Style Conference Part 8

AIM Code School By: Vanessa Kasun

Let's go ahead and work on the schedule page of our website. We will take what we learned about tables and apply that information here. Follow the directions and please read through each step. Double check any syntax errors if your code is not working correctly.

We'll begin our Schedule page by opening up the schedule.html file and adding a **<section>** element with a class attribute value of row, much like we've done with all of the other subpages. Within this new **<section>** element let's also add a ** ** element with a class attribute value of ***container***.

(remember you should already have the nav, header, and intro information on this page already, if you don't please go ahead and do that)

```
<section class="row">
  <div class="container">
    ...
  </div>
</section>
```

Within the new section we'll add three tables, one for each day of the conference. The tables will display the events of each day using three columns and multiple rows and will include a table head and table body.

To get started let's outline the structure of the first table, including the **<table>, <thead>**, and **<tbody>** elements.

```
<section class="row">
  <div class="container">

    <table>
      <thead>
        ...
      </thead>
      <tbody>
        ...
      </tbody>
    </table>

  </div>
</section>
```

Let's create a new section in our **main.css** file to add some additional styles.

In our new section of styles specifically for the Schedule page (which will appear just below the styles for the Speakers page), let's set our **<table>** elements to have a **width** of 100% and a **bottom margin** of 44 pixels.

Then, using the ***:last-child*** pseudo-class selector to identify the last **<table>** element within the section, let's set its **bottom margin** to 0 pixels. Doing so prevents this table from conflicting with the **bottom padding** belonging to the **<section>** element with a class attribute value of **row**.

So far, the new section within our **main.css** file looks like this:

```
/*
  Schedule
*/

table {
  margin-bottom: 44px;
  width: 100%;
}
table:last-child {
  margin-bottom: 0;
}
```

Now let's add some data to our table. We'll begin with the first day of our conference, the workshop day on August 24.

Within the **<thead>** element of the table let's add a **<tr>** element. The first cell within the row will be a **<th>** element noting the focus of the day: "Workshops" for this specific day. Since the **<th>** element is the heading for the row we'll also add the **scope** attribute with a value of **row** to it.

After the **<th>** element comes a **<td>** element with the date, "August 24th" in this case. Now, more often than not we'll have three columns, the first being a table heading that identifies a time of day and the second two columns being regular table cells that identify speakers for that given time. Since this row doesn't feature two separate speakers we'll want to add the **colspan** attribute with a value of 2 to the **<td>** element, forcing it to span two columns.

Our code for the table now looks like this:

```
<table>
  <thead>
    <tr>
      <th scope="row">
        Workshops
      </th>
      <td colspan="2">
        August 24th
      </td>
    </tr>
  </thead>
  <tbody>
    ...
  </tbody>
</table>
```

On the **<th>** element, and all subsequent **<th>** elements, we'll add the **scope** attribute with a value of **row** to semantically identify this element as the header for the row. Then within the **<th>** element let's add a **<time>** element that shows the time of the first activity of the day, "8:30 AM" in this case. We'll also include a **datetime** attribute on the **<time>** element with a value noting the time in hours, minutes, and seconds, **08:30:00.**

In the <**td>** element that follows the **<th>** element we'll include the activity name (since there aren't any speakers at this time), which is "Registration" in this case. Since there is only one activity at this time we'll also include the **colspan** attribute with a value of 2 on the **<td>** element.

In all, the code for our first table looks like this:

```
<table>
  <thead>
    <tr>
      <th scope="row">
        Workshops
      </th>
      <td colspan="2">
        August 24th
      </td>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">
        <time datetime="08:30:00">8:30 AM</time>
      </th>
      <td colspan="2">
        Registration
      </td>
    </tr>
  </tbody>
</table>
```

For the second row within the **<tbody>** element let's add a **<tr>** element just below our previous row. Then let's add a **<th>** element with the **scope** attribute with a value of row, and again add a **<time>** element with the appropriate time and ***datetime*** attribute value within that **<th>** element.

After the **<th>** element let's add two **<td>** elements for the two speakers presenting at that time. Directly inside each **<td>** element we'll add an **<a>** element, which will link back to where that speaker is positioned on the Speakers page. Remember, we added **id** attributes with each speaker's name to the parent elements for each speaker. Using that **id** attribute value preceded by the **speakers.html** filename and a pound/hash sign, **#,** we can link directly to that speaker's talk description and biography on the Speakers page.

