Client-Side Web Programming: JavaScript (Part 3)

Robert M. Dondero, Ph.D.

Princeton University

Objectives

- We will cover:
  - JavaScript libraries for client-side web programming
  - jQuery

Agenda

- **AJAX via XMLHttpRequest enhancements**
- JavaScript libraries
- jQuery

AJAX Enhancements

- Recall **<u>PennyAjax1</u>** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - **penny.py**
  - **index.html**

PennyAjax1/penny.py (Page 1 of 1)

```
#!/usr/bin/env python

#----------------------------------------------------------------------
# penny.py
# Author: Bob Dondero
#----------------------------------------------------------------------

import json
import flask
import database

#----------------------------------------------------------------------

app = flask.Flask(__name__)

#----------------------------------------------------------------------

@app.route('/', methods=['GET'])
@app.route('/index', methods=['GET'])
def index():

    return flask.send_file('index.html')

#----------------------------------------------------------------------

@app.route('/searchresults', methods=['GET'])
def search_results():

    author = flask.request.args.get('author')
    if author is None:
        author = ''
    author = author.strip()

    if author == '':
        books = []
    else:
        books = database.get_books(author) # Exception handling omitted

    json_doc = json.dumps(books)
    response = flask.make_response(json_doc)
    response.headers['Content-Type'] = 'application/json'
    return response
```

blank (Page 1 of 1)

This page is intentionally blank.

PennyAjax1/index.html (Page 1 of 2)

```
 1: <!DOCTYPE html>
 2: <html>
 3:    <head>
 4:       <title>Penny.com</title>
 5:    </head>
 6:
 7:    <body>
 8:
 9:       <hr>
10:       Good <span id="ampmSpan"></span> and welcome to
11:       <strong>Penny.com</strong>
12:       <hr>
13:
14:       <h1>Author Search</h1>
15:       Please enter an author name:
16:       <input type="text" id="authorInput" autoFocus>
17:       <hr>
18:       <div id="resultsDiv"></div>
19:
20:       <hr>
21:       Date and time: <span id="datetimeSpan"></span><br>
22:       Created by <a href="https://www.cs.princeton.edu/~rdondero">
23:       Bob Dondero</a>
24:       <hr>
25:
26:       <script>
27:
28:          'use strict';
29:
30:          function getAmPm() {
31:             let dateTime = new Date();
32:             let hours = dateTime.getHours();
33:             let amPm = (hours < 12) ? 'morning' : 'afternoon';
34:             let ampmSpan = document.getElementById('ampmSpan');
35:             ampmSpan.innerHTML = amPm;
36:          }
37:
38:          function getDateTime() {
39:             let dateTime = new Date();
40:             let datetimeSpan =
41:                document.getElementById('datetimeSpan');
42:             datetimeSpan.innerHTML = dateTime.toLocaleString();
43:          }
44:
45:          function escape(s) {
46:             s = s.replace('&', '&amp;');
47:             s = s.replace('<', '&lt;');
48:             s = s.replace('>', '&gt;');
49:             s = s.replace('"', '&quot;');
50:             s = s.replace("'", '&apos;');
51:             return s;
52:          }
53:
54:          function convertToHtml(books) {
55:             let html = '';
56:             for (let book of books) {
57:                html += escape(book.isbn) + ': ';
58:                html += '<strong>';
59:                html += escape(book.author);
60:                html += '</strong>: ';
61:                html += escape(book.title) + '<br>';
62:             }
63:             return html;
64:          }
65:
```

PennyAjax1/index.html (Page 2 of 2)

```
66:          function handleResponse() {
67:             if (this.status !== 200) {
68:                alert('Error: Failed to fetch data from server');
69:                return;
70:             }
71:             let books = JSON.parse(this.response);
72:             let html = convertToHtml(books);
73:             let resultsDiv = document.getElementById('resultsDiv');
74:             resultsDiv.innerHTML = html;
75:          }
76:
77:          function handleError() {
78:             alert('Error: Failed to fetch data from server');
79:          }
80:
81:          function getResults() {
82:             let authorInput = document.getElementById('authorInput');
83:             let author = authorInput.value;
84:             let encodedAuthor = encodeURIComponent(author);
85:             let url = '/searchresults?author=' + encodedAuthor;
86:             let request = new XMLHttpRequest();
87:             request.onload = handleResponse;
88:             request.onerror = handleError;
89:             request.open('GET', url);
90:             request.send();
91:          }
92:
93:          function setup() {
94:             getAmPm();
95:             window.setInterval(getAmPm, 1000);
96:             getDateTime();
97:             window.setInterval(getDateTime, 1000);
98:             let authorInput = document.getElementById('authorInput');
99:             authorInput.addEventListener('input', getResults);
100:         }
101:
102:         document.addEventListener('DOMContentLoaded', setup);
103:
104:      </script>
105:   </body>
106: </html>
```

AJAX Enhancements

- PennyAjax1 app is a ***single page app (SPA)***
- SPAs are enabled by AJAX

AJAX Enhancements

- **Problem**:
  - Code to convert JavaScript data structure to HTML doc is ugly, inefficient
- **Solution:**
  - Use a JavaScript **template engine,** such as ***Mustache***

