Lab 8

Today you'll be using Tornado to make a simple version of Brian Kane's YouTube Doubler. The user will be presented with a form to choose two videos and start times, and upon submission, the server will return a page that plays the two videos simultaneously.

Begin by downloading the lab 8 starter code.

Nice to meet you, Tornado
Let's examine the files we have.

`lab8.py` is the web server Python code. Edit it in your favorite text editor as if you were going to edit some HTML from previous labs.
Note: Python is an indentation-based language, meaning your code has to be neatly indented in order to function properly. You should be fine if you follow what's given, just be careful not to mix tabs and spaces willy-nilly.

The `templates` folder contains the HTML templates that Tornado will fill out when you call `self.render`.

The `static` folder contains files that don't change, like CSS. If you want to add CSS or JavaScript, you can put them in this folder.

Let's begin by starting our Tornado server.

1. Open a Terminal and navigate to the folder where you saved `lab8.py`.
2. Install Tornado. Run the command: `~cs198-tr/tornado/setup`
   This automates the installation steps from http://www.tornadoweb.org/.
3. Start your web server: `python lab8.py`
   If you want to stop it, press `CTRL+C` in the terminal.
4. Open a browser and go to http://localhost:8888/. You should see a mostly blank starter page, but not an error.
5. Your web server is working!

Part 1: Forming a form
Open `templates/index.html` in your favorite editor. We want to add a form that lets the user specify the input info for two videos. Each video should have a text input for the YouTube video ID, and another text input for the start time.

The video ID is the 11-character string of nonsense at the end of a YouTube link. For example, if you're watching `http://www.youtube.com/watch?v=WQgLNUq0ktI`, the video ID (what you'll enter into your form for this lab) is `WQgLNUq0ktI`.

Add the HTML that creates a form with those four inputs, and a submit button (refer to the lecture slides for an example). Choose whatever action you want (something like "/double" will suffice), and set the HTTP method to "get". This is because we can copy/paste and bookmark GET forms, but not POST forms, which is useful if you want to send a doubler link to a friend.

Make sure each input has `name` and `type` attributes specified, or you won't be able to access the form data on the server side.

Part 2: Routes
Towards the end of lab8.py, you'll see the section for routes. Currently there's only a single entry for `MainHandler`, which will display `index.html`.

Add another route that matches the action you chose for your form in Part 1, and hook it up to the right `Handler`.

Part 3: "It's logic, Spock. I thought you'd like that."
Right now the handler doesn't do anything, so you'll probably get an error if you try to submit the form in your browser. Let's add some logic that makes use of the user's form input.

Looking at the render call, we see that the `double.html` template uses two variables, `v1_url` and `v2_url`. In the HTML, we see that those two variables are passed as the `src` attribute of the YouTube `iframes`.

Your task is to assign those two variables, somehow incorporating the user's form data. Refer to the lecture for an example of how to access and use form data, and look through `lab8.py` for any additional things that might help you.

Part 4: Templates
Your last task is to use the `v1_url` and `v2_url` variables to actually show the YouTube players on the page. Open up `templates/double.html` and fill in the two `divs` with actual players.

Refer to this YouTube API blog post for the HTML that embeds a player:
http://apiblog.youtube.com/2010/07/new-way-to-embed-youtube-videos.html

You should use `v1_url` and `v2_url` as the `src` for the two players. There's an example of how to dynamically insert things into the page in the `<head>` of the template.

That's it! Try doubling some videos.

Submission
Since we no longer have static pages, you won't be able to leave a server running at the usual place. Instead, just leave your files in lab8, so that we can view the directory listing (lec8.py, templates, static) when visiting `http://inst.eecs.berkeley.edu/~cs198-xx/lab8/`. We'll download the files and run our own Tornadoes.

Tips
- What happens if the user leaves a text box blank? You might want to provide default values, especially for the start time.
- If you have any syntax errors, Python will tell you (in the terminal where you ran `python lab8.py`).
- PLEASE MUTE YOUR SPEAKERS OR USE HEADPHONES. There's nothing worse than 30 Macs blasting nyan cat during lab.