Within the **<a>** element we'll include an <**h4>** element with the speaker's name followed by the talk title.

The code for the first two workshops looks like this: (remember that id value needs to match those of the speakers you created. **EX ."fake-name"**)

```
    <tr>
      <th scope="row">
        <time datetime="08:30:00">8:30 AM</time>
      </th>
      <td colspan="2">
        Registration
      </td>
    </tr>
    <tr>
      <th scope="row">
        <time datetime="09:00:00">9:00 AM</time>
      </th>
      <td>
        <a href="speakers.html#vanessa-kasun">
          <h4>Vanessa Kasun</h4>
       React from Scratch. Creating a React App from Ground up!
        </a>
      </td>
      <td>
        <a href="speakers.html#joshua-burden">
          <h4>Joshua Burden</h4>
         Lights! Camera! Interaction! Design Inspiration from Filmmakers
        </a>
      </td>
    </tr>
  </tbody>
</table>
```

From here, repeat this pattern for each activity speaker to finish the table and then add the next two tables for the second two days of the conference. Keep in mind that the table head will always include a table heading noting the events of the day and a table cell spanning two columns showing the date.

Then, within the body of each table, every row will have a table heading that shows the time of day. After the table heading will be an activity, a speaker, or multiple speakers. Activities without speakers will reside within a single **<td>** element that spans two columns. If only one speaker is presenting at a certain time, that speaker will reside within a single <td> element that spans two columns as well, **<a>** and **<h4>** elements and all.

If there are two speakers for a given time then each speaker will reside within his or her own **<td>** element, just as before.

The full code for all three tables can be found at the end of this exercise. For reference, the table for the first day, which includes lunch and two more speakers, looks like this:

```
<tr>
  <th scope="row">
    <time datetime="09:00:00">9:00 AM</time>
  </th>
  <td>
    <a href="speakers.html#vanessa-kasun">
      <h4>Vanessa Kasun</h4>
     React from Scratch. Creating a React App from Ground up!
    </a>
  </td>
  <td>
    <a href="speakers.html#joshua-burden">
      <h4>Joshua Burden</h4>
     Lights! Camera! Interaction! Design Inspiration from Filmmakers
    </a>
  </td>
</tr>
  <th scope="row">
    <time datetime="12:30:00">12:30 PM</time>
  </th>
  <td colspan="2">
    Lunch
  </td>
</tr>
<tr>
  <th scope="row">
    <time datetime="14:00">2:00 PM</time>
  </th>
  <td>
    <a href="speakers.html#david-tarvin">
      <h4>David Tarvin</h4>
      Crafty Coding: Generating Knitting Patterns
    </a>
  </td>
  <td>
    <a href="speakers.html#nate-decker">
      <h4>Nate Decker</h4>
      From Muppets to Mastery: Core UX Principles from Mr. Jim Henson
    </a>
  </td>
</tr>
```

Now that our tables are taking shape, it's time to add a little style to them. Let's begin by adding some general styles to the **<th>** and **<td>** elements. For both the **<th>** and **<td>** elements let's add a **bottom padding** of 22 pixels and a **vertical alignment** of top. For **<th>** elements specifically let's add a **right padding** of 45 pixels, a **text alignment** of right, and a **width** of 20%. Then, for **<td>** elements let's add a **width** of 40%.

Below our existing table and schedule styles, our code should look like this:

```
th,
td {
  padding-bottom: 22px;
  vertical-align: top;
}
th {
  padding-right: 45px;
  text-align: right;
  width: 20%;
}
td {
  width: 40%;
}
```

Next, let's style the table head and the elements within the table head. We'll set a **line-height** of 44 pixels on the **<thead>** element only, and a **color** of #648880 and a **font-size** of 24 pixels on all **<th>** elements nested within a **<thead>** element. Our new styles include the following:

```
thead {
  line-height: 44px;
}
thead th {
  color: #648880;
  font-size: 24px;
}
```

The table head is looking good, so let's also add some styles for the table body. We'll begin with **<th>** elements nested within the **<tbody>** element: changing their color, adding some **font-** and **text-**based styles, and adding some **top padding.**

```
tbody th {
    color: #a9b2b9;
    font-size: 14px;
    font-weight: 400;
    padding-top: 22px;
    text-transform: uppercase;
}
```

We'll also add some styles to **<td>** elements nested within the **<tbody>** element, beginning with a top border and padding. We'll style the **<td>** elements that span only one column by adding 15 pixels of right padding to those that form the left column and 15 pixels of left padding to those that form the right column. Doing so puts a total of 30 pixels of padding between the two columns while keeping each cell the same size. We don't need to apply any left or right padding to the **<td>** elements that span two columns.