AJAX Enhancements

- See **<u>PennyAjax2</u>** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjax2/index.html (Page 1 of 2)

```
 1: <!DOCTYPE html>
 2: <html>
 3:    <head>
 4:       <title>Penny.com</title>
 5:    </head>
 6:
 7:    <body>
 8:
 9:       <hr>
10:       Good <span id="ampmSpan"></span> and welcome to
11:       <strong>Penny.com</strong>
12:       <hr>
13:
14:       <h1>Author Search</h1>
15:       Please enter an author name:
16:       <input type="text" id="authorInput" autoFocus>
17:       <hr>
18:       <div id="resultsDiv"></div>
19:
20:       <hr>
21:       Date and time: <span id="datetimeSpan"></span><br>
22:       Created by <a href="https://www.cs.princeton.edu/~rdondero">
23:       Bob Dondero</a>
24:       <hr>
25:
26:       <script src=
27:          "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
28:       </script>
29:
30:       <!-- <script src="/static/mustache.min.js"></script> -->
31:
32:       <script>
33:
34:          'use strict';
35:
36:          function getAmPm() {
37:             let dateTime = new Date();
38:             let hours = dateTime.getHours();
39:             let amPm = (hours < 12) ? 'morning' : 'afternoon';
40:             let ampmSpan = document.getElementById('ampmSpan');
41:             ampmSpan.innerHTML = amPm;
42:          }
43:
44:          function getDateTime() {
45:             let dateTime = new Date();
46:             let datetimeSpan =
47:                document.getElementById('datetimeSpan');
48:             datetimeSpan.innerHTML = dateTime.toLocaleString();
49:          }
50:
51:          function convertToHtml(books) {
52:             let template = `
53:                {{#books}}
54:                   {{isbn}}:
55:                   <strong>{{author}}</strong>:
56:                   {{title}}
57:                   <br>
58:                {{/books}}
59:             `;
60:             let map = {books: books};
61:             let html = Mustache.render(template, map);
62:             return html;
63:          }
64:
65:          function handleResponse() {
```

PennyAjax2/index.html (Page 2 of 2)

```
66:             if (this.status !== 200) {
67:                alert('Error: Failed to fetch data from server');
68:                return;
69:             }
70:             let books = JSON.parse(this.response);
71:             let html = convertToHtml(books);
72:             let resultsDiv = document.getElementById('resultsDiv');
73:             resultsDiv.innerHTML = html;
74:          }
75:
76:          function handleError() {
77:             alert('Error: Failed to fetch data from server');
78:          }
79:
80:          function getResults() {
81:             let authorInput = document.getElementById('authorInput');
82:             let author = authorInput.value;
83:             let encodedAuthor = encodeURIComponent(author);
84:             let url = '/searchresults?author=' + encodedAuthor;
85:             let request = new XMLHttpRequest();
86:             request.onload = handleResponse;
87:             request.onerror = handleError;
88:             request.open('GET', url);
89:             request.send();
90:          }
91:
92:          function setup() {
93:             getAmPm();
94:             window.setInterval(getAmPm, 1000);
95:             getDateTime();
96:             window.setInterval(getDateTime, 1000);
97:             let authorInput = document.getElementById('authorInput');
98:             authorInput.addEventListener('input', getResults);
99:          }
100:
101:         document.addEventListener('DOMContentLoaded', setup);
102:
103:      </script>
104:   </body>
105: </html>
```

AJAX Enhancements

- **Problem:**
  - Server will respond to requests in arbitrary order
- **Solution:**
  - Abort previous request

AJAX Enhancements

- See **<u>PennyAjax3</u>** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjax3/index.html (Page 1 of 2)

```
 1: <!DOCTYPE html>
 2: <html>
 3:    <head>
 4:       <title>Penny.com</title>
 5:    </head>
 6:
 7:    <body>
 8:
 9:       <hr>
10:       Good <span id="ampmSpan"></span> and welcome to
11:       <strong>Penny.com</strong>
12:       <hr>
13:
14:       <h1>Author Search</h1>
15:       Please enter an author name:
16:       <input type="text" id="authorInput" autoFocus>
17:       <hr>
18:       <div id="resultsDiv"></div>
19:
20:       <hr>
21:       Date and time: <span id="datetimeSpan"></span><br>
22:       Created by <a href="https://www.cs.princeton.edu/~rdondero">
23:       Bob Dondero</a>
24:       <hr>
25:
26:       <script src=
27:          "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
28:       </script>
29:
30:       <script>
31:
32:          'use strict';
33:
34:          function getAmPm() {
35:             let dateTime = new Date();
36:             let hours = dateTime.getHours();
37:             let amPm = (hours < 12) ? 'morning' : 'afternoon';
38:             let ampmSpan = document.getElementById('ampmSpan');
39:             ampmSpan.innerHTML = amPm;
40:          }
41:
42:          function getDateTime() {
43:             let dateTime = new Date();
44:             let datetimeSpan =
45:                document.getElementById('datetimeSpan');
46:             datetimeSpan.innerHTML = dateTime.toLocaleString();
47:          }
48:
49:          function convertToHtml(books) {
50:             let template = `
51:                {{#books}}
52:                   {{isbn}}:
53:                   <strong>{{author}}</strong>:
54:                   {{title}}
55:                   <br>
56:                {{/books}}
57:             `;
58:             let map = {books: books};
59:             let html = Mustache.render(template, map);
60:             return html;
61:          }
62:
63:          function handleResponse() {
64:             if (this.status !== 200) {
65:                alert('Error: Failed to fetch data from server');
```

