Client-Side Web Programming: JavaScript (Part 4)

Robert M. Dondero, Ph.D.

Princeton University

Objectives

- We will cover:
  - React concepts
  - React examples
  - jQuery vs. React

Agenda

- **React concepts**
- React simple examples
- React examples
- jQuery vs. React

React Concepts

Jordan
Walke

Note: **React** == **React.js** == **ReactJS**

React Concepts

- The fundamental idea…

- jQuery (or no client-side library)
  - HTML code contains JavaScript code
- React
  - HTML code is generated by JavaScript code

React Concepts

- Two styles of React programming:
  - **Class-based**
    - (pro) Easier to understand?
    - (con) Semi-deprecated
  - **Functional**
    - (pro) More succinct
    - (con) Harder to understand?

- We'll cover functional…

React Concepts

- Key concept: ***components***
  - Pgmmer defines components
  - Each component is defined as a function
  - Components can be arranged hierarchically
  - Each component:
    - Can have **state**
    - Can accept properties (**props**) from parent component
    - Returns a (partial) DOM tree

React Concepts

- Key concept: ***virtual DOM***
  - Corresponding to the browser DOM tree, React maintains a *virtual DOM tree*
  - For each browser DOM node, there is a *virtual* DOM node

React Concepts

- Key concept: ***virtual DOM*** (cont.)
  - At initial rendering, and when the state of a component changes:
    - Component (typically) returns a (partial) DOM tree
    - React updates the virtual DOM tree with the component's (partial) DOM tree
    - React compares the updated virtual DOM tree with the previous version to determine diffs
    - Using diffs, React *reconciles* the virtual DOM tree and browser DOM tree
      - Updates the smallest possible browser DOM subtree

React Concepts

- Key supporting technology: ***JSX (JavaScript XML)***
  - Allows embedding of HTML-like code (actually, XML code) in JavaScript code

React Concepts

- Key concept: ***React bundles***
  - Typically React programmers use **WebPack**, **Next.js**, **Vite**, … to create a **React bundles**
  - Here, for simplicity, we will not create React bundles
    - But see next lecture

Agenda

- React concepts
- **React simple examples**
- React examples
- jQuery vs. React

React Simple Examples

- See **reacthello.html**

reacthello.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Hello</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //-------------------------------------------------------------

         function App() {
            return (
               <div>
                  Hello, world.
               </div>
            );
         }

         //-------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>

```

blank (Page 1 of 1)

```
This page is intentionally blank.
```

React Simple Examples

- See **reacthello.html** (cont.)
  - Things to note:
    - Overall structure
    - (Minimal) use of HTML
    - Use of JSX
    - Defining a component
      - Which returns a partial DOM tree

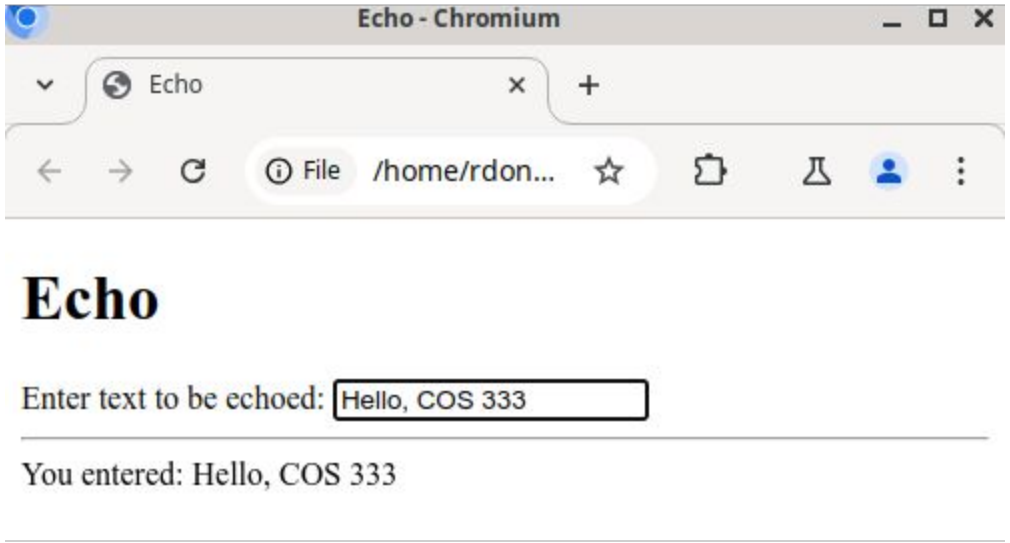
React Simple Examples

- See **reacthello.html** (cont.)
  - Things to note:
    - `React.StrictMode`
      - "Lets you find common bugs in your components early during development."
      - "Use StrictMode to enable additional development behaviors and warnings for the component tree inside."

https://react.dev/reference/react/StrictMode

React Simple Examples

- See **reactecho1.html**



reactecho1.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Echo</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //-------------------------------------------------------------

         function App() {
            const [enteredText, setEnteredText] = React.useState('');

            return (
               <div>
                  <h1>Echo</h1>
                  Enter text to be echoed: 
                  <input
                      type='text'
                      onInput={ function (event) {
                         setEnteredText(event.target.value);
                      }}
                      autoFocus
                  />
                  <hr />
                  You entered: {enteredText}
               </div>
            );
         }

         //-------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>
```

