香港中文大學
The Chinese University of Hong Kong

SPA AND ROUTING

CSCI2720 2022-23 Term 1

Building Web Applications

Dr. Chuck-jee Chau
chuckjee@cse.cuhk.edu.hk

OUTLINE

- Page-based navigation
- Single-page apps
- Routing in app
- React-router

FROM WEB PAGES TO WEB APPLICATIONS

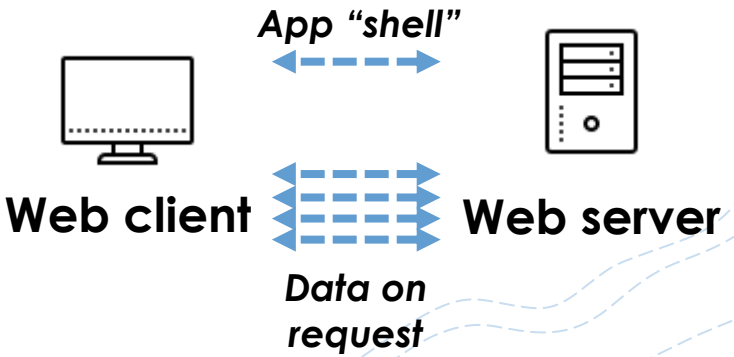
- The web originated as a system for managing ***static content pages with hyperlinks***
  - Consider your experience with Wikipedia articles
- And then the web evolved with the use of CSS, JavaScript, and all kinds of ***dynamic contents***
- ***"Web Applications"***
  - Presenting your contents/services in an environment like native applications, yet in a browser

PAGE-BASED NAVIGATION

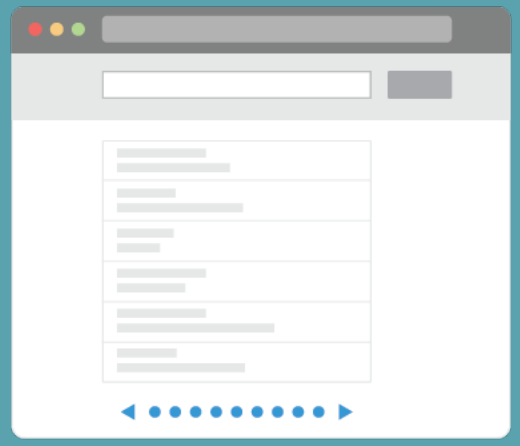
- Bandwidth waste by page structures and styling being ***reloaded for every page***
  - Caching may help
- The loading of a page (*e.g.*, blanking out) gives a poor experience for the user
  - *"White screen of hell"*
  - ***User Experience*** (UX) became a term so ubiquitous when the web becomes part of everyone's life

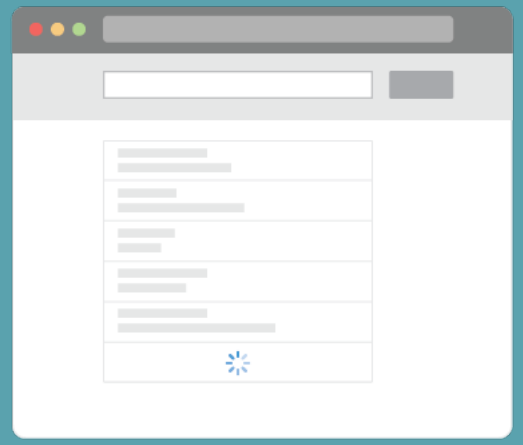
SINGLE-PAGE APPLICATION (SPA)

- Ajax techniques became widely used around 2005
  - Asynchronous retrieval of data, without reloading
  - Now developers are gradually shifting to use the *Fetch API*
- The paradigm of SPA easily got popular
  - *Initial load*: The page framework (HTML), styles (CSS) and code (JS)
  - *Subsequent loads*: Only ***fragments of data*** required to be displayed



DIFFERENT VIEWS OF SPA





See: https://uxplanet.org/ux-infinite-scrolling-vs-pagination-1030d29376f1

Pagination

- Contents are displayed based on links
- Good for locating items (on a certain page), with a ***sense of control***
- E.g., Google Search

Infinite scrolling

- Contents are displayed without a finishing-line
- New contents are loaded slightly ahead
- Good for ***content discovery***, and optimized for mobile app experience
- E.g., Instragram posts

ISSUES OF SPA

- Page performance on various devices
- Search and location of items on page
- Scroll bar does not reflect amount of actual data
- Lacking a page footer

➔ Consideration of ***user experience***!

