## Windows

### visual c++ building tools

All steps to install build tools:

1. [Download build tools](https://www.visualstudio.com/thank-you-downloading-visual-studio/?sku=BuildTools)
2. Download layout files for offline installation:

Possible workloads are specified [here](https://docs.microsoft.com/en-us/visualstudio/install/workload-component-id-vs-build-tools). Following command contains workloads for msbuild and C++ build tools.

Run (exe file will have some version numbers in name):

vs\_BuildTools.exe --layout D:\vs2019offline --add Microsoft.VisualStudio.Workload.MSBuildTools --add Microsoft.VisualStudio.Workload.VCTools --includeRecommended --lang zh-CN

1. Copy D:\vs2019offline to box on which you want to install it.
2. Open mmc and import all certificates from c:\BT2017offline\certificates to "Trusted Root Certification Authorities" of computer (not current user)
3. Run (exe file will have some version numbers in name): c:\BT2017offline\vs\_BuildTools.exe --noweb
4. Continue clicking next ...

### Android Studio

<https://developer.android.com/studio#downloads>

Android SDK

使用Android Studio 安装 Android SDK

Notice:

1. 安装过程中, 如果提示使用 proxy, 设置为 http, 127.0.0.1:1080
2. 设置 android sdk 下载目录为有写入权限的目录, 如 D:\android\studio\_sdk

## Standalone Android SDK Tool

什么是SDK

SDK：（software development kit）软件开发工具包。是软件开发工程师用于为特定的软件包、软件框架、硬件平台、操作系统等建立应用软件的开发工具的集合。Android SDK 指的是Android专属的软件开发工具包。

安装SDK前准备工作

### 下载并安装 java sdk

**java jre**

选择语言英语, 所有下载, 下载离线安装包

<https://www.java.com/en/download/manual.jsp>

**java jdk**

底部导航 [Developers](https://www.java.com/en/download/faq/develop.xml) > [JDK downloads](http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html#javasejdk) > [Java SE 8u211 / Java SE 8u212](https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html) > JDK download >

|  |  |  |
| --- | --- | --- |
| Windows x86 | 202.62 MB | [jdk-8u211-windows-i586.exe](https://www.oracle.com/technetwork/java/javase/downloads/javascript: void(0)) |
| Windows x64 | 215.29 MB | [jdk-8u211-windows-x64.exe](https://www.oracle.com/technetwork/java/javase/downloads/javascript: void(0)) |

**注: java jdk需要在注册并登录账号后才能下载**

1. 新建c:/java/jdk, c:/java/jre, 分别安装到 jdk 和 jre 到这两个目录中

#### 设置JAVA环境变量

**添加**

变量名：Path

变量值：

C:\ProgramData\Oracle\Java\javapath

%java\_home%\bin

%java\_home%\jre\bin

OR

C:\ProgramData\Oracle\Java\javapath;%java\_home%\bin;%java\_home%\jre\bin

**新建**

变量名：JAVA\_HOME

变量值：C:\java\jdk

**新建**

变量名：ClassPath

变量值：.;%JAVA\_HOME%\lib\dt.jar;%JAVA\_HOME%\lib\tools.jar;

#### 确定已经安装完成

命令行分别输入: java, javac, java -version, javac -version 进行测试

### 安装 android sdk

下载地址：http://tools.android-studio.org/index.php/sdk

installer\_r24.4.1-windows.exe

安装位置 D:\android\standalone\_sdk

#### 设置环境变量

**新建**

变量名：ANDROID\_HOME

变量值：D:\android\standalone\_sdk

**添加**

变量名：Path

变量值：

%ANDROID\_HOME%\platform-tools

%ANDROID\_HOME%\tools

OR

%ANDROID\_HOME%\platform-tools;%ANDROID\_HOME%\tools;

### 下载 sdk

运行 "D:\android\standalone\_sdk\SDK Manager.exe" 打开 Android SDK Manager

设置代理

Tools > Options > Proxy > mirrors.neusoft.edu.cn:80

Others > Check "Force htts:// ... sources to be fetched using http://..

下载 Packages

Tools >

Android SDK Tools

Android SDK Platform-tools

Android SDK Build-tools (newest version)

Android 9

Android 8.1.0

...

Extras > Check all

### 移动端自动化控制工具详讲

#### Android 开发工具使用

什么是adbadb ( Android Debug Bridge)是一个通用命令行工具，其允许您与模拟器实例或连接的 Android 设备进行通信。它可为各种设备操作提供便利，如安装和调试应用。

adb工具的工作原理

启动一个 adb 客户端时，此客户端首先检查是否有已运行了 adb 服务器进程。如果没有，它将启动服务器进程。当服务器启动时，它与本地 TCP 端口 5037 绑定，并侦听从 adb 客户端发送的命令

所有 adb 客户端均使用端口 5037 与 adb 服务器通信。

注意事项使用adb命令工具，需要移动客户端（手机）开启开发者模式，并允许USB调试

adb start-server

\* daemon not running; starting now at tcp:5037

\* daemon started successfully

adb version

Android Debug Bridge version 1.0.41

Version 28.0.3-5475833

Installed as D:\android\standalone\_sdk\platform-tools\adb.exe

adb devices

List of devices attached

127.0.0.1:62001 device

启动安卓开发者模式, 允许 USB 调试

#### 升级 noxplayer 的 adb 版本

打开

D:\android\standalone\_sdk\platform-tools

复制 adb.exe, AdbWinApi.dll, AdbWinUsbApi.dll, 三个文件, 覆盖

\Program Files\Nox\bin 文件夹中的三个同名文件

复制 D:\android\standalone\_sdk\platform-tools\adb.exe, 重命名为 nox\_adb.exe, 覆盖

\Program Files\Nox\bin 文件夹中的同名文件

覆盖前注意备份原文件

确认 nox/bin 中的 adb.exe 版本和 android\standalone\_sdk\platform-tools\adb.exe 两个文件的版本相同

adb.exe –version

D:\Program Files\Nox\bin\adb.exe --version

#### adb.exe 安装应用

D:\Program Files\Nox\bin\adb.exe

cd D:\Program Files\Nox\bin\

adb.exe devices -l

D:\Program Files\Nox\bin>adb.exe devices -l

List of devices attached

127.0.0.1:62001 device product:scorpio model:Mi\_Note\_2 device:shamu

如果没有列出来连接的设备, 重启 nox 然后再次运行此命令.

D:\Program Files\Nox\bin>adb.exe devices -l

List of devices attached

127.0.0.1:62025 device product:dream2qltezh model:google\_Pixel\_2 device:dream2qltechn

127.0.0.1:62001 device product:greatqltesq model:SM\_N950U device:shamu

使用 adb 安装应用

adb.exe install 把 apk 包拖动到命令行窗口中, 会自动生成地址

D:\Program Files\Nox\bin>adb.exe install D:\David\Downloads\Douguolatest69404.apk

进入到 adb.exe 所在的目录, 执行

adb.exe -s 127.0.0.1:62001 install D:\David\Downloads\Douguolatest69404.apk

进入手机 shell 交互环境

adb -s 127.0.0.1:62001 shell

cd /data/app

ls

com.douguo.recipe-1

com.seebaby-1

com.ss.android.ugc.aweme-1

com.bbk.appstore-1

com.tal.kaoyan

包名:

