5.3.6 Convert Trash into a Droppable

Currently, the only way to delete a task is to delete all tasks at once. In Lesson 2, however, you added a `<div>` element with the attribute `id="trash"`. Now would be a good time to make use of this trash placeholder.

Refer to the jQuery UI website again. Specifically, look over the **Droppable widget documentation (https://api.jqueryui.com/droppable/)** and read up on some of the options and events that come with it. Documentation can be overwhelming at times, so get used to skimming! In particular, narrow in on the descriptions for `accept`, `tolerance`, and `drop`.

After you've spent some time with the documentation, return to the `script.js` file in Taskmaster and add the following block of code:

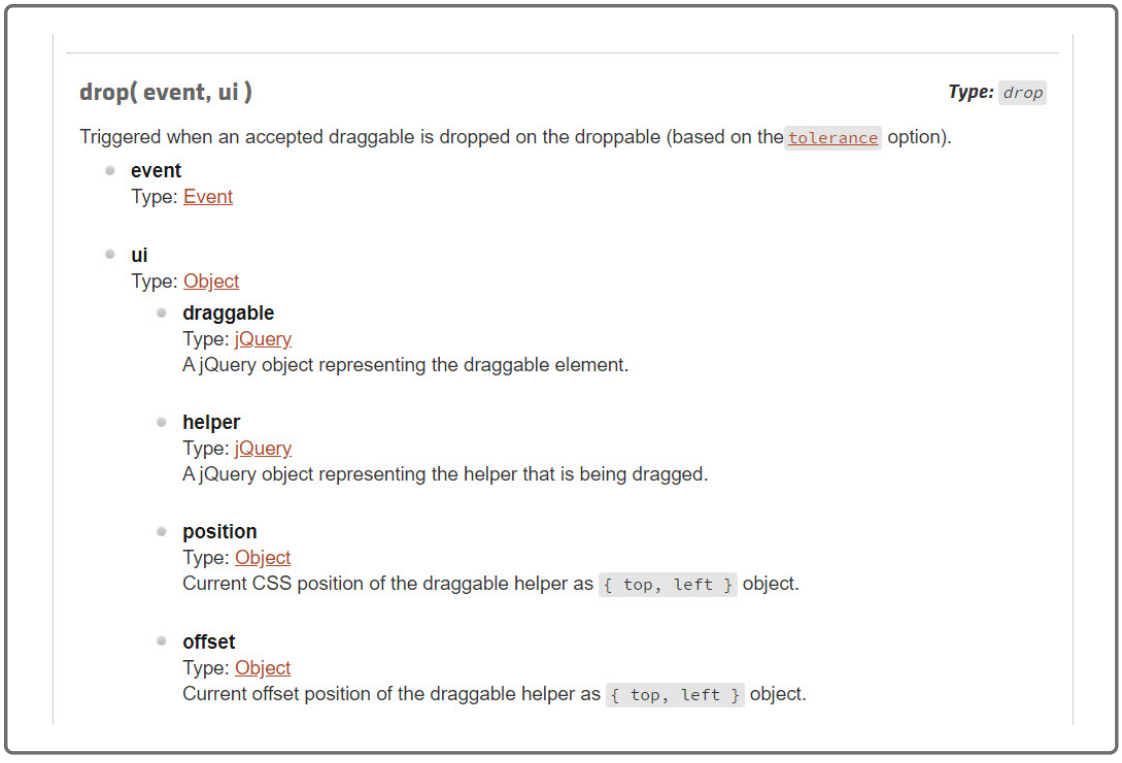
```
$("#trash").droppable({
  accept: ".card .list-group-item",
  tolerance: "touch",
  drop: function(event, ui) {
    console.log("drop");
  },
  over: function(event, ui) {
    console.log("over");
  },
  out: function(event, ui) {
```

```
    console.log("out");
  }
});
```

Save and test the app in the browser, watching the console to see when the various events occur. Like the Sortable widget, some of these events are only worth capturing for styling purposes. We'll save those for later. The `drop()` method is the most important, because that means a user is trying to delete a task.

How do you know which task the user wants to delete? You wouldn't be able to use `this` here: `this` would simply refer to the `#trash` element that the event belongs to. The `event.target` property wouldn't help, either, because `#trash` is also the target of the drop.

Look closely at the documentation for `drop`. You should see something like the image below:



drop(event, ui) *Type:* `drop`

Triggered when an accepted draggable is dropped on the droppable (based on the `tolerance` option).

- **event**
  Type: Event
- **ui**
  Type: Object
  - **draggable**
    Type: jQuery
    A jQuery object representing the draggable element.
  - **helper**
    Type: jQuery
    A jQuery object representing the helper that is being dragged.
  - **position**
    Type: Object
    Current CSS position of the draggable helper as `{ top, left }` object.
  - **offset**
    Type: Object
    Current offset position of the draggable helper as `{ top, left }` object.

jQuery UI has a second function parameter, which they've decided to call `ui`. This variable is an object that contains a property called `draggable`.

According to the documentation, `draggable` is "a jQuery object representing the draggable element." If that's true, then we should be able to call DOM methods on it!

In the `drop()` method, add the following line:

```
ui.draggable.remove();
```

jQuery's `remove()` method works just like a regular JavaScript `remove()` and will remove that element from the DOM entirely.

Save and test again. Dropping a task onto the trash dropzone should delete/remove it from the DOM. Now that the task has been removed, do we need to call `saveTasks()` again? Surprisingly, no! Removing a task from any of the lists triggers a sortable `update()`, meaning the sortable calls `saveTasks()` for us. Refresh the browser to verify that the task is, in fact, gone.

Great job—you've successfully braved unfamiliar documentation to implement some cool new features. Everything that jQuery UI provides can still be achieved with plain JavaScript, but this library sped up the development process significantly.

Before we consider this GitHub issue done, we'll make some final adjustments in the next step to accommodate mobile devices.