

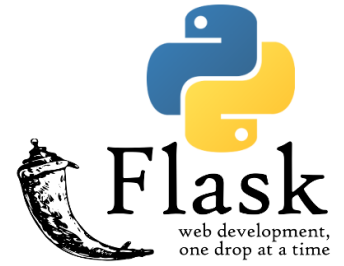


# Pass The Flask

Quickly Create IBM i Web Apps and Services  
with Python and Flask



# Your Presenter



**Richard Schoen**

[richard@richardschoen.net](mailto:richard@richardschoen.net)

IBM i, Windows and Linux  
Software Development Expert

# Richards Journey with IBM i System Integration



- Integrate S/36 to DOS Turbo Pascal screen -HLLAPI (1988)
- Windows 3.1, Microsoft Visual C/C++, Visual Basic 3 (1993)
- APPC programming with PC Support to access RPG (1995)
- TCP/IP, Email, FTP from Windows to IBM i (1997)
- Sockets programming with VB and RPG (2002)
- Web development and web services (1997 –current)
- IBM i Access ODBC/OLEDB/ADO.Net (2000-current)
- JT400 java API converted to .Net with IKVM (2005-current)
- HTTP URL API –(AKA REST) (2004 –current)
- XMLSERVICE –universal open source IBM i DB access (2012-current)
- Editor choice: RDI, Eclipse, Visual Studio, Visual Studio Code, Notepad++, etc...
- (1984–today : Green screen to cellphones) The greatest computing era !!

# Session Topics



- Why consider Python and Flask for IBMi ?
- Intro to Python and Flask
- Develop web application from desktop
- Add a web service layer
- Deploy to IBM i
- Edit/maintain from IBM i

# Why Consider Python and Flask



- Application modernization
- Multi-platform web development
- Very easy to install and get started
- Become productive quickly
- Tons of example code, libraries and tutorials
- Integrates or extends existing IBM i apps
- Python can be used for all kinds of IBM i utility programs
- Python code can be called from RPG/CL easily - PYRUN

# Potential IBM i Use Cases



- Query and present data
- Graphical dashboards
- Spool file management
- Job management
- Browser based Upload/download of IFS files
- Web based editor for source and documents
- IFS file viewer – git and other data

# Opinion – A Few Things I Don't Like



- No variable typing
- No pre-compiler for program checking during editing
- This means more QA testing !!

# Flask and RPG Similarities



- Single mainline program
- Top down structure of main app
- Outputs HTML screens/pages similar to EXFMT
- Functions are similar to sub-procedures



# Directly Accessing IBM i DB2 Data with Flask



- ODBC
- iToolkit (Uses XMLSERVICE)
- Web service calls
- We will be using IBM i Access ODBC Driver

# What is Flask ?



## Flask

Flask is a lightweight [WSGI](#) web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around [Werkzeug](#) and [Jinja](#) and has become one of the most popular Python web application frameworks.

Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

# Flask Acronyms Terms



- Flask – Web application framework
- WSGI – Web server gateway interface (similar to CGI)
- Jinja – Full featured HTML template engine for Python
- Werkzeug – A comprehensive WSGI web application library

## Why Not Django ?



**Flask** provides *simplicity, flexibility* and *fine-grained control*. It is *unopinionated* (it lets you decide how you want to implement things).

**Django** provides an *all-inclusive* experience: you get an *admin panel, database interfaces, an ORM,* and directory structure for your apps and projects out of the box.

<https://www.codementor.io/@garethdwyer/flask-vs-django-why-flask-might-be-better-4xs7mdf8v>