考研邦 com.tal.kaoyan

豆果美食 com.douguo.recipe

掌通家园 com.seebaby

抖音 com.ss.android.ugc.aweme

vivo 应用商店 com.bbk.appstore-1

oppo 应用商店 com.oppo.market-1

删除包时要使用包名来删除

# 退出安卓交互环境

exit

# 使用 adb 删除应用

adb -s 127.0.0.1:62001 uninstall com.seebaby

如果 adb devices 时没有显示出连接的设备, 可能是因为先进行设备的连接, 才启动了 adb server, 此时可以手动连接

adb connect 127.0.0.1:62001

# 查看系统应用包

adb shell pm list package

vivo 应用商店

package:com.bbk.appstore

oppo 应用商店

package:com.oppo.market

# 把本机的文件复制到安卓系统中

adb push C:\apk\yeshen\_test.txt /sdcard

# 把安卓设备中的文件复制到电脑中

adb pull /sdcard/screen.png C:\apk\

# 使用 adb 进行截图

adb shell screencap /sdcard/test.png

adb pull /sdcard/test.png C:\apk\

#### uiautomator 工具使用

什么是uiautomator

Android 4.3发布的时候发布的测试工具uiautomator是用来做UI测试的。也就是普通的手工测试，点击每个控件元素 看看输出的结果是否符合预期。比如 登陆界面 分别输入正确和错误的用户名密码然后点击登陆按钮看看是否能否登陆以及是否有错误提示等

uiautomator工具的组成

uiautomatorviewer – 一个图形界面工具来扫描和分析应用的UI控件。存放在tools目录打开

D:\android\standalone\_sdk\tools\uiautomatorviewer.bat

点击 Device Screenshot (with compress) 来获取设备的截图

高级版uiautomatorviewer-master

替换 D:\android\standalone\_sdk\tools 中的同名文件和文件夹

uiautomator – 一个测试的Java库，包含了创建UI测试的各种API和执行自动化测试的引擎。

使用方法

1. 开启真机（或模拟器）的USB调式模式后连接电脑2. 打开设备上的App界面3. 启动UI Automatorviewer4. 获取对应页面的元素(多设备连接时要选择需要获取元素的设备)

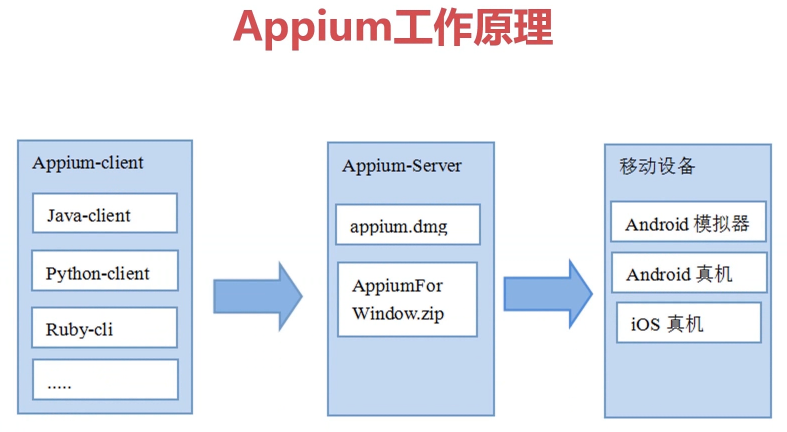
#### Appium自动化工具详讲

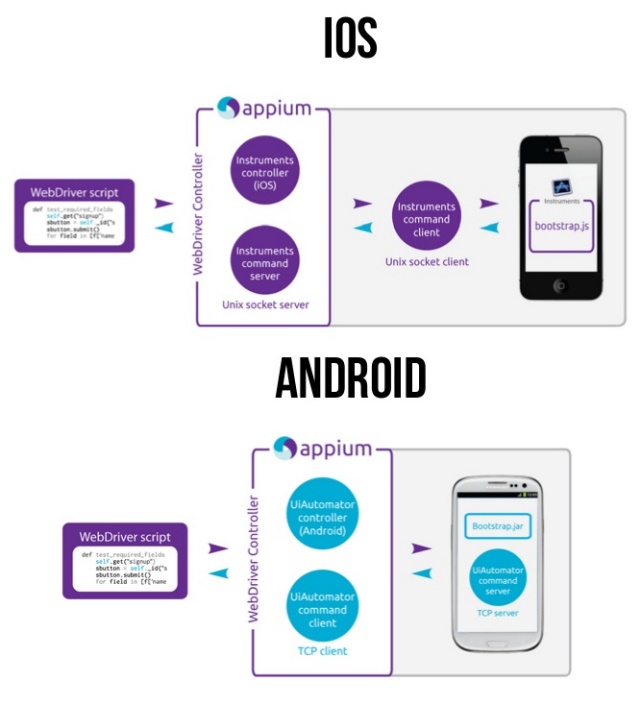
appium

Appium是一个开源测试自动化框架，可用于原生，混合和移动Web应用程序测试。 它使用WebDriver协议驱动iOS，Android应用程序。

## appium

appium选择了Client/Server的设计模式



Appium架构 

下载

<http://appium.io/>

appium sample code:

<https://github.com/appium/appium/tree/master/sample-code>

appium setup guide in english

<https://github.com/appium/appium/blob/master/docs/en/about-appium/getting-started.md>

appium documentation

<http://appium.io/documentation.html?lang=en>

chinese documentation

<http://appium.io/docs/cn/about-appium/intro/>

Appium 中文文档小组

<https://testerhome.com/appium-doc-cn>

多语言支持

Ruby https://github.com/appium/ruby\_lib

Python https://github.com/appium/python-client

Java https://github.com/appium/java-client

JavaScript (Node.js) https://github.com/admc/wd

Objective C https://github.com/appium/selenium-objective-c

PHP https://github.com/appium/php-client

C# (.NET) https://github.com/appium/appium-dotnet-driver

RobotFramework https://github.com/jollychang/robotframework-appiumlibrary

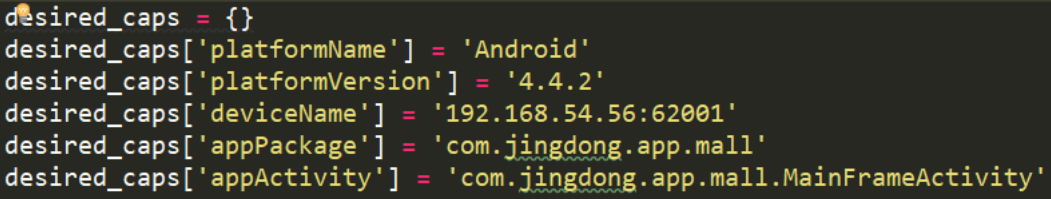
windows 下载安装

<https://github.com/appium/appium-desktop/releases>

启动服务, port 4723

[Appium] Appium REST http interface listener started on 0.0.0.0:4723

Start Inspector Session

Desired Capabilitydesired capability的功能是配置Appium会话。他们告诉Appium服务器您想要自动化的平台和应用程序。 

Desired Capability的值-公用



Desired Capability的值-android



Desired Capability的值-IOS



获取

appPackage

appActivity

D:\android\standalone\_sdk\build-tools\28.0.3\aapt.exe

aapt.exe dump badging D:\David\Desktop\code\apks\Douguolatest69404.apk

aapt.exe dump badging D:\David\Desktop\code\apks\Douguolatest69404.apk | find "launchable-activity"

# launchable-activity: name='com.douguo.recipe.MainActivity' label='' icon=''

**Vivo 应用商店**

查看包名

D:\android\standalone\_sdk\build-tools\28.0.3>aapt.exe dump badging D:\David\Desktop\vivo\_appstore.apk | find "package"

package: name='com.bbk.appstore' versionCode='6420' versionName='8.10.2.0'

