COMP4021
Internet Computing

Using HTML Forms

Gibson Lam

Using HTML Forms

- You have started to create some server-side programs using Node.js

- Those programs return information only based on the path of the request, i.e.:

$$\texttt{app.get(}\boxed{\texttt{"/useful\_data"}}\texttt{, ...);}$$

- To make them more useful, you can use HTML forms to send data to them from the browser

HTML Forms

- HTML forms work like paper forms for filling in information

- An HTML form contains some form elements enclosed within `<form>…</form>`

- When the form is submitted, the data inside it is sent to the server, to the destination set in the `<form>` tag

An HTML Form Example

- Here is a simple form example:

```
<form method="get" action="/target">
    <p>Firstname:
    <input type="text" name="firstname"></p>
    <p>Lastname:
    <input type="text" name="lastname"></p>
    <p><input type="submit"></p>
</form>
```

Firstname: [   ]

Lastname: [   ]

[Submit]

Attributes in the Form

- The `<form>` tag contains two attributes:

  method   The HTTP method used by
            the HTTP request, which can
            be GET or POST

  action   The URL of the destination

- In the previous example, the HTTP method is GET and the destination is the relative URL `/target`

```
<form method="get" action="/target">
```

Form Elements

- You can use many form elements for data filling such as:
  - Text inputs
  - Number inputs
  - Dropdown boxes

- These inputs usually have a name attribute

- You also use buttons to perform some actions such as submitting the form

Username:

Password:

Message:

A cat sat on a mat.

☐ Important!

Type:
◉ Request ○ Inquiry

Role:
Year 1

Email:

Simple Text Inputs

- You can create simple text inputs using `<input>`, for example:

  - Text field:

    ```
    <input type="text"
           name="username">
    ```

    Username:

  - Password field:

    ```
    <input type="password"
           name="password" maxlength="12">
    ```
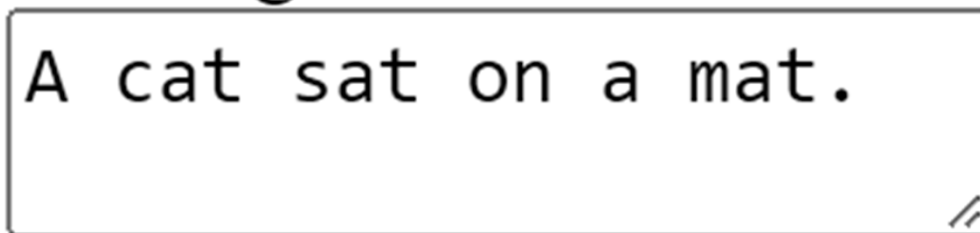
    Password:

  If you want to, you can use the `maxlength` *attribute to restrict the length of the input*

Text Area

- Instead of a single-line text box, you can make a text area using `<textarea>`, like this:

```
<textarea name="message">A cat sat
on a mat.</textarea>
```

Message:

```
A cat sat on a mat.
```

- One way to make a bigger box is to adjust the `cols` (no. of columns) and `rows` (no. of rows) attributes

Making a Text Area Bigger

- For example, you can make the text area to have 80 columns and 25 rows:

Message:

A cat sat on a mat.

```
<textarea
  name="message"
  cols="80"
  rows="25">A cat sat on a mat.</textarea>
```

Checkboxes and Radio Buttons 1/2

- Checkboxes and radio buttons are similar form elements, except that radio buttons can be grouped together

- They both can have a `value` attribute which is sent when the form is submit

- Here is an example of a checkbox:

```
<input type="checkbox"
       name="important">
Important!
```
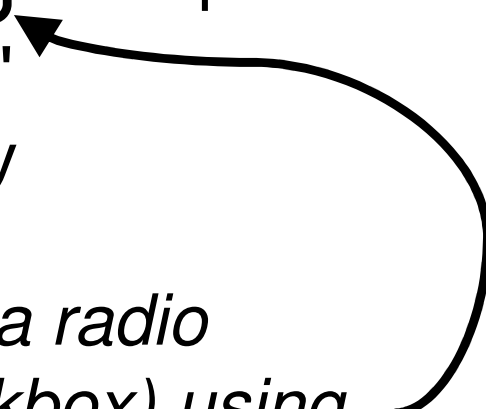
☐ Important!

