FRESNO STATE
Discovery. Diversity. Distinction.

Web Programming (CSci 130)

Department of Computer Science
College of Science and Mathematics
California State University Fresno
H. Cecotti

HTML5

- Learning outcomes
  - More Functionalities of HTML5
    - Lists
    - Tables
    - Video
  - WAI-ARIA
- Remark:
  - Outline of books/tutorials
    - Some books separate HTML5 from CSS3
    - Some books go by type of elements, using both HTML5 and CSS3

Lists

- Type of lists
  - Ordered lists
  - Unordered lists
  - Items <li> bla bla
- Examples in HTML5…
  - See HTML files in Canvas

```
<ol>
    <li>Make sure you have unplugged the
    →lamp from the wall socket.</li>
    <li>Unscrew the old bulb.</li>
    <li>Get the new bulb out of the
    →package.</li>
    <li>Check the wattage to make sure
    →it's correct.</li>
    <li>Screw in the new bulb.</li>
    <li>Plug in the lamp and turn it
    →on!</li>
</ol>
```

```
<ul>
    <li>New or improved features marked
    →with a solid bullet.</li>
    <li>One-click page layout</li>
    <li>Spell checker for 327 major
    →languages</li>
    <li>Image retouching plug-in</li>
    <li>Special HTML filters</li>
    <li>Unlimited Undo's and Redo's</li>
    <li>Automatic book writing</li>
</ul>
```

```
<h1>The Great American Novel</h1>
<ol>
    <li>Introduction
      <ol>
        <li>Boy's childhood</li>
        <li>Girl's childhood</li>
      </ol>
    </li>
    <li>Development
      <ol>
        <li>Boy meets Girl</li>
        <li>Boy and Girl fall in love
        →</li>
        <li>Boy and Girl have fight
        →</li>
      </ol>
    </li>
    <li>Climax
      <ol>
        <li>Boy gives Girl ultimatum
          <ol>
            <li>Girl can't believe
            →her ears</li>
            <li>Boy is indignant at
            →Girl's indignance</li>
          </ol>
        </li>
        <li>Girl tells Boy to get
        →lost</li>
      </ol>
    </li>
    <li>Denouement</li>
    <li>Epilogue</li>
</ol>
```

Tables

- Examples in HTML5…
  - ➢ See HTML files in blackboard

```
...
<body>
<table>
    <caption>Quarterly Financials for
    → 1962-1964 (in Thousands)</caption>
    <tr>
       <th scope="col">1962</th>
       <th scope="col">1963</th>
       <th scope="col">1964</th>
    </tr>
    <tr>
       <td>$145</td>
       <td>$167</td>
       <td>$161</td>
    </tr>
    <tr>
       <td>$140</td>
       <td>$159</td>
       <td>$164</td>
    </tr>
    <tr>
       <td>$153</td>
       <td>$162</td>
       <td>$168</td>
    </tr>
    <tr>
       <td>$157</td>
       <td>$160</td>
       <td>$171</td>
    </tr>
</table>
</body>
</html>
```

```
<table>
    <caption>Quarterly Financials for
    → 1962-1964 (in Thousands)</caption>
    <thead> <!-- table head -->
       <tr>
          <th scope="col">Quarter</th>
          <th scope="col">1962</th>
          <th scope="col">1963</th>
          <th scope="col">1964</th>
       </tr>
    </thead>
    <tbody> <!-- table body -->
       <tr>
          <th scope="row">Q1</th>
          <td>$145</td>
          <td>$167</td>
          <td>$161</td>
       </tr>
       <tr>
          <th scope="row">Q2</th>

          <td>$140</td>
          <td>$159</td>
          <td>$164</td>
       </tr>
       ... Q3 and Q4 rows ...
    </tbody>
    <tfoot> <!-- table foot -->
       <tr>
          <th scope="row">TOTAL</th>
          <td>$595</td>
          <td>$648</td>
          <td>$664</td>
       </tr>
    </tfoot>
</table>
```

```
<table>
    <caption>TV Schedule</caption>
    <thead> <!-- table head -->
       <tr>
          <th scope="rowgroup">Time</th>
          <th scope="col">Mon</th>
          <th scope="col">Tue</th>
          <th scope="col">Wed</th>
       </tr>
    </thead>
    <tbody> <!-- table body -->
       <tr>
          <th scope="row">8 pm</th>
          <td>Staring Contest</td>
          <td colspan="2">Celebrity Hoedown
          → </td>
       </tr>
       <tr>
          <th scope="row">9 pm</th>
          <td>Hardy, Har, Har</td>
          <td>What's for Lunch?</td>
          <td rowspan="2">Movie of the Week
          → </td>
       </tr>
       <tr>
          <th scope="row">10 pm</th>
          <td>Healers, Wheelers &amp;amp;
          → Dealers</td>
          <td>It's a Crime</td>
       </tr>
    </tbody>
</table>
```