arrow.js (Page 1 of 1)

```
//-----------------------------------------------------------------------
// arrow.js
// Author: Bob Dondero
//-----------------------------------------------------------------------

'use strict';

function square1(i) {return i * i;}
let square2 = function(i) {return i * i;};
let square3 = (i) => {return i * i;};
let square4 = (i) => i * i;
let square5 = i => i * i;

function prod1(i, j) {return i * j;}
let prod2 = function(i, j) {return i * j;};
let prod3 = (i, j) => {return i * j;};
let prod4 = (i, j) => i * j;

function greet1() {process.stdout.write('hi\n');}
let greet2 = function() {process.stdout.write('hi\n');};
let greet3 = () => {process.stdout.write('hi\n');};
let greet4 = () => process.stdout.write('hi\n');

function main() {
   process.stdout.write(String(square1(5)) + '\n');
   process.stdout.write(String(square2(5)) + '\n');
   process.stdout.write(String(square3(5)) + '\n');
   process.stdout.write(String(square4(5)) + '\n');
   process.stdout.write(String(square5(5)) + '\n');

   process.stdout.write(String(prod1(5, 6)) + '\n');
   process.stdout.write(String(prod2(5, 6)) + '\n');
   process.stdout.write(String(prod3(5, 6)) + '\n');
   process.stdout.write(String(prod4(5, 6)) + '\n');

   greet1();
   greet2();
   greet3();
   greet4();
}

if (require.main === module)
   main();
```

React Simple Examples

- See **reactecho1.html** (cont.)
  - Things to note:
    - ***useState hook***
      - Defines a **state** variable and a function that can be called to change its value
    - When a component's **state** changes, React:
      - Uses state to create and return a (partial) DOM tree
      - Update virtual DOM tree with partial DOM tree
      - Compares new virtual DOM tree with old
      - Reconciles virtual DOM tree with browser DOM tree

Aside: Arrow Functions

- ***Arrow function def expressions***
  - Informally ***arrow functions***
  - Arrow functions vs ordinary functions:
    - Often more succinct
    - Same semantics **- mostly!!!**
  - See **Appendix 1** for more information

Aside: Arrow Functions

- See **arrow.js**

```
$ node arrow.js
25
25
25
25
25
30
30
30
30
hi
hi
hi
$
```

reactecho1.html (Page 1 of 1)

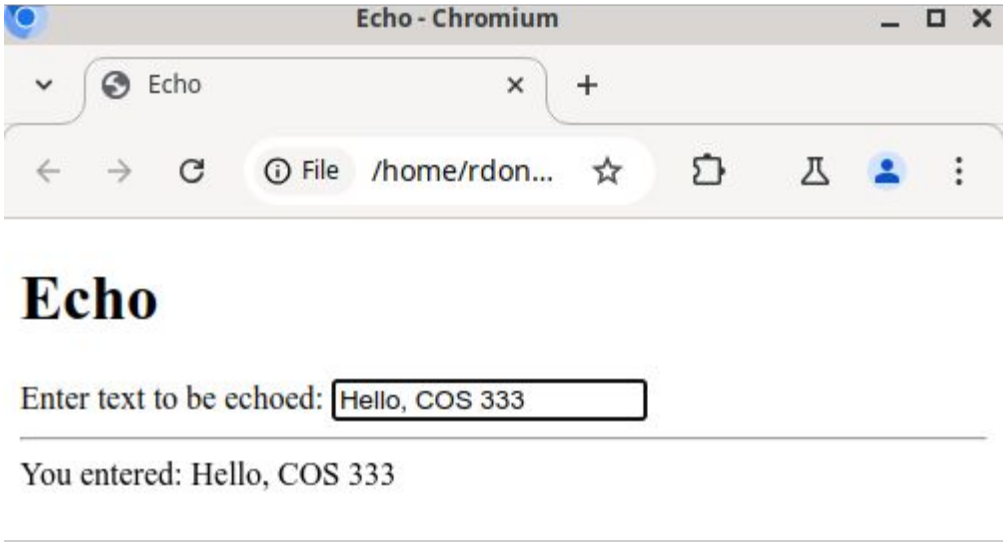
```
<!DOCTYPE html>
<html>
   <head>
      <title>Echo</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //-------------------------------------------------------------

         function App() {
            const [enteredText, setEnteredText] = React.useState('');

            return (
               <div>
                  <h1>Echo</h1>
                  Enter text to be echoed: 
                  <input
                      type='text'
                      onInput={ function (event) {
                         setEnteredText(event.target.value);
                      }}
                      autoFocus
                  />
                  <hr />
                  You entered: {enteredText}
               </div>
            );
         }

         //-------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>
```

arrow.js (Page 1 of 1)

```
//-----------------------------------------------------------------------
// arrow.js
// Author: Bob Dondero
//-----------------------------------------------------------------------

'use strict';

function square1(i) {return i * i;}
let square2 = function(i) {return i * i;};
let square3 = (i) => {return i * i;};
let square4 = (i) => i * i;
let square5 = i => i * i;

function prod1(i, j) {return i * j;}
let prod2 = function(i, j) {return i * j;};
let prod3 = (i, j) => {return i * j;};
let prod4 = (i, j) => i * j;

function greet1() {process.stdout.write('hi\n');}
let greet2 = function() {process.stdout.write('hi\n');};
let greet3 = () => {process.stdout.write('hi\n');};
let greet4 = () => process.stdout.write('hi\n');

function main() {
   process.stdout.write(String(square1(5)) + '\n');
   process.stdout.write(String(square2(5)) + '\n');
   process.stdout.write(String(square3(5)) + '\n');
   process.stdout.write(String(square4(5)) + '\n');
   process.stdout.write(String(square5(5)) + '\n');

   process.stdout.write(String(prod1(5, 6)) + '\n');
   process.stdout.write(String(prod2(5, 6)) + '\n');
   process.stdout.write(String(prod3(5, 6)) + '\n');
   process.stdout.write(String(prod4(5, 6)) + '\n');

   greet1();
   greet2();
   greet3();
   greet4();
}

if (require.main === module)
   main();
```

