香港中文大學
The Chinese University of Hong Kong

CSCI2720 2023-24 Term 1: Building Web Applications

Lab 3: Form and Fetch

Dr Colin Tsang

CSE account for students

- CSE account is now provided for non-CSE students to use lab computer and CSE server.
- You should have received an email from the CSE tech team regarding this issue.
- You can use them during this term, and files on Desktop will be retained throughout.
- For CSE account problems, please contact techteam@cse.cuhk.edu.hk using your CUHK email account.

Outline

- Understand a sample
  - Bootstrap components with form
- Improve a web form
  - Basic input and textarea
  - Radio buttons
  - Button and event
- JavaScript fetch
  - Local web server
  - Save and load the comments

Skeleton code

- Download the HTML file *lab_03.html* from BlackBoard.
  - Read the code comments and try to understand it.

- Just a set of ** ** with two boxes in a flex layout
  - An SVG circle
  - A paragraph with a heading
  - The beginning of a form

Form

- You will work on this file with your favourite text editor.
- You can find an HTML form **<form>** below the list of comments with:
  - Email
  - Comment
    - Paragraph of text using **<textarea>**
  - Color (to be used by SVG)
    - You can try other colors later

Form

- The email and comment boxes came from this website: https://getbootstrap.com/docs/5.2/forms/form-control/

- Look at the classes *form-label* and *form-control*
- Observe the correspondence between *for* of **<label>** and *id* of **<input>**

Form

- A color choice is available between comment box
  - Only the radio button for the red color is built for you
  - Your task is to provide more choices of color
    - Repeat *div*/*input*/*label* with different *id* but same *name*.

- You can try different Bootstraps
  - See: https://getbootstrap.com/docs/5.0/forms/checks-radios/

colintsang@cuhk.edu.hk

This lab is so fun!

Add your comment:

Email address

name@example.com

Comment

- Red
- Blue
- Yellow
- Pink

Add comment

Button and Event

- A button is built for submitting the comment
- It is linked to an *onclick* event (as an attribute) for this button
- When this button is clicked on, the JS engine will run the **processform()** function

```
<button type="button" class="btn btn-primary" onclick="processform()">Add comment</button>
```