查看 launchable-activity

D:\android\standalone\_sdk\build-tools\28.0.3>aapt.exe dump badging D:\David\Desktop\vivo\_appstore.apk | find "launchable-activity"

launchable-activity: name='com.bbk.appstore.ui.AppStore' label='V-Appstore' icon=''

oppo 应用商店

aapt.exe dump badging D:\David\Desktop\oppo\_app\_store.apk | find "launchable-activity"

launchable-activity: name='com.oppo.market.activity.MainActivity' label='' icon=''

进入 adb shell, 输入以下命令, 打开对应的 app

adb shell

logcat | grep cmp=

多输入几个空行, 与启动的其它参数分隔开, 再在安卓模拟器中启动应用

查找以下内容

考研帮

I/ActivityManager( 1748): START u0 {act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.tal.kaoyan/.ui.activity.SplashActivity bnds=[149,618][290,848]} from uid 1000 on display 0

W/ActivityManager( 1748): Unable to start service Intent { act=com.google.android.gms.analytics.service.START cmp=com.google.android.gms/.analytics.service.AnalyticsService (has extras) } U=0: not found

W/ActivityManager( 1748): Unable to start service Intent { act=com.google.android.gms.analytics.service.START cmp=com.google.android.gms/.analytics.service.AnalyticsService (has extras) } U=0: not found

I/ActivityManager( 1748): START u0 {cmp=com.tal.kaoyan/.ui.login.LoginActivity (has extras)} from uid 10038 on display 0

考研帮

{

"platformName": "Android",

"platformVersion": "5.1.1",

"deviceName": "127.0.0.1:62001",

"appPackage": "com.tal.kaoyan",

"appActivity": "com.tal.kaoyan.ui.activity.SplashActivity"

}

掌通家园

I/ActivityManager( 1748): START u0 {act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.seebaby/.login.ui.activity.LauncherActivity bnds=[8,618][149,848] (has extras)} from uid 1000 on display 0

I/ActivityManager( 1748): START u0 {cmp=com.seebaby/.parent.home.ui.activity.MainActivity} from uid 10037 on display 0

W/ActivityManager( 1748): Unable to start service Intent { cmp=com.autonavi.minimap/com.amap.api.service.AMapService (has extras) } U=0: not found

W/ActivityManager( 1748): Unable to start service Intent { cmp=com.autonavi.minimap/.LBSConnectionService (has extras) } U=0: not found

{

"platformName": "Android",

"platformVersion": "5.1.1",

"deviceName": "127.0.0.1:62001",

"appPackage": "com.seebaby",

"appActivity": "com.seebaby.login.ui.activity.LauncherActivity"

}

豆果美食

I/ActivityManager( 1748): START u0 {act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.douguo.recipe/.MainActivity bnds=[431,388][572,618] (has extras)} from uid 1000 on display 0

I/ActivityManager( 1748): START u0 {cmp=com.douguo.recipe/.HomeActivity} from uid 10034 on display 0

W/ActivityManager( 1748): Unable to start service Intent { cmp=com.autonavi.minimap/com.amap.api.service.AMapService (has extras) } U=0: not found

W/ActivityManager( 1748): Unable to start service Intent { cmp=com.autonavi.minimap/.LBSConnectionService (has extras) } U=0: not found

{

"platformName": "Android",

"platformVersion": "5.1.1",

"deviceName": "127.0.0.1:62001",

"appPackage": "com.douguo.recipe",

"appActivity": "com.douguo.recipe.MainActivity"

}

Vivo 应用商店

I/ActivityManager( 1682): START u0 {act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.bbk.appstore/.ui.AppStore bnds=[431,388][572,618]} from uid 1000 on display 0

I/ActivityManager( 1682): START u0 {flg=0x14000000 cmp=com.bbk.appstore/.ui.NewInstallAppActivity} from uid 10035 on display 0

I/ActivityManager( 1682): START u0 {flg=0x14000000 cmp=com.bbk.appstore/.ui.AppStoreTabActivity} from uid 10035 on display 0

Oppo 应用商店

I/ActivityManager( 1764): START u0 {act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.oppo.market/.activity.MainActivity bnds=[186,770][362,1050]} from uid 1000 on display 0

I/ActivityManager( 1764): START u0 {flg=0x4000000 cmp=com.oppo.market/a.a.a.agk} from uid 10033 on display 0

微信

I/ActivityManager( 1771): START u0 {act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.tencent.mm/.ui.LauncherUI bnds=[186,770][362,1050] (has extras)} from uid 1000 on display 0

安卓模拟器设置: 显示触摸操作, 指针位置

adb.exe kill-server 关闭 adb server

Appium自动化工具实战

我们要实现登录考研帮app，点击 研讯 按钮，并对里面的信息进行滑动操作

工具 1、一台手机或者安卓模拟器 2、uiautomatorviewer 3、python

app应用数据抓取实战进阶

### 夜神模拟器

下载地址

<https://www.yeshen.com/>

夜神多开器

android 7 抖音无法正常运行

夜神系统设置 > 启动 root > 预设手机型号

豆果美食下载

<https://m.douguo.com/show/normaldown>

<https://i1.douguo.com/apk/Douguolatest69404.apk?v=20190605>

抖音下载

<https://www.douyin.com/>

掌通家园

官网: <http://www.51110.com/>

下载: <http://web.szy.cn/ztjy-share/download>

进入 noxplayer 安装目录

D:\Program Files\Nox\bin

adb.exe devices -l

List of devices attached

127.0.0.1:62001 device product:LGM-V300K model:LGM\_V300K device:shamu transport\_id:3

#### 为 app 抓取创建一个新的虚拟环境

mkvirtualenv app

安装包

### genymotion模拟器

下载地址

https://www.genymotion.com/download-trial/

Genymotion部署文件时出错

genymotion设备无法运行ARM指令。我们可以安装一个工具来解决这个问题

解决方法：

1. 用Android Studio 创建一个ARM的虚拟机；（当然这个不是你想要的）
2. 下载Genymotion-ARM-Translation-Librarities工具转换包

https://pan.baidu.com/s/1kUAftyR

将下载号的工具包直接拖拽到Genymotion中，然后提示重启模拟器；

### vmware workstation

下载:

0daydown, 六维, byr

#### download ubuntu, centos iso file

ubuntu

https://launchpad.net/ubuntu/+cdmirrors

centos

http://isoredirect.centos.org/centos/7/isos/x86\_64/

#### 安装 open-vm-tools

ubuntu

apt search open-vm

open-vm-tools/bionic-updates,now 2:10.3.5-7~ubuntu0.18.04.1 amd64 [installed]

open-vm-tools-desktop/bionic-updates 2:10.3.5-7~ubuntu0.18.04.1 amd64

open-vm-tools-dev/bionic-updates 2:10.3.5-7~ubuntu0.18.04.1 amd64

sudo apt install open-vm-tools open-vm-tools-desktop open-vm-tools-dev

centos

yum update

yum search open-vm

open-vm-tools.x86\_64 : Open Virtual Machine Tools for virtual machines hosted on VMware

open-vm-tools-desktop.x86\_64 : User experience components for Open Virtual Machine Tools

open-vm-tools-devel.x86\_64 : Development libraries for Open Virtual Machine Tools

