Flask Web

1 初识Flask

时间：2010年4月1日。

作者：Arimin Ronacher。

Flask是使用Python编写的Web微框架。Web框架可以让我们不用关心底层的请求响应处理，更方便高效地编写Web程序。因为Flask核心简单且易于扩展，所以被称作微框架(micro framework)。Flask有两个主要依赖，一个是WSGI(Web Server Gateway Interface，Web服务器网关接口)工具集——Werkzeug(http://werkzeug.pocoo.org/)，另一个是Jinja2模板引擎(http://jinja.pocoo.org/)。Flask只保留了Web开发的核心功能，其他的功能都由外部扩展来实现，比如数据库集成、表单认证、文件上传等。如果没有合适的扩展，你甚至可以自己动手开发。Flask不会替你做决定，也不会限制你的选择。总之，Flask可以变成任何你想要的东西，一切都由你做主。

> WSGI(Web Server Gateway Interface)是Python中用来规定Web服务器如何与Python Web程序进行沟通的标准。

1.1 搭建开发环境

1.1.1 Pipenv工作流

Pipenv是基于pip的Python包管理工具，它和pip的用法非常相似，可以看作pip的加强版，它的出现解决了旧的pip+virtualenv+requirements.txt的工作方式的弊端。具体来说，它是 pip、Pipfile和Virtualenv的结合体，它让包安装、包依赖管理和虚拟环 境管理更加方便，使用它可以实现高效的Python项目开发工作流。

1.安装pip和Pipenv

```
pip3 --version
```

如果报错，那么你需要自己安装pip。最简单的方式是下载并使用 Python执行get-pip.py文件(https://bootstrap.pypa.io/get-pip.py)。

常用命令：`pip3 install <package>`

这会从PyPI(Python Package Index，Python包索引)上下载并安装指定的包。

> PyPI(https://pypi.org)是一个Python包的在线仓库，截至2018年5 月，共有13万多个包存储在这里。后面我们会学习如何编写自己的Flask扩展，并把它上传到PyPI上。到时你就可以使用上面这条命令安装自己编写的包。

在Linux或macOS系统中使用sudo以全局安装：

```
sudo pip3 install pipenv
```

> 如果你不想全局安装，可以添加--user选项执行用户安装(即pip install--user pipenv)，并手动将用户基础二进制目录添加到PATH环境变量中，具体可参考https://docs.pipenv.org/install/#installing-pipenv 。
>
> PyPI中的包名称不区分大小写。出于方便的考虑，后面的安装命令 都将使用小写名称。:可以使用下面的命令检查Pipenv是否已经安装：

```
zounuo@Kens-Air ~ % pipenv --version
pipenv, version 2022.12.19
```

2.创建虚拟环境

在Python中，虚拟环境(virtual enviroment)就是隔离的Python解释器环境。通过创建虚拟环境，你可以拥有一个独立的Python解释器环境。这样做的好处是可以为每一个项目创建独立的Python解释器环境， 因为不同的项目常常会依赖不同版本的库或Python版本。使用虚拟环境可以保持全局Python解释器环境的干净，避免包和版本的混乱，并且可以方便地区分和记录每个项目的依赖，以便在新环境下复现依赖环境。

虚拟环境通常使用Virtualenv来创建，但是为了更方便地管理虚拟环境和依赖包，我们将会使用集成了Virtualenv的Pipenv。首先确保我们当前工作目录在示例程序项目的根目录，即helloflask文件夹中，然后使用`pipenv install`命令为当前的项目创建虚拟环境：

```
zounuo@Kens-Air helloflask % pipenv install
Creating a virtualenv for this project...
Pipfile: /Users/zounuo/study/source/Flask/code/helloflask/Pipfile
Using /Library/Frameworks/Python.framework/Versions/3.11/bin/python3 (3.11.1) to create virtualenv...
⠸ Creating virtual environment...created virtual environment CPython3.11.1.final.0-64 in 594ms
  creator CPython3Posix(dest=/Users/zounuo/.local/share/virtualenvs/helloflask-NpdXn8fN, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/Users/zounuo/Library/Application Support/virtualenv)
    added seed packages: pip==22.3.1, setuptools==65.6.3, wheel==0.38.4
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator

✔ Successfully created virtual environment!
Virtualenv location: /Users/zounuo/.local/share/virtualenvs/helloflask-NpdXn8fN
Installing dependencies from Pipfile.lock (733065)...
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

这会为当前项目创建一个文件夹，其中包含隔离的Python解释器环 境，并且安装pip、wheel、setuptools等基本的包。因为示例程序仓库里包含Pipfile文件，所以这个文件中列出的依赖包也会一并被安装。

> 默认情况下，Pipenv会统一管理所有虚拟环境。虚拟环境文件夹会在`~/.local/share/virtualenvs/`目录下创建。如果你想在项目目录内创建虚拟环境文件夹，可以设置环境变量PIPENV_VENV_IN_PROJECT，这时名为.venv的虚拟环境文件夹将在项目根目录被创建。
>
> 虚拟环境文件夹的目录名称的形式为`“当前项目目录名+一串随机字符”`，比如helloflask-5Pa0ZfZw。

你可以通过--three和--two选项来声明虚拟环境中使用的Python版本 (分别对应Python3和Python2)，或是使用--python选项指定具体的版本号。同时要确保对应版本的Python已经安装在电脑中。

在单独使用Virtualenv时，我们通常会显式地激活虚拟环境。在Pipenv中，可以使用pipenv shell命令显式地激活虚拟环境:

```
 $ pipenv shell
 Loading .env environment variables...
 Launching subshell in virtual environment. Type 'exit' to return.
```