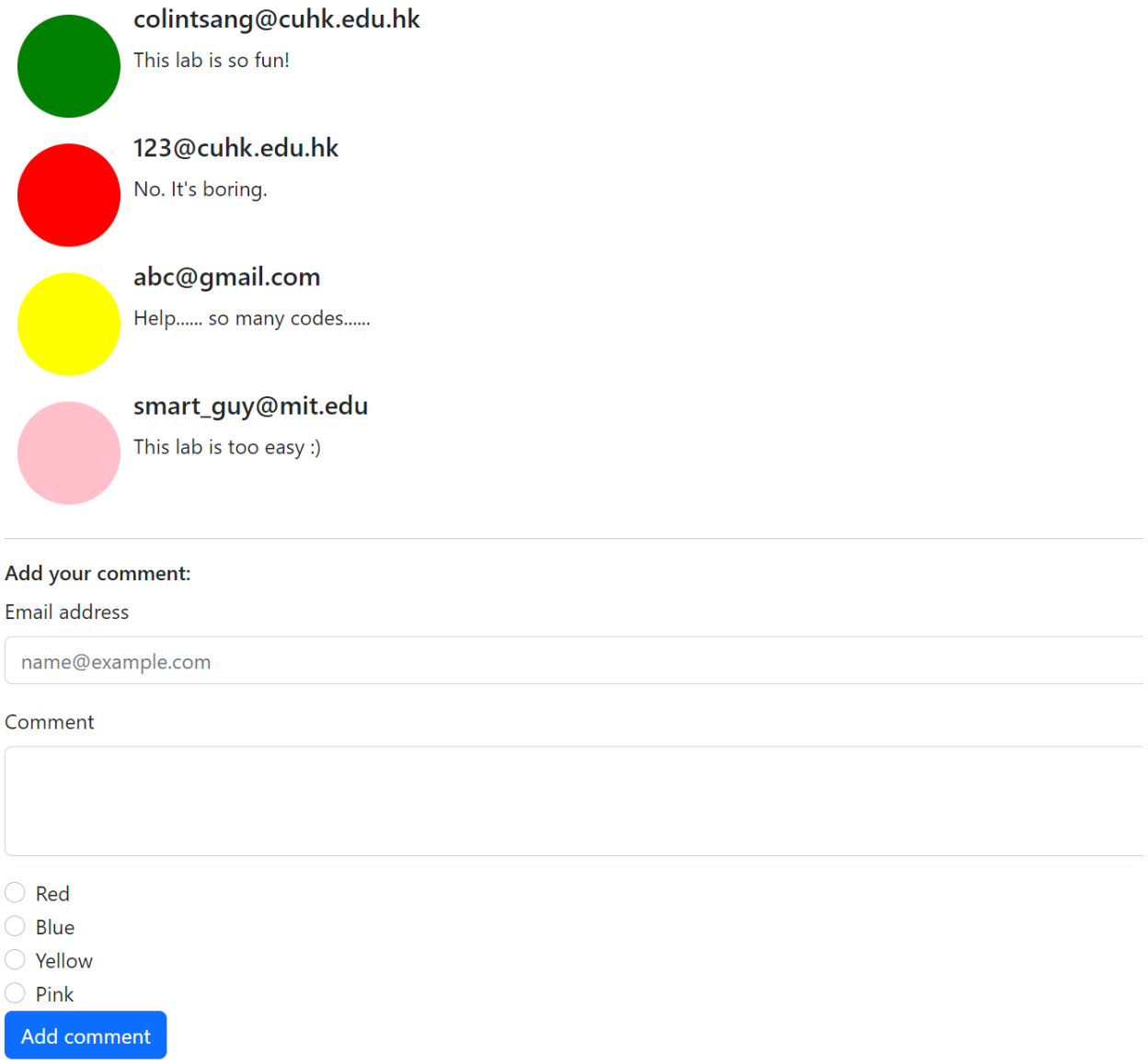
Setup JS

- Create a new external JS file to be used, e.g., *script.js*
  - `<script src="script.js"></script>`
- In the JS file, create your event handler **processform()**
  - Test the button in console, does it work?
  -
```
function processform() {
    console.log("testing");
}
```

Event handler

- Your task is to use the **processform()** function to perform this:
- When the user adds a new comment, it will appear on top of the website.

- *This task is similar to Problem 3 - Task 2 in your assignment one.*



Event handler – prepare comment element

- Set up a new element

```
let newComment = document.createElement("div");
let element = '<div><svg height="100" width="100"><circle cx="50"
cy="50" r="40"></svg></div><div><h5></h5><p></p></div>';
newComment.innerHTML = element;
```

- Set the classes of the div and its children div's. className is used here

```
newComment.className = "d-flex";
newComment.querySelectorAll("div")[0].className = "flex-shrink-0";
newComment.querySelectorAll("div")[1].className = "flex-grow-1";
```

Event handler – prepare comment element

- Increment the comment id. Note that *#comments* refer to the id *comments*.

```
let lastComment = document.querySelector("#comments").lastElementChild;
newComment.id = 'c' + (Number(lastComment.id.substr(1))+1);
```

Event handler – apply contents from form

- Change contents <h5> and <p> according to form input with id

```
    newComment.querySelector("h5").innerHTML =
document.querySelector("#new-email").value;

    newComment.querySelector("p").innerHTML =
document.querySelector("#new-comment").value;
```

- Get the color choice from the radio buttons

```
    let color = document.querySelectorAll("input[name=new-
color]:checked")[0].value;
```

Event handler – add a new comment

- Change the fill color of the SVG circle

```
newComment.querySelector("circle").setAttribute("fill",
color);
```

- Append it to the div #comments

```
document.querySelector("#comments").appendChild(newComment);
```

- Reset the form to clear the contents

```
document.querySelector("form").reset();
```

colintsang@cuhk.edu.hk

This lab is so fun!

good_student@cuhk.edu.hk

I followed all the steps and finally finished!

Add your comment:

Email address

name@example.com

Comment

- Red
- Blue
- Yellow
- Pink

Add comment

Before we move to the next topic…

- In this add comment task, the new content is generated entirely by JS
  - In fact, you can build the whole website by JS in the first place

- All new comments will be lost after refresh

- If you forgot SVG:
  - https://developer.mozilla.org/en-US/docs/Web/SVG/Tutorial/Basic_Shapes

JS Fetch

- From BlackBoard, download the *lab_03_file.txt*.
  - Place it in the same folder as your *lab_03.html*.