open-vm-tools-test.x86\_64 : Test utilities for Open Virtual Machine Tools

sudo yum install -y open-vm-tools open-vm-tools-desktop open-vm-tools-devel

查看是否安装成功

ls /etc/init.d | grep open-vm

## 抓包工具

### charles

下载: 0daydown.com

下载java

https://www.java.com/en/download/manual.jsp

激活

运行以下命令生成激活码

java -jar keygen.jar

设置代理端口

proxy > proxy settings > port:8888

设置抓取https请求

proxy > ssl proxy settings > \*:443

help > ssl proxying > install charles root certification

手机端:

help > ssl proxying > install charles root certification on a mobile client

手机端访问

<http://chls.pro/ssl>

下载证书, 安装

charles-proxy-ssl-proxying-certificate.pem

或者在本地查找 Charles 证书, 复制到手机中

C:\Users\David\AppData\Roaming\Charles\data\ca\charles-proxy-ssl-proxying-certificate.pem

chrome 使用 SwitchyOmega 添加情景模式

代理协议: http, 代理服务器: 127.0.0.1, 代理端口: 8888

### fiddler

https://telerik-fiddler.s3.amazonaws.com/fiddler/FiddlerSetup.exe

<https://www.telerik.com/download/fiddler>

设置fiddler port为 8889

Options > Connections > Fiddler listens on port: 8889

Options > HTTPS > 抓取并解析 https 流量

Options > HTTPS > Actions > 信任根证书

查看证书: chrome > settings > 证书 > 受信任的根证书颁发机构

**注意:** fiddler 设置好之后要重启才能生效.

菜单与工具栏

Edit > Copy 选择性复制

Edit > Find session 查找

Edit > Mark 以某种颜色标记

Rules, 设置抓包的选项, 如过滤 403, 过滤图片请求, 过滤连接请求等

TextWizard 文本转换工具

工具栏: 只抓取某个进程的网络流量

工具栏: Decode 自动解析加密请求

QuickExec

<https://docs.telerik.com/fiddler/knowledgebase/quickexec>

对某个网站单独设置断点

请求关

bpu <https://www.baidu.com>

取消:

bpu

bpafter <https://www.baidu.com>

取消:

bpafter

chrome 浏览器 使用 SwitchyOmega 设置代理

情景模式名称: fiddler, 协议: http, 服务器 127.0.0.1, 端口: 8889

### 手机抓包

**真实手机**

pc 启动热点, 查看热点ip地址, 注意这里不是本地连接的ip地址

无线局域网适配器 本地连接\* 2:

连接特定的 DNS 后缀 . . . . . . . :

本地链接 IPv6 地址. . . . . . . . : fe80::bc0f:f1f2:161e:f83c%14

IPv4 地址 . . . . . . . . . . . . : 192.168.137.1

子网掩码 . . . . . . . . . . . . : 255.255.255.0

默认网关. . . . . . . . . . . . . :

手机连接热点

手机设置热点的代理. fiddler 抓包软件的端口是 8889, 设置手机连接的 wifi 热点的代理地址为: 192.168.137.1:8889

在电脑上启动 fiddler, Tools > Options > HTTPS > from remote clients only 只抓取远程设置的流量

手机访问 192.168.137.1:8889, 安装 fiddler ssl 根证书.

安卓模拟器

开启安卓模拟器的桥接模式

查看 本地连接的 ip 地址

IPv4 地址: 172.18.15.227

IPv4 DNS 服务器: 106.75.117.13

117.50.18.177

制造商: Realtek

描述: Realtek PCIe GbE Family Controller

驱动程序版本: 10.27.511.2018

物理地址(MAC): 48-4D-7E-BF-3A-6D

设置 fiddler 只抓取远端的流量, 设置 fiddler 端口号为 8889

设置浏览器的起始页为空白页, 否则一直提示证书错误, 无法跳转到下载证书的页面

访问

<http://172.18.15.227:8889/>

下载证书并安装

锁屏 密码: abc123

WeinXin\_Article 安卓模拟器

设置代理:

172.18.15.224, 8889

### mitmproxy

可以加载自定义的 python 脚本

Man-in-the-middle attack

基于 python

Windows 需要安装 Visual C++ building tools

mkvirtualenv app

pip install ipython

安装 mitmproxy

pip install mitmproxy

pip install requests

pip install lxml

组件:

mitmproxy (windows 不支持)

mitmdump 把抓包结果保存到文件中, 加载 python 脚本文件

mitmweb 以图形界面的方式显示抓包结果

使用:

查看版本

mitmproxy -version

# mitmproxy's console interface is not supported on windows, run mitmdump or mitmweb

mitmdump --version

mitmweb --version

chrome 浏览器 使用 SwitchyOmega 设置代理

情景模式名称: fiddler, 协议: http, 服务器为安装 mitmproxy 的主机的 ip 地址, 如: 127.0.0.1, 端口: 8080

切换到 mitmproxy 代理

windows cmd 中启动 mitmweb

linux terminal 中启动 mitmproxy

指定端口启动

windows: mitmweb -p 8889

linux: mitmproxy -p 8889

指定端口启动 mitmproxy, mitmweb 之后, 都要重新设置安卓 wifi 的代理, 设置与指定端口相同的端口. 并重新安装 mitmproxy 的证书

加载 python 脚本

linux mitmdump -p 8889 -s test.py

访问 mitm.it 来安装 mitmproxy 证书

windows 证书安装到 "受信任的根证书颁发机构" 中

安卓模拟器安装mitmproxy 证书, 安装之后要重启 模拟器

windows 使用 mitmdump 把抓包信息保存到文件中

mitmdump -w baidu.txt

Linux 中直接使用 mitmproxy 命令来启动 mitmproxy server

mitmproxy -p 8889

命令行 参数 端口号

### Packet Capture android

https://www.coolapk.com/apk/app.greyshirts.sslcapture

http://123.125.9.49/dd.myapp.com/16891/EDCE2FBBF748DF1D1D0CA3D6F00C9BC3.apk?mkey=5b84334b6fca44dd&f=184b&fsname=app.greyshirts.sslcapture\_1.4.7\_23.apk&csr=db5e&from\_tracker=y&cip=111.202.98.40&proto=http

ubuntu18.04更换国内源

1.备份原始源文件source.list

桌面打开终端，执行命令：

sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak

2.修改源文件source.list

（1）终端执行命令：sudo chmod 777 /etc/apt/source.list更改文件权限使其可编辑；

（2）执行命令：sudo gedit /etc/apt/source.list打开文件进行编辑；

（3）删除原来的文件内容，复制下面的任意一个到其中并保存（常用的是阿里源和清华源，推荐阿里源）；

<https://developer.aliyun.com/mirror/>

## Ubuntu python 开发环境配置

python源码安装

### 更新系统

apt update

apt upgrade

apt install vim

启动远程ssh连接

SSH 登录时出现如下错误：Disconnected:No supported authentication methods available

Permission denied (publickey,gssapi-keyex,gssapi-with-mic).

sshd[10826]: Connection closed by XX.XX.XX.XX.

Disconnected: No supported authentication methods available.