Break

- Create some tables:

Country	Food/Drinks/ Tobacco	Clothing/ Footwear	Leisure/ Education
Ireland	28.91%	6.43%	2.21%
Italy	16.36%	9.00%	3.20%
Spain	18.80%	6.51%	1.98%
Sweden	15.77%	5.40%	3.22%
Turkey	32.14%	6.63%	4.35%

	average		other category
	height	weight	
males	height	0.003	*yyy*
females	1.9	0.002	xxx

Video

- Example

```
<body>
<video src="myvideo.ext"></video>
</body>
```

- To support all HTML5-capable browsers
  - ➢ need to supply video in at least 2 different formats: MP4 and WebM (sponsored by google).
- Using video with multiple sources

```
<body>
    <video controls="controls">
        <source src="paddle-steamer.mp4"
        → type="video/mp4">
        <source src="paddle-steamer.webm"
        → type="video/webm">
        <p>Sorry, your browser doesn't
        → support the video element</p>
    </video>
</body>
```

Video

```
<body>
    <video src="paddle-steamer.webm"
    → autoplay="autoplay" loop="loop">
    → </video>
</body>
```

```
<video src="paddle-steamer.webm"
→ poster="paddle-steamer-poster.jpg"
→ controls="controls"></video>
```

- Attributes and descriptions
  - **src**: specifies the URL to the video file.
  - **autoplay**: automatically starts playing the video as soon as it can.
  - **controls**: adds the browser's default control set to the video.
  - **muted**: mutes the video's audio (not currently supported by any browser).
  - **loop**: Plays the video in a loop.
  - **poster**: Specifies an image file to display (instead of the first frame of the video) when it loads. It takes a URL to the required image file.
  - **width**: width of the video in pixels.
  - **height**: height of the video in pixels.
  - **preload**: Hints to the browser how much of the video it is to load. 3 different values:
    - **none**: it doesn't load anything.
    - **metadata**: loads only the video's metadata (length and dimensions).
    - **auto:** it lets the browser decide what to do (default setting).

Audio

- 5 main audio codecs for audio files
  - Ogg Vorbis: .ogg file extension
    - Free and open-source software project → Xiph.Org Foundation
    - Supported by Firefox 3.5+, Chrome 5+, and Opera 10.5+.
  - MP3: .mp3 file extension
    - MPEG-1 Audio Layer III or MPEG-2 Audio Layer III
      - perceptual coding: remove components of sound that are considered to be beyond the hearing capabilities
    - Supported by Safari 5+, Chrome 6+, Internet Explorer 9+, and iOS.
  - WAV: .wav file extension
    - Waveform Audio File Format
      - MS and IBM audio file format standard for storing an audio bitstream on PC computers
    - Supported by Firefox 3.6+, Safari 5+, Chrome 8+, and Opera 10.5+.
  - AAC: .aac file extension
    - Advanced Audio Coding (AAC): proprietary audio coding standard for lossy digital audio compression
    - Supported by Safari 3+, Internet Explorer 9+, iOS 3+, and Android 2+.
  - MP4: .mp4 extension
    - Supported by Safari 3+, Chrome 5+, Internet Explorer 9+, iOS 3+, and Android 2+.

Audio

- Need to be in 2 different formats to ensure support across **all** HTML5-capable browsers
  - best formats: Ogg Vorbis & MP3
- Example
  - <body>
  - <audio src="macarena.ogg"></audio>
  - </body>
- Provide multiple audio sources

```
<body>
    <audio controls="controls">
        <source src="piano.ogg" type=
        → "audio/ogg">
        <source src="piano.mp3" type=
        → "audio/mp3">
    </audio>
</body>
```

```
<body>
    <audio controls="controls">
        <source src="piano.ogg" type=
        → "audio/ogg">
        <source src="piano.mp3" type=
        → "audio/mp3">
    <a href="piano.mp3">Download the
    → audio</a>
    </audio>
</body>
```

Cannot play the file, give the link to download it

Audio

- Main audio attributes
  - **src**: specifies the URL to the audio file
  - **autoplay**: Automatically starts playing the audio as soon as it can
  - **controls**: Adds the browser's default control set to the audio
  - **muted**: Mutes the audio (not currently supported by any browser).
  - **loop**: Plays the audio in a loop.
  - **preload**: Hints to the browser how much of the audio it is to load
    - **none**: doesn't load anything.
    - **metadata**: loads only the audio's metadata (e.g. length)
    - **auto**: lets the browser decide what to do (default)

Special case with flash fallbacks

- Video
  - Example:

```
<video controls="controls">
    <source src="paddle-steamer.mp4"
    → type="video/mp4">
    <source src="paddle-steamer.webm"
    → type="video/webm">
    <object type="application/
    → x-shockwave-flash" data=
    → "player.swf?videoUrl=paddle-
    → steamer.mp4&controls=true">
        <param name="movie" value=
        → "player.swf?videoUrl=paddle-
        → steamer.mp4&controls=true" />
    </object>
</video>
```