<https://wiki.python.org/moin/WebFrameworks>

# Flask Hello World



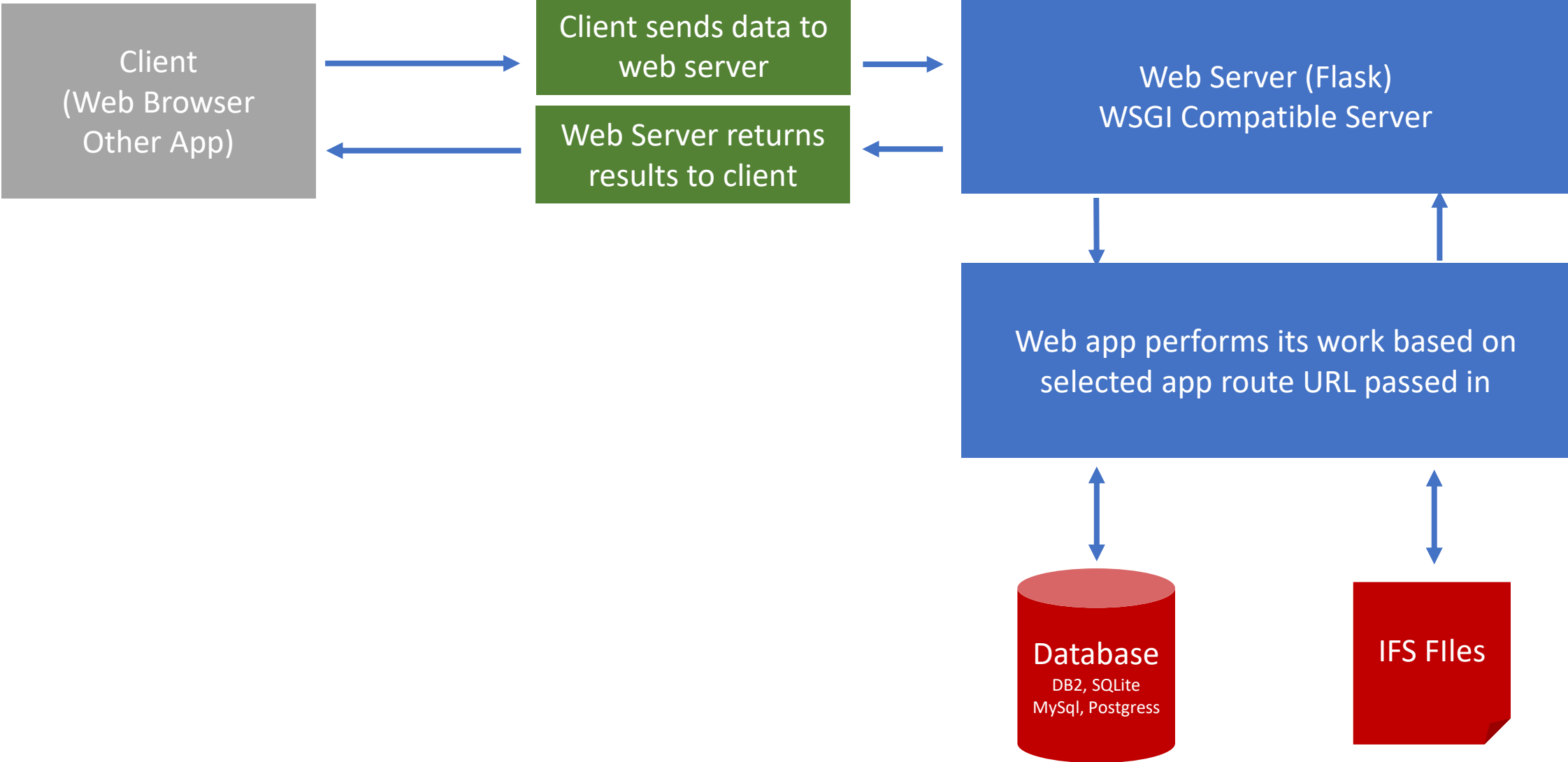
```
from flask import Flask, escape, request

app = Flask(__name__)

@app.route('/')
def hello():
    name = request.args.get("name", "World")
    return f'Hello, {escape(name)}!'
```

```
$ env FLASK_APP=hello.py flask run
* Serving Flask app "hello"
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

# Flask Web Application Flow



# Flask Web Server Deployment Options



- Gunicorn
- Nginx
- uWSGI
- Gevent
- Twisted Web
- FastCgi
- Apache – mod\_wsgi
- Flask site deployment options

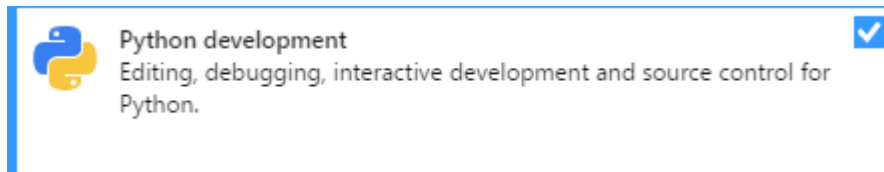
<https://flask.palletsprojects.com/en/1.1.x/deploying>

# Desktop Python/Flask Development




- Install Visual Studio Community 2019 with Python

<https://visualstudio.microsoft.com/vs/community>



- Install IBM i Access – ACS Windows App Package (ODBC)

<https://www.ibm.com/support/pages/ibm-i-access-client-solutions>

ACS Windows App Pkg English (64bit)	IBMiAccess_v1r1_WindowsAP_En glish.zip	53311376 B	<a href="#">Download</a> 
-------------------------------------	---	------------	--



# Other Development IDEs



- Visual Studio Code

<http://code.visualstudio.com>

- PyCharm – JetBrains

<http://www.jetbrains.com/pycharm>

- Eclipse

<https://www.eclipse.org>

- PyDev Eclipse plugin

<http://www.pydev.org>

- LiClipse

<http://www.liclipse.com/index.html>

# VS Develop Demo



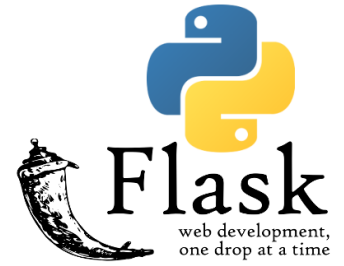
# Demo

# Visual Studio Flask App Structure



- `runserver.py` – launches flask app web server
- `__init__.py` - first file loaded reads config file
