Advanced Programming with Python

Session 1

Pepe García jgarciah@faculty.ie.edu

Advanced Programming with Python. Session 1

Professor

Pepe García

jgarciah@faculty.ie.edu

Ask me anything

About the course

- 15 sessions

About the course

- 15 sessions
- 1 workgroup assignment

About the course

- 15 sessions
- 1 workgroup assignment
- Individual work (I'll be grading the Python part of your Term Integration Project)

Syllabus

SESSION 1 (FACE TO FACE)

Course presentation. in this session we will introduce the course, the syllabus, materials we're going to use, and grading system. Backend development in Python. We will understand what are the latest tools of Backend development in Python, from libraries for creating web servers, to others helping with the creation of development environments such as pipenv.

SESSION 2, 3, & 4 (FACE TO FACE)

HTTP in Python

We will use this session to understand the basics of web servers in Python using Flask, and how to create HTTP clients with Python.
We will learn the following subjects:

- HTTP Request-Response cycle

SESSION 2, 3, & 4 (FACE TO FACE)

HTTP in Python

We will use this session to understand the basics of web servers in Python using Flask, and how to create HTTP clients with Python.
We will learn the following subjects:

- HTTP Request-Response cycle
- HTTP Status codes

SESSION 2, 3, & 4 (FACE TO FACE)

HTTP in Python

We will use this session to understand the basics of web servers in Python using Flask, and how to create HTTP clients with Python.
We will learn the following subjects:

- HTTP Request-Response cycle
- HTTP Status codes
- Flask routing

SESSION 2, 3, & 4 (FACE TO FACE)

HTTP in Python

We will use this session to understand the basics of web servers in Python using Flask, and how to create HTTP clients with Python.
We will learn the following subjects:

- HTTP Request-Response cycle
- HTTP Status codes
- Flask routing
- Rendering JSON

SESSION 5 & 6 (FACE TO FACE)

HTML Templating

How do we use HTML templates with Flask.

SESSION 7 (FACE TO FACE)

Web servers - Authentication

In this session we will learn about how to implement authentication in web applications.

SESSION 8 (FACE TO FACE)

Connecting to databases

In this session we will learn how to make our Python applications connect to databases. The professor will create a video for the students to follow.

SESSION 9 (FACE TO FACE)

Connecting to databases

In this session we will review what we did in the last async session about databases, and learn a bit more about how render data from the database to HTML.

SESSION 10 (FACE TO FACE)

Case: building a Twitter clone in Python

We will use this session to do some hands-on work. We will tackle a small project in class in which we will create a Twitter clone with Python.

SESSION 11 (FACE TO FACE)

Group assignment

In this session students will do a group assignment. We will have time in class for working on it and ask questions.

SESSION 12 & 13(FACE TO FACE)

Analytical web applications - Dash

In this session we will introduce a new framework for data oriented web applications, Dash. With Dash we will be able to construct data rich applications in an easy way.

SESSION 14 (FACE TO FACE)

Deployment

In this session we will learn how to deploy our flask applications to the cloud.

SESSION 15 (FACE TO FACE)

QA session

In this session we will do a whirlwind tour over what we have learned in the course and we will have time to answer questions students may have

Grading criteria

Criteria	Score
Class participation	**15%**
Workgroups	**35%**
Individual work	**50%**

Joining the new organization

Pick yourself from the list!

https://classroom.github.com/a/yhHvHkKT

Intro

What are we really going to learn in this course?

Let's draw!