- In the HTML file, create a new button for *load file*
  - It can be next to the *add comment* button
  - Create an *onclick* event for it

```
<button type="button" class="btn btn-primary" onclick="loadfile()">Load File</button>
```

  - Setup the event handler **loadfile()** and test the button

```
function loadfile(){
    alert('testing');
}
```

JS Fetch

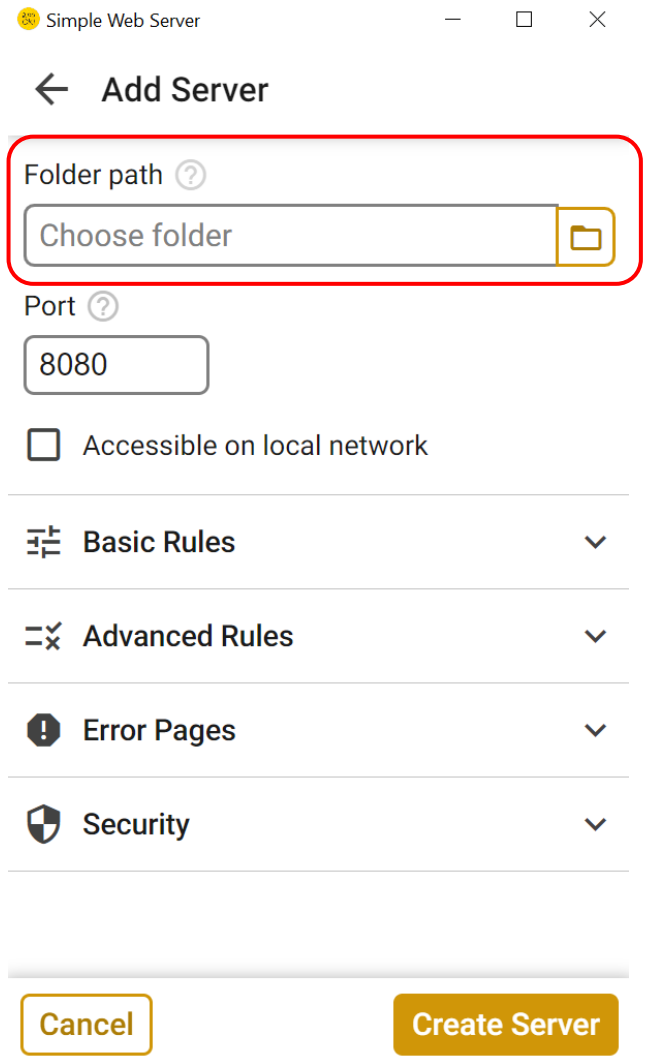
- Now, try using this **laodfile()** function for loading.
- Can you obtain the contents in *lab_03_file_txt* ?

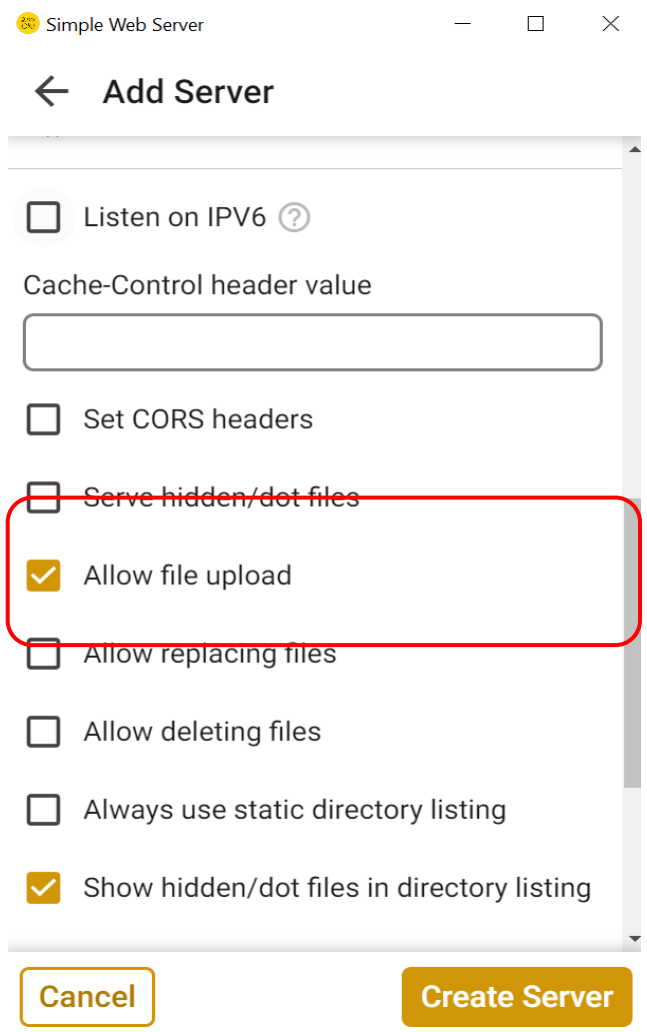
- You will not be able to fetch a file in this way.
  - Google Chrome's security setting doesn't allow using **fetch()** on local file system.
  - In this case, you must upload your file onto a web server to try, but it is inconvenient for the developers.

