Class 21: Interactivity with Snippets

- Review: Implementing Static Interactivity
- Coding Interactivity with Code Snippets
- Coding a Gallery with Code Snippets
- Selecting Snippets
- Troubleshooting Tips

Copyright 2026, Kyle J. Harms

Review: Implement Static Interactivity

✅ Phase I

1. Plan, then **code** *all* interactivity and non-interactivity HTML + CSS.
2. Plan, then **code** initial/default page state using *only* **CSS classes**.

Phase II

3. Plan, then **code** listening for user events in JavaScript.
4. Plan, then **code** responding to events by adding/removing **CSS classes**.

Review: Phase I Implementation Example

1. Plan, then **code** *all* interactivity and non-interactivity HTML + CSS.
2. Plan, then **code** initial/default page state using *only* **CSS classes**.

```
<dialog open id="cookie-banner" class="banner">
  <p>This site remembers the tickets in your cart.</p>
  <button id="cookie-accept-button" class="button">Got it</button>
</dialog>
```

```
.hidden {
  display: none !important;
}
```

Review: Interactivity JavaScript

New JavaScript File:

`scripts/cookie-banner.js`

Load JavaScript:

```
  <script src="scripts/jquery-4.0.0.min.js"></script>
  <script src="scripts/cookie-banner.js"></script>
</body>
```

Implement Static Interactivity

✅ Phase I

1. Plan, then **code** *all* interactivity and non-interactivity HTML + CSS.
2. Plan, then **code** initial/default page state using *only* **CSS classes**.

Phase II

3. Plan, then **code** listening for user events in JavaScript.
4. Plan, then **code** responding to events by adding/removing **CSS classes**.

Interactivity with Code Snippets

Code Snippets

Snippets are code **examples** with some steps missing.
(i.e. completion problem)

Idea: Code interactivity through **example-based programming**.

Example: When the user clicks on an element:

```
$("TODO: CSS selector").on("click", function () { // ← click event

  TODO: response to event

}) // ← click event
```

Why Snippets?

There is no pre-req for this course. While *you* may know programming, **many** of your peers may not have learned it yet.

Example-based programming (snippets) is an **evidence-based** approach to program without writing the syntax yourself using **completion problems** (examples with key steps missing).

- Each snippet is a **building block**.
- Assemble the *building blocks* to create your interactivity.

Example Interactivity Snippets

Event Snippet: On-Click Element

```
$("TODO: CSS selector").on("click", function () { // ← click event
  // ↓ action snippets ↓
  TODO: action snippet(s)
  // ↑ action snippets ↑
}) // ← click event
```

Action Snippet: Add CSS Class to Element

```
$("TODO: CSS selector").addClass("TODO: CSS class name")
```

Action Snippet: Remove CSS Class from Element

```
$("TODO: CSS selector").removeClass("TODO: CSS class name")
```

Programming with Snippets

1. ✅ **Plan** your interactivity using pseudocode.
2. Copy pseudocode into `.js` file.
3. Format the pseudocode as JavaScript comments (select it, `Ctrl+/` or `Cmd+/`).

```
// This is a comment
```

Example:

```
// when #buttonB clicked:
//   add .hidden to #itemA
//   remove .hidden from #itemB
//   add .hidden to #itemC
```

Programming with Snippets (Autocomplete)

4. Use Codespaces' **autocomplete** to add the snippet for **each line** of pseudocode.
    1. Start typing the **type** of snippet: `event:`, `action:`, or `if:`
    2. Select the desired snippet from the autocomplete list.
    3. Complete the **first** `TODO` in the snippet.
    4. Press `TAB` to move to the next `TODO` and complete it.
    5. Repeat until all `TODO`s are completed.

Change *only* the `TODO`s. Edit anything else = 0 credit. No leniency.

Snippet Types

`event:` When

```
event: click element
event: page ready
event: resize browser window
event: modal closed
```

`action:` Do

```
action: add css class
action: remove css class
action: open modal
action: close modal
action: debug console message
```

`if:` Choose

```
if: browser narrow or wide
if: element has css class
```

`event:` snippets hold `action:` **and** `if:` snippets.
`if:` snippets hold `action:` snippets.

`if:` Snippets

An `if:` snippet doesn't **do** anything.

It **chooses** which `action:` snippets execute.