解决方法:

vim /etc/ssh/sshd\_config

PasswordAuthentication yes

service ssh restart (ubuntu)

或

systemctl restart sshd (centos)

修改 linux 显示的主机名称

centos

vim /etc/sysconfig/network

HOSTNAME=主机名

ubuntu

vim /etc/hostname

vim /etc/hosts

将127.0.0.1 对应的老主机名更换为新的主机名

修改更新源

<https://opsx.alibaba.com/mirror>

备份

sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak

vim /etc/apt/sources.list

替换

archive.ubuntu.com

为

mirrors.aliyun.com

使用sed批量替换

sudo sed -i 's/mirrors.shu.edu.cn/mirrors.aliyun.com/g' /etc/apt/sources.list

ubuntu设置root密码

sudo passwd

su root

### 使用 apt-get 安装 python3

apt install python3

查看版本

python3 -V

openssl openssl-server

### 安装支持包

python3.6

参考

<https://tecadmin.net/install-python-3-6-ubuntu-linuxmint/>

sudo apt-get install build-essential checkinstall

sudo apt-get install libreadline-gplv2-dev libncursesw5-dev libssl-dev libsqlite3-dev tk-dev libgdbm-dev libc6-dev libbz2-dev

python3.7

参考

<https://tecadmin.net/install-python-3-7-on-ubuntu-linuxmint/>

sudo apt-get install build-essential checkinstall

sudo apt-get install libreadline-gplv2-dev libncursesw5-dev libssl-dev libsqlite3-dev tk-dev libgdbm-dev libc6-dev libbz2-dev libffi-dev zlib1g-dev

### 源码安装 python3.6.8

mkdir /tmp/Python36

cd /tmp/Python36

wget https://www.python.org/ftp/python/3.6.8/Python-3.6.8.tgz

tar xvf Python-3.6.8.tgz

cd /tmp/Python36/Python-3.6.8

# use --enable-optimizations option with configure command to enable additional supports like SSL, bz2 support.

./configure --enable-optimizations

make -j 4

# make altinstall is used to prevent replacing the default python binary file /usr/bin/python .

sudo make altinstall

ubuntu 使用py3来代替python3

vim ~/.bash\_profile

添加代码:

alias py3='python3'

alias py2='python2'

升级 pip

pip3.6 install --upgread pip

由于系统中没有安装过pip, 升级pip后会把pip, pip3, pip3.6 都指向pip3.6

which python3.6

# /usr/local/bin/python3.6

which pip

which pip3

# 使用国内pip源

https://pypi.douban.com/simple 豆瓣

https://mirrors.ustc.edu.cn/pypi/web/simple/ 中国科学技术大学

https://mirrors.aliyun.com/pypi/simple/ 阿里云

https://pypi.tuna.tsinghua.edu.cn/simple/ 清华大学

mkdir ~/.pip

vim ~/.pip/pip.conf

# windows在 %HOMEPATH%\pip\pip.ini

修改内容为：

[global]

index-url = https://pypi.douban.com/simple

[install]

trusted-host=pypi.douban.com

# 安装虚拟环境

apt install python-virtualenv

安装 pip3

sudo apt install python3-pip

升级pip3

pip3 install --upgrade pip

pip3 install --upgrade setuptools

pip3 intall virtualenv

pip3 install virtualenvwrapper

pip3 install virtualenvwrapper-win (windows)

sudo find / -name virtualenvwrapper.sh

# /usr/local/bin/virtualenvwrapper.sh

# 配置virtualwrapper环境变量

sudo vim ~/.bashrc

#

export WORKON\_HOME=$HOME/.virtualenvs # 虚拟环境创建的地方

export VIRTUALENVWRAPPER\_PYTHON=/usr/local/bin/python3.6 # 指定虚拟使用的python解释器路径

source /usr/local/bin/virtualenvwrapper.sh # 每次登陆用户自动执行下脚本

# 使配置生效.

source ~/.bashrc

# 指定创建虚拟环境时的python版本

mkvirtualenv --python=/usr/bin/python3 python3\_scrapy

mkvirtualenv --python=C:\Python36\python.exe python3\_spider

windows

在创建虚拟环境时可以指定python的版本.

mkvirtualenv -p C:\Python35\python.exe py3

mkvirtualenv --python=C:\Python35\python.exe python3\_scrapy‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬

# 查看所有可用的虚拟环境

lsvirtualenv

workon

# 进入到虚拟环境所在的目录

cdvirtualenv my\_env

# 退出虚拟环境

deactivate

# 删除某个虚拟环境

rmvirtualenv my\_env

rm -rf python3\_scrapy

virtualenvwrapper新建的虚拟环境默认放在C:\Users\David\Envs\中.

可以修改此默认保存的位置

新建系统环境变量, 变量名: WORKON\_HOME, 变量值E:\Envs\

Linux virtualenvwrapper配置

sudo find / -name virtualenvwrapper.sh

# /home/david/.local/bin/virtualenvwrapper.sh

# /usr/local/bin/virtualenvwrapper.sh

# 配置virtualwrapper环境变量

Add three lines to your shell startup file (.bashrc, .profile, etc.)

sudo vim ~/.bashrc

# WORKON\_HOME设置虚拟环境保存的目录, $HOME为用户的主目录, 即设置虚拟环境保存在用户主目录下的.virtualenvs目录下.

PROJECT\_HOME 开发的项目存放的位置

VIRTUALENVWRAPPER\_PYTHON 指定虚拟使用的python解释器路径

source /usr/local/bin/virtualenvwrapper.sh 表示每次登陆用户自动执行下脚本, 如果是非root用户, 则类似是 source /home/david/.local/bin/virtualenvwrapper.sh

export WORKON\_HOME=$HOME/.virtualenvs

export PROJECT\_HOME=$HOME/Devel

export VIRTUALENVWRAPPER\_PYTHON=/usr/bin/python3

source /usr/local/bin/virtualenvwrapper.sh

# 使配置生效

source ~/.bashrc

**ubuntu安装twisted 出错**

Running setup.py bdist\_wheel for Twisted ... error

sudo apt update

sudo apt install build-essential python3.6-dev

### docker

#### 安装教程

<https://docs.docker.com/install/linux/docker-ce/ubuntu/>

#### 手动重启 docker 守护进程

Once Docker is installed, you need to start the Docker daemon. Most Linux distributions use systemctl to start services. If you do not have systemctl, use the service command.

systemctl:

sudo systemctl start docker

service:

sudo service docker start

#### 安装完成后配置

<https://docs.docker.com/install/linux/linux-postinstall/>

# 修改docker配置文件, 使用国内的 docker 加速器

vim /etc/docker/daemon.json

{

"registry-mirrors": ["https://dlfa9xic.mirror.aliyuncs.com"]

}

**以非root用户登录**

1. Create the docker group.

sudo groupadd docker

1. Add your user to the docker group.

sudo usermod -aG docker $USER

1. logout or restart on virtual machine for changes to take effect
2. Verify that you can run docker commands without sudo.

docker run hello-world

**设置系统启动时启动docker服务**

sudo systemctl enable docker

To disable this behavior, use disable instead.

sudo systemctl disable docker

**设置Docker daemon 监听外部连接**

# 创建/etc/systemd/system/docker.service.d目录

sudo mkdir /etc/systemd/system/docker.service.d

# 创建一个/etc/systemd/system/docker.service.d/http-proxy.conf文件

sudo vim /etc/systemd/system/docker.service.d/http-proxy.conf

Configuring remote access with systemd unit file

1. Use the command to open an override file for docker.service in a text editor.

sudo systemctl edit docker.service

1. Add or modify the following lines, substituting your own values.

[Service]

ExecStart=

ExecStart=/usr/bin/dockerd -H fd:// -H tcp://127.0.0.1:2375

[Service]

ExecStart=

ExecStart=/usr/bin/dockerd -H tcp://0.0.0.0:2375 -H unix:///var/run/docker.sock --data-root="/data/docker-container/lib/docker"

# 修改 Docker Root Dir 路径

[Service]

ExecStart=

ExecStart=/usr/bin/dockerd -H fd:// -H tcp://127.0.0.1:2375 --data-root=/data/docker-container/lib/docker