- For *local development*, you can install a local web server.
- There are multiple ways to set up a local web server.

Simple web server

- Download and install: https://simplewebserver.org/
- Set up a new local server
  - You just need to *choose a folder* to start it.
  - Put all your files in this lab there.
  - In *Advanced Rules*, click *Allow File Upload*
  - "Port" 8080 is a point of entry by default





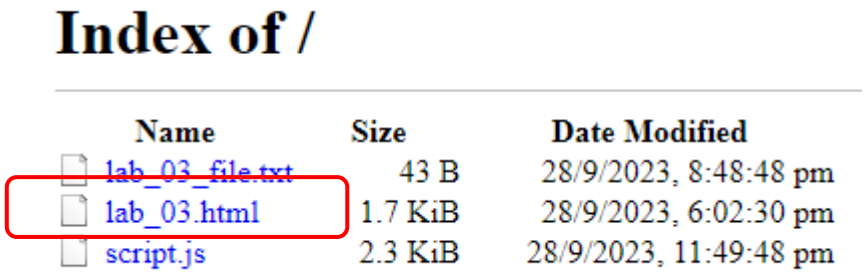
Simple web server

- Visit your server at: http://localhost:8080/
  - This address refers to localhost
  - Or: http://127.0.0.1:8080/, they are the same.
  - Include your folder names if necessary

- Many other ways can build a local server:
  - Most of them have security concerns.
  - Most of them do not support save/upload with PUT/GET/POST.
  - For consistency, you are recommended to use the simple web server in this lab and the assignment

Download file by Fetch

- Now, open your HTML file from the local web server.
- Do not open it directly, as it will open it as a local file instead of a server file.



- And as usual, use your text editor to update the code.
- The only difference here is that we need to view the result from the server, not locally.

Download file by Fetch

- In your script.js:

```
function loadfile(){
    let contentElement = document.getElementById("new-comment");

    fetch("lab_03_file.txt") // or absolute address http://127.0.0.1:8080/lab_03_file.txt
      .then(response => response.text())
      .then(data => {
        contentElement.textContent = data;
      })
      .catch(error => {
        console.error("Error fetching data:", error);
      });
}
```

colintsang@cuhk.edu.hk

This lab is so fun!

Add your comment:

Email address

name@example.com

Comment

This is a testing sentence to be displayed.

