

# Dropdown menus

- Common desire
- Can be done many ways
  - Pure CSS or with JS
- Surprisingly tricky

# **Base Concept of Implementation**

- Menu placed over content, doesn't move content
- Menu is initially hidden
- Showing/not showing reacts to user
- Menu stays open while submenu is navigated

# Menu overlapping page content

- Overlap means `position: absolute;`
  - May mean `position: relative;` on parent
    - Create **positioned container**
    - To set menu position

# Menu options initially hidden

## Multiple options

- `display: none;`
  - Removes from accessibility
- `height/max-height: 0;`
  - Still in content
  - Visible to Assistive Tech
  - Not visible to users

# Menu shows/not shows

JS can handle interactivity

- But we haven't gotten there yet

Pure CSS has to react to "state"

- pseudo-selectors:
  - `focus`/`focus-within`/`hover`/`active`
  - `hover` has a11y problems

# HTML structure

- MANY options
  - Makes googling hard
- Common pattern
  - Menu is `<ul>`
  - Each item may have text and a `<ul>`
  - "Text" may or may not be a link
  - Submenu `<ul>` contains `<a>` links

# Sample

```
<ul class="menu">
  <li>
    <button type="button">Sleep</button>
    <ul>
      <li><a href="#">Paw over face</a></li>
      <li><a href="#">Head down</a></li>
      <li><a href="#">Curled up</a></li>
      <li><a href="#">On back</a></li>
    </ul>
  </li>
  <li>
    <button type="button">Eat</button>
    <ul>
      <li><a href="#">Scarf</a></li>
      <li><a href="#">Puke</a></li>
      <li><a href="#">Yowl if 10% of food is gone</a></li>
      <li><a href="#">Bury food</a></li>
    </ul>
  </li>
</ul>
```

# Why buttons?

- Could be anything to make it work (span, etc)
- Buttons have a11y and keyboard benefits
- Notice `type="button"`
  - They don't DO anything when pressed
    - JS could, but we aren't using any
  - Buttons let us navigate with keyboard
    - Or touch to focus (mobile!)
  - We can style them to not look weird



# Styling needs

CSS will be complex :(

- Style menu and submenu and links
- Position and hide submenu
- Show submenu when condition is met

# CSS complications

A11y?

- No `:hover`
  - At least not by itself
- Only some elements navigate with keyboard
  - Thus why we had buttons
- Need `:focus-within` to work w/keyboard
- Never remove outline/focus indicator

# Basics

```
.menu {  
  display: flex;  
  flex-direction: row;  
  justify-content: space-evenly;  
  
  padding: 0;  
  margin: 0;  
  
  list-style: none;  
}
```

```
.menu button { /* Button look like link */  
  border: none;  
  
  background: inherit; /* New property! (to us) */  
  color: blue;  
  
  cursor: pointer; /* Another new one! */  
  text-decoration: underline;  
}
```

# Convert to Menus

```
.menu > li {  
  position: relative; /* positioned container */  
  
  flex-grow: 1; /* New! */  
  /* justify-content on parent now pointless */  
}  
  
.menu ul {  
  position: absolute; /* position OVER text */  
  
  max-height: 0; /* Hide menu */  
  padding: 0;  
  margin: 0;  
  
  overflow: hidden; /* Hide menu contents */  
  
  background-color: white; /* see-thru bad */  
  
  list-style: none;  
}
```

# Make visible

- When the main main `<li>` is hovered
- select the ul inside it
- And change the `max-height`

```
.menu li:hover ul { /* woah */  
  max-height: initial;  
}
```

# Keyboard friendly

`:hover` doesn't work for keyboard/tables

- Also act when the button has `focus`

```
.menu li:hover ul,  
.menu button:focus ~ ul { /* wut? */  
  max-height: initial;  
}
```

- Select `<ul>` that is sibling of button with focus

# Keyboard navigation

Great, but that breaks when the keyboard is used to navigate into the menu

```
.menu li:hover ul,  
.menu li:focus-within ul {  
  max-height: initial;  
}
```

- Select `<ul>` inside the main menu `<li>` that has focus inside it

# Clean up

- CSS classes are better
  - Flat specificity
  - More communicative
  - You have to translate, not copy :)
- Padding on dropdown menu
- Line-height on dropdown menu
- Distinguish behavior for mobile vs desktop



# Advanced variations

Many ways to use CSS based on state

- Allow changes even without JS
- Example: "Hamburger" Menu
  - Can use a hidden "checkbox"

JS can be better choice for a11y!

- Semantic HTML is key though

# Summary - Dropdown Menu

- Start with Semantic HTML
- Consider multiple navigation
  - Assistive Technology
  - Mouse
  - Keyboard
  - Touch
- Buttons (controls) and links (nav)
  - Built-in options
  - Can style differently
  - `<button type="button">` = no submit attempt

# Summary - CSS Problem Solving

- Do one layer/problem at a time
- You can answer simple questions
  - Make everything a simple question
  - Just a lot of them

# Summary - Dropdown positioning

- Dropdown was absolute position
  - Required positioned container
    - with `position: relative;`
- Overlaps rest of page
- `background-color`, `padding` to fix

# Summary - Hiding

- set `max-height` to hide
  - ...in this case
  - Multiple options
- Keep content on page for a11y
- set `overflow: hidden;` to truly hide
  - combine with `max-height`

# Summary - Showing hidden elements

- Use `:hover`, `:focus-within`, `:checked`, `:active`
- Combine to select element you want to change
- `:has()` coming Very Soon to help as well!
- Don't hide the focus outline!

# Summary - Lessons

CSS often does not have "cut-and-paste" answers

- Understand how to solve the needs
- Different answers are "best"
  - Depends on specific details