React Simple Examples

- See **reactecho2.html** (cont.)
  - Things to note:
    - Uses arrow functions exclusively



reactecho2.html (Page 1 of 1)

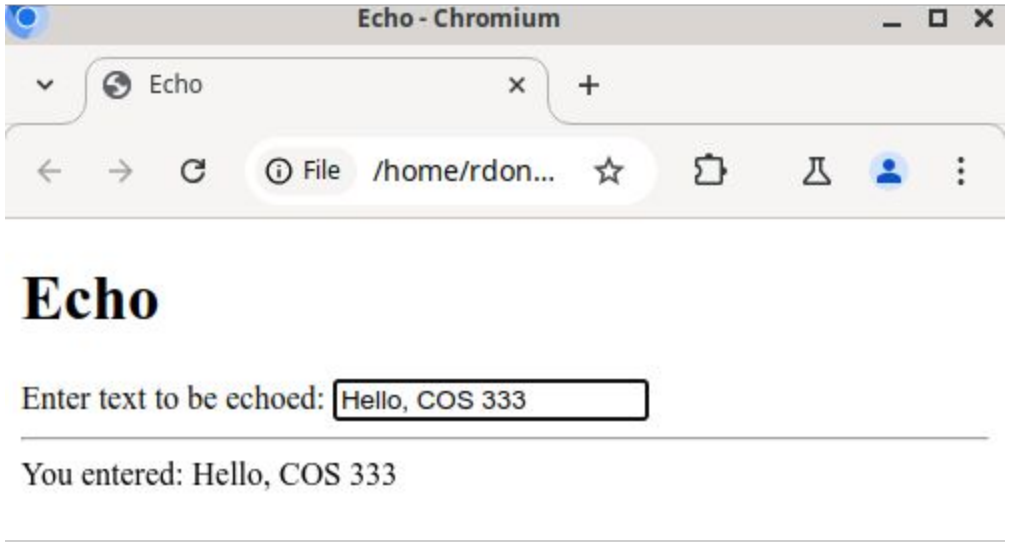
```
<!DOCTYPE html>
<html>
   <head>
      <title>Echo</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //---------------------------------------------------------------

         const App = () => {
            const [enteredText, setEnteredText] = React.useState('');

            return (
               <div>
                  <h1>Echo</h1>
                  Enter text to be echoed: 
                  <input
                      type='text'
                      onInput={ (event) => {
                         setEnteredText(event.target.value);
                      }}
                      autoFocus
                  />
                  <hr />
                  You entered: {enteredText}
               </div>
            );
         }

         //---------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>
```

reactecho3.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Echo</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //---------------------------------------------------------------

         function App() {
            const [enteredText, setEnteredText] = React.useState('');

            return (
               <div>
                  <h1>Echo</h1>
                  Enter text to be echoed: 
                  <input
                      type='text'
                      onInput={ (event) => {
                         setEnteredText(event.target.value);
                      }}
                      autoFocus
                  />
                  <hr />
                  You entered: {enteredText}
               </div>
            );
         }

         //---------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>
```

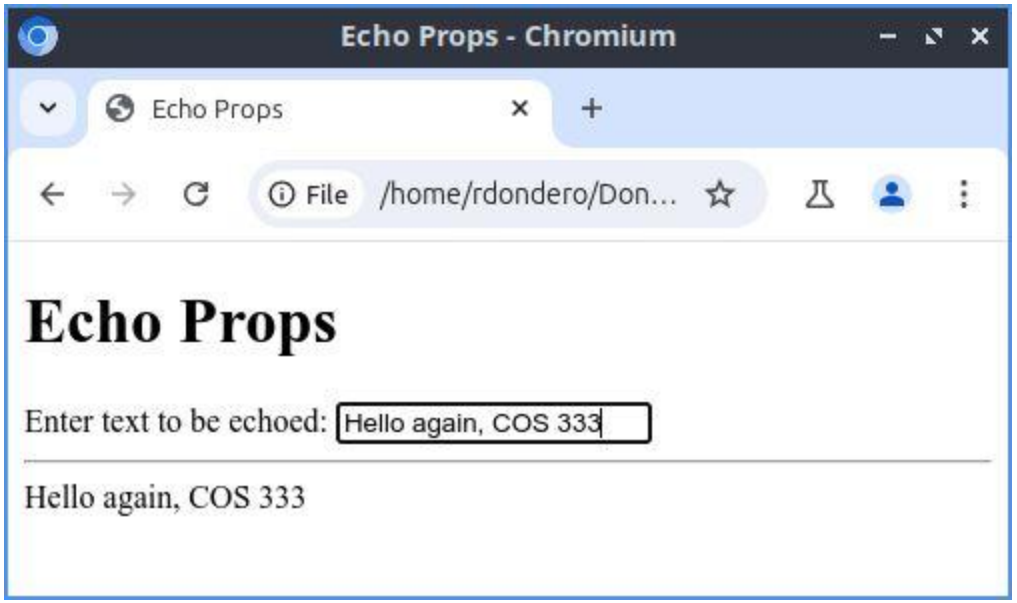
React Simple Examples

- See **reactecho3.html** (cont.)
  - Things to note:
    - Uses arrow functions as callbacks



React Simple Examples

- See **reactecho4.html**



reactecho4.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Echo Props</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //---------------------------------------------------------------

         function EchoInput(props) {
            return (
               <div>
                  <h1>Echo Props</h1>
                  Enter text to be echoed: 
                  <input
                      type='text'
                      onInput={ (event) => {
                         props.callback(event.target.value);
                      }}
                      autoFocus
                  />
               </div>
            );
         }

         //---------------------------------------------------------------

         function EchoOutput(props) {
            return (
               <div>
                  {props.txt}
               </div>
            );
         }

         //---------------------------------------------------------------

         function Echo() {
            const [enteredText, setEnteredText] = React.useState('');

            return (
               <div>
                  <EchoInput callback={setEnteredText} />
```