Checkboxes and Radio Buttons 2/2

- Here is a group of radio buttons, having the same name:

Type:
◉ Request ○ Inquiry

```
<input type="radio" name="type"
      value="request" checked > Request
<input type="radio" name="type"
      value="inquiry"> Inquiry
```
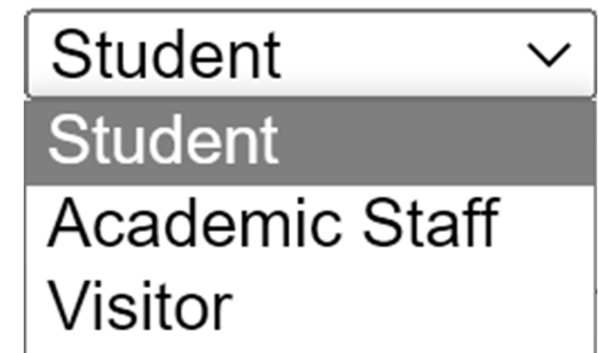
You can select a radio button (or checkbox) using the checked attribute

Drop Down Lists

- Using a combination of `<select>` and `<option>`, you can make a drop down selection

- For example, here is a drop down list with three options:

```
<select name="role">
  <option>Student</option>
  <option>Academic Staff</option>
  <option>Visitor</option>
</select>
```
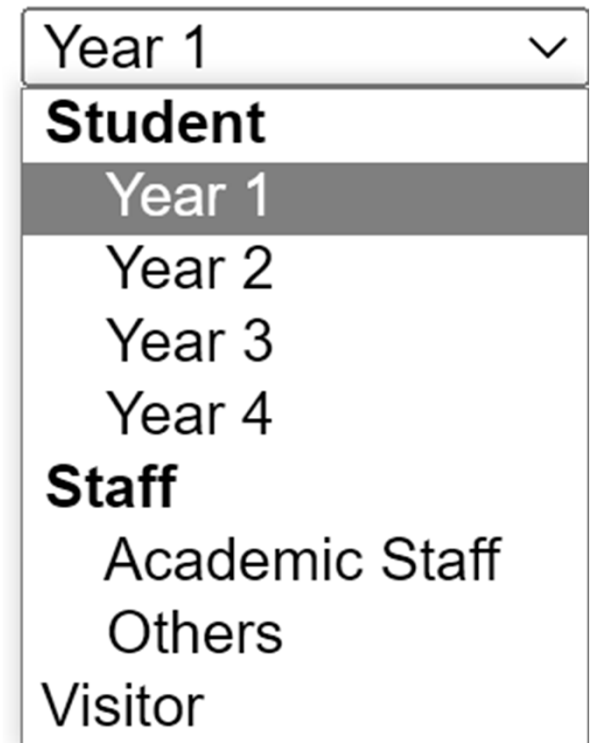

Role:

Student ⌄

Student
Academic Staff
Visitor

Option Groups

- You can also arrange the options in a structural way using option groups, for example:

```html
<select name="role">
  <optgroup label="Student">
    <option>Year 1</option>
    <option>Year 2</option>
    <option>Year 3</option>
    <option>Year 4</option>
  </optgroup>
  <optgroup label="Staff">
    <option>Academic Staff</option>
    <option>Others</option>
  </optgroup>
  <option>Visitor</option>
</select>
```

Role:

Year 1 ⌄
Student
Year 1
Year 2
Year 3
Year 4
Staff
Academic Staff
Others
Visitor

Hidden Fields

- Hidden fields are text fields that do not visually shown on a webpage, i.e.:

```
<input type="hidden" name="from"
        value="http://www.ust.hk">
```

- You use them to send some useful data to the server

- Typically, they are created by the server-side programs, or by JavaScript

HTML5 Inputs

- The form elements we have seen so far are the common and older ones

- HTML5 introduces a collection of new input fields by giving you different input types:

  - `type="email"`
  - `type="url"`
  - `type="number"`
  - `type="range"`

  - `type="date"`
  - `type="time"`
  - `type="color"`

Email and URL Inputs

- Email and URL input fields looks just like ordinary text fields, as shown below:

```
<input type="email"
       name="email">
<input type="url"
       name="website">
```

Email:

Website:

- They are particularly useful for mobile devices when showing the keypad, or for data validation

Input Fields Validation

- If you enter some invalid email or URL into the fields, when you submit the form, the fields will validate the content for you, like these:

Email:

Gibson Lam

! Please include an '@' in the email address. 'Gibson Lam' is missing an '@'.

