



# Intro to Flask

Presented By Jeremie Bornais

Sample code available at: [flask.winhacks.ca](https://flask.winhacks.ca)

# A Little About Me

- President of the UWindsor Computer Science Society
- Co-Founder of WinHacks and BorderHacks
- Software developer intern at Assent and Green Shield Canada
- Research assistant
- Project Lead: Google Developer Student Clubs
- 10X Hackathon participant

[jeremie.bornais.ca](http://jeremie.bornais.ca)

[github.com/jere-mie](https://github.com/jere-mie)

[linkedin.com/in/jeremie-bornais](https://linkedin.com/in/jeremie-bornais)



# Agenda

## About Flask

Overview of the framework, what it is, how people use it, why you should consider using it.

## Installation

Simple step by step installation instructions.

## Your First App

The structure of a Flask app, and getting it up and running.

## Routes and Templates

Adding new routes to the app, rendering templates, inheriting from templates, static files.

## Next Steps

WTForms, SQLAlchemy, User Authentication, and many other ways to extend your app.

## Interactive Demo

We'll be making our own Flask app from scratch!

# About Flask

What it is and why you should use it

# What is Flask?

- A micro web framework written in Python
- Used for coding the back end of websites
- Very light by design
- Has many additional addons that can be used in addition to it (WTForms, SQLAlchemy, Flask-Login. etc.)

# Why use it?

- Very simple to get started
- Many great addons
- Hackathon-friendly
- Scales well
- More customizable, let's you decide how you want to solve certain problems

# Installation

Just a “pip install” away!

# “pip install flask”

- Python 3.5+ is required to install Flask
- Pip must also be installed to be able to install it
- Venv is recommended to make handling libraries easier
- Simply run “*pip install flask*” to install it!
  - You may need to use “pip3” instead of “pip” if you’re on a Linux or Unix system

Link to download Python and pip: [python.org](https://python.org)



# Your First App

Boilerplate code to get you started

# The bare minimum

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello, World!'
if __name__ == "__main__":
    app.run(debug=True)
```

1. Save this file to `app.py`
2. Run `python app.py`
  - a. You may need to run `python3 app.py` instead
3. Go to <http://127.0.0.1:5000/>

# Common structure of a Flask app

- `app.py`
- `secrets.json`
- `static/`
  - `style.css`
  - `script.js`
  - ...
- `templates/`
  - `layout.html`
  - `home.html`
  - ...

# Routes and Templates

Add pages, use real HTML files

# Common functions

## **render\_template()**

Used to render an HTML template file in the 'templates' folder

## **url\_for()**

Used to find the URL of a particular route, often used for links and specifying file sources (ex. css files)

## **redirect()**

Used to redirect the user to a different URL or route. Often used with `url_for()` to redirect to different routes on the website

# Your own routes

```
@app.route('/about')
```

```
def about():
```

```
    return render_template('about.html')
```

```
@app.route('/contact')
```

```
def contact():
```

```
    return render_template('contact.html')
```

# Passing in data to routes

```
@app.route('/posts/<post_id>')  
def posts(post_id):  
    # do something with post_id  
    return render_template('about.html')
```

Linking this route with `url_for()`:

```
<a href="{ url_for('posts', post_id=4) }">View Post</a>
```

# Templates

## In layout.html:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width">
  </head>
  <body>
    {% block content %}{% endblock %}
  </body>
</html>
```

## In home.html:

```
{% extends 'layout.html' %}
{% block content %}
  <h1>Hello World!</h1>
{% endblock %}
```



# Templates

## Using a for loop:

```
{% for contact in contacts %}  
  <h3>{{ contact }}</h3>  
{% endfor %}
```

## Using an if statement:

```
{% if current_user.is_authenticated %}  
  <h3>Welcome User!</h3>  
{% else %}  
  <h3>You Must Login!</h3>  
{% endif %}
```

# Next Steps

Extending your app's functionality

# Common packages used with Flask

## Flask-WTF

Allows for the easy creation and handling of web forms in flask.

## Flask-SQLAlchemy

Simple extension that allows you to connect to a variety of SQL databases with the same code.

## Flask-Login

Handles user sessions, makes authentication a breeze.

## Gunicorn

When used in conjunction with a reverse proxy (like caddy), allows for the easy deployment of the app.

## Flask-Uploads

Makes handling user file uploads simple and secure.

## bcrypt

Used for hashing and checking passwords, makes authentication much more secure.

# Interactive Demo

Let's make something!

# Questions?

**THANK YOU FOR JOINING US!**

**[discord.win hacks.ca](https://discord.win hacks.ca)**