reactecho4.html (Page 2 of 2)

```
                  <hr />
                  <EchoOutput txt={enteredText} />
               </div>
            );
         }

         //---------------------------------------------------------------

         function App() {
            return (
               <Echo />
            );
         }

         //---------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>
```

React Simple Examples

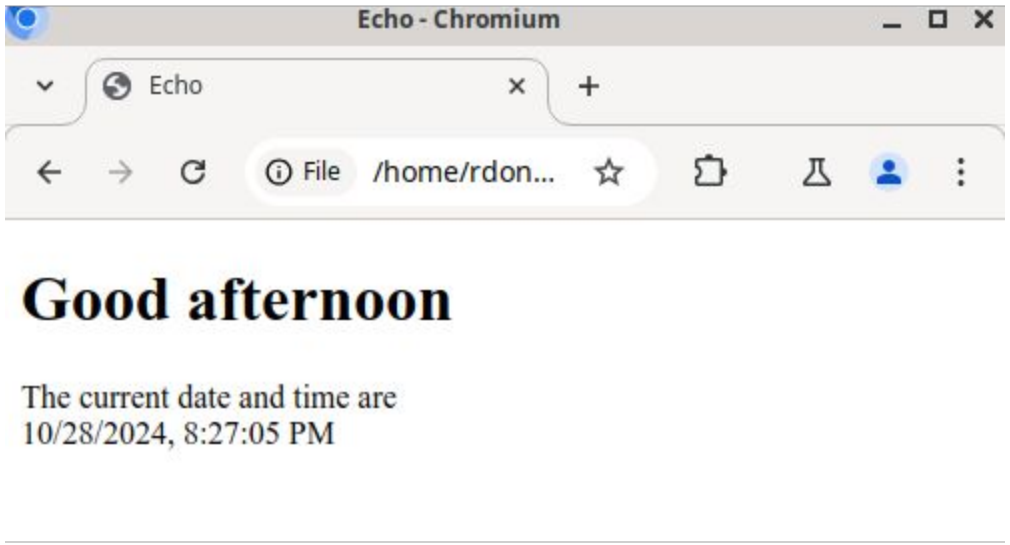
- See **reactecho4.html** (cont.)
  - Things to note:
    - Components arranged in a tree (hierarchy)
    - Use of a **prop** to send data **downward**
      - From parent component to child component
      - From `Echo` to `EchoOutput`
    - Use of a **prop** to send data **upward**
      - From child component to parent component
      - From `EchoInput` to `Echo`
      - Prop must be a callback function

React Simple Examples

- **Question:**
  - How does a **parent** component send data to a **child** component?
- **Answer:**
  - Props
- **Question:**
  - How does a **child** component send data to its **parent** component?
- **Answer:**
  - Props that reference callback functions

React Simple Examples

- See **reactdatetime.html**



reactdatetime.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Echo</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //-------------------------------------------------------------

         function App() {
            const [datetime, setDatetime] = React.useState(new Date());

            function updateDatetime() {
               window.setInterval(
                  () => {setDatetime(new Date());},
                  1000
               );
            }

            React.useEffect(updateDatetime, []);

            let hours = datetime.getHours();
            let ampm = (hours < 12) ? 'morning' : 'afternoon';
            return (
               <div>
                  <h1>Good {ampm}</h1>
                  The current date and time are<br/>
                  {datetime.toLocaleString()}
               </div>
            );
         }

         //-------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
```

reactdatetime.html (Page 2 of 2)