SPA: PROS AND CONS

What's good?	*What's bad?*
Efficient use of bandwidth	Browser history is not trivial, not easy for bookmarking
Separation of view and data ➔ easier for maintenance	Search engine optimization: contents not easily retrieved by robots
Imitating native applications and improved user experience without obvious reloading and waiting	JavaScript dependent ➔ thin server, thick client

ROUTING IN THE APP

- Modern JS frameworks easily support SPA by allowing change of contents in ***components***
- Routing is a fundamental feature in frameworks
  - To display different contents basing on the where the user is in the application
  - *See: http://rhymedcode.net/javascript-framework/angular-vs-react-vs-vue-routing/*

FILE-SYSTEM ROUTING

- By default, a web server simply serves everything under a specified directory as "**/**" (root) of the URL
  - e.g., *https://www.cse.cuhk.edu.hk/academics/ug-course-list/* could be pointing to **`index.html`** (default filename) in this directory list
  - All directories and files are served, basing on file permissions on the server
    - Admins often set permission **711** (others ***no read***, ***no write***, only ***execute***) to the directory, so a list of files will not be shown if accessing the directory
- Even with the routing methods mentioned later, it is still possible to let React serve static contents using the **`/public`** directory
  - Usually for media or data files

```
/
+ about
+ academics
  + ug-course-list
    - index.html
    ...
+ research
+ people
...
```

STATIC AND DYNAMIC ROUTING

Static routing

- Routing before rendering takes place during ***initialization***
- Allowing inspection and matching of routes earlier
- Used by Express, Angular, Vue, …

Dynamic routing

- Routing takes place as the app is ***rendering***
- A component is rendered if a path is matched as a prop
- Possible for responsive routes
- Used by React-router

See: *https://reactrouter.com/core/guides/philosophy/dynamic-routing*

HISTORY API

- In the browser, the URLs visited by the user can be found in the `window.history` object
  - `history.back()` loads the previous page
  - `history.forward()` loads the next page (if user went back before)
  - `history.go(num)` loads the specific item (*num*=-1, -2, …) in the history list
- It is possible to change the browser's current visited page with `window.location`
  - `location.assign()` loads a new URL
  - `location.replace()` replaces the current URL from the history with a new one
  - *See: https://developer.mozilla.org/en-US/docs/Web/API/Window/location*

HISTORY API

- New items can be pushed into the history stack with JS
  - `history.pushState(`*`data`*`, `*`title`*`, `***`url`***`)`
  - This allows SPA to enhance navigation since users can click on the familiar Back button
  - *See: https://css-tricks.com/using-the-html5-history-api/#aa-an-example-using-pushstate-and-ajax*
- The history stack also provides a `history.replaceState()` method, as well as a **`onpopstate`** event
  - **`onpopstate`** is triggered when the user clicks on Back or Forward buttons
  - *See: https://developer.mozilla.org/en-US/docs/Web/API/Window/popstate_event*

REACT-ROUTER

- Installing **`react-router-dom`** with either…
  - Linking from CDN, e.g., unpkg.com
    Note: this is fading out from the current version v.6.4
    - *https://unpkg.com/history@5/umd/history.development.js*
    - *https://unpkg.com/react-router@6/dist/umd/react-router.development.js*
    - *https://unpkg.com/react-router-dom@6/dist/umd/react-router-dom.development.js*
    - Set up the relevant components before using:

```
const {BrowserRouter, Routes, Route, Link} = ReactRouterDOM;
```

  - Using npm (e.g., with **`create-react-app`**):

```
npm install react-router-dom
```

    - Set up the relevant components before using:

```
import { BrowserRouter, Routes, Route, Link} from "react-router-dom";
```

REACT-ROUTER

- It has made a lot of changes to syntax from v4 to v5, to v5.1, to v6, and further to v6.4
- When referring to online tutorials, beware of version differences!
  - *Sometimes "traditional" syntax could be supported as well, even if not found in modern tutorials…*

REACT-ROUTER

https://stackblitz.com/edit/chuckjee-react-router

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import { BrowserRouter,
  Routes,
  Route,
  Link } from 'react-router-dom';

class App extends React.Component {
  render() {
    return (
      <BrowserRouter>
        <div>
          <ul>
            <li><Link to="/">Home</Link></li>
            <li><Link to="/about">About</Link></li>
          </ul>
          <hr/>
        <Routes>
         <Route path="/" element={<Home/>} />
         <Route path="/about" element={<About/>} />
        </Routes>
        </div>
      </BrowserRouter>
    );
  }
}

class Home extends React.Component {
  render() {
    return <h2>Home</h2>;
  }
}

class About extends React.Component {
  render() {
    return <h2>About</h2>;
  }
}

const root = ReactDOM.createRoot(
                document.querySelector("#app"));
root.render(<App/>);
```

MORE TO DISCOVER…

- There are more useful features of React-router
  - URL/query parameters
  - Nesting
  - 404 error page
  - Sidebar
- Learn from examples!
  - Some may not correspond to the updated version though
  - *https://github.com/remix-run/react-router/tree/dev/examples*

READ FURTHER…

React-router Docs (v6.3)

https://reactrouter.com/docs/en/v6

Ultimate React Router v6 Guide

https://blog.webdevsimplified.com/2022-07/react-router/