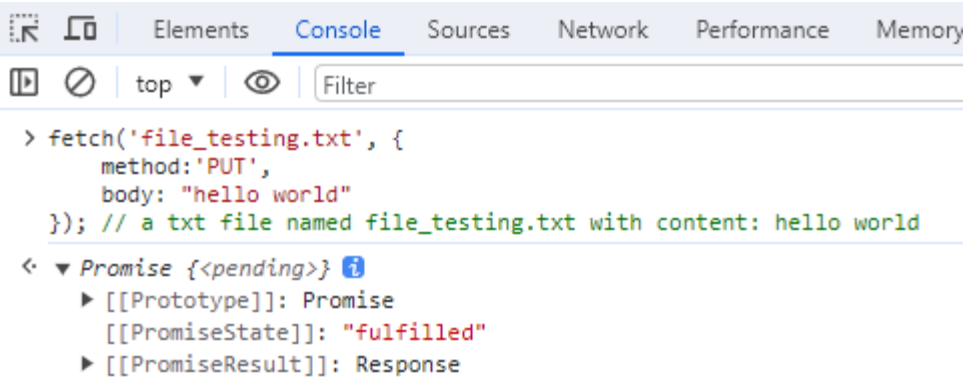
- Red
- Blue
- Yellow
- Pink

Add comment | Load File

Writing to the server

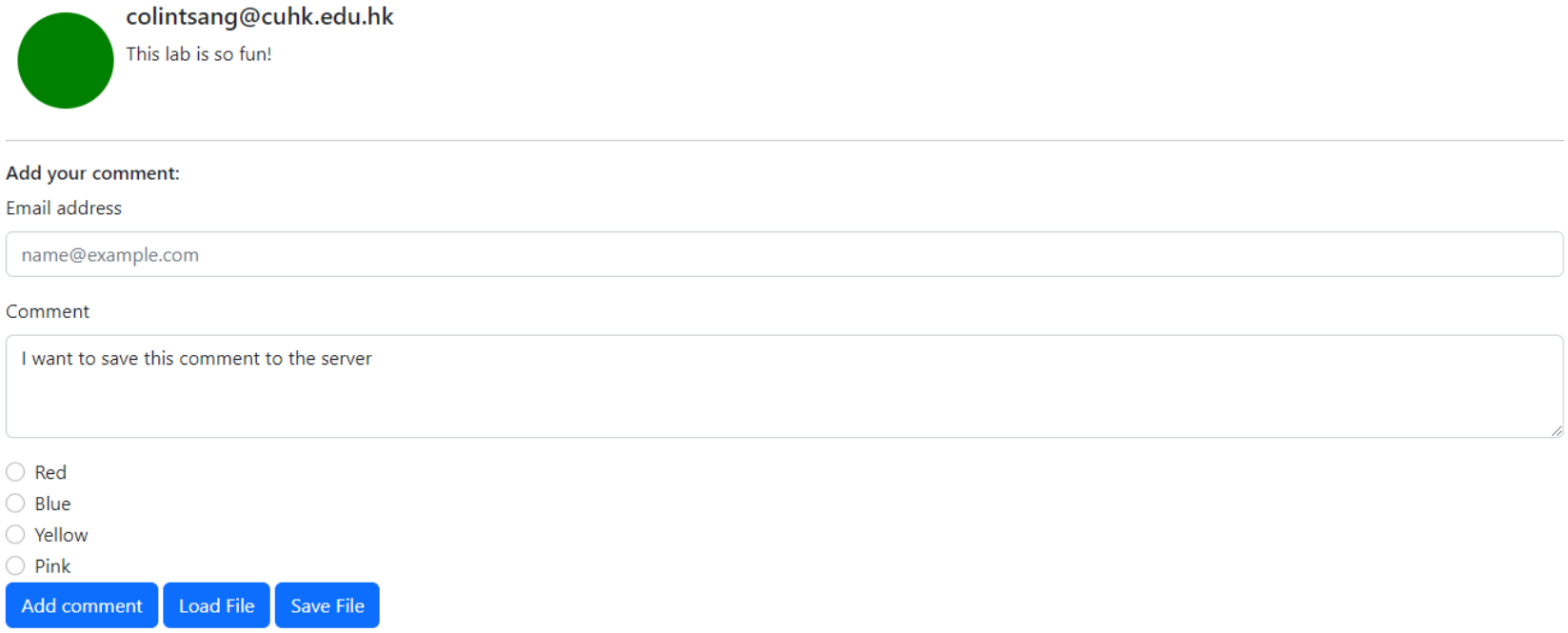
- Users can upload data to the server too.
  - A specific script is needed to handle the data.

- The *PUT* method is the easiest way to upload data, as no script is needed.
- You can use fetch directly in the console.
- You will see a new file appear in your server (i.e., your selected folder in *simple web server*).



Save file by fetch

- Can you create a save file button, so that when it is clicked, the comment is uploaded to the server as a txt file?



Save file by fetch

- Hints:
  - Use the *.value* property to extract the data from the HTML element.
  - Try to use **fetch()** with the *PUT* method following the example, but what should be in the *body* this time?

- Solution code will be available on BlackBoard later.
- There are many other ways to finish **processform()**, **loadfile()**, and **savefile()**.
- Try not to copy my code. Do it by yourself !

Submission

- No submission is needed for labs
- What you have done for this lab maybe useful for your assignments and project
- Please keep your own file safely