1. Save the file.
2. Reload the systemctl configuration.

sudo systemctl daemon-reload

1. Restart Docker.

sudo systemctl restart docker.service

1. Check to see whether the change was honored by reviewing the output of netstat to confirm dockerd is listening on the configured port.

sudo netstat -lntp | grep dockerd

tcp 0 0 127.0.0.1:2375 0.0.0.0:\* LISTEN 3758/dockerd

## Centos python 开发环境配置

### python

python3.6

sudo yum install gcc openssl-devel bzip2-devel

参考

<https://tecadmin.net/install-python-3-6-on-centos/>

python3.7

sudo yum install gcc openssl-devel bzip2-devel libffi-devel

参考

<https://tecadmin.net/install-python-3-7-on-centos/>

### docker

安装教程

<https://docs.docker.com/install/linux/docker-ce/centos/>

安装完成后配置

<https://docs.docker.com/install/linux/linux-postinstall/>

# 1. 安装 redis

wget http://download.redis.io/redis-stable.tar.gz

tar xvzf redis-stable.tar.gz

cd redis-stable

make && make install

# make之后可以使用make test进行测试. 然后再make install

# apt install tcl

# make test

which redis-server

# /usr/local/bin/redis-server

which redis-cli

# /usr/local/bin/redis-cli

启动 redis

redis-server

查看redis是否启动, 重新打开一个终端执行redis-cli连接到redis-server

$ redis-cli

redis 127.0.0.1:6379> ping

PONG

redis 127.0.0.1:6379> set mykey somevalue

OK

redis 127.0.0.1:6379> get mykey

"somevalue"

# 使用redis\_init\_script脚本启动redis

# Create a directory where to store your Redis config files and your data:

sudo mkdir /etc/redis

sudo mkdir /var/redis

# Copy the init script that you'll find in the Redis distribution under the utils directory into /etc/init.d.

# 将redis安装目录的utils下的redis\_init\_script拷贝到/etc/init.d目录下, 放在/etc/init.d下，主要是将redis作为一个系统的daemon进程去运行的，每次系统启动，redis进程一起启动

# We suggest calling it with the name of the port where you are running this instance of Redis. For example:

17839

sudo cp ./redis-stable/utils/redis\_init\_script /etc/init.d/redis\_6379

sudo cp ./redis-stable/utils/redis\_init\_script /etc/init.d/redis\_17839

# (optional) 赋予脚本执行权限

chmod a+x /etc/init.d/redis\_6379

chmod a+x /etc/init.d/redis\_17839

# Edit the init script.

sudo vim /etc/init.d/redis\_6379

sudo vim /etc/init.d/redis\_17839

# Make sure to modify REDISPORT accordingly to the port you are using. Both the pid file path and the configuration file name depend on the port number.

# 根据需要修改端口号

REDISPORT=6379

# REDISPORT=17839

# 根据需要修改安装的redis目录

EXEC=/usr/local/bin/redis-server

CLIEXEC=/usr/local/bin/redis-cli

PIDFILE=/var/run/redis\_${REDISPORT}.pid

# 可以修改配置文件目录，也可以按照这个目录在linux上创建

CONF="/etc/redis/${REDISPORT}.conf"

# Copy the template configuration file you'll find in the root directory of the Redis distribution into /etc/redis/ using the port number as name, for instance:

sudo cp redis.conf /etc/redis/6379.conf

sudo cp ./redis-stable/redis.conf /etc/redis/6379.conf

sudo cp ./redis-stable/redis.conf /etc/redis/17839.conf

# Create a directory inside /var/redis that will work as data and working directory for this Redis instance:

sudo mkdir /var/redis/6379

sudo mkdir /var/redis

sudo mkdir /var/redis/17839

# Edit the configuration file, making sure to perform the following changes:

sudo vim /etc/redis/6379.conf

sudo vim /etc/redis/17839.conf

1. 注释掉 bind 127.0.0.1 允许远程登录 redis

2. Change the port accordingly. In our example it is not needed as the default port is already 6379. 设置redis监听的端口号

3. Set daemonize to yes (by default it is set to no). 让redis以daemon进程启动

4. Set the pidfile to /var/run/redis\_6379.pid (modify the port if needed). 设置redis的pid文件.

5. Set your preferred loglevel. 设置日志等级

6. Set the logfile to /var/log/redis\_6379.log

7. Set the dir to /var/redis/6379 (very important step!) 设置持久化文件的存储位置（该目录需要有足够的磁盘空间）

8. requirepass 设置登录密码

9. Finally add the new Redis init script to all the default runlevels using the following command:

sudo update-rc.d redis\_6379 defaults

sudo update-rc.d redis\_17839 defaults

# 出错信息, 但不影响正常使用

root@ubuntu:~# sudo update-rc.d redis\_6379 defaults

insserv: Script redis\_6379 is broken: incomplete LSB comment.

insserv: missing `Required-Start:' entry: please add even if empty.

insserv: missing `Required-Stop:' entry: please add even if empty.

You are done! Now you can try running your instance with: 启动redis

sudo /etc/init.d/redis\_6379 start

sudo /etc/init.d/redis\_17839 start

验证redis是否启动成功

ps -ef | grep redis

ps -aux | grep redis

没有设置登录密码的redis的关闭

/etc/init.d/redis\_6379 stop

登录带有密码验证的redis-server

redis-cli -p 17839 -a Xzz@8481

或者

redis-cli -p 17839

auth password

# 设置有密码验证的redis的关闭

方法一:

redis-cli -p 17839 -a password shutdown

方法二:

redis-cli -p 17839

auth password

shut down

quit

Make sure that everything is working as expected:

1. Try pinging your instance with redis-cli.

2. Do a test save with redis-cli save and check that the dump file is correctly stored into /var/redis/6379/ (you should find a file called dump.rdb).

3. Check that your Redis instance is correctly logging in the log file.

4. If it's a new machine where you can try it without problems make sure that after a reboot everything is still working.

Note: In the above instructions we skipped many Redis configuration parameters that you would like to change, for instance in order to use AOF persistence instead of RDB persistence, or to setup replication, and so forth. Make sure to read the example redis.conf file (that is heavily commented) and the other documentation you can find in this web site for more information.

Install SQL Server

To configure SQL Server on Ubuntu, run the following commands in a terminal to install the mssql-server package.

Import the public repository GPG keys:

wget -qO- https://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -

Register the Microsoft SQL Server Ubuntu repository:

sudo add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/16.04/mssql-server-2017.list)"

Tip

If you want to try SQL Server 2019 , you must instead register the Preview (2019) repository. Use the following command for SQL Server 2019 installations:

sudo add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/16.04/mssql-server-preview.list)"

Run the following commands to install SQL Server:

sudo apt-get update

sudo apt-get install -y mssql-server

After the package installation finishes, run mssql-conf setup and follow the prompts to set the SA password and choose your edition.

sudo /opt/mssql/bin/mssql-conf setup

Tip

The following SQL Server 2017 editions are freely licensed: Evaluation, Developer, and Express.

Note

Make sure to specify a strong password for the SA account (Minimum length 8 characters, including uppercase and lowercase letters, base 10 digits and/or non-alphanumeric symbols).

Once the configuration is done, verify that the service is running:

systemctl status mssql-server

If you plan to connect remotely, you might also need to open the SQL Server TCP port (default 1433) on your firewall.

