Intermediate Python

Unit 3: Introduction to Flask

Overview

- What is Flask?
- Why are we learning this?
- The Internet, aka "the web"
- Writing Flask applications

What is Flask?

Flask is a Python library.

Flask is a library that lets us create **web applications** and **websites/webpages**.

Flask is a web framework.

Flask helps us build web servers.

Flask helps us build web servers that power our web applications.

A note on terminology

The terms **web application**, **website**, and **webpage** are all interchangeable and refer to a website that is accessed with a web browser.

Why are we learning this?

Why are we learning this?

Much of our world is powered by the web.

Why are we learning this?

Even when we're not browsing *the web* on our *browsers*, we're likely on the web.

Why are we learning this?

Everything is connected to the web: your phone, your watch, even your fridge might even be connected to the web.

Why are we learning this?

But the primary use of the web is still the usage of webpages, and this is what we'll be learning about.

Why are we learning this?

Being able to create programs that rely on *the web* or *networking* is an important part of being a software engineer.

The Internet

What is The Internet?

The Internet is a global network of billions of computers and electronic devices that are able to talk to each other.

Talking to each other

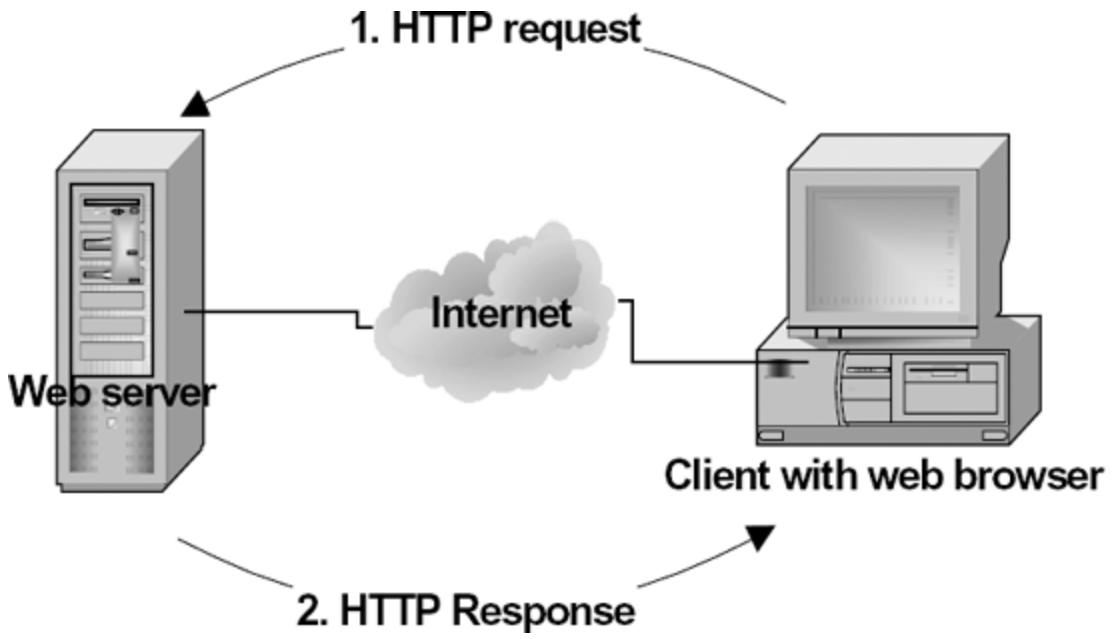
What is meant by "talking to each other" is simply the act of sending and receiving messages.

Talking to each other

The first computer sends a **request** for some data and the second computer **responds** to the request.

Terminology

- **Request**: a message sent by a computer, the sender, to another computer, the receiver.
- **Response**: a response to a message sent back from the receiver to the sender.


1. HTTP request
Internet
Web server
Client with web browser
2. HTTP Response

Let's jump into the code

Sample Flask application

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Let's break this down

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Imports

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Using imported code

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

`__name__`

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Creating an application

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Running an application

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Functions

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Whatever our function returns will be the response sent back to the client.

Whatever our function returns will be what is displayed in our browser.

Decorators

```
import flask

app = flask.Flask(__name__)

@app.get("/")
def index():
    return "Hello, world"

app.run()
```

Decorators

Decorators allow us to add functionality to out functions.