问题汇总

1. uv、conda、venv 都有哪些区别（https://github.com/camel-ai/owl/blob/main/README_zh.md）

Python 的语法

with 有什么用?

用来管理资源。防止内存泄露。

内部实现：上下文管理器。

使用场景：

1. 文件操作、网络连接、数据库连接、锁操作

使用方式

```
with expression(上下文管理器对象) [as variable(可选。上下文管理器的返回值赋值给变量)]:
    pass
```

为什么会有 `__init__.py` 文件？有什么用

用来标识包的，有了这个文件就可以被导入。

通过四个下划线包裹，可以声明python的关键魔术方法。类似于关键字。

https://mp.weixin.qq.com/s/oX6JZS_INj9M8Ntkh4Jp9Q

python的上下文管理器是什么概念

python 中的url路径是怎么处理的?

---

python mac 环境安装

```
# 安装 pyenv
brew install pyenv
# 安装 python 版本
pyenv install 3.9.17
# 配置环境变量
echo 'eval "$(pyenv init --path)"' >> ~/.zshrc
echo 'eval "$(pyenv init -)"' >> ~/.zshrc
# 设置全局版本
pyenv global 3.9.17
```

```
# 安装 pipx
brew install pipx
# 安装 uv
pipx install uv
# 配置环境变量
echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.zshrc
```

python windows 环境安装

```
# 安装 Pyenv
官方安装教程：https://github.com/pyenv-win/pyenv-win/blob/master/docs/installation.md#git-commands
注：文中有标注powershell可能有一些报错，需要解决一下。
# powershell 执行策略修改 - 对于需要进行签名验证的情况适用
Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass
# 安装 uv
powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"
```

Python入门

> python 工程化： https://pyloong.github.io/pythonic-project-guidelines/

了解Python

1. 过去十年各个编程语言的趋势(TIOBE)
2. 编译型语言(直接编译成CPU理解的机器码)和解释型语言(需要一行一行翻译)的理解
3. 优点: 优雅,明确,简单.而且拥有很多封装好的类库和别人的第三方库
4. 缺点: (1)运行速度相较比较慢 (2)代码是不能加密的,发布程序就是发布源码

安装环境

1. Anaconda： 官方推荐只会让我们下载python的包，但事实上的使用包括数据挖掘和机器学习等等都需要用到其他一些第三方的包！！
    1. Anaconda是一个编译好可以直接安装的库，里面有很多封装好的常用的包(包括python的包在内)
    2. 下载地址： www.continuum.io
    3. 安装（windows）： 注意勾选将python加到环境变量
    4. 验证安装（windows）：用python命令来验证
    5. Anaconda自带了idle编辑器（ide），通过idle可以呼出
2. pycharm
    1. pycharm，跟idea一个公司的比较高效的python编辑器

2. 下载pycharm： https://www.jetbrains.com/pycharm/(可以用社区版的，够用)
3. 安装过程和idea差不多
4. 配置python.exe目录的环境

Python实战（汇率兑换）

Python的基本语法

1. 缩进（tab键）
2. 注释（单行注释用"#"，多行数字用""" """三引号来包括）
3. input()输入，print()输出，eval()将字符串转换成数字，
4. 声明变量不用数据类型
5. 操纵字符串：不仅可以正向索引里面的单个字符，还可以反向索引，s='python',t可以用s[2]和s[-4]
6. 操纵字符串：区间索引s[A:B]可以截取字符串（从位置A到B，**不包括B**）
7. 条件判断if的语法：if __ : pass elif __ :pass else: pass（pass占位符，if后面的条件不用加引号，else if直接写成elif）
8. 调试，直接在文件上打断点就可以了

as

Python 框架

gradio —— python 的 ui框架

python的版本和命令行

`py -3` —— windows 比较好的启动器

`python | python3` —— linux上的命令行工具

`pip | pip3` —— 包管理工具

python2 & python3 不兼容

设置全局的镜像源

```
# 设置全局镜像源
pip3 config set global.index-url [url]
# 查看配置
pip3 config list
# 镜像源地址
清华：https://pypi.tuna.tsinghua.edu.cn/simple
阿里云：http://mirrors.aliyun.com/pypi/simple/
中国科技大学 https://pypi.mirrors.ustc.edu.cn/simple/
```

创建虚拟环境，并激活

```
python -m venv [venvName]
source [venvName]/bin/activate  # Linux/macOS
[venvName]\Scripts\activate     # Windows
```

安装依赖命令 - 使用 `requirements.txt`

```
pip3 install -r requirements.txt
```

windows上安装 Python 环境：https://blog.csdn.net/yan111111114/article/details/140684394