PennyAjax3/index.html (Page 2 of 2)

```
66:                return;
67:             }
68:             let books = JSON.parse(this.response);
69:             let html = convertToHtml(books);
70:             let resultsDiv = document.getElementById('resultsDiv');
71:             resultsDiv.innerHTML = html;
72:          }
73:
74:          function handleError() {
75:             alert('Error: Failed to fetch data from server');
76:          }
77:
78:          let request = null;
79:
80:          function getResults() {
81:             let authorInput = document.getElementById('authorInput');
82:             let author = authorInput.value;
83:             let encodedAuthor = encodeURIComponent(author);
84:             let url = '/searchresults?author=' + encodedAuthor;
85:             if (request !== null)
86:                request.abort();
87:             request = new XMLHttpRequest();
88:             request.onload = handleResponse;
89:             request.onerror = handleError;
90:             request.open('GET', url);
91:             request.send();
92:          }
93:
94:          function setup() {
95:             getAmPm();
96:             window.setInterval(getAmPm, 1000);
97:             getDateTime();
98:             window.setInterval(getDateTime, 1000);
99:             let authorInput = document.getElementById('authorInput');
100:            authorInput.addEventListener('input', getResults);
101:         }
102:
103:         document.addEventListener('DOMContentLoaded', setup);
104:
105:      </script>
106:   </body>
107: </html>
```

AJAX Enhancements

- **Problem**:
  - Server could be overwhelmed with requests
- **Solution:**
  - ***Debounce*** the requests

AJAX Enhancements

- See **<u>PennyAjax4</u>** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjax4/index.html (Page 1 of 2)

```
 1: <!DOCTYPE html>
 2: <html>
 3:    <head>
 4:       <title>Penny.com</title>
 5:    </head>
 6:
 7:    <body>
 8:
 9:       <hr>
10:       Good <span id="ampmSpan"></span> and welcome to
11:       <strong>Penny.com</strong>
12:       <hr>
13:
14:       <h1>Author Search</h1>
15:       Please enter an author name:
16:       <input type="text" id="authorInput" autoFocus>
17:       <hr>
18:       <div id="resultsDiv"></div>
19:
20:       <hr>
21:       Date and time: <span id="datetimeSpan"></span><br>
22:       Created by <a href="https://www.cs.princeton.edu/~rdondero">
23:       Bob Dondero</a>
24:       <hr>
25:
26:       <script src=
27:          "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
28:       </script>
29:
30:       <script>
31:
32:          'use strict';
33:
34:          function getAmPm() {
35:             let dateTime = new Date();
36:             let hours = dateTime.getHours();
37:             let amPm = (hours < 12) ? 'morning' : 'afternoon';
38:             let ampmSpan = document.getElementById('ampmSpan');
39:             ampmSpan.innerHTML = amPm;
40:          }
41:
42:          function getDateTime() {
43:             let dateTime = new Date();
44:             let datetimeSpan =
45:                document.getElementById('datetimeSpan');
46:             datetimeSpan.innerHTML = dateTime.toLocaleString();
47:          }
48:
49:          function convertToHtml(books) {
50:             let template = `
51:                {{#books}}
52:                   {{isbn}}:
53:                   <strong>{{author}}</strong>:
54:                   {{title}}
55:                   <br>
56:                {{/books}}
57:             `;
58:             let map = {books: books};
59:             let html = Mustache.render(template, map);
60:             return html;
61:          }
62:
63:          function handleResponse() {
64:             if (this.status !== 200) {
65:                alert('Error: Failed to fetch data from server');
```

PennyAjax4/index.html (Page 2 of 2)

```
66:                return;
67:             }
68:             let books = JSON.parse(this.response);
69:             let html = convertToHtml(books);
70:             let resultsDiv = document.getElementById('resultsDiv');
71:             resultsDiv.innerHTML = html;
72:          }
73:
74:          function handleError() {
75:             alert('Error: Failed to fetch data from server');
76:          }
77:
78:          let request = null;
79:
80:          function getResults() {
81:             let authorInput = document.getElementById('authorInput');
82:             let author = authorInput.value;
83:             let encodedAuthor = encodeURIComponent(author);
84:             let url = '/searchresults?author=' + encodedAuthor;
85:             if (request !== null)
86:                request.abort();
87:             request = new XMLHttpRequest();
88:             request.onload = handleResponse;
89:             request.onerror = handleError;
90:             request.open('GET', url);
91:             request.send();
92:          }
93:
94:          let timer = null;
95:
96:          function debouncedGetResults() {
97:             clearTimeout(timer);
98:             timer = window.setTimeout(getResults, 500);
99:          }
100:
101:         function setup() {
102:            getAmPm();
103:            window.setInterval(getAmPm, 1000);
104:            getDateTime();
105:            window.setInterval(getDateTime, 1000);
106:            let authorInput = document.getElementById('authorInput');
107:            authorInput.addEventListener('input', debouncedGetResults);
108:         }
109:
110:         document.addEventListener('DOMContentLoaded', setup);
111:
112:      </script>
113:   </body>
114: </html>
```