```
   </body>
</html>
```

React Simple Examples

- See **reactdatetime.html** (cont.)
  - Things to note:
    - ***useEffect hook***
      - `React.useEffect(f);`
        - » Call `f()` at initial render and every subsequent render
      - `React.useEffect(f, []);`
        - » Call `f()` at initial render
      - `React.useEffect(f, [statevar]);`
        - » Call `f()` at initial render and when `statevar` changes

Agenda

- React concepts
- React simple examples
- **React examples**
- jQuery vs. React

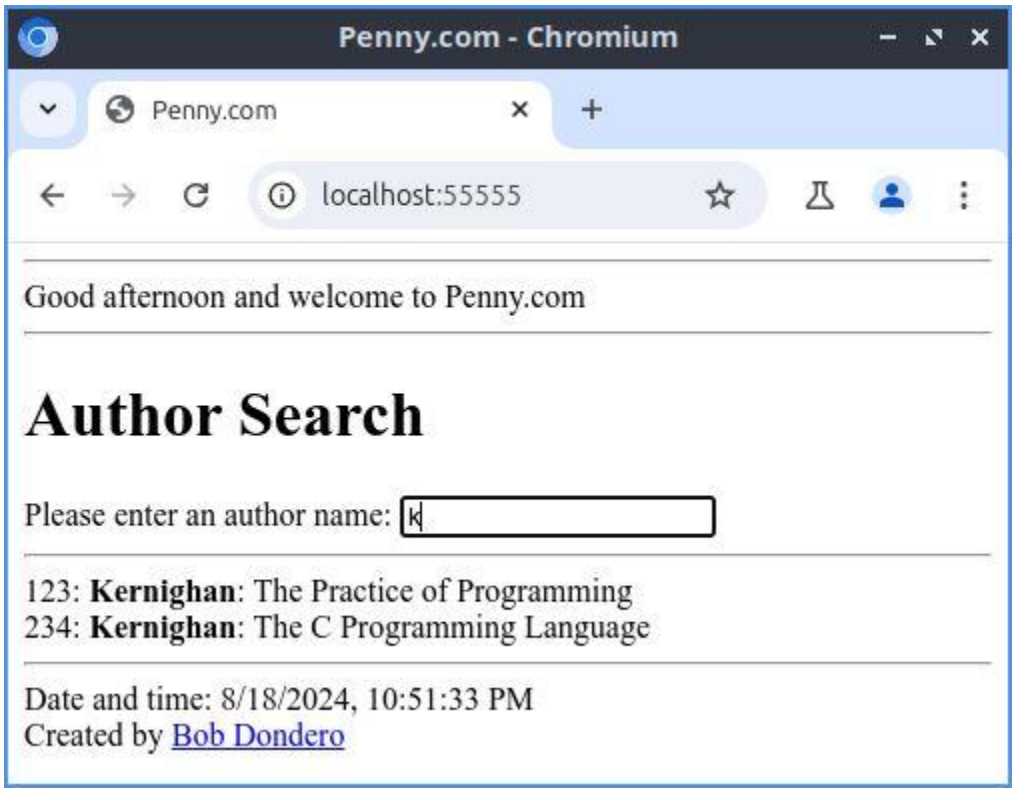
React Examples

- See **PennyReact1** app

Thanks, in part, to Liam Esparraguara ('24)

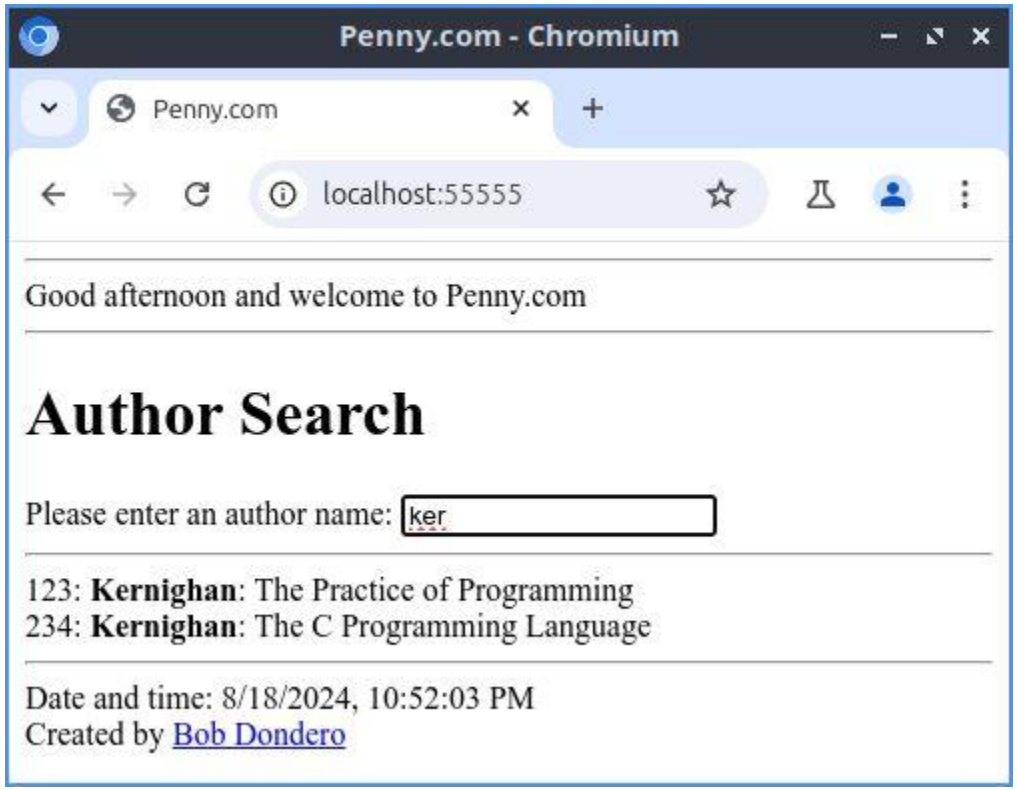
React Examples

- See **PennyReact1** app (cont.)



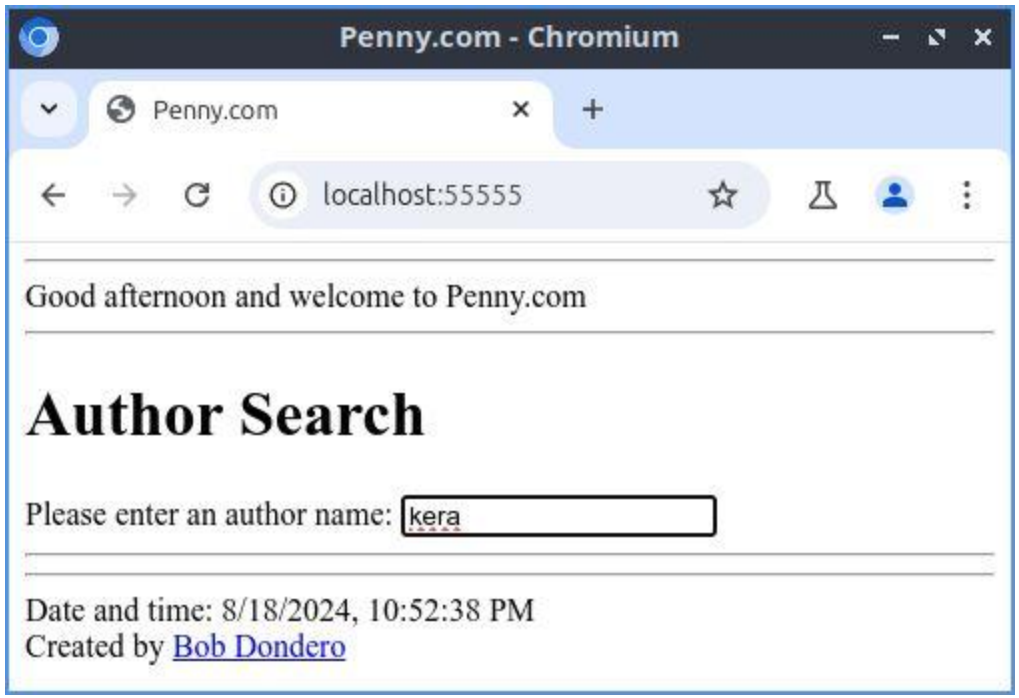
React Examples

- See **PennyReact1** app (cont.)



React Examples

- See **PennyReact1** app (cont.)



React Examples

- See **PennyReact1** app (cont.)
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyReact1/index.html (Page 1 of 3)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //-------------------------------------------------------------

         function PennyHeader() {
            const [datetime, setDatetime] = React.useState(new Date());

            function updateHeader() {
               window.setInterval(
                  () => {setDatetime(new Date());},
                  1000
               );
            }

            React.useEffect(updateHeader, []);

            let hours = datetime.getHours();
            let ampm = (hours < 12) ? 'morning' : 'afternoon';
            return (
               <div>
                  <hr />
                  Good {ampm} and welcome to Penny.com
                  <hr />
               </div>
            );
         }

         //-------------------------------------------------------------

         function PennySearch() {
            const [author, setAuthor] = React.useState('');
            const [books, setBooks] = React.useState([]);

            function fetchBooks() {
               function handleResponse() {
                  if (this.status !== 200) {
                     alert('Error: Failed to fetch data from server');
                     return;
```