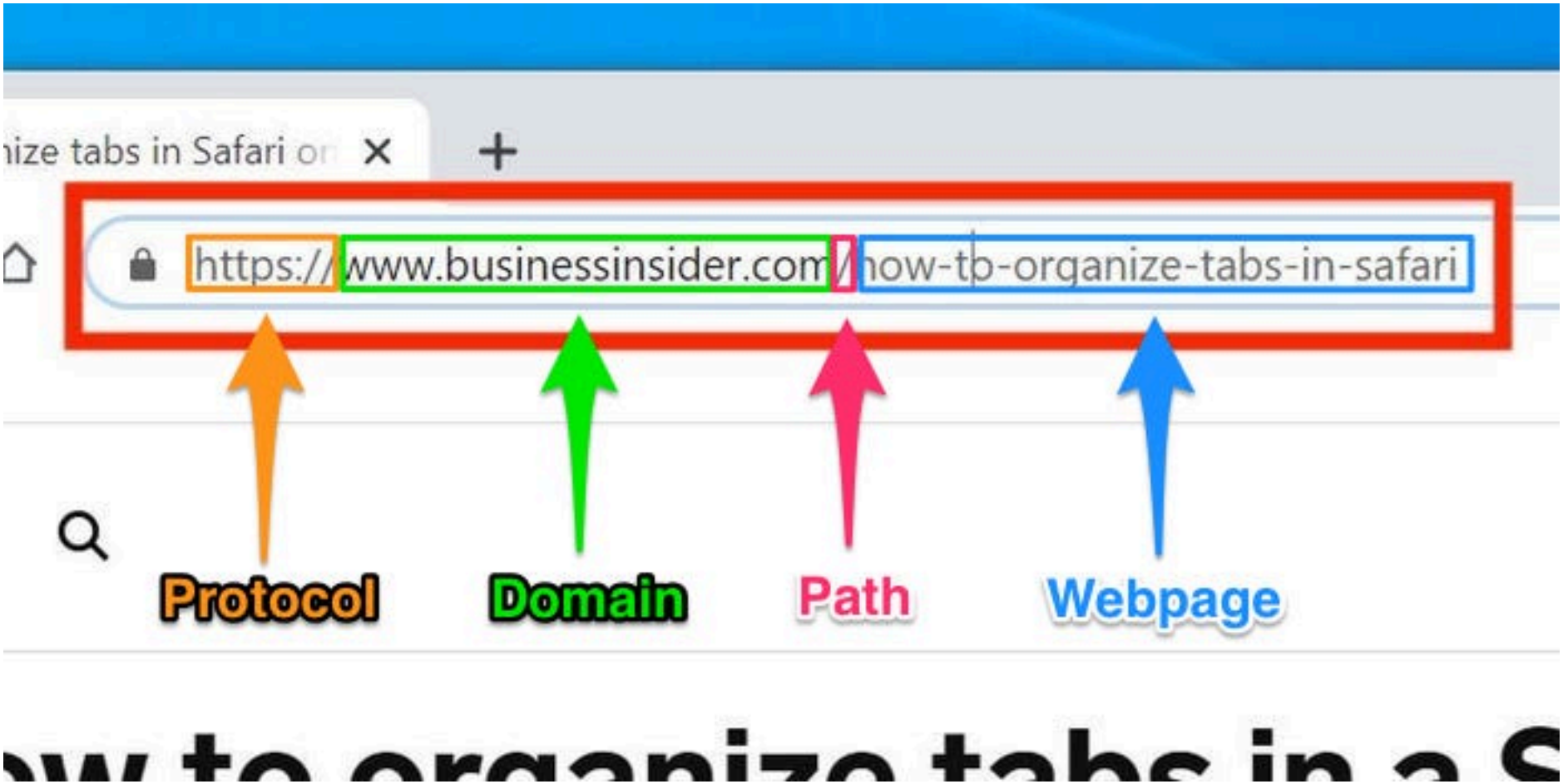
HTTP

HTTP

HTTP is a request-response protocol. HTTP **clients send requests** and HTTP **servers answer with responses**.

HTTP. URLs

Uniform Resource Locators.



HTTP

Depending on the intention of the request, HTTP describes different methods:

HTTP

Depending on the intention of the request, HTTP describes different methods:

method	**intention**
GET	read to a resource
POST	update a resource
PUT	create a resource
DELETE	delete a resource

HTTP

We will use `flask` in this course to learn and create web servers in Python.

HTTP

We will use `flask` in this course to learn and create web servers in Python.

`flask` is bundled in Anaconda already, so we don't need to download it.

HTTP - flask

example: simple flask application

how do we use ‘flask‘?

HTTP - flask

```
from flask import Flask

app = Flask("simplest server")

@app.route("/hello")
def hello():
    return "hello from the web!"

app.run()
```

HTTP routes

Our flask server can handle different routes by adding more handlers to it:

```
@app.route("/hello")
def hello():
    return "hi!"

@app.route("/goodbye")
def hello():
    return "bye!"
```