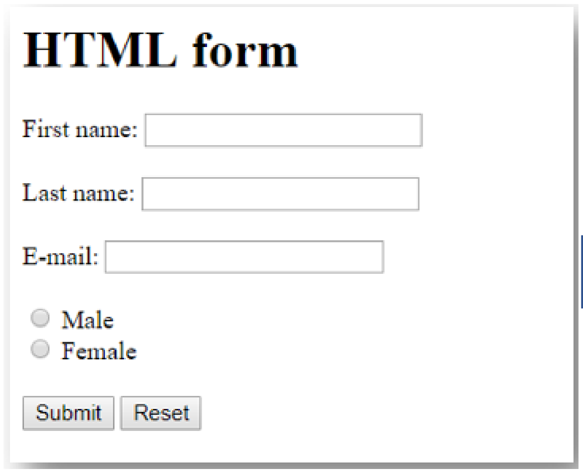
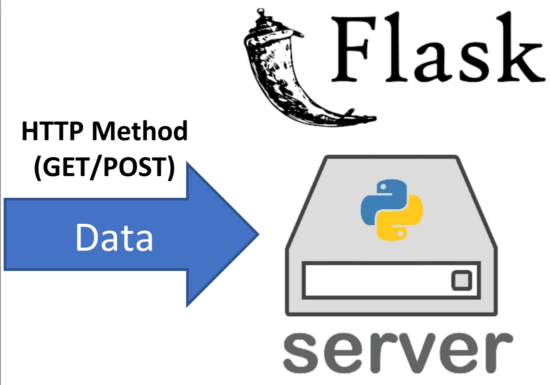
Lesson 17 – HTML Forms Processing

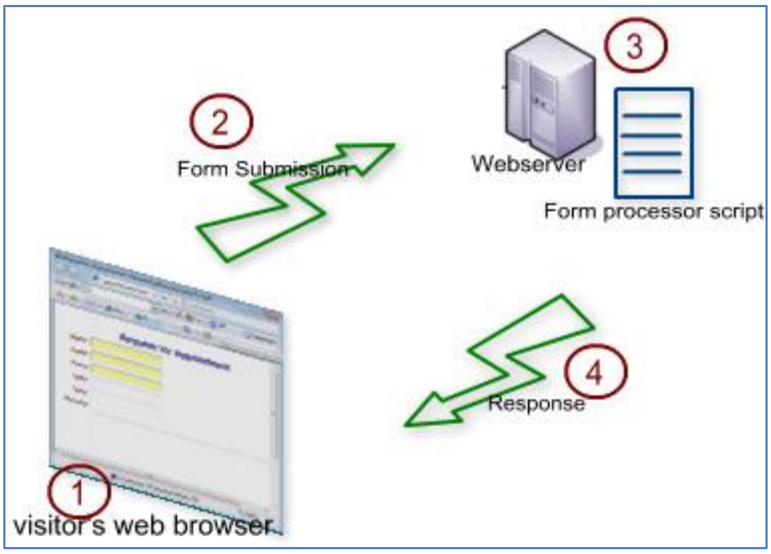
What is an HTML form?



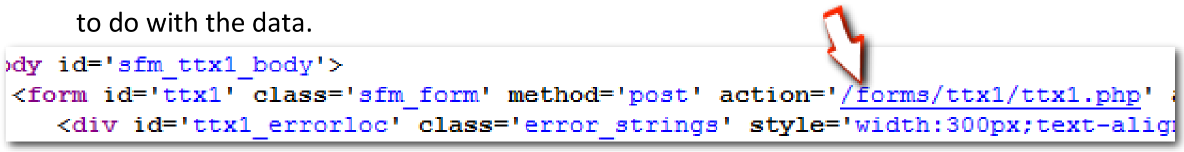


How is an HTML Form processed on the server? [1]

1. ***Your visitor loads the form page in her web browser.***
   - The browser sends a request to the web server. The web server returns the HTML page that contains the form. The HTML page returned can be an HTML file on the server or an HTML page generated by a script running on the web server (like, a PHP or Python script). The HTML page can contain references to Cascading Style Sheets, files and images. The web browser downloads them as well. Then the page is displayed.



2. ***Your visitor fills the form and submits it. The submission data are sent to the web server.***
   - Once the visitor has submitted the form, the form data is sent to the web server. In the form, the author of the form has to mention an 'action' URL that tells the browser where to send the form submission data. The 'action' usually points to the URL of a script that knows what to do with the data.



```
dy id='sfm_ttx1_body'>
<form id='ttx1' class='sfm_form' method='post' action='/forms/ttx1/ttx1.php'
   <div id='ttx1_errorloc' class='error_strings' style='width:300px;text-alig
```

3. ***The web server processes the request.***
    - The web server passes the form submission data to the form processor script (mentioned by the 'action'). The form processor script is written in languages like PHP, ASP, Perl or Python. The form processor script may send the data by email or save it into a database or a file.
4. ***A response is sent back to the browser.***
    - The form processor script sends a response indicating the success or failure of the form processing operations back to the server. The response could be to re-direct to another page.

Task 0:

- Read the tutorial on HTML Forms & complete exercises.
- Read an overview of the basic Hypertext Transfer Protocol (HTTP) Methods.

Task 1:

- Send data to the Python Flask Webserver via an HTML form.

Instructions:

0. If Flask is not installed, run `pip install flask` from the command line.
    - If the system fails to run the installation, follow the instructions provided here.
1. **Download** a script **forms_processing_example.py** from NAS.
2. Open the script in any text editor (such as Notepad++) and **examine the code**.
3. **Start the webserver** using the command `python forms_processing_example.py`
    - If the system fails to run the script, follow the instructions provided here.
4. Open an index **webpage** in your **browser** at http://localhost:80
5. **Submit data** to the server using the **GET method**.
    - Look at the response page URL - Do you see the data sent???
6. Return to an index page at http://localhost:80
7. **Submit data** to the server using the **POST method**
    - Look at the response page URL - What is the difference???

Task 2:

- Create an online HTML calculator with actions bound to buttons (+ - * /) that can send the inputs to the server, process it and return the results.

Instructions:

1. Download a script **calculator.py** from NAS.
2. Open the script in any text editor (such as Notepad++) and **examine the code**.
3. **Start the webserver** using the command `python calculator.py`
4. Open the **calculator page** in your **browser** at http://localhost:80/calculator
5. Edit the code in **calculator.py** in order to add support for other arithmetic operations (- / % ^):
    a. First, add a new **submit button** to the calculator page HTML with an operation name.
    b. Then implement the added operation inside the **compute()** function.
    c. **Check** if the implementation works at http://localhost:80/calculator
    d. Implement the remaining operations (- / % ^).
