Web Design – Lecture-12

INSTRUCTOR: VIDHI KESARIA

Northeastern University
Seattle

Class Schedule

- 1) Website Architecture, Design, Strategy and Planning and Creating web pages – Semantic HTML4/5
- 2) Design using CSS (covers CSS3)
- 3) Responsive web design,
- 4)Making the web page Interactive - JavaScript (Datatypes, Inline, Embedded and external JavaScript, variables, operators, loops, functions, arrays, objects
- 5) Working with DOM Model and JavaScript Events
- 6) Using Jquery & Hosting on AWS
- 7) Assignment presentation with demo.
- 8) Local storage, cookies, Personalization & wireframing for project
- 9) Design Style guide, Using LESS Preprocessor + Project work

10) Performance + Project work

- 11) Review styled website with actual content, learn about interactive JavaScript + Project work
- 12) Accessibility Techniques & Tools + Project work
- 13)SEO Techniques + Project work
- 14) Written exams, Final project submission & Demo

Accessibility

- Equal access and opportunity to people with a diverse range of
  - hearing,
  - movement,
  - sight,
  - cognitive abilities
- Availability/accessibility
- Usability/ accessibility

- Not just about compliance but part of user experience

Design Considerations

- Managing Flow
  - A user must be able to use the keyboard to complete all interactions.
  - Order of the flow (forms, section content, etc)
  - Proximity with related content
- Visual Interactions
  - Mindful Redundancy (info from color is also available through headings, subheadings, etc)

S/w Assistive Tools & Techniques

- Voice recognition (Productivity, efficiency /disability)
  - Dictation (avoid typing)
  - Command control (auto interaction – see it & say it)
- Screen magnifiers (low visibility)
- Screen reader (blind)
  - Say everything mode
  - Traditional keyword navigation
  - Move based on types of objects (eg. press H for headings, F for forms)

Personas – Vision issues

- Screen Reader & Screen Maginifier
  - Techniques:
    - Using semantic markup. Links, Form fields, Buttons, Headings, Lists, etc.
    - Interfaces that are accessible with keyboards
    - Text alternative for visuals (eg. summary in form of table for a pie-chart, text for progress bar)

Personals – Hearing issues

- Audio files (eg. Podcast)
  - Text alternative – Transcript
- Video files
  - captions for audio portion
  - Transcript

Personas – Cognitive issues

Issues

- Memory
- Problem solving
- Attention
- Focus
- literacy and reading,
- visual or verbal comprehension

Strategies

- Design for forgetfulness (eg. inline error msg)
- Simple plain language (avoiding jargons)
- Predictable interface (consistency in navigation, steps,

Basic Techniques

- “alt” attributes used for all descriptive images
- Responsive web design (relative units & fluid layouts)
- Way to skip menus
- Accessible Forms
- Accessible Tables
- Color brightness/contrast

Basic Techniques - Forms

- Elements within forms should each have a label, and ensure that each label is explicitly associated with its corresponding form element.
- If a form element does not have a visible label, then use the title attribute to describe its purpose.
- Group together form fields that are similar or related using the fieldset and legend elements.
- Use the HTML element optgroup to group options in a selection list, where it is logical to do so or where the grouping helps the user navigate a long list.
- Test that all elements receive focus in a logical order, including controls.
- Make it clear in a machine-readable way which fields are required in the form.

Tables

- Do not use tables for layouts — tables are for tabular information and data.
- Ensure the information in tables is linearised and makes sense for people using a screen reader.
- Provide a caption that identifies the table using the caption element.
- Provide a summary that gives an overview of the table's structure using the summary attribute.
- Ensure that row and/or column headers are marked using the <th> element.
- Create tables that have a simple structure. Avoid complex tables with merged or split cells and multiple header rows, where possible.
- For data tables that have multiple row and/or column headers and/or merged cells, associate header and data cells using the id and headers attributes.
- Identify structural groups of rows and groups of columns.
- Use the scope attribute to identify whether a header cell is for a row, column, or group of rows or columns.

Skip content

- Allow users to skip blocks of content by adding a link or links to either:
- Use WAI-ARIA landmark roles to identify key regions within a webpage.
- Using section headings to group blocks of content and convey the structure of the content and ensure headings are marked using appropriate heading elements.
- Use only an appropriate amount of 'skip to' and 'skip over' links to avoid complexity

Input Assistance

- Clearly identify required fields.
- Ensure required fields are indicated by more than one method, not only visually. It is preferable that required fields are signified in text as well as any other method.
- Ensure that all additional cues denoting a required form field are within the label element.
- Let users know when an error has occurred and how to correct them.
- Let users check, correct, and/or cancel information submission processes.
- Provide text guidance on how to submit information correctly.
- Provide context-sensitive help where possible.
- Provide error-correcting suggestions where appropriate.
- Provide all of these options in proportion to the consequences of user input error