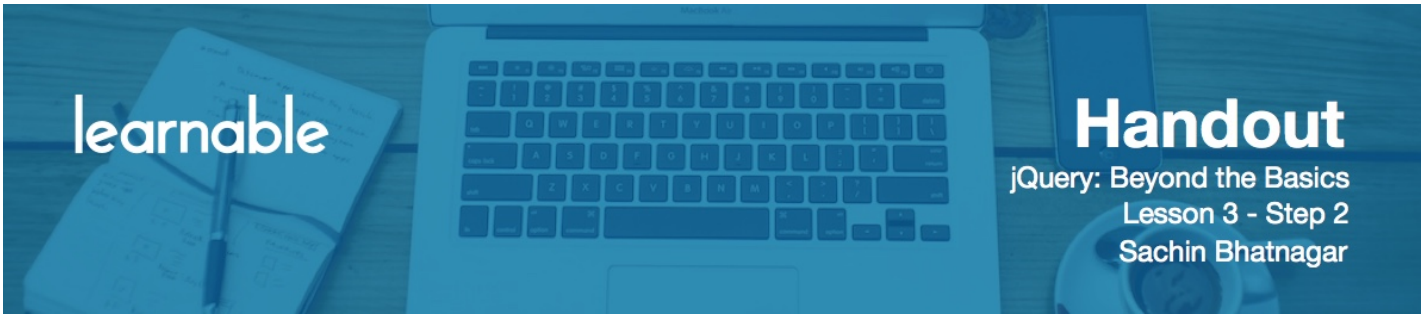

Introduction to loading partials using AJAX

It's time to put theory into practice. The first AJAX usage scenario that we'll examine involves loading HTML content from separate files onto a single page.

Imagine we have a website with a section that has to display content from four different HTML files. Individual files that are used in a scenario like this are known as **partials**. To load content from these files into a given section on the page, we can use jQuery's `load` method:

```
$(<selector>).load( "content.html", function(response, status, xhr) {} );
```

The `<selector>` here represents an element on the page into which the content needs to be placed once loaded using AJAX.

The first argument is the file which should be loaded. An optional callback function lets us access three properties:

- `response`, which also provides access to the content that has been loaded in
- `status`, which informs us whether the AJAX call was successful or not
- `xhr`, which provides us with access to the underlying `XMLHTTPRequest` object.

These arguments are typically used in advanced usage scenarios.

Loading HTML content with AJAX

Begin by opening the *loadingPartials_Begin.htm* file from the accompanying archive.

For all AJAX examples to work properly, make sure that the files are placed in the root of your web server, and served using HTTP. This should be done typically over localhost or the IP of the serving machine. You'll also see a folder called *partials* which contains four HTML files representing excerpts from four short stories.

So we want to load partials into the content area represented by a `div` tag when the user clicks on their links in the navigation section. We also have an animated *.gif* image which we'll use while the content is being loaded to indicate the loading process.

I've already created variables that represent the `contentPane` and the throbber image, and a click event listener for the hyperlinks. Note the use of `evt.preventDefault` and `evt.stopPropogation` to prevent the browser from navigating to the partial pages when the user clicks on these links.

```
var throbber = $("#throbber");
var contentPane = $(".contentPane");
$(".navlinks").on("click", function(evt) {
  evt.preventDefault();
  evt.stopPropagation();
  contentPane
      .fadeOut(200)
      .queue(function() {
        throbber.fadeIn(100);
        $(this).dequeue();
      })
      .load($(this).attr("href"), function(response, status, xhr) {
        if(status === "success") {
          throbber.fadeOut(100);
          $(this).fadeIn(1000);
        } else {
```

```
          alert(status);
        }
      });
});
```

We utilize jQuery's `load` method and pass the URL of the partial file to load.

The `contentPane` fades out before loading the content and the throbber image is displayed inside the `queue` function.

Inside the callback function for the `load` method we check the response status and either hide the throbber and display the `contentPane` again or display an error.

Go on and check the result in the browser!