PennyReact1/index.html (Page 2 of 3)

```
                  }
                  let books = JSON.parse(this.response);
                  setBooks(books);
               }

               function handleError(request) {
                  if (request.statusText !== 'abort')
                     alert('Error: Failed to fetch data from server');
               }

               let encodedAuthor = encodeURIComponent(author);
               let url = '/searchresults?author=' + encodedAuthor;
               let request = new XMLHttpRequest();
               request.onload = handleResponse;
               request.onerror = handleError;
               request.open('GET', url);
               request.send();
               return () => {request.abort();}
            }

            React.useEffect(fetchBooks, [author]);

            let timer = null;

            function debouncedSetAuthor(author) {
               clearTimeout(timer);
               timer = setTimeout(()=>{setAuthor(author);}, 500);
            }

            return (
               <div>
                  <h1>Author Search</h1>
                  Please enter an author name: 
                  <input
                      type='text'
                      onInput={ (event) => {
                         debouncedSetAuthor(event.target.value);
                      }}
                      autoFocus
                  />
                  <hr />
                  {books.map((book) => (
                     <div key={book.isbn}>
                        {book.isbn}: 
                        <strong>{book.author}</strong>: 
                        {book.title}
                        <br />
                     </div>
                  ))}
               </div>
            );
         }

         //-------------------------------------------------------------

         function PennyFooter() {
            const [datetime, setDatetime] = React.useState(new Date());

            function updateFooter() {
               window.setInterval(
                  () => {setDatetime(new Date());},
                  1000
               );
            }

```

PennyReact1/index.html (Page 3 of 3)

```
            React.useEffect(updateFooter, []);

            return (
               <div>
                  <hr />
                  Date and time: {datetime.toLocaleString()}
                  <br />
                  Created by 
                  <a href="https://www.cs.princeton.edu/~rdondero">
                  Bob Dondero</a>
               </div>
            );
         }

         //---------------------------------------------------------------

         function App() {
            return (
               <div>
                  <PennyHeader />
                  <PennySearch />
                  <PennyFooter />
               </div>
            );
         }

         //---------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>

```

blank (Page 1 of 1)

```
1: This page is intentionally blank.
```

React Examples

- **Problem** (not really, but let's pretend)
  - The PennySearch function is too long
  - The PennySearch component is too complex
- **Solution**
  - Factor the PennySearch function into subordinate functions
  - Factor the PennySearch component into child components

React Examples

- See **PennyReact2** app (cont.)
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyReact2/index.html (Page 1 of 3)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <div id="root"></div>

      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
      </script>

      <script type="text/babel">

         'use strict';