We'll add all of these horizontal paddings using the **:first-of-type**, **:last-of-type**, and **:only-of-type** pseudo-class selectors. These pseudo-class selectors work very similarly to the **:last-child** pseudo-class selector we've used before.

The **:first-of-type** pseudo-class selector will select the first element of its type within a parent element. In our case, the **td:first-of-type** selector will select the first **<td>** element within a **<tr>** element. Then, the **:last-of-type** pseudo-class selector will select the last element of its type within a parent element.

Again, in our case, the **td:last-of-type** selector will select the last **<td>** element within a **<tr>** element. Lastly, the **:only-of-type** pseudo-class selector will select an element if it's the only element of its type within a parent element. Here, the **td:only-of-type** selector will only select a **<td>** element if it's the only **<td>** element within a **<tr>** element, such as when a **<td>** element spans two columns.

Our styles are a little complex, but they're flexible in addressing the needs of our table. These new styles include the following:

```
tbody td {
    border-top: 1px solid #dfe2e5;
    padding-top: 21px;
}

tbody td:first-of-type {
    padding-right: 15px;
}

tbody td:last-of-type {
    padding-left: 15px;
}

tbody td:only-of-type {
    padding-left: 0;
    padding-right: 0;
}
```

Next, let's adjust a few of the styles on existing elements to clean up the design. We'll start by making all of the links within the table a medium gray. If we target only the **<a>** elements within a table, our headings with the speaker's name within the links will remain green, while the talk titles will be gray, creating a nice contrast between the two.

While we're adjusting the styles of the entries for the talks, let's also remove the **bottom margin** on the **<h4>** elements within the table, allowing the speaker's name to sit closer to her or his title. We can implement these styles with the following code:

```
table a {
    color: #888;
}

table h4 {
    margin-bottom: 0;
}
```

Lastly, let's create some visual contrast among the different types of activities happening throughout the day. All of the talks look good with our latest changes. For all of the other activities, such as registration, lunch, and breaks (which are within the table body) as well as for the date (which is within the table header) let's use a subtle gray.

We'll do so by creating a new class, ***schedule-offset***, and assigning a **color** of ***#a9b2b9*** to it.

```
.schedule-offset {
    color: #a9b2b9;
}
```

Once the class is in place, let's add it to all of the **<td>** elements that span two columns and include either the day's date or a designated activity—registration, lunch, or a break. Looking back to our table for the first day, the workshops day, the HTML will look like this:

```
    <tr>
      <th scope="row">
        Workshops
      </th>
      <td class="schedule-offset" colspan="2">
        August 24th
      </td>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">
        <time datetime="08:30:00">8:30 AM</time>
      </th>
      <td class="schedule-offset" colspan="2">
        Registration
      </td>
    </tr>
    <tr>
      <th scope="row">
        <time datetime="09:00:00">9:00 AM</time>
      </th>
      <td>
        <a href="speakers.html#vanessa-kasun">
          <h4>Vanessa Kasun</h4>
       React from Scratch. Creating a React App from Ground up!
        </a>
      </td>
      <td>
        <a href="speakers.html#joshua-burden">
          <h4>Joshua Burden</h4>
          Lights! Camera! Interaction! Design Inspiration from Filmma
        </a>
      </td>
    </tr>
      <th scope="row">
        <time datetime="12:30:00">12:30 PM</time>
      </th>
      <td class="schedule-offset" colspan="2">
        Lunch
      </td>
```

Tables, which may appear simple on the surface, can be quite complex, and that is the case with our Styles Conference schedule. The good news is that our schedule is now complete, and it's looking great.

Take a look at the source code to make sure your HTML structure is correct!

[Schedule Page Source Code]