AJAX Enhancements

- **Bonus:**
  - Debouncing reduces the need to abort requests

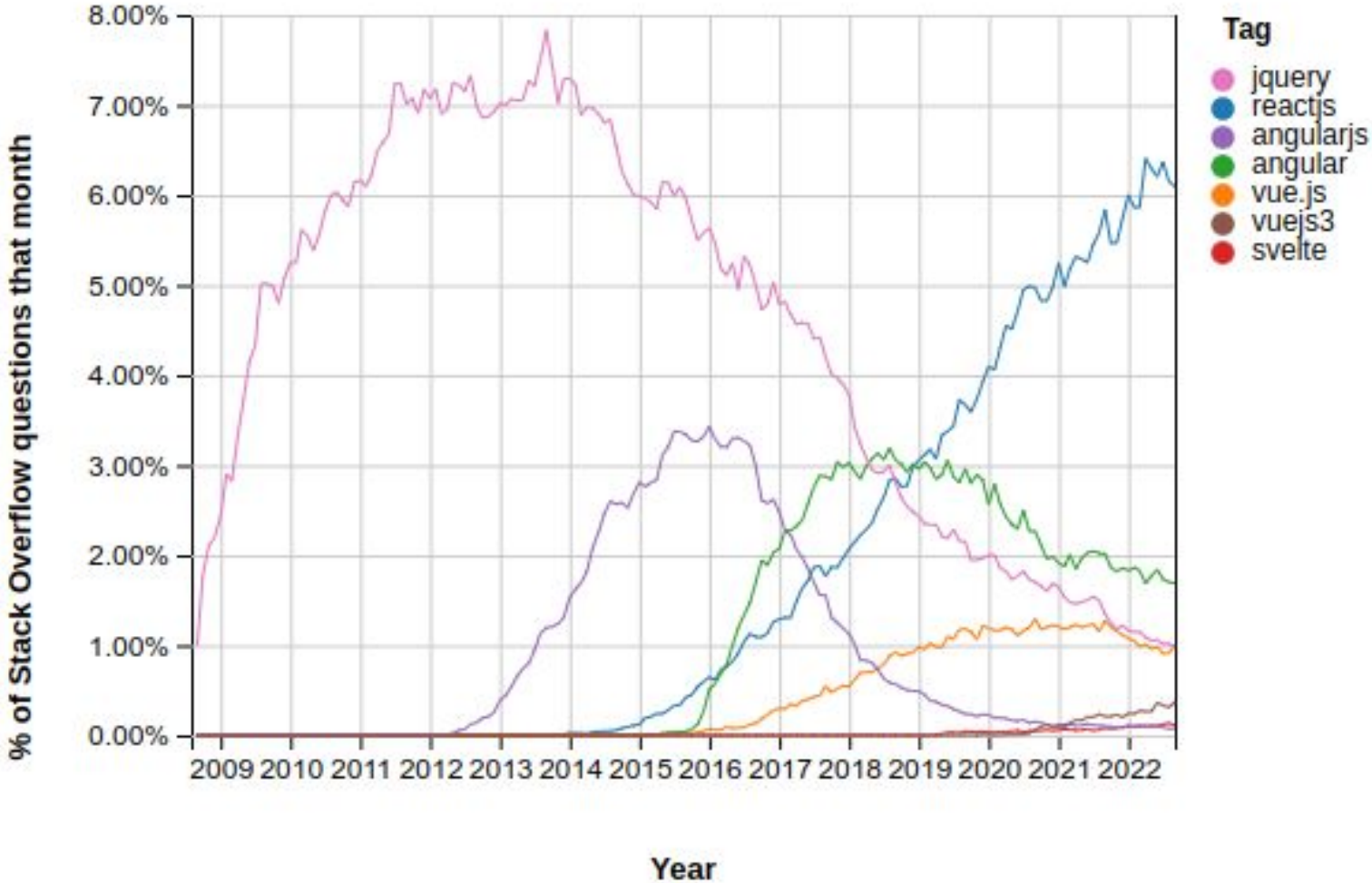
Agenda

- AJAX via XMLHttpRequest enhancements
- **JavaScript libraries**
- jQuery

JavaScript Libraries

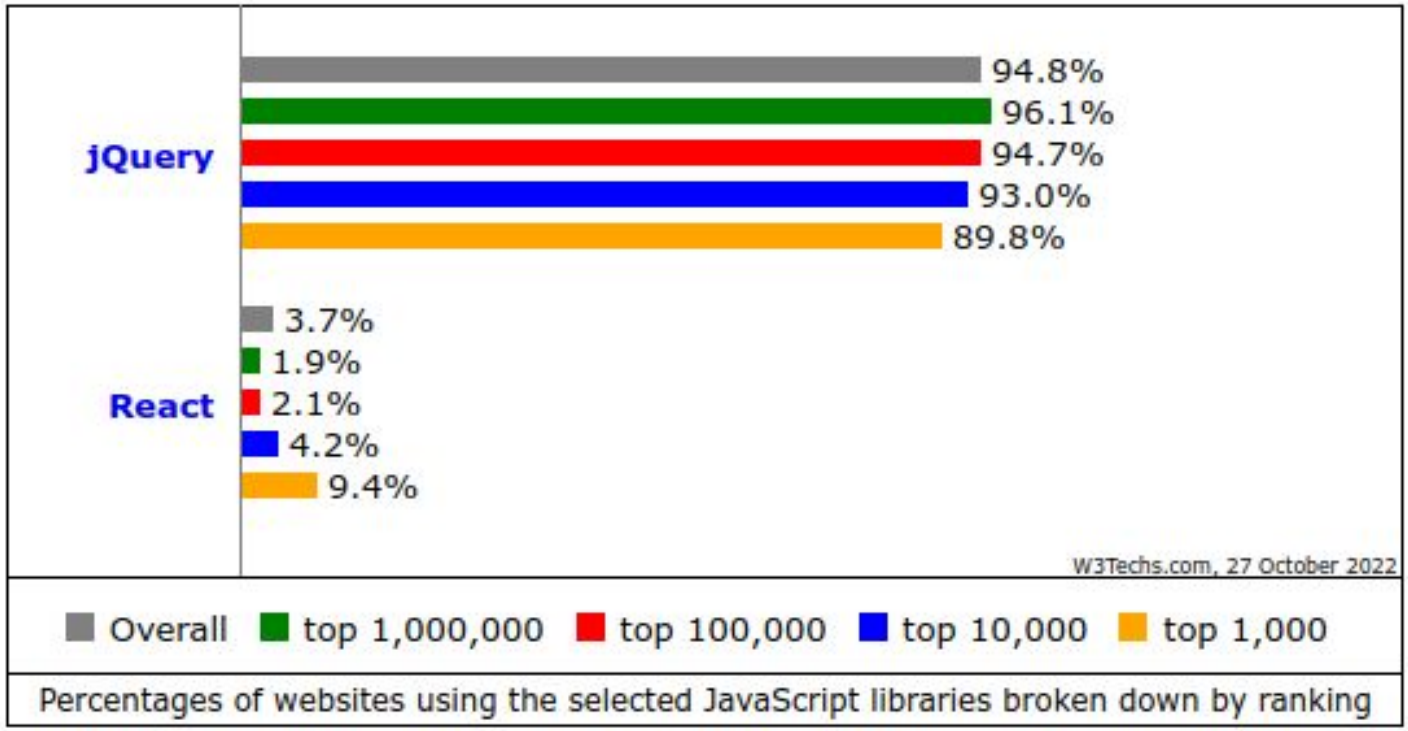
- **Problem**:
  - JavaScript/AJAX code uses common patterns and often is repetitive
- **Solution:**
  - Use a JavaScript **library**
    - **jQuery**, **React**, AngularJS, Angular, vue.js, vue3, svelte, …

JavaScript Libraries



As of Oct 2022, according to https://insights.stackoverflow.com/trends?tags=reactjs%2Cvue.js%2Cangular%2Csvelte%2Cangularjs%2Cvuejs3%2Cjquery

JavaScript Libraries



As of October 2022, according to
https://w3techs.com/technologies/comparison/js-jquery,js-react

