files选项卡 → Python Script，填写如下模板代码：

|  |
| --- |
| # -\*- coding:utf-8 -\*-  # @ProjectName: ${PROJECT\_NAME}  # @Email : 276517382@qq.com  # @FileName : ${NAME}.py  # @DATETime : ${DATE} ${TIME}  # @Author : 笑看风云 |

常见的预定义变量：

* ${PROJECT\_NAME} 当前项目的名称。
* ${NAME} 在文件创建过程中在“新建文件”对话框中指定的新文件的名称。
* ${USER} 当前用户的登录名。
* ${DATE} 当前的系统日期。
* ${TIME} 当前系统时间。
* ${YEAR} 今年。
* ${MONTH} 当月。
* ${DAY} 当月的当天。
* ${HOUR} 目前的小时。
* ${MINUTE} 当前分钟。
* ${PRODUCT\_NAME} 将在其中创建文件的IDE的名称

### 配置Go开发环境

#### 安装Go插件

|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| Go | JetBrains |
| Go Template | JetBrains |
|  |  |
|  |  |

#### 配置Go文件模板

File → Settings → Editor → File and Code Templates → Files选项卡 → Go Application、Go Scratch、Go File，填写如下模板代码：

|  |
| --- |
| // -\*- coding:utf-8 -\*-  // @ProjectName: ${PROJECT\_NAME}  // @Email : 276517382@qq.com  // @FileName : ${NAME}.py  // @DATETime : ${DATE} ${TIME}  // @Author : 笑看风云 |

### 安装PHP插件

|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| PHP | JetBrains |
| PHP Annotations | espend\_de |
| PHP composer.json support | psliwa |
| Laravel | Daniel Espendiller |
| Blade | JetBrains |
| LaravelStorm | SmartBit8 |
|  |  |

### 安装Flutter插件

|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| Dart | JetBrains |
| Flutter | flutter.dev |
|  |  |
|  |  |

### 安装Lua插件

|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| Lua | sylvanaar |
|  |  |
|  |  |
|  |  |

### 安装NodeJS插件

|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| NUnitJS | Fabio Montanari |
|  |  |
|  |  |
|  |  |

### 安装Vue插件

|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| Vue.js | JetBrains |
| Vue Componer Creator | Fabio Gianini |
| IntelliVue | TwelveTone LLC |
|  |  |

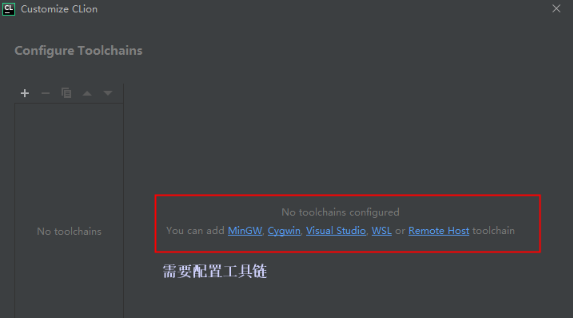
### 安装Julia插件

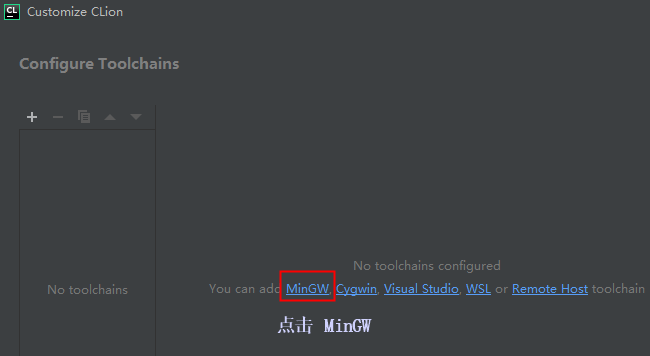
|  |  |
| --- | --- |
| 插件名 | 插件作者 |
| Julia | ice1000 |
|  |  |