Website:

HKUST

! Please enter a URL.

Number and Range Inputs

- You can use spinners and sliders for number adjustment, e.g.:

```
<input type="number" name="age" value="15"
      min="0" max="100" step="1">
<input type="range" name="height"
      min="50" max="200" step="5">
```

Age:

15

Height:

Date and Time Input

- You use the date and time input fields to give you date and time control

```
<input type="date"
       name="exam-date">
<input type="time"
       name="exam-time">
```

Colour Input

- Colour input shows a box with a colour inside, as shown below:

Your Favourite Colour:
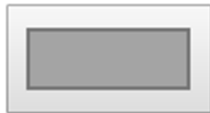
```
<input type="color"
             name="favourite">
```

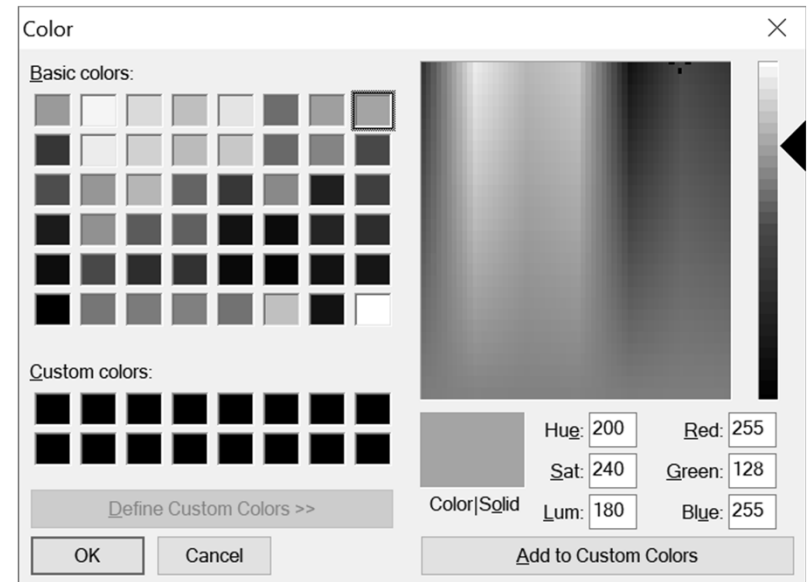- The major advantage for the input field is that you will get a colour picker if you click on the field

Additional HTML5 Attributes

- There are some new attributes for form elements from HTML5 such as:
  - `placeholder` — This is a 'hint' of what you need to do for a text field
  - `required` — Setting the field as a required form field

Favourite course:

Type COMP4021 here!

! Please fill out this field.

```
<input type="text" name="course"
  required
  placeholder="Type COMP4021 here!">
```

  - `autofocus` — Setting the field to be the focus when the page is loaded

Using a Form Button

- A form typically has a submit button, which can be created using this HTML:

  `<input type="submit">`

- If the submit button is clicked, the form will then send the data to the server

- You can also use `<button>...</button>` to create a submit button, i.e.:

`<button type="submit">Submit</button>`

Sending Data to the Server

- A form can send the data to the server using the GET method or POST method
  - When using the GET method, the form data is appended to the end of the URL
  - When using the POST method, the form data is sent in the HTTP request body

- We will focus on the GET method in this discussion

The Query String

- The form data sent using the GET method is called the *query string*

- It is text data containing name and value pairs separated by ampersands (&)

