Un gran poder conlleva una gran responsabilidad

html forms

Python/Flask

Last talk that doesn’t involve databases
Good news:

Lots of people who know stuff in this class. (ask here or post on piazza)
Schedule

❖ today: flask & forms
❖ next class: flask and postgresQL
❖ next Tuesday: RAT
❖ next Thursday: 1st Lightning round (need access to secure network and good familiarity w/ postgres.)
git clone https://github.com/zacharski/AlienAbductions2.git

Alien Abduction
Code disponible par courrier git

forms
Esto es fácil.

This is easy

But here is an even simpler example…
One ‘submit’ button

- The world’s simplest form …
и результат

and the result

and now the code behind the magic…
The code ...

<form method="post" action="/simple2">
<h2>Don't Press This Button.</h2>
<p>
  <input type="submit" value="Press Me" />
</p>
</form>
The code ...

```html
<form method="post" action="/simple2">
  <h2>Don't Press This Button.</h2>
  <p>
    <input type="submit" value="Press Me" />
  </p>
</form>
```
<form

- method="get": data limited to 256 characters; not secure
- method="post": unlimited data; more secure
From the HTML specs

- If the processing of a form is idempotent (i.e. it has no lasting observable effect on the state of the world), then the form method should be GET. Many database searches have no visible side-effects and make ideal applications of query forms.

- If the service associated with the processing of a form has side effects (for example, modification of a database or subscription to a service), the method should be POST.
<form method = "post" action = "/simple2">
<h2>Don't Press This Button.</h2>
<p>
    <input type="submit" value="Press Me" />
</p>
</form>
and the result

```python
@app.route('/simple2', methods=['POST'])
def simple2():
    return render_template('simple2.html')
```
The world’s second simplest form

- Wow. A web page that allows for user input. How novel!
The method so far

<form method = "post" action = "/something">
    ...
    <input type="submit" value="text on button"/>
  
</form>
When a user presses the submit button where does she go?

<form method = "post" action = "/simple4">
  <h2> Don't Press This Button. </h2>
  <p>Name:  
    <input type="text" id="someName" name="name" />
  </p>
  <p>
    <input type="submit" value="Press Me" />
  </p>
</form>
<input type="text" id="someName" name="name" />

- create a text input box
- unique identifier for this HTML element (used client side)
- name used by server when this form is submitted
then on server...

```python
@app.route('/simple4', methods=['POST'])
def simple4():
    return render_template('simple4.html',
                           name=request.form['name'])
```
forms

```html
<input type="number" name="day" min="1" max="31" step="1" value="1" size="3"/>
```
Lorem Ipsum Dolor

**checkboxes**

```python
print(request.form['bald'])
'checked'
```

[Image of a form with checkboxes and fields for first name, last name, date abducted, location, and appearance. The form also includes a slider for how scary the abduction is, with options for not scary and scary.]
checkboxes

```python
print(request.form.getlist('appearance'))
'checked'
```
print(request.form.getlist('appearance'))
[u'bald', u'almond eyes', u'blue', u'fingers']
Your task (in a team)

- clone the alien abduction repository
- get all values to display on the report2 page
- add additional input field for extra xp
- the more people that demo in a team, the more xp.