Agenda

- AJAX via XMLHttpRequest enhancements
- JavaScript libraries
- **jQuery**

jQuery

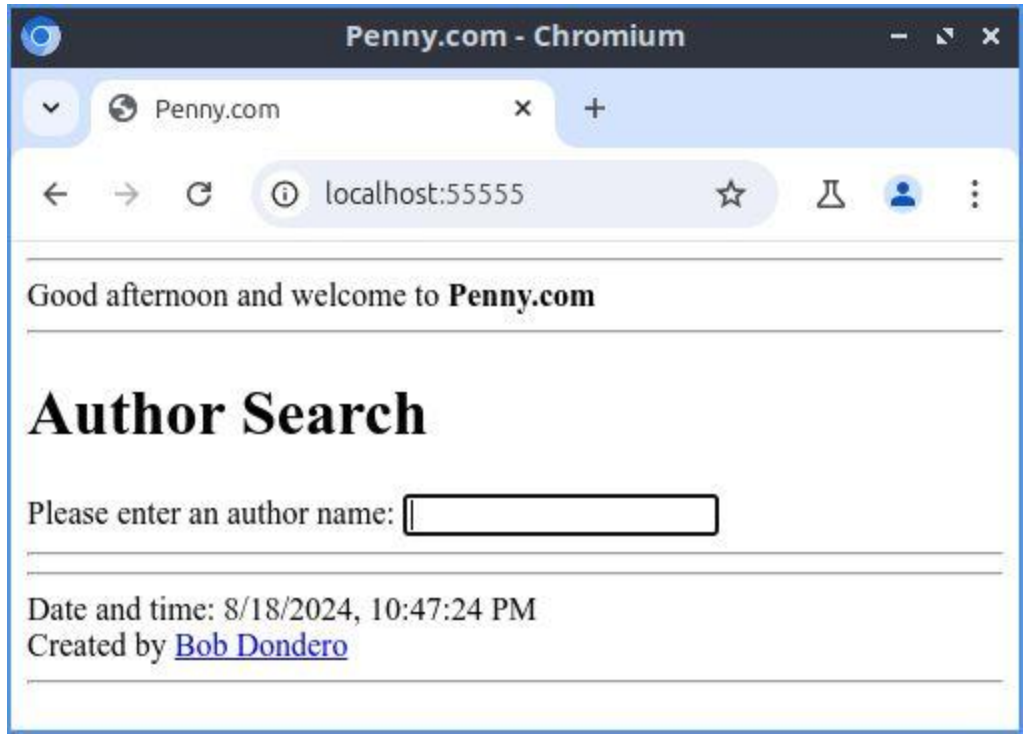
John Resig

jQuery

- jQuery statement syntax:
  - **`jQuery(`*`selector`*`).`*`action`*`()`**
  - **`$(`*`selector`*`).`*`action`*`()`**
    - **`$`**
      - The jQuery function
    - ***`selector`***
      - Selects DOM node(s)
      - As in CSS; covered soon
    - ***`action`*`()`**
      - Specifies an action to be performed on the selected DOM node(s)