- Here is a simple query string in an URL:

```
https://www.google.com/
        search?q=superman&tbm=isch
```
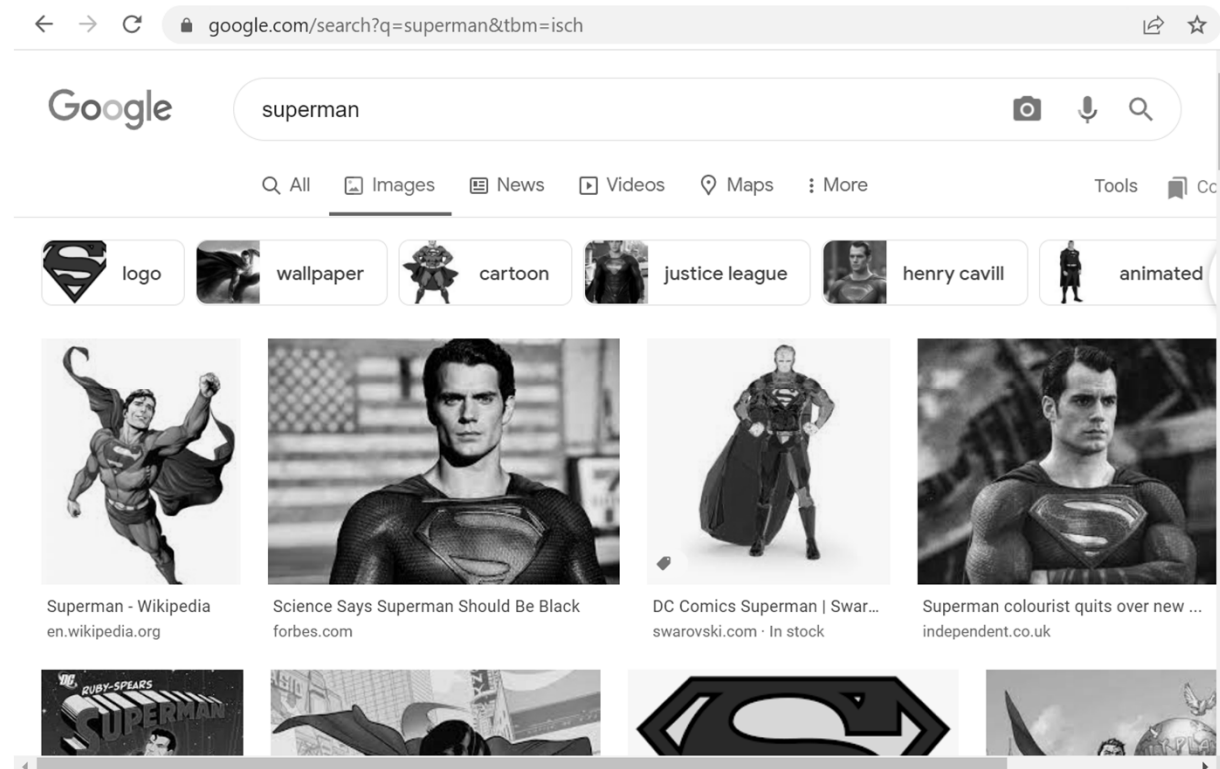
*The question mark (?) separates
the URL and the query string*

The Google Query String

`https://www.google.com/`
`        search?q=superman&tbm=isch`

- The Google query string has two parameters:
    - `q` is the search term
    - `tbm` indicates the search is an image search (`isch`)

Query String From a Form

- Using this form as an example:

  Firstname: Gibson

  Lastname: Lam

  Submit

- If the form is submitted, i.e. clicking on the submit button, the query string will look like this:

```
.../target?firstname=Gibson&lastname=Lam
```

The name and value pairs come from the form elements

Reading Query String in Express

- You can easily get the query string in Express using the query object from the request, e.g.:

```
app.get("/target", (req, res) => {
  const { firstname, lastname } = req.query;
  res.json({ firstname, lastname });
});
```

- The query object contains the name/value pairs of the query string

Using the Example

- In the example, if you enter these values in the form and then click 'Submit', you will see this output from the browser:

Firstname: | Tai Man

Lastname: | Chan

Submit

← → C ⓘ localhost:8000/target?firstname=Tai+Man&lastname=Chan

```
{"firstname":"Tai Man","lastname":"Chan"}
```