Pseudocode Plan

```
when #hamburger clicked:
  if #nav has .hidden:
    remove .hidden from #nav
  else:
    add .hidden to #nav
```

Snippets Plan

```
event: click element
  if: element has css class
    action: remove css class
  (else)
    action: add css class
```

How to: Snippet Autocomplete – Type

1. Type the **type** of snippet: `event:` , `action:` , or `if:`

```
6   // when #buttonB is clicked:
7   //    add .hidden to #itemA
8   //    remove .hidden from #itemB
9   //    add .hidden to #itemC
10  ✨
11  event:
12
13  // whe
14  //    a
15  //    add .hidden from #itemB
```

```
event: click element                    eventOnClick
event: page load                        eventOnPageLoaded
event: resize browser window            eventOnWindowResize
```

How to: Snippet Autocomplete – Select

2. Select the desired snippet from the autocomplete list.

```
6   // when #buttonB is clicked:
7   //   add .hidden to #itemA
8   //   remove .hidden from #itemB
9   //   add .hidden to #itemC
10
11  $("TODO: CSS selector").on("click", function() { // ← click event
12
13    // ↓ action snippets ↓
14    TODO: action snippet(s)
15    // ↑ action snippets ↑
16
17  }); // ← click event
18
19
```

How to: Snippet Autocomplete – Tab

3. Complete the **first** `TODO` in the snippet.
4. Press `TAB` to move to the next `TODO` and complete it.

```
6   // when #buttonB is clicked:
7   //   add .hidden to #itemA
8   //   remove .hidden from #itemB
9   //   add .hidden to #itemC
10
11  $("#buttonB").on("click", function() { // ← click event
12
13    // ↓ action snippets ↓
14    TODO: action snippet(s)
15    // ↑ action snippets ↑
16
17  }); // ← click event
18
19
```

How to: Snippet Autocomplete – Complete

5. Repeat until all `TODO`s are completed.

```
// when #buttonB is clicked:
//   add .hidden to #itemA
//   remove .hidden from #itemB
//   add .hidden to #itemC

$("#buttonB").on("click", function() { // ← click event

  // ↓ action snippets ↓
  $("#itemA").addClass("hidden");
  $("#itemB").removeClass("hidden");
  $("#itemC").addClass("hidden");

  // ↑ action snippets ↑

}); // ← click event

```

Demo: Gallery Snippets

✅ Event Snippet: On-Click Element

```
event: click element
```

✅ Action Snippet: Add CSS Class

```
action: add css class
```

✅ Action Snippet: Remove CSS Class

```
action: remove css class
```

Pseudocode Plan

```
when #buttonA clicked:
  remove .hidden from #itemA
  add .hidden to #itemB
  add .hidden to #itemC

when #buttonB clicked:
  add .hidden to #itemA
  remove .hidden from #itemB
  add .hidden to #itemC

when #buttonC clicked:
  add .hidden to #itemA
  add .hidden to #itemB
  remove .hidden from #itemC
```

Demo: Debug Console Message

```
when the page loads:
  log "page ready" to the console
```

`event:` Snippets

```
event: page ready
```

`action:` Snippets

```
action: debug console message
```

Demo: Is the Screen Wide or Narrow?

```
when the page loads:
  if window is narrow:
    log "page ready on narrow screen" to the console
  else (if window is wide):
    log "page ready on wide screen" to the console
```

`event:` Snippets

```
event: page ready
```

`if:` + `action:` Snippets

```
if: browser narrow or wide
```

```
action: debug console message
```

Activity: Pseudocode → Snippets

Identify the snippets needed to implement this on your **handout** sheet.

```
when #cookie-accept-button clicked:
  add .hidden to #cookie-banner
```

`event:` Snippets

```
event: click element
```

```
event: page ready
```

```
event: resize browser window
```

`action:` Snippets

```
action: add css class
```

```
action: remove css class
```

`if:` Snippets

```
if: browser narrow or wide
```

```
if: element has css class
```

Activity: Pseudocode → Snippets

Working with a peer, **implement** the following pseudocode using snippets in the activity codespace.

Pseudocode Plan

```
when #cookie-accept-button clicked:
  add .hidden to #cookie-banner
```

Snippets Plan

```
event: click element
  action: add css class
```

Example: Cookie Banner Snippets

```
// when #cookie-accept-button clicked:
//   add .hidden to #cookie-banner

$("#cookie-accept-button").on("click", function () { // ← click event

  // ↓ action snippets ↓
  $("#cookie-banner").addClass("hidden")
  // ↑ action snippets ↑

}) // ← click event
```

Professionalism: Troubleshooting Your Own Code

In industry, you will troubleshoot your own code; there are no office hours.