         //---------------------------------------------------------------

         function PennyHeader() {
            const [datetime, setDatetime] = React.useState(new Date());

            function updateHeader() {
               window.setInterval(
                  () => {setDatetime(new Date());},
                  1000
               );
            }

            React.useEffect(updateHeader, []);

            let hours = datetime.getHours();
            let ampm = (hours < 12) ? 'morning' : 'afternoon';
            return (
               <div>
                  <hr />
                  Good {ampm} and welcome to Penny.com
                  <hr />
               </div>
            );
         }

         //---------------------------------------------------------------

         function PennyInput(props) {
            return (
               <div>
                  <h1>Author Search</h1>
                  Please enter an author name: 
                  <input
                      type='text'
                      onInput={ (event) => {
                         props.callback(event.target.value);
```

PennyReact2/index.html (Page 2 of 3)

```
                      }}
                      autoFocus
                  />
                  <hr />
               </div>
            );
         }

         //---------------------------------------------------------------

         function PennyOutput(props) {
            return (
               <div>
                  {props.books.map((book) => (
                     <div key={book.isbn}>
                        {book.isbn}: 
                        <strong>{book.author}</strong>: 
                        {book.title}
                        <br />
                     </div>
                  ))}
               </div>
            );
         }

         //---------------------------------------------------------------

         function PennySearch() {
            const [author, setAuthor] = React.useState('');
            const [books, setBooks] = React.useState([]);

            function fetchBooks() {
               function handleResponse() {
                  if (this.status !== 200) {
                     alert('Error: Failed to fetch data from server');
                     return;
                  }
                  let books = JSON.parse(this.response);
                  setBooks(books);
               }

               function handleError(request) {
                  if (request.statusText !== 'abort')
                     alert('Error: Failed to fetch data from server');
               }

               let encodedAuthor = encodeURIComponent(author);
               let url = '/searchresults?author=' + encodedAuthor;
               let request = new XMLHttpRequest();
               request.onload = handleResponse;
               request.onerror = handleError;
               request.open('GET', url);
               request.send();
               return () => {request.abort();}
            }

            React.useEffect(fetchBooks, [author]);

            let timer = null;

            function debouncedSetAuthor(author) {
               clearTimeout(timer);
               timer = setTimeout(()=>{setAuthor(author);}, 500);
            }
```

PennyReact2/index.html (Page 3 of 3)

```
            return (
               <div>
                    <PennyInput callback={debouncedSetAuthor} />
                    <PennyOutput books={books} />
               </div>
            );
         }

         //-------------------------------------------------------------

         function PennyFooter() {
            const [datetime, setDatetime] = React.useState(new Date());

            function updateFooter() {
               window.setInterval(
                  () => {setDatetime(new Date());},
                  1000
               );
            }

            React.useEffect(updateFooter, []);

            return (
               <div>
                  <hr />
                  Date and time: {datetime.toLocaleString()}
                  <br />
                  Created by 
                  <a href="https://www.cs.princeton.edu/~rdondero">
                  Bob Dondero</a>
               </div>
            );
         }

         //-------------------------------------------------------------

         function App() {
            return (
               <div>
                  <PennyHeader />
                  <PennySearch />
                  <PennyFooter />
               </div>
            );
         }

         //-------------------------------------------------------------

         let domRoot = document.getElementById('root');
         let reactRoot = ReactDOM.createRoot(domRoot);
         reactRoot.render(
            <React.StrictMode>
               <App />
            </React.StrictMode>
         );

      </script>
   </body>
</html>
```

blank (Page 1 of 1)

1: This page is intentionally blank.

Agenda

- React concepts
- React simple examples
- React examples
- **jQuery vs. React**

jQuery vs. React

- Repeating the fundamental idea…

- **jQuery**
  - HTML code contains JavaScript code
- **React**
  - HTML code is generated by JavaScript code

jQuery vs. React

- **jQuery**
  - HTML code contains JavaScript code
  - Modularity by **technologies**
- **React**
  - HTML code is generated by JavaScript code
  - Modularity by **components**

jQuery vs. React

- Commentary:
  - Is jQuery code more maintainable?
  - Is React code more reusable?

Summary

- We have covered:
  - React concepts
  - React examples
  - jQuery vs. React
- See also:
  - **Appendix 1**: Arrow functions

Appendix 1:
Arrow Functions

Arrow Functions

- Recall from JavaScript lectures…
- **Question**: How is `this` bound within a function `f()`?
- **Answer**: Depends upon how `f()` is called:

Function Call	Binding of `this`
`f()`	In `f()`, `this` is `undefined`
`o.f()`	In `f()`, `this` is bound to `o`
`new f()`	In `f()`, `this` is bound to a new empty object

Arrow Functions

- Some terms for this lecture:
  - **Ordinary function**: a non-arrow function
  - **Ordinary variable**: a non-`this` variable

Arrow Functions

- ***Arrow function def expressions***
  - Informally ***arrow functions***
  - Arrow functions vs ordinary functions:
    - More succinct
    - Same semantics - **mostly!!!**

Aside: setInterval & setTimeout

In browsers:

```
window.setInterval(f, ms);
// Call f every ms milliseconds
```

We have seen

```
window.setTimeout(f, ms);
// Call f after ms milliseconds
```

We have seen

In Node.js:

```
setInterval(f, ms);
// Call f every ms milliseconds
```

```
setTimeout(f, ms);
// Call f after ms milliseconds
```

We'll use now

Arrow Functions

- **Fact 1**: In an ordinary function…
  - The value of `this` is determined **dynamically**
    - Based upon the call
    - `o.f()`
      - In the function `this` is bound to `o`
    - `f()`
      - In the function `this` is undefined

Arrow Functions

```
$ node arrow1.js
undefined
$
```

- See **arrow1.js**
  - Notes:
    - Global code calls `main()`
    - `main()` calls `blueCar.writeColor()`
    - `blueCar.writeColor()` calls `setTimeout()`
    - `setTimeout()` calls given ordinary function
      - As `f()`, not as `o.f()`
    - In ordinary function, `this` is undefined

arrow1.js (Page 1 of 1)

```
 1: //-------------------------------------------------------------------------
 2: // arrow1.js
 3: // Author: Bob Dondero
 4: //-------------------------------------------------------------------------
 5:
 6: 'use strict';
 7:
 8: class Car {
 9:
10:    constructor(color) {
11:       this._color = color;
12:    }
13:
14:    writeColor() {
15:       // Error: this is undefined.
16:       setTimeout(
17:          function () { process.stdout.write(this._color + '\n'); },
18:          2000
19:       );
20:    }
21: }
22:
23: function main() {
24:    let blueCar = new Car('blue');
25:    blueCar.writeColor();
26: }
27:
28: if (require.main === module)
29:    main();
```

arrow2.js (Page 1 of 1)

```
 1: //-------------------------------------------------------------------------
 2: // arrow2.js
 3: // Author: Bob Dondero
 4: //-------------------------------------------------------------------------
 5:
 6: 'use strict';
 7:
 8: class Car {
 9:
10:    constructor(color) {
11:       this._color = color;
12:    }
13:
14:    writeColor() {
15:       let self = this;
16:       setTimeout(
17:          function () { process.stdout.write(self._color + '\n'); },
18:          2000
19:       );
20:    }
21: }
22:
23: function main()
24: {
25:    let blueCar = new Car('blue');
26:    blueCar.writeColor();
27: }
28:
29: if (require.main === module)
30:    main();
```

Arrow Functions

- **Fact 2**: In an ordinary function…
  - The value of an ordinary variable is determined **statically**
    - Based upon program block structure

Arrow Functions

```
$ node arrow2.js
blue
$
```

- See **arrow2.js**
  - Notes:
    - Global code calls `main()`
    - `main()` calls `blueCar.writeColor()`
    - `blueCar.writeColor()` calls `setTimeout()`
    - `setTimeout()` calls given ordinary function
      - As `f()`, not as `o.f()`
    - In ordinary function, `this` is undefined
      - But the ordinary function doesn't use `this`!

arrow1.js (Page 1 of 1)

```
 1: //-------------------------------------------------------------------------
 2: // arrow1.js
 3: // Author: Bob Dondero
 4: //-------------------------------------------------------------------------
 5:
 6: 'use strict';
 7:
 8: class Car {
 9:
10:    constructor(color) {
11:       this._color = color;
12:    }
13:
14:    writeColor() {
15:       // Error: this is undefined.
16:       setTimeout(
17:          function () { process.stdout.write(this._color + '\n'); },
18:          2000
19:       );
20:    }
21: }
22:
23: function main() {
24:    let blueCar = new Car('blue');
25:    blueCar.writeColor();
26: }
27:
28: if (require.main === module)
29:    main();
```

arrow2.js (Page 1 of 1)

```
 1: //-------------------------------------------------------------------------
 2: // arrow2.js
 3: // Author: Bob Dondero
 4: //-------------------------------------------------------------------------
 5:
 6: 'use strict';
 7:
 8: class Car {
 9:
10:    constructor(color) {
11:       this._color = color;
12:    }
13:
14:    writeColor() {
15:       let self = this;
16:       setTimeout(
17:          function () { process.stdout.write(self._color + '\n'); },
18:          2000
19:       );
20:    }
21: }
22:
23: function main()
24: {
25:    let blueCar = new Car('blue');
26:    blueCar.writeColor();
27: }
28:
29: if (require.main === module)
30:    main();
```

Arrow Functions

- **Fact 3**: In an arrow function…
  - The value of `this` (and any ordinary variable) is determined **statically**
    - Based upon program block structure

Arrow Functions

```
$ node arrow3.js
blue
$
```

- See **arrow3.js**
  - Notes:
    - Global code calls `main()`
    - `main()` calls `blueCar.writeColor()`
    - `blueCar.writeColor()` calls `setTimeout()`
    - `setTimeout()` calls given arrow function
      - As `f()`, not as `o.f()`
    - In arrow function, `this` is bound to `blueCar`

arrow3.js (Page 1 of 1)

```
 1: //-----------------------------------------------------------------------
 2: // arrow3.js
 3: // Author: Bob Dondero
 4: //-----------------------------------------------------------------------
 5:
 6: 'use strict';
 7:
 8: class Car {
 9:
10:    constructor(color) {
11:       this._color = color;
12:    }
13:
14:    writeColor() {
15:       setTimeout(
16:          () => { process.stdout.write(this._color + '\n'); },
17:          2000
18:       );
19:    }
20: }
21:
22: function main() {
23:    let blueCar = new Car('blue');
24:    blueCar.writeColor();
25: }
26:
27: if (require.main === module)
28:    main();
```

Arrow Functions

- **Question**: Why use arrow functions?
- **Answer 1**: They're often more succinct
- **Answer 2**: `this` is defined statically

- Arrow functions often are appropriate as callback functions