```
zounuo@Kens-Air helloflask % pipenv shell
Launching subshell in virtual environment...
 . /Users/zounuo/.local/share/virtualenvs/helloflask-NpdXn8fN/bin/activate
zounuo@Kens-Air helloflask %  . /Users/zounuo/.local/share/virtualenvs/helloflask-NpdXn8fN/bin/activate
```

提示：当执行`pipenv shell`或`pipenv run`命令时，Pipenv会自动从项目目录下的.env文件中加载环境变量。

Pipenv会启动一个激活虚拟环境的子shell，会发现命令行提示符前添加了虚拟环境名"(虚拟环境名称)$"，比如：

```
(helloflask) zounuo@Kens-Air helloflask %
```

退出：`exit`。

注意：在Windows系统中使用pipenv shell激活虚拟环境时，虽然激活成功，但是命令行提示符前不会显示虚拟环境名称。

除了显式地激活虚拟环境，Pipenv还提供了一个pipenv run命令，这个命令允许你不显式激活虚拟环境即可在当前项目的虚拟环境中执行命令，比如:

```
pipenv run python hello.py
```

这会使用虚拟环境中的Python解释器，而不是全局的Python解释器。事实上，和显式激活/关闭虚拟环境的传统方式相比，pipenv run是更推荐的做法，因为这个命令可以让你在执行操作时不用关心自己是否激活了虚拟环境。为了方便书写，本书后面涉及的诸多命令会直接写出，省略前面的虚拟环境名称。在实际执行时，你需要使用pipenv shell激活虚拟环境后执行命令，或是在命令前加入pipenv run，后面不再提示。

3.管理依赖

一个程序通常会使用很多的Python包，即**依赖(dependency)**。而 程序不仅仅会在一台电脑上运行，程序部署上线时需要安装到远程服务器上，而你也许会把它分享给朋友。如果你打算开源的话，就可能会有更多的人需要在他们的电脑上运行。为了能顺利运行程序，他们不得不记下所有依赖包，然后使用pip或Pipenv安装，这些重复无用的工作当然应该避免。在以前我们通常使用pip搭配一个requirements.txt文件来记录依赖。但requirements.txt需要手动维护，在使用上不够灵活。Pipfile的出现就是为了替代难于管理的requirements.txt。

在创建虚拟环境时，如果项目根目录下没有Pipfile文件，pipenv install命令还会在项目文件夹根目录下创建Pipfile和Pipfile.lock文件，前者用来记录项目依赖包列表，而后者记录了固定版本的详细依赖包列表。当我们使用Pipenv安装/删除/更新依赖包时，Pipfile以及Pipfile.lock 会自动更新。

> 使用`pipenv graph`查看当前环境下的依赖情况，或是在 虚拟环境中使用`pip list`命令查看依赖列表。
>
> 当需要在一个新的环境运行程序时，只需要执行`pipenv install`命令。`Pipenv`就会创建一个新的虚拟环境，然后自动从`Pipfile`中读取依赖并安装到新创建的虚拟环境中。

1.1.2 安装Flask

使用`pipenv install`命令安装Flask：

```
$ pipenv install flask
Installing flask...
...
Successfully installed Jinja2-2.10 MarkupSafe-1.0 Werkzeug-0.14.1 click-6.7 flask-1.
```

Pipenv会自动帮我们管理虚拟环境，所以在执行pipenv install安装Python包时，无论是否激活虚拟环境，包都会安装到虚拟环境中。后面 我们都将使用Pipenv安装包，这相当于在激活虚拟环境的情况下使用pip安装包。只有需要在全局环境下安装/更新/删除包，我们才会使用pip。

从上面成功安装的输出内容可以看出，除了Flask包外，同时被安装的还有5个依赖包，它们的主要介绍如表1-1所示。

表1-1　Flask的依赖包

名称与版本	说　明	资　源
Jinja2（2.10）	模板渲染引擎	主页：http://jinja.pocoo.org/ 源码：https://github.com/pallets/jinja 文档：http://jinja.pocoo.org/docs/
MarkupSafe（1.0）	HTML 字符转义（escape）工具	主页：https://github.com/pallets/markupsafe
Werkzeug（0.14.1）	WSGI 工具集，处理请求与响应，内置 WSGI 开发服务器、调试器和重载器	主页：http://werkzeug.pocoo.org/ 源码：https://github.com/pallets/werkzeug 文档：http://werkzeug.pocoo.org/docs/
click（6.7）	命令行工具	主页：https://github.com/pallets/click 文档：http://click.pocoo.org/6/
itsdangerous（0.24）	提供各种加密签名功能	主页：https://github.com/pallets/itsdangerous 文档：https://pythonhosted.org/itsdangerous/

> 包括Flask在内，Flask的5个依赖包都由Pocoo团队 (http://www.pocoo.org/)开发，主要作者均为Armin Ronacher(http://lucumr.pocoo.org/)，这些项目均隶属于Pallets项目 (https://www.palletsprojects.com/)。
>
> 本书使用了最新版本的Flask(1.0.2)，如果你还在使用旧版本，请使用下面的命令进行更新：
>
> ```
> pipenv update flask
> ```
>
> 如果你手动使用`pip`和`virtualenv`管理包和虚拟环境，可以使用`--upgrade`或`-U`选项(简写时U为大写)来更新包版本：`pip install -U <包名称>`

`pipenv --venv`：查看项目对应的虚拟环境路径。Linux或macOS系统下的路径类似~/.local/share/virtualenvs/helloflask- kSN7ec1K/bin/python或~/.virtualenvs/helloflask-kSN7ec1K/bin/python。

1.1.3 集成开发环境

IDE(Integrated Development Environment)：集成开发环境

例：Pycharm。

1.2 Hello,Flask!