- Python 的运行工具也叫解释器

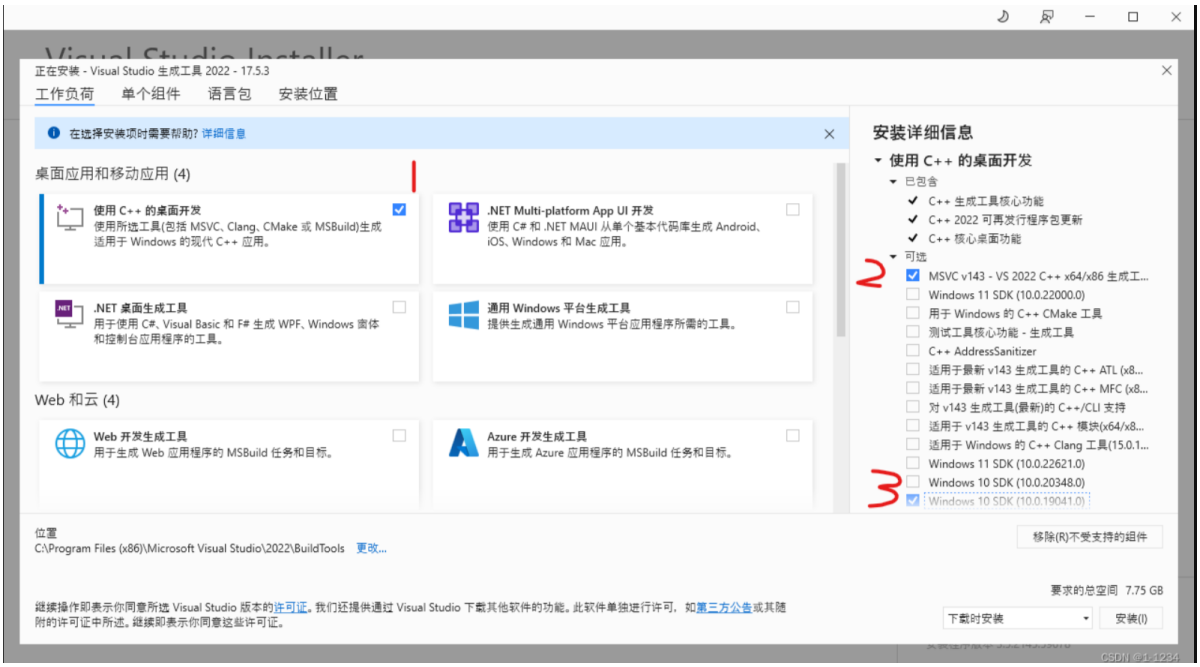
包管理工具

- 使用 pip —— 传统使用
- 使用 uv —— 比较新的东西，可能效率会更高

问题解决

1 安装pandas 依赖的时候报错 Microsoft Visual C++ 14.0 or greater is required

下载微软安装的下载器，然后点击安装对应套件



---

python 入门

环境安装

windows

- 直接官网下载安装包 （https://www.python.org/downloads/windows/）

依赖管理

1. 传统的方式 `pip` —— `requirements.txt`

使用这个命令进行依赖安装，会将依赖信息记录到这个.txt文件中

```
pip install -r requirements.txt
```

2. 使用 `Pipenv`

Pipenv 是一个专门为 Python 项目设计的依赖管理工具，它结合了 Pip 和 Virtualenv 的功能。
使用 Pipenv 可以自动创建和管理虚拟环境，并生成 Pipfile 和 Pipfile.lock 文件来记录依赖。
安装依赖时，可以使用以下命令：`pipenv install`

3. 使用 `Poetry`

Poetry 使用 pyproject.toml 文件来管理依赖和项目配置。
安装依赖时，可以使用以下命令：`poetry install`

4. 使用 `Conda`

Conda 是一个开源的包管理系统和环境管理系统，特别适用于科学计算和数据科学领域。
Conda 使用 environment.yml 文件来管理依赖。
安装依赖时，可以使用以下命令：`conda env create -f environment.yml`

5. 使用 virtualenv 和 venv：

virtualenv 和 venv 是 Python 内置的虚拟环境管理工具，可以创建隔离的 Python 环境。
你可以手动创建虚拟环境并使用 pip 安装依赖。
创建虚拟环境的命令：`python -m venv myenv`
激活虚拟环境后，使用 pip 安装依赖：`pip install package_name`

问题1： 什么是虚拟环境（venv）

> python3 -m venv langchain

- 解决不同项目依赖不同版本的python的适配问题。 —— 使用`pycharm`创建项目会默认创建虚拟环境。

问题2： 什么是pip

- 包管理工具。配置镜像源`pip config set global.index-url https://mirrors.aliyun.com/pypi/simple/`