**1、Flask-WTF简介**

我将使用[Flask-WTF](http://packages.python.org/Flask-WTF)插件来处理本应用中的Web表单，它对[WTForms](https://wtforms.readthedocs.io/)进行了浅层次的封装以便和Flask完美结合。这是本应用引入的第一个Flask插件，但绝不是最后一个。插件是Flask生态中的举足轻重的一部分，Flask故意设计为只包含核心功能以保持代码的整洁，并暴露接口以对接解决不同问题的插件。

Flask插件都是常规的Python三方包，可以使用pip安装。 那就继续在你的虚拟环境中安装Flask-WTF吧：

(venv) $ pip install flask-wtf

2、在本章中，我还要用到两个。 第一个是[Flask-SQLAlchemy](http://packages.python.org/Flask-SQLAlchemy)，这个插件为流行的[SQLAlchemy](http://www.sqlalchemy.org/)包做了一层封装以便在Flask中调用更方便，类似SQLAlchemy这样的包叫做[Object Relational Mapper](http://en.wikipedia.org/wiki/Object-relational_mapping)，简称ORM。 ORM允许应用程序使用高级实体（如类，对象和方法）而不是表和SQL来管理数据库。 ORM的工作就是将高级操作转换成数据库命令。

SQLAlchemy不只是某一款数据库软件的ORM，而是支持包含[MySQL](https://www.mysql.com/)、[PostgreSQL](https://www.postgresql.org/)和[SQLite](https://www.sqlite.org/)在内的很多数据库软件。简直是太强大了，你可以在开发的时候使用简单易用且无需另起服务的SQLite，需要部署应用到生产服务器上时，则选用更健壮的MySQL或PostgreSQL服务，并且不需要修改应用代码（译者注：只需修改应用配置）。

3、我将在本章中介绍的第二个插件是[Flask-Migrate](https://github.com/miguelgrinberg/flask-migrate)。 这个插件是[Alembic](https://bitbucket.org/zzzeek/alembic)的一个Flask封装，是SQLAlchemy的一个数据库迁移框架。 使用数据库迁移增加了启动数据库时候的一些工作，但这对将来的数据库结构稳健变更来说，是一个很小的代价。

## 4、Flask-Login简介

在本章中，我将向你介绍一个非常受欢迎的Flask插件[Flask-Login](https://flask-login.readthedocs.io/)。 该插件管理用户登录状态，以便用户可以登录到应用，然后用户在导航到该应用的其他页面时，应用会“记得”该用户已经登录。它还提供了“记住我”的功能，允许用户在关闭浏览器窗口后再次访问应用时保持登录状态。

* is\_authenticated: 一个用来表示用户是否通过登录认证的属性，用True和False表示。
* is\_active: 如果用户账户是活跃的，那么这个属性是True，否则就是False（译者注：活跃用户的定义是该用户的登录状态是否通过用户名密码登录，通过“记住我”功能保持登录状态的用户是非活跃的）。
* is\_anonymous: 常规用户的该属性是False，对特定的匿名用户是True。
* get\_id(): 返回用户的唯一id的方法，返回值类型是字符串(Python 2下返回unicode字符串).