## CLion

### CLion配置

第一次启动 CLion 的时候，弹出如下对话框：







点击 Download 之后，浏览器自动打开<http://mingw-w64.org/doku.php/download/mingw-builds> 页面，如下图所示：



下载后的文件为 mingw-w64-install.exe 安装软件，由于安装过程太费时，所以我们下载压缩包更省时。

#### 下载 MinGW 压缩包

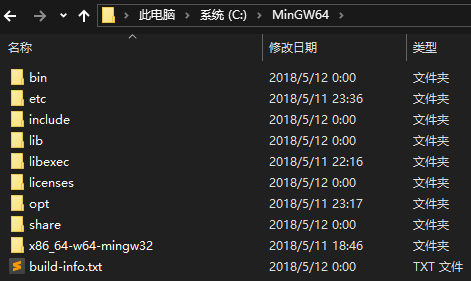
打开 <https://sourceforge.net/projects/mingw-w64/files/> 找到如下内容：



下载后，是一个 x86\_64-8.1.0-release-posix-sjlj-rt\_v6-rev0.7z 压缩包，我们把这个压缩包解压缩到如下目录：

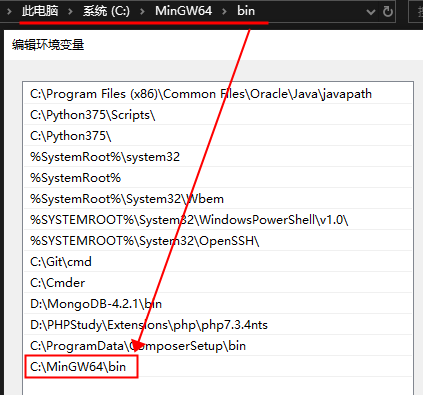
|  |
| --- |
| C:\MinGW64 |

目录结构如下截图：



#### 配置 MinGW 环境变量

我们需要把 bin 目录配置到系统的 path 环境变量中，如下图所示：



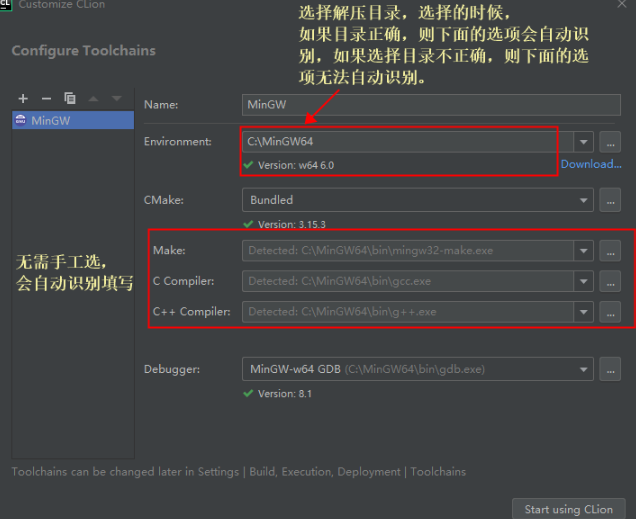
#### 验证 MinGW 是否配置正确

环境变量配置完成之后，我们需要验证配置是否正确，我们在任意位置（桌面）打开cmd命令行，输入如下命令：

|  |
| --- |
| C:\Users\Administrator\Desktop  λ gcc -v  Using built-in specs.  COLLECT\_GCC=gcc  COLLECT\_LTO\_WRAPPER=C:/MinGW64/bin/../libexec/gcc/x86\_64-w64-mingw32/8.1.0/lto-wrapper.exe  Target: x86\_64-w64-mingw32  Configured with: ../../../src/gcc-8.1.0/configure --host=x86\_64-w64-mingw32 --build=x86\_64-w64-mingw32 --target=x86\_64-w64-mingw32 --prefix=/mingw64 --with-sysroot=/c/mingw810/x86\_64-810-posix-sjlj-rt\_v6-rev0/mingw64 --enable-shared --enable-static --enable-targets=all --enable-multilib --enable-languages=c,c++,fortran,lto --enable-libstdcxx-time=yes --enable-threads=posix --enable-libgomp --enable-libatomic --enable-lto --enable-graphite --enable-checking=release --enable-fully-dynamic-string --enable-version-specific-runtime-libs --enable-sjlj-exceptions --disable-libstdcxx-pch --disable-libstdcxx-debug --enable-bootstrap --disable-rpath --disable-win32-registry --disable-nls --disable-werror --disable-symvers --with-gnu-as --with-gnu-ld --with-arch-32=i686 --with-arch-64=nocona --with-tune-32=generic --with-tune-64=core2 --with-libiconv --with-system-zlib --with-gmp=/c/mingw810/prerequisites/x86\_64-w64-mingw32-static --with-mpfr=/c/mingw810/prerequisites/x86\_64-w64-mingw32-static --with-mpc=/c/mingw810/prerequisites/x86\_64-w64-mingw32-static --with-isl=/c/mingw810/prerequisites/x86\_64-w64-mingw32-static --with-pkgversion='x86\_64-posix-sjlj-rev0, Built by MinGW-W64 project' --with-bugurl=https://sourceforge.net/projects/mingw-w64 CFLAGS='-O2 -pipe -fno-ident -I/c/mingw810/x86\_64-810-posix-sjlj-rt\_v6-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86\_64-zlib-static/include -I/c/mingw810/prerequisites/x86\_64-w64-mingw32-static/include' CXXFLAGS='-O2 -pipe -fno-ident -I/c/mingw810/x86\_64-810-posix-sjlj-rt\_v6-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86\_64-zlib-static/include -I/c/mingw810/prerequisites/x86\_64-w64-mingw32-static/include' CPPFLAGS=' -I/c/mingw810/x86\_64-810-posix-sjlj-rt\_v6-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86\_64-zlib-static/include -I/c/mingw810/prerequisites/x86\_64-w64-mingw32-static/include' LDFLAGS='-pipe -fno-ident -L/c/mingw810/x86\_64-810-posix-sjlj-rt\_v6-rev0/mingw64/opt/lib -L/c/mingw810/prerequisites/x86\_64-zlib-static/lib -L/c/mingw810/prerequisites/x86\_64-w64-mingw32-static/lib '  Thread model: posix  gcc version 8.1.0 (x86\_64-posix-sjlj-rev0, Built by MinGW-W64 project)  C:\Users\Administrator\Desktop  λ |