At this point, SQL Server is running on your Ubuntu machine and is ready to use!

iptables -A INPUT -p tcp --dport 1433 -j ACCEPT

iptables -A INPUT -p tcp --dport 1433 -j ACCEPT

iptables -A INPUT -p tcp --dport 1433 -j ACCEPT

### Scrapy-Splash 的使用

https://cuiqingcai.com/5428.html

Scrapy-Splash的安装分为两部分。

一个是Splash服务的安装，具体是通过Docker，安装之后，会启动一个Splash服务，我们可以通过它的接口来实现JavaScript页面的加载。

另外一个是Scrapy-Splash的Python库的安装，安装之后即可在Scrapy中使用Splash服务。

1. 相关链接

GitHub：https://github.com/scrapy-plugins/scrapy-splash

PyPI：https://pypi.python.org/pypi/scrapy-splash

使用说明：https://github.com/scrapy-plugins/scrapy-splash#configuration

Splash官方文档：http://splash.readthedocs.io

2. 安装Splash

Scrapy-Splash会使用Splash的HTTP API进行页面渲染，所以我们需要安装Splash来提供渲染服务。这里通过Docker安装，在这之前请确保已经正确安装好了Docker。

**pull image**

docker pull scrapinghub/splash

**Start the container:**

docker run -it -p 8050:8050 scrapinghub/splash

Splash is now available at 0.0.0.0 at port 8050 (http).

安装完成之后，会有类似的输出结果：

2017-07-03 08:53:28+0000 [-] Log opened.

2017-07-03 08:53:28.447291 [-] Splash version: 3.0

2017-07-03 08:53:28.452698 [-] Qt 5.9.1, PyQt 5.9, WebKit 602.1, sip 4.19.3, Twisted 16.1.1, Lua 5.2

2017-07-03 08:53:28.453120 [-] Python 3.5.2 (default, Nov 17 2016, 17:05:23) [GCC 5.4.0 20160609]

2017-07-03 08:53:28.453676 [-] Open files limit: 1048576

2017-07-03 08:53:28.454258 [-] Can't bump open files limit

2017-07-03 08:53:28.571306 [-] Xvfb is started: ['Xvfb', ':1599197258', '-screen', '0', '1024x768x24', '-nolisten', 'tcp']

QStandardPaths: XDG\_RUNTIME\_DIR not set, defaulting to '/tmp/runtime-root'

2017-07-03 08:53:29.041973 [-] proxy profiles support is enabled, proxy profiles path: /etc/splash/proxy-profiles

2017-07-03 08:53:29.315445 [-] verbosity=1

2017-07-03 08:53:29.315629 [-] slots=50

2017-07-03 08:53:29.315712 [-] argument\_cache\_max\_entries=500

2017-07-03 08:53:29.316564 [-] Web UI: enabled, Lua: enabled (sandbox: enabled)

2017-07-03 08:53:29.317614 [-] Site starting on 8050

2017-07-03 08:53:29.317801 [-] Starting factory <twisted.web.server.Site object at 0x7ffaa4a98cf8>

这样就证明Splash已经在8050端口上运行了。

验证:

1. 在浏览器中打开http://localhost:8050，即可看到Splash的主页。
2. 在终端中运行

curl localhost:8050

或

curl 127.0.0.1:8050

1. 远程访问 splash 服务

当然，Splash也可以直接安装在远程服务器上。我们在服务器上以守护进程运行Splash即可，命令如下：

docker run -dit -p 8050:8050 scrapinghub/splash

这里多了-d参数，它代表将Docker容器以守护态运行，这样在中断远程服务器连接后，不会终止Splash服务的运行。

打开 linux 防火墙的 8050 端口, 这时, 就能在浏览器中通过 linux\_ip:8050 或 curl linux\_ip:8050 来访问 splash 服务了.

3. Scrapy-Splash python cli 的安装

成功安装Splash之后，接下来再来安装其Python库，命令如下：

pip3 install scrapy-splash

Docker splash 帮助信息

docker run -p 8050:8050 scrapinghub/splash --help

Usage: splash [options]

Options:

-h, --help show this help message and exit

-f LOGFILE, --logfile=LOGFILE

log file

-m MAXRSS, --maxrss=MAXRSS

exit if max RSS reaches this value (in MB or ratio of

physical mem) (default: 0)

--proxy-profiles-path=PROXY\_PROFILES\_PATH

path to a folder with proxy profiles

--js-profiles-path=JS\_PROFILES\_PATH

path to a folder with javascript profiles

--no-js-cross-domain-access

disable support for cross domain access when executing

custom javascript (default)

--js-cross-domain-access

enable support for cross domain access when executing

custom javascript (WARNING: it could break rendering

for some of the websites)

--allowed-schemes=ALLOWED\_SCHEMES

comma-separated list of allowed URI schemes (default:

http,https,data,ftp,sftp,ws,wss)

--filters-path=FILTERS\_PATH

path to a folder with network request filters

--xvfb-screen-size=XVFB\_SCREEN\_SIZE

screen size for xvfb (default: 1024x768)

--disable-private-mode

disable private mode (WARNING: data may leak between

requests)

--disable-xvfb disable Xvfb auto start

--disable-lua-sandbox disable Lua sandbox

--disable-browser-caches

disables in-memory and network caches used by webkit

--lua-package-path=LUA\_PACKAGE\_PATH

semicolon-separated places to add to Lua package.path.

Each place can have a ? in it that's replaced with the

module name.

--lua-sandbox-allowed-modules=LUA\_SANDBOX\_ALLOWED\_MODULES

semicolon-separated list of Lua module names allowed

to be required from a sandbox.

--strict-lua-runner

enable additional internal checks for Lua scripts

(WARNING: for debugging only)

-v VERBOSITY, --verbosity=VERBOSITY

verbosity level; valid values are integers from 0 to 5

(default: 1)

--version print Splash version number and exit

-p PORT, --port=PORT port to listen to (default: 8050)

-i IP, --ip=IP binded ip listen to (default: 0.0.0.0)

-s SLOTS, --slots=SLOTS

number of render slots (default: 20)

--max-timeout=MAX\_TIMEOUT

maximum allowed value for timeout (default: 90.0)

--disable-ui disable web UI

--disable-lua disable Lua scripting

--argument-cache-max-entries=ARGUMENT\_CACHE\_MAX\_ENTRIES

maximum number of entries in arguments cache (default:

500)

splash volume

request filter

<https://splash.readthedocs.io/en/latest/api.html#request-filters>

Splash supports filtering requests based on [Adblock Plus](https://adblockplus.org/) rules. You can use filters from [EasyList](https://easylist.adblockplus.org/en/) ( https://easylist.to/ ) to remove ads and tracking codes (and thus speedup page loading), and/or write filters manually to block some of the requests (e.g. to prevent rendering of images, mp3 files, custom fonts, etc.)

To activate request filtering support start splash with --filters-path option:

python3 -m splash.server --filters-path=/etc/splash/filters

The folder --filters-path points to should contain .txt files with filter rules in Adblock Plus format. You may download easylist.txt from [EasyList](https://easylist.adblockplus.org/en/)and put it there, or create .txt files with your own rules.

For example, let’s create a filter that will prevent custom fonts in ttf and woff formats from loading (due to qt bugs they may cause splash to segfault on Mac OS X):

! put this to a /etc/splash/filters/nofonts.txt file

! comments start with an exclamation mark

.ttf|

.woff|

To use this filter in a request add filters=nofonts parameter to the query:

curl 'http://localhost:8050/render.png?url=http://domain.com/page-with-fonts.html&filters=nofonts'

You can apply several filters; separate them by comma:

curl 'http://localhost:8050/render.png?url=http://domain.com/page-with-fonts.html&filters=nofonts,easylist'

If default.txt file is present in --filters-path folder it is used by default when filters argument is not specified. Pass filters=none if you don’t want default filters to be applied.