Audio

```
<audio controls="controls">
    <source src="piano.ogg" type=
    → "audio/ogg">
    <source src="piano.mp3" type=
    → "audio/mp3">
    <object type="application/
    → x-shockwave-flash"
        data="player.swf?audioUrl=
        → piano.mp3&controls=true">
        <param name="movie" value=
        → "player.swf?audioUrl=
        → piano.mp3&controls=true" />
    </object>
</audio>
```

Youtube

- Integrate video
  - Example:
    - <iframe width="560" height="315" src="https://www.youtube.com/embed/I0RBwuv79UM" frameborder="0" allow="autoplay; encrypted-media" allowfullscreen></iframe>

- Create a page where you add
  - A video
    - Use youtube and internet to extract a video file from a youtube link
  - Some music
    - Use youtube and internet to extract the audio from a youtube link
  - Some youtube video

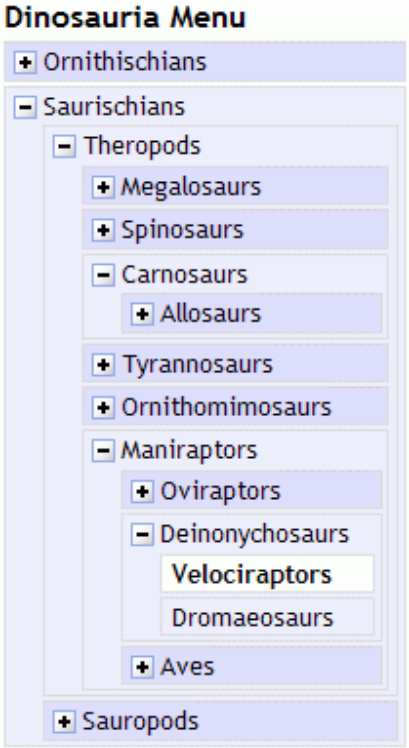
WAI-ARIA

- **Challenge**
  - User interface controls and content updates
    - Often not accessible to users with disabilities
      - Example: screen reader users, people who cannot use a mouse or other pointing device
- **WAI-ARIA**
  - Web Accessibility Initiative – Accessible Rich Internet Applications
  - ARIA roles, states, and properties are analogous to a CSS for assistive technologies. For screen reader users, ARIA controls the rendering of their non-visual experience
  - Addition of
    - role, property, and state information to dynamic web applications
  - for developers of web applications, web browsers, assistive technologies, and accessibility evaluation tools
- Further reading
  - https://www.w3.org/WAI/standards-guidelines/aria/
  - https://www.w3.org/TR/wai-aria-practices/#intro

WAI-ARIA

- Rationale:
  - ➢Web sites are increasingly using more advanced and complex user interface controls
- Problem
  - ➢To provide an accessible UX to people with disabilities
    - ○ →assistive technologies need to be able to interact with these controls.
    - ○ Example: drag-and-drop with only the keyboard and no mouse

  - ➢information that the assistive technologies need **not** available with most current Web technologies ☹



WAI-ARIA

- Example
  - role

```
...
<body>

<!-- Start page container -->
<div id="container">
    <header role="banner">
       ...
       <nav role="navigation">
          ... [ul with links] ...
       </nav>
    </header>

    <!-- Column One when CSS applied -->
    <div id="content" role="main">
       <article>
          ...
       </article>

       <article>
          ...
       </article>

       ... [more sections as desired] ...
    </div>
    <!-- end column one -->

    <!-- Column Two when CSS applied -->
    <div id="sidebar">
       <aside role="complementary">
          ...
       </aside>

       <aside role="complementary">
          ...
       </aside>

       ... [more sections as desired] ...
    </div>
    <!-- end column two -->

    <footer role="contentinfo">
       ...
    </footer>
</div>
<!-- end page container -->

</body>
</html>
```

WAI-ARIA

- Attributes
  - Added to markup like the roles **but** a range of ARIA attributes available for use.
  - ARIA attributes: prefixed with **aria-**
  - 2 types of attributes
    - states
      - value of states are bound to change as a result of user interaction
    - Properties
      - value of states are bound to change as a result of user interaction

```
<span role="checkbox"
      aria-checked="true"
      tabindex="0"
      id="simulatedcheckbox">
</span>
```

  - Example
    - a state is aria-checked
    - → used to show the state of elements that are emulating interactive elements
      - Example: checkboxes and radio buttons but are not the native elements themselves

Conclusion

- HTML for the structure
  - Tags represent a semantic
  - Tags are **meaningful**
  - Remark
    - Don’t be tricked by the default presentation in the browser, it is meaningless. The presentation will come with CSS
- Next week
  - **CSS3**: Let’s put some color and style in the pages ☺