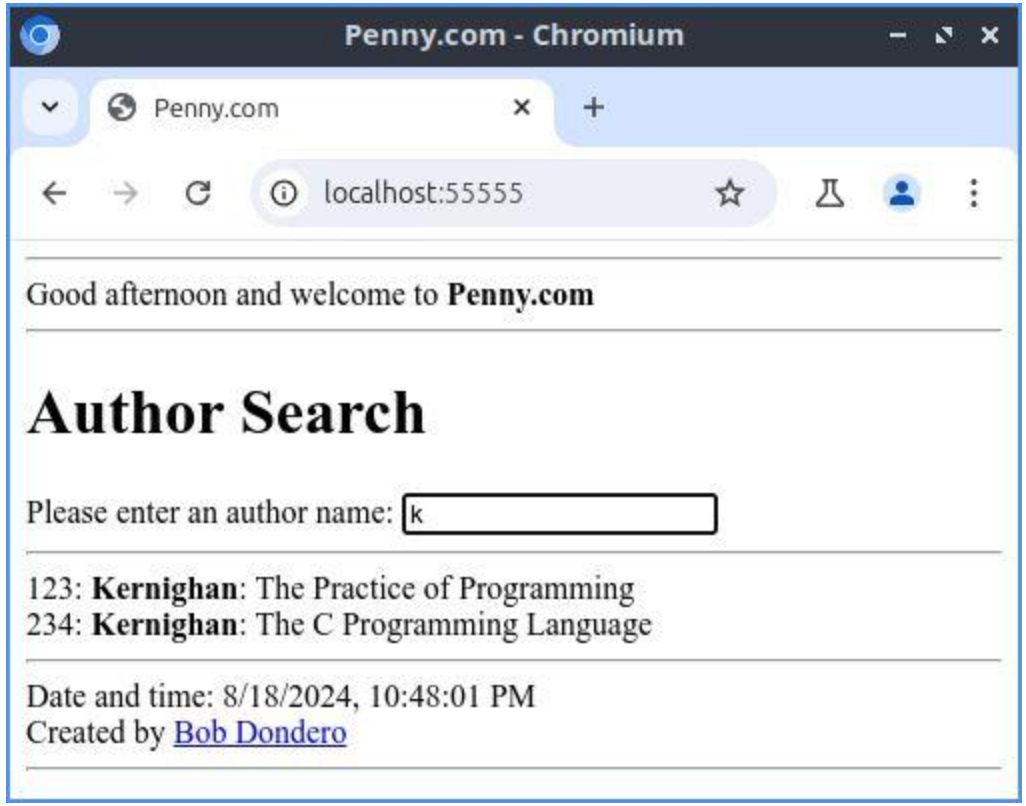
jQuery

- See **<u>PennyJQuery</u>** app



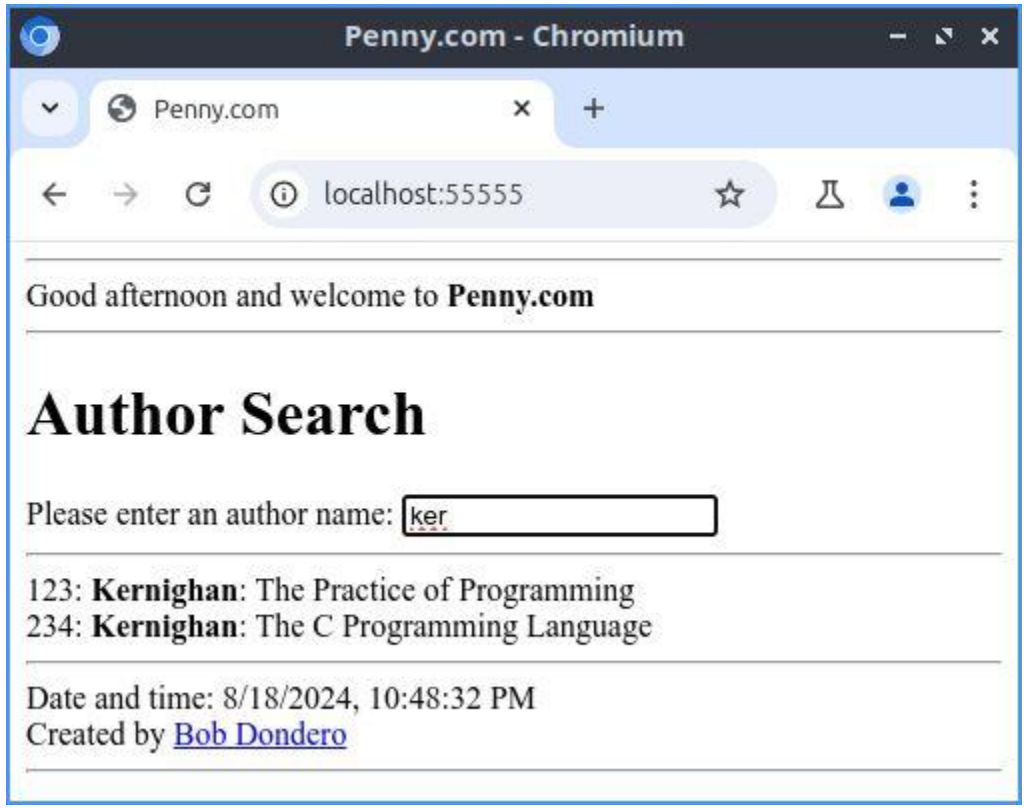
jQuery

- See **<u>PennyJQuery</u>** app



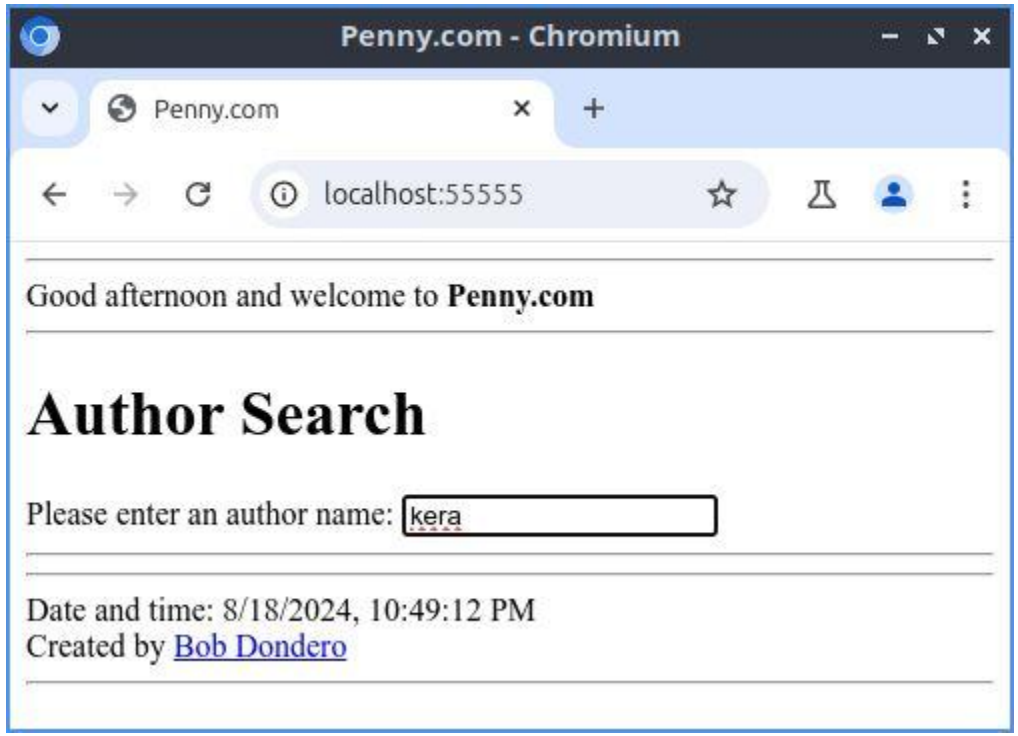
jQuery

- See **PennyJQuery** app



jQuery

- See **<u>PennyJQuery</u>** app



jQuery

- See **<u>PennyJQuery</u>** app (cont.)
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyJQuery/index.html (Page 1 of 2)

```
 1: <!DOCTYPE html>
 2: <html>
 3:    <head>
 4:       <title>Penny.com</title>
 5:    </head>
 6:
 7:    <body>
 8:       <hr>
 9:       Good <span id="ampmSpan"></span> and welcome to
10:       <strong>Penny.com</strong>
11:       <hr>
12:
13:       <h1>Author Search</h1>
14:       Please enter an author name:
15:       <input type="text" id="authorInput" autoFocus>
16:       <hr>
17:       <div id="resultsDiv"></div>
18:
19:       <hr>
20:       Date and time: <span id="datetimeSpan"></span><br>
21:       Created by <a href="https://www.cs.princeton.edu/~rdondero">
22:       Bob Dondero</a>
23:       <hr>
24:
25:       <script src=
26:          "https://cdn.jsdelivr.net/npm/jquery@3.7.1/dist/jquery.min.js">
27:       </script>
28:
29:       <script src=
30:          "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
31:       </script>
32:
33:       <script>
34:
35:          'use strict';
36:
37:          function getAmPm() {
38:             let dateTime = new Date();
39:             let hours = dateTime.getHours();
40:             let amPm = 'morning';
41:             if (hours >= 12)
42:                amPm = 'afternoon';
43:             $('#ampmSpan').html(amPm);
44:          }
45:
46:          function getDateTime() {
47:             let dateTime = new Date();
48:             $('#datetimeSpan').html(dateTime.toLocaleString());
49:          }
50:
51:          function convertToHtml(books) {
52:             let template = `
53:                {{#books}}
54:                   {{isbn}}:
55:                   <strong>{{author}}</strong>:
56:                   {{title}}
57:                   <br>
58:                {{/books}}
59:             `;
60:             let map = {books: books};
61:             let html = Mustache.render(template, map);
62:             return html;
63:          }
64:
65:          function handleResponse(books) {
```

PennyJQuery/index.html (Page 2 of 2)

```
66:             let html = convertToHtml(books);
67:             $('#resultsDiv').html(html);
68:          }
69:
70:          function handleError(request) {
71:             if (request.statusText !== 'abort')
72:                alert('Error: Failed to fetch data from server');
73:          }
74:
75:          let request = null;
76:
77:          function getResults() {
78:             let author = $('#authorInput').val();
79:             let encodedAuthor = encodeURIComponent(author);
80:             let url = '/searchresults?author=' + encodedAuthor;
81:             if (request !== null)
82:                request.abort();
83:             let requestData = {
84:                type: 'GET',
85:                url: url,
86:                success: handleResponse,
87:                error: handleError
88:             };
89:             request = $.ajax(requestData);
90:          }
91:
92:          let timer = null;
93:
94:          function debouncedGetResults() {
95:             clearTimeout(timer);
96:             timer = window.setTimeout(getResults, 500);
97:          }
98:
99:          function setup() {
100:            getAmPm();
101:            window.setInterval(getAmPm, 1000);
102:            getDateTime();
103:            window.setInterval(getDateTime, 1000);
104:            $('#authorInput').on('input', debouncedGetResults);
105:         }
106:
107:         $('document').ready(setup);
108:
109:      </script>
110:   </body>
111: </html>
```

jQuery

- jQuery summary…

jQuery

- jQuery makes accessing the DOM easier

Without jQuery:

```
let ampmSpan =
   document.getElementById('ampmSpan');
