School of Computer Science

Web and Database Computing 2019

Lecture 10: Serving Static and Dynamic Content

adelaide.edu.au

seek LIGHT

Setting up our server

Introduction to the HTTP server side

So far we have only been using the client

- All our files have been local and accessed directly
- All our code has run in the browser

We'll now start using a server

We will be using

Node.js

in this course

(NOT php)

What you'll need

NodeJS 10

- Windows & Mac users, download at https://nodejs.org/en/download/
- Linux users follow instructions at https://nodejs.org/en/download/package-manager/

Building a server with Express/Node.js

Node.js is becoming increasingly popular for web applications

- Javascript
  - One language client and server
  - Event driven
  - Familiar programming style
- Asynchronous
  - Performance (does not have to wait for slow operations to complete)
  - Using 'call back' functions (called when the operation completes, this allows the server to deal with other tasks until the call back is triggered.)

Express

Express is a web applications framework for Node.js

- Its purpose is to make it easy to write web applications in Node.js

Setting up a basic Express server is easy!

- The express generator will set up your Node.js directory structure for you
  - Install using `npm install -g express-generator` (may require sudo/admin privleges)
  - Run the command `express`
- We can then modify the files that the express generator created for us.

Setting up a basic webserver using Express

Demo

- Use the `express` command to setup our basic server

What has express created for us?

- **`bin`** contains program files used to run our server
- **`public`** contains files we can serve statically
- **`routes`** contains Node.js code files for serving dynamic content
- **`views`** contains template files
- **`app.js`** is the main Node.js application file
- **`package.json`** contains a metadata about our app

Preparing our basic Express server

Demo

- Use the `npm install` command to install needed files for our basic server

What has npm created for us?

- `node_modules` contains library files used to run our server
- `package-lock.json` locks the versions of libraires used

Serving files statically

Demo

- Start the server with `npm start`
- Your server is accessible at http://localhost:3000
- Place files in the public folder to make them available

What was being served?

- Files placed in the public folder would be available at the corresponding path
  - A file created at `public/test.html` will be available at http://localhost:3000/test.html
  - A file created at `public/css/test.css` will be available at http://localhost:3000/css/test.css

Quiz!

Refresher:

- 5 questions in the next 5 slides
  These do **not** appear in the PDF of the slideshow
- Answers in the online quiz visible after all 3 attempts
- 3 attempts at the quiz
- Keep highest mark
- Can be completed any time in the next 24h
- 0.5% of your final grade

Q1

Which of the given are advantages of client-side dynamic programming?

Q2

Which of the given is correct Javascript for a FOR loop that iterates over an array and prints the result to the debugging console?

Q3

Consider this HTML:

```html
<p id="p1" class="a">Paragraph 1</p>
<p id="p2" class="a b">Paragraph 2</p>
<p id="p3" class="b">Paragraph 3</p>
```

Which of the given Javascript statements will select ONLY the paragraph `Paragraph 2`?

Q4

Which of the given JavaScript conditionals will evaluate to **true**

Q5

Which of the following is correct Javascript to create a `<div>` element and add it as the last element in the body?

< /quiz >

Serving Dynamic Content

Dynamic view counter

Say we want to create a counter that shows how many times a button has been pushed.

- Write a basic HTML page
  - Include a counter and a button that increments the counter.
- What will happen if we now serve this from express as a static page?
- What if we wanted the counter to represent the number of times the page was loaded rather than the number of times the user pressed the button?
- What if we wanted the counter to represent the number of times anyone in the world pressed the button on the page?

State information

The information needs to be remembered somewhere!

Options:

- Store the value of the variable on the web server in our Javascript
- Save the value in a file on the server and read/write as needed
- Store the value in a database and query/update as needed

However the data is stored, it needs to be inserted into the web page before we send it to the client.

Our express files

- **`routes`** contains Node.js code files for serving dynamic content
- **`routes/index.js`** contains basic dynamic routes

Routes

Routes are special functions that we can define on our server to perform actions when a given path is requested.

```
router.get('/some/path', function(req, res) {
    // Do stuff
});
```

http://expressjs.com/en/api.html#router.METHOD

Our first route

Demo

- Modify the `routes/index.js` file to add a new custom `GET` route at `/count` that prints the number of page visits

Sending a response

We don't want to just print the output, we want to send an HTTP response to the client!

- The `res` object that express passes to our function helps us here.
    - `router.get('/test', function(req,`**`res`**`) {`
    - `res` represents the HTTP response that express sends.
- One of the methods of the response object res, is the send method.
    - `res.send([body])`
    - The body is the HTTP response body. Can be a string (html web page), or data.

```
router.get('/test', function(req, res) {
    res.send("This is a test");
});
```

Let's use `res.send()` to send our message to the browser instead of the console.

http://expressjs.com/en/api.html#res

A route that sends a response

Demo

- Modify the `routes/index.js` file's route at `/count` to send the number of page visits as a response.

THE UNIVERSITY
of ADELAIDE

What's happening

Due:

- Prac Exercises 1, 2, 3 all DUE Friday if not already completed

This week:

- Introduction to AJAX

Further learning:

- Download and install Node.js
- Try setting up your first Express server.
- Keep using HTML and CSS in your forum posts.