Reviewing JS-Cart

Goal: Practice how to mingle

- Events
- State
- HTML updates/changes

...with front end JS

Different than server-side

- not request/response

Key Concepts

- State + Project Architecture
- Managing HTML
- Separating Concerns
- Source of Truth

What is Architecture?

- Planning how the pieces connect

Key concept there is *planning*

- Advanced developers will tell you the majority of coding involves no typing of code

Data Models and Structures

> *Bad programmers worry about the code. Good programmers worry about data structures and their relationships.*
>
> - *Linus Torvalds, Creator of Linux kernel and Git*

- What does your inventory look like?
- What does your cart data look like?
- What impact do these decisions have?

Project Files and Structure

- Where is your "main code"?
  - Runs on page load
  - Imports and uses other code
- If I'm looking for a bit of code
  - Do I know what file to look in?
- If I have a variable/function/file name
  - Do I know what it represents from the name?

Improving your architecture

- Get past a project by
    - Get it working by the deadline
- Improve as a coder by
    - Not being content with "working"
    - Notice "pain points"
        - Find better solutions
    - Ask questions
        - Seek answers
    - Pace yourself
        - Can't learn it All

Understand not just the How, but the Why

Following a best practice

- Makes you good to work with

Understanding WHY the best practice

- Makes you great to work with
- Able to solve changes
- Able to apply nuance
    - say and understand "it depends"

6250 Assignments are Tough!

You should have questions

- I can answer some
- Other answers come with practice

Intent is to get you used to tackling problems where

- You know enough to make a working solution
- But not so much that you have no questions
- This mimics what should happen on the job
- Use the uncertainty to get better

Generating HTML for JS-Cart

- I should have given some example pictures
    - My bad
    - I counted on familiar ecommerce sites
    - I simplified, but it made confusing
- BUT good practice for jobs
    - Many jobs will have a design to imitate
    - Other times you have to make functional first

Practice turning limited information into a plan

When you have certain needs

- What data models enable it?
- What elements does a page need?
- What is semantic vs styling?
- What concerns do you want to separate?
- What to plan before you start coding?
- What to code first?

JS Cart Example

- Define data models
    - Write and export them
- Define base HTML for product page
- Write initial display
    - Confirm it works
- Write add to cart
- Define base HTML for cart view
- Write View Cart
- Write Edit Quantity option
- Write Checkout option

Separation of Concerns

- Data Models
    - Is the quantity in the inventory or the cart?
- Generation of HTML?
    - No shared scope between files
    - Requires passing state to generating function
        - That's good!

Example Passing State

Example trimmed down for space

```
export const products = [
  { name: 'Meow is the time', price: 0.99 },
  { name: 'Brutal Fluff', price: 3.14 },
];
```

```
export const generateProductHtml = (state) => {
   const html = state.products.map( (cat, index) => `
   <li class="cat">
      <span class="cat-name">${cat.name}</span>
      <span class="cat-price">$${cat.price.toFixed(2)}</span>
      <button type="button"
        class="cat-add"
        data-index=${index}
      >Add to Cart</button>
    </li>
    `;
    return `<ul class="cats">${html}</ul>`;
});
```

Source of Truth

Code should always have a **single source of truth**

- Otherwise when sources of truth don't agree
    - Subtle bug
    - Disagreement more likely

A single source of truth can still be wrong

- A bug
- But a more OBVIOUS bug

Truth in JS Cart

What is in your cart data?

- Is it a copy of the inventory data?

Ideal is cart only has its own data + references

- `quantity`
- Product index (if products is an array)
- Product key (if products is an object)

When cart needs name, pic, and price

- All pulled from the single source of truth
    - products

Parts of JS Cart should "feel" wrong

- Event to State to Render should feel good
    - But likely still new and unfamiliar
- But rendering can feel clumsy
    - Writing HTML in JS
    - Replacing a LOT of HTML for any state change

These are the right responses!

You are learning state management

A separated state is

- Best practice
- Able to handle changes with minimal growing complexity
- Unnatural and inhuman

You have to learn to think this way

- Learning isn't instant

Writing HTML feels clumsy

Writing HTML in JS

- Not ideal

There are templating libraries

- Make it a little easier
- Do the same thing

We are skipping such libraries

- Understand what it is doing
- Will jump past to React very soon

Rendering feels wasteful

We rewrite a LOT of HTML on ANY state change

We could track which HTML depends on which state

- Write wrapper functions to change that state
- Trigger re-render of just those parts of HTML
- Would be a lot of work
    - Lots of edge cases and bugs to fix
- People have already done this work
    - Such as React
- For now we focus on learning OTHER aspects
    - So don't worry about efficiency for now