Yes, AI can do it for you.

However, you still need to know how to troubleshoot your own code to be a successful developer.

The best way to learn to troubleshoot is to practice it!

Troubleshooting Interactivity

1. Did you load the JavaScript file in the HTML?
2. Is the JavaScript loaded as the very last element in the `<body>`?
3. Is jQuery loaded **before** your JavaScript file?
4. Do all of your interactivity elements have a *unique* ID? (`id=""`)
5. Does your code match the snippet **exactly**? (Only the `TODO`s change.)
6. Did you forget the `#` in the **selector** in the snippet?
7. Did you mistakenly add a `.` to the **class name** in the add/remove class snippet?

Troubleshooting Interactivity

8. Are you adding a class when you should be removing one?
9. Are you removing a class when you should be adding one?
10. Does the CSS class exist in your CSS file?
11. If the CSS class styling is subtle, temporarily make it loud:
    `background-color: red;`
12. Use `action: debug console message` to test that an **event** works.
13. Is the HTML and CSS correct? Validate it!
14. Are there errors in the JavaScript code? Check the console.

Activity: Troubleshoot the Cookie Banner

1. Did you load the JavaScript file in the HTML?
2. Is the JavaScript loaded as the very last element in the `<body>`?
3. Is jQuery loaded **before** your JavaScript file?
4. Do all of your interactivity elements have a *unique* ID? (`id=""`)
5. Does your code match the snippet **exactly**? (Only the `TODO`s change.)
6. Did you forget the `#` in the **selector** in the snippet?
7. Did you mistakenly add a `.` to the **class name** in the add/remove class snippet?
8. Are you adding a class when you should be removing one?
9. Are you removing a class when you should be adding one?
10. Does the CSS class exist in your CSS file?
11. If the CSS class styling is subtle, temporarily make it loud: `background-color: red;`
12. Use `action: debug console message` to test that an **event** works.
13. Is the HTML and CSS correct? Validate it!
14. Are there errors in the JavaScript code? Check the console.

JavaScript Errors and Validation

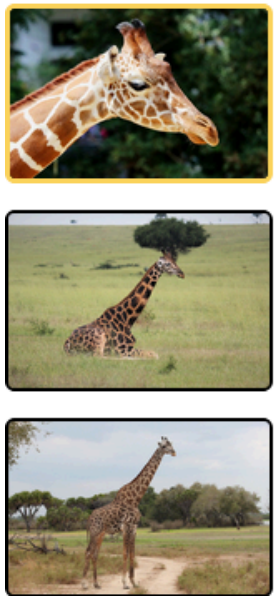
There is no **validator** for JavaScript.

You must **run** your code and test it to check for errors.

Test your interactivity and then check the developer tools, **console tab**, for error messages.

No errors? Your code is valid! (Warnings are okay.)

Demo: JavaScript Validation





Our friend, Jeff. *Source*

```
when #buttonA clicked:
  remove .hidden from #itemA
  add .hidden to #itemB
  add .hidden to #itemC

when #buttonB clicked:
  add .hidden to #itemA
  remove .hidden from #itemB
  add .hidden to #itemC

when #buttonC clicked:
  add .hidden to #itemA
  add .hidden to #itemB
  remove .hidden from #itemC
```

Activity: Cookie Banner JavaScript Validation

1. Open the Developer Tools **Console** tab.
2. Refresh the page.
3. Test the functionality of the interactivity.
4. When done testing, check the Console tab for any JavaScript errors. (warnings are okay.)

   Note: Ignore errors for Codespaces (unrelated to your website).

   All errors should list **your** JavaScript file as the *source*.

Gotcha: Assignments Require Snippets

You are **required** to implement the snippets using the method taught in this class.

You may insert snippets and replace the `TODO`s. You may **not** modify the snippet code (including comments) in any way.

If you modify the snippet code, including comments, or submit non-snippet JavaScript code you will receive **0 credit** for the assignment. No leniency.

What's Next

Today: Milestone 3 Due
Today: Homework 8 Released

Wednesday: Accessible Interactivity