ampmSpan.innerHtml = amPm;
```

With jQuery:

```
$('#ampmSpan').html(amPm);
```

=> access by id

jQuery

- jQuery makes AJAX easier

Without jQuery:

```
function handleResponse() {
   if (this.status !== 200) {
      alert('Error: Failed to fetch data from server');
         return;
   }
   let books = JSON.parse(this.response);
   let html = convertToHtml(books);
   let resultsDiv = document.getElementById('resultsDiv');
   resultsDiv.innerHTML = this.responseText;
}
function getResults() {
   ...
   request = new XMLHttpRequest();
   request.onload = handleResponse;
   request.onerror = handleError;
   request.open('GET', url);
   request.send();
}
```

jQuery

- jQuery makes AJAX easier (cont.)

With jQuery:

```
function handleResponse(books) {
   let html = convertToHtml(books);
   $('#resultsDiv').html(html);
}

function getResults() {  ...
   let requestData = {
      type: 'GET',
      url: url,
      success: handleResponse,
      error: handleError
   };
   request = $.ajax(requestData);
}
```

jQuery parses JSON automatically

jQuery

- jQuery pros
  - Easy to learn
    - Especially if you know CSS
    - Lots of web info
  - Easy to use
  - **Handles a wide variety of browsers**

jQuery

- jQuery cons
  - Less necessary with current browsers
  - Incompatible with client-side libraries that use a **virtual DOM** (e.g., React)

Aside: AJAX Implementations

AJAX Implementation	Browser Built-In or Library?	Async Mechanism	COS 333 Coverage
XMLHttpRequest	Built-in	Callbacks	Last lecture
fetch & AbortController	Built-in	Promises	Last lecture appendix
Axios	Library	Promises	None
jQuery	Library	Callbacks (or promises)	This lecture

Aside: AJAX Implementations

AJAX Implementation	Firefox	Chrome
XMLHttpRequest	12+ (2012)	31+ (2013)
fetch	39+ (2015)	42+ (2015)
AbortController	57+ (2017)	66+ (2018)
Axios	12+ (2012)	31+ (2013)
jQuery	12+ (2012)	31+ (2013)

Summary

- We have covered:
  - JavaScript libraries for client-side web programming
  - jQuery