Only related resources are filtered out by request filters; ‘main’ page loading request can’t be blocked this way. If you really want to do that consider checking URL against Adblock Plus filters before sending it to Splash (e.g. for Python there is adblockparser library).

不下载图片

If you want to stop downloading images check [‘images’](https://splash.readthedocs.io/en/latest/api.html#arg-images) parameter. It doesn’t require URL-based filters to work, and it can filter images that are hard to detect using URL-based patterns.

Warning

在有大量的过滤规则的情况下, 如使用 easylist.txt 的情况下, 使用 re2 来代替 python 默认的 re 模块.

pip install re2

It is very important to have [pyre2](https://github.com/axiak/pyre2) library installed if you are going to use filters with a large number of rules (this is the case for files downloaded from [EasyList](https://easylist.adblockplus.org/en/)).

Without pyre2 library splash (via [adblockparser](https://github.com/scrapinghub/adblockparser)) relies on re module from stdlib, and it can be 1000x+ times slower than re2 - it may be faster to download files than to discard them if you have a large number of rules and don’t use re2. With re2 matching becomes very fast.

Make sure you are **not using** re2==0.2.20 installed from PyPI (it is broken); use the latest version.

## sql server for linux on docker

docker hub:

<https://hub.docker.com/_/microsoft-mssql-server>

SQL Server on Linux7

<https://docs.microsoft.com/zh-cn/sql/linux/sql-server-linux-overview?view=sql-server-2017>

<https://docs.microsoft.com/zh-cn/sql/linux/tutorial-restore-backup-in-sql-server-container?view=sql-server-2017>

sql server cli

<https://github.com/dbcli/mssql-cli/blob/master/doc/installation/linux.md#centos-7>

<https://github.com/twright-msft/mssql-node-docker-demo-app>

install and usage on docker

<https://docs.microsoft.com/zh-cn/sql/linux/quickstart-install-connect-docker?view=sql-server-2017&pivots=cs1-bash>

1. Pull the SQL Server 2017 Linux container image from Docker Hub.

docker pull mcr.microsoft.com/mssql/server:2017-latest

docker pull mcr.microsoft.com/mssql/server

2. run docker

docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=yourStrong(!)Password' \

--name 'sqlserver' \

-p 1433:1433 \

-v sqlserverdata:/var/opt/mssql \

-d mcr.microsoft.com/mssql/server:2017-latest

docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=DreamingTech@163.com' \

--name 'sqlserver' \

-p 1433:1433 \

-v sqlserverdata:/var/opt/mssql \

-d mcr.microsoft.com/mssql/server:2017-latest

sudo docker ps -a

**Change the SA password**

Use docker exec to run sqlcmd to change the password using Transact-SQL. Replace <YourStrong!Passw0rd> and <YourNewStrong!Passw0rd> with your own password values.

sudo docker exec -it sql1 /opt/mssql-tools/bin/sqlcmd \

-S localhost -U SA -P 'yourStrong(!)Password' \

-Q 'ALTER LOGIN SA WITH PASSWORD="<YourNewStrong!Passw0rd>"'

docker compose 安装

<https://docs.docker.com/compose/install/#install-compose>

1. Run this command to download the current stable release of Docker Compose:

sudo curl -L "https://github.com/docker/compose/releases/download/1.24.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

2. Apply executable permissions to the binary:

sudo chmod +x /usr/local/bin/docker-compose

3. Test the installation.

docker-compose --version

docker-compose version 1.24.0, build 1110ad01

Note: If the command docker-compose fails after installation, check your path. You can also create a symbolic link to /usr/bin or any other directory in your path.

For example:

sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

# docker 运行爬虫

所需要的镜像:

docker pull mcr.microsoft.com/mssql/server:2017-latest

docker pull redis:alpine

docker pull redis:latest

docker pull mongo:latest

docker pull python:3.6-alpine

docker pull bash:latest

docker pull mysql:latest

docker pull python:3.6-slim

docker pull python:latest

docker pull mysql:5.7

docker pull scrapinghub/splash:latest

docker pull ubuntu

docker pull centos

# 运行 ubuntu 容器

docker run -it ubuntu bash

python scrapy 运行环境

Dockerfile-slim

```

FROM python:3.6-slim

RUN mkdir /root/.pip && cd /root/.pip \

&& touch pip.conf \

&& echo "[global]" > pip.conf \

&& echo "index-url=https://pypi.douban.com/simple" >> pip.conf \

&& echo "[install]" >> pip.conf \

&& echo "trusted-host=pypi.douban.com" >> pip.conf \

&& pip install --upgrade pip \

&& pip install --upgrade setuptools \

&& pip install --upgrade wheel

RUN sed -i 's/deb.debian.org/mirrors.163.com/g' /etc/apt/sources.list \

&& sed -i 's/security.debian.org/mirrors.163.com/g' /etc/apt/sources.list \

&& apt-get update && apt-get -y install \

gcc default-libmysqlclient-dev

RUN pip install requests lxml scrapy scrapy-redis redis mysqlclient

WORKDIR /data/movie\_spider

```

说明: scrapy 代码使用 volume 映射到 /data 目录下

WORKDIR 设置工作目录为 /data/movie\_spider

在 docker-compose.yaml 文件中设置运行的命令为 scrapy crawl actress

这样, python docker 容器创建之后就会自动执行 scrapy crawl actress 的命令来运行爬虫

运行 Dockerfile 生成自定义镜像

docker build /home/david -f /home/david/Dockerfile-slim -t python:3.6-slim-scrapy

docker-compose-one-python.yaml

version: '3.6'

services:

# 生成的服务的名称

python\_1:

build:

# 指定 Dockerfile 所在的路径, 如果镜像不存在, 会使用此 Dockerfile 自动构建

context: /home/david

# 指定 Dockerfile 的名称

dockerfile: Dockerfile-slim

# build 生成的镜像的 tag 信息, 如果镜像已存在, 就不执行构建镜像的操作

image: python:3.6-slim-scrapy

depends\_on:

- redis

restart: always

volumes:

- /home/david/data:/data

# 与 Dockerfile 中的 WORKDIR 功能相同, 指定工作目录

working\_dir: /data/movie\_spider

# 生成的容器的名称

container\_name: "python\_1"

# 容器运行时执行的命令

command: ["scrapy", "crawl", "actress"]

networks:

- scrapynet

redis:

image: redis

ports:

- 6379:6379

container\_name: "redis"

networks:

- scrapynet

volumes:

- jav\_redis\_data:/data

command:

redis-server

restart: always

networks:

scrapynet:

driver: "bridge"

volumes:

jav\_redis\_data:

driver: "local"

运行 docker-compose 文件

docker-compose -f docker-compose-one-python.yaml up -d

docker-compose -f docker-compose-ten-python.yaml down

version 2: redis.Spider 类爬虫

lpush movie\_start\_urls 'https://javzoo.com/cn/actresses' 'https://avme.pw/cn/actresses'

version 3: crawlSpider 类爬虫

version 4: seeds 类爬虫

# ubuntu 安装 git

root 用户

For Ubuntu, this PPA provides the latest stable upstream Git version

add-apt-repository ppa:git-core/ppa

apt update; apt install git

<https://github.com/pyenv/pyenv/wiki/Common-build-problems>

sudo apt-get install -y make build-essential libssl-dev zlib1g-dev libbz2-dev \

libreadline-dev libsqlite3-dev wget curl llvm libncurses5-dev libncursesw5-dev \

xz-utils tk-dev libffi-dev liblzma-dev python-openssl git