说明我们的配置没有问题。

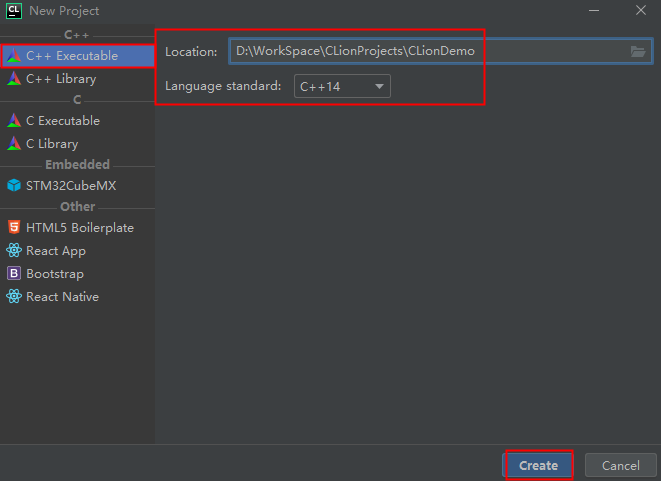
安装完 MinGW 之后，我们继续配置 CLion 如下图所示：

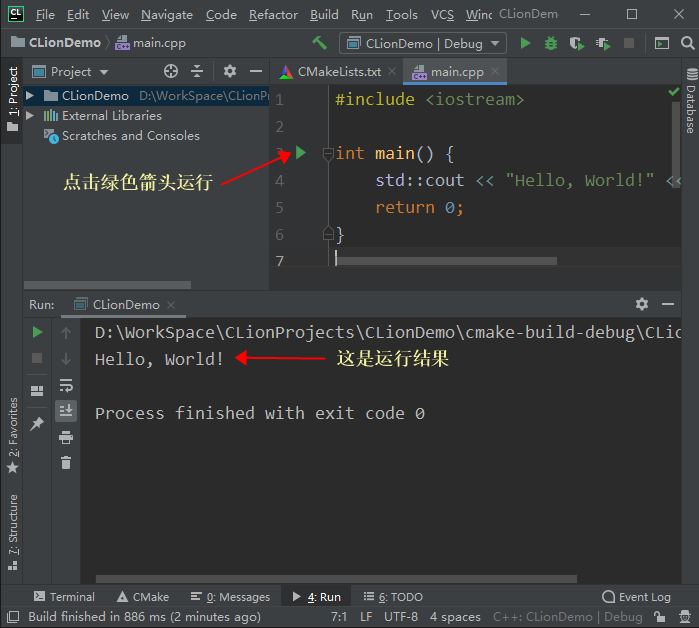


配置完成之后，点击 Start using CLion即可。

### 创建项目

过程如下图所示：





## PyCharm

### PyCharm使用GitHub

步骤一：

File → Settings → Version Control → GitHub → Add account:

Server: github.com

Login: 276517382@qq.com

Password: 密码

Connection timeout: 5000 seconds

用户名: zhenglushan

步骤二：

File → Settings → Version Control → Git:

Path to Git executable: C:\Git\bin\git.exe

步骤三：

选中项目 → VCS → Import into Version Control → Share Project on GitHub → Repository name: 项目名称 → Private 选中(一定要选中 Private 私有)

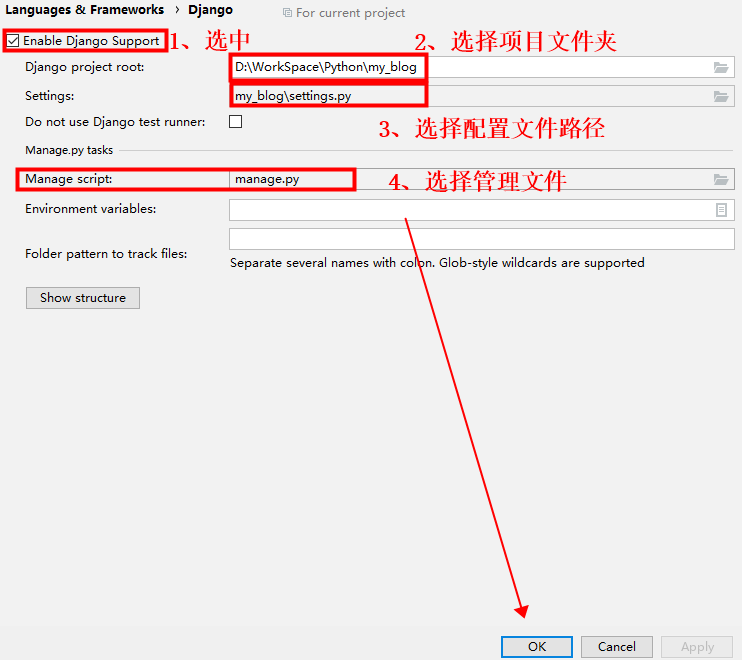
### 开发Django项目时，没有代码提示

有时候我们**并非通过**PyCharm来直接创建Django项目，因此在写Django模板文件时，经常没有代码提示，怎么办呢？

我们只需要把该项目配置为Django项目即可。

过程如下：

PyCharm → File → Settings → Language & Frameworks → Django：



### PyCharm 代码格式化快捷键修改

File → Settings → Keymap → Main menu → Code → Reformat Code:

Ctrl + Alt + /

### PyCharm 代码补全快捷键查看

File → Settings → Keymap → Main menu → Code → Completion → Cyclic Expand Word:

Alt + /