- `config.py` – config file
- `views.py` – main program
- HTML templates directory
- HTML static content
- Classes: `dbsqlite.py`, `dbibmi.py`

# Deploy Python/Flask on IBMi



- Install Open Source Package Management

<https://www.ibm.com/support/pages/getting-started-open-source-package-management-ibm-i-ac>

- Install Yum Packages

# Install Python3 Yum Packages



- Install Python3
- <https://www.ibm.com/support/pages/getting-started-open-source-package-management-ibm-i-ac>
- Install unixODBC Drivers
- Install IBM i Access PASE ODBC – ACS PASE App Package (ODBC)

<https://www.ibm.com/support/pages/ibm-i-access-client-solutions>

ACS PASE App Pkg	IBMiAccess_v1r1_PASE_AP.zip	8577229 B	<a href="#">Download</a> ↓
------------------	-----------------------------	-----------	----------------------------

# Installing unixODBC via Yum Packages



- Install Unix ODBC Driver via Yum
- unixODBC
- unixODBC-devel
- Used for ODBC access by IBM i Access ODBC driver

# Download New IBM i Access PASE ODBC Driver



- Download from IBM i Access Client Solutions site  
<http://www-01.ibm.com/support/docview.wss?uid=isg3T1026805>
- Released May 2019
- Unzip `ibm-iaccess-1.1.0.xx-0.ibm7.2.ppc64.rpm` from `IBMiODBC_driver.zip` (Where `xx` is the version in the ZIP file)
- Upload `ibm-iaccess-1.1.0.xx-0.ibm7.2.ppc64.rpm` to IFS root dir `/ibm-iaccess-1.1.0.xx-0.ibm7.2.ppc64.rpm` using your favorite file transfer tool.

# Install New IBM i Access PASE ODBC Driver



- Start pase terminal on IBMi via: `CALL QP2TERM` and then run command list below to install the IBM i Access ODBC driver

```
cd /
```

```
PATH=/QOpenSys/pkgsrc/bin
```

```
export PATH
```

```
rpm -i ibm-iaccess-1.1.0.xx-0.ibm7.2.ppc64.rpm (xx is ver in zip file)
```

- Once install has completed you will be notified on the command line.



# Check the default \*LOCAL ODBC Data Source



- View and verify the default \*LOCAL ODBC data source
- Edit file `/qopensys/etc/odbc.ini` using a PC editor or following CL command: `EDTF '/qopensys/etc/odbc.ini'`
- If the \*LOCAL ODBC entry it looks like below you are good to go

### IBM provided DSN - do not remove this line ###

[\*LOCAL]

Description = Default IBM i local database

Driver = IBM i Access ODBC Driver

System = localhost

UserID = \*CURRENT

### Start of DSN customization

### End of DSN customization

### IBM provided DSN - do not remove this line ###

## Edit the \*LOCAL ODBC Data Source – (Optional)



- Edit the \*LOCAL ODBC data source to support your \*LOCAL system. (`SYS1` is our example local system database name.)
- Run `WRKRDBDIRE` to identify the appropriate system entry tied to the local DB2 database entry location: \*LOCAL.
- Edit file `/qopensys/etc/odbc.ini` using a PC editor or following CL command: `EDTF '/qopensys/etc/odbc.ini'`
- Add or edit the following entries for \*LOCAL data source and set `SYS1` as the database. (Use your own Database name instead of `SYS1`)

`[*LOCAL]`

`Description = local`

`Driver = IBM i Access ODBC Driver`

`System = localhost`

`Naming = 0`

`Database = SYS1`

# Test \*LOCAL ODBC Connectivity with isql utility



- Start pasc terminal on IBMi via: `CALL QP2TERM` and then run command list below to test the IBMi Access ODBC connectivity
- `cd /`
- `PATH=/QOpenSys/pkgsrc/bin`
- `export PATH`
- `isql '*LOCAL' user1 pass1` (Use your own IBMi user/password)
- If connect succeeds it looks something like this:

```
+-----+
| Connected!
| sql-statement
| help[tablename]
| quit
+-----+
```

# Test \*LOCAL ODBC Connectivity with isql utility



- Run a sample SQL statement: `select * from qiws.qcustcdt`
- If this works you should see a list of records on screen like this:

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| CUSNUM | LSTNAM | INIT| STREET    | CITY | STATE| ZIPCOD | CDTLMT| CHGCOD| BALDUE | CDTDUE +-
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 938472 | Henning | G K | 4859 Elm Ave | Dallas| TX  | 75217 | 5000 | 3    | 37.00 | 1234.56 |
| 839283 | Jones   | B D | 21B NW 135 St| Clay  | NY  | 13041 | 400  | 1    | 100.00 | 0      |
| 392859 | Vine    | S S | PO Box 79    | Broton| VT  | 5046  | 700  | 1    | 439.00 | 0      |
| 938485 | Johnson | J A | 3 Alpine Way | Helen | GA  | 30545 | 9999 | 2    |         |         |
```

- Your IBM i Access PASE ODBC driver is ready to be used

# Sample ODBC Connection Strings



- Listed below is a sample connection string:

```
DSN=*LOCAL;UID=user1;PWD=user1
```

- At some point IBM will hopefully PTF the ability to specify \*CURRENT as the data source user so that user/password is not required in DSNs
- Alternative DSN-less PC style connection string  
Driver={IBM i Access ODBC Driver}; System=localhost;Uid=user1;  
Pwd=pass1;
- [https://www.ibm.com/support/knowledgecenter/en/ssw\\_ibm\\_i\\_74/rzaik/connectkeywords.htm](https://www.ibm.com/support/knowledgecenter/en/ssw_ibm_i_74/rzaik/connectkeywords.htm)

# Deploy to IBM i



- Upload app to IFS directory /pythonmagic
- Log in to SSH shell and run following bash commands
- **cd /pythonmagic**
- Edit config.py if needed for connection strings and file paths for SQLite or IBM i Access Database
- **python3 runserver.py (Start Flask server)**
- Test app from browser

# Useful Python Web Links



- <https://palletsprojects.com/p/flask/>
- <https://www.python.org/>
- <https://www.fullstackpython.com/flask.html> Install Python, Flask
- <http://effbot.org/pyfaq/why-was-python-created-in-the-first-place.htm>

# Sample App Links



- IBM i Sample Flask Project

<https://github.com/richardschoen/FlaskWebProjectIBMi>



# What you learned



- How to develop web apps with Flask / Visual Studio
- How to use IBM i Access ODBC connectivity
- How to deploy Flask applications on IBM i
- Get started !!
- Questions: [richard@richardschoen.net](mailto:richard@richardschoen.net)