thinkvitamin.com

Handlebars.js Part 2

Built-In Helpers

Handlebars includes a few built in helpers that make life easier. They are `{{#each}}`, `{{#if}}`, and {{#unless}}.

The *each* Helper

The `{{#each}}` helper iterates over each item in an array. Here's an example.

```
<script id="each-template" type="text/x-handlebars-template">
  {{#each people}}
    ... output person's info here...
  {{/each}}
</script>
```

The above template would iterate over each item in the array named people and output the content of the block.

The *if* Helper

The `{{#if}}` helper does just what you'd expect. It allows you to implement an if block in your code. The `if` helper outputs the block that it contains if the value given to it is truthy.

One tricky aspect of the helper, though, is that Handlebars doesn't support conditional statements, so code like `{{#if x > y}}` isn't possible. That's on purpose. Our take is that any tricky logic like that can be wrapped up into a helper to make sure that your template stays nice and clean.

Here's an example:

```
<script id="each-template" type="text/x-handlebars-template">
  {{#if people}}
    ... output person's info here...
  {{/if}}
</script>
```

That template would only output the inside of the block if people was truthy, so it wouldn't output if people was `null`, `0`, `false`, or `undefined`. Probably a more appropriate if statement in the above example would be `{{#if people.length}}`, so that the block would also not be displayed if a people array is present, but empty.

The *unless* Helper

The `{{#unless}}` helper is basically just the opposite of if. It only outputs the contained block if the given expression is false. So, for example:

```
<script id="each-template" type="text/x-handlebars-template">
  {{#unless people.length}}
    There aren't any people.
  {{/unless}}
</script>
```

The above template would only output the sentence *There aren't any people* if `people.length` evaluates to a falsy value like `null`, `0`, `false`, or `undefined`.

The *else* Expression

Handlebars.js includes a special expression, `{{else}}`, that can be used with any block helper to represent what should be output if the given expression evaluates to a falsy value. Here's an example of how to use it:

```
<script id="each-template" type="text/x-handlebars-template">
  {{#if people.length}}
    ... output person's info here...
  {{else}}
    There aren't any people.
  {{/if}}
</script>
```

Partials

Partials come in handy when you have a chunk of a Handlebars.js template that you need to use in a few different contexts. The `Handlebars.registerPartial` method registers a partial. It takes the name of the partial as its first argument and either a template source string or a compiled template as its second argument. The fact that it accepts a compiled template as the second argument is actually pretty useful. That allows you, for example, to use the partial in a loop that outputs a list but also append items to the list later using the partial's template function.

To use a partial from a template, simply include `{{> partialName}}`. Here's an example of using a partial:

```
<script id="people-template" type="text/x-handlebars-template">
  {{#each people}}
    {{> person}}
  {{/each}}
</script>

<script id="person-partial" type="text/x-handlebars-template">
  <div class="person">
    <h2>{{first_name}} {{last_name}}</h2>
    <div class="phone">{{phone}}</div>
    <div class="email"><a href="mailto:{{email}}">{{email}}</a></di
    <div class="since">User since {{member_since}}</div>
  </div>
</script>

<script type="text/javascript">
  $(document).ready(function() {
    var template = Handlebars.compile($("#people-template").html())
    Handlebars.registerPartial("person", $("#person-partial").html(

    template(yourData);
  }
</script>
```

Writing Customer Helpers

One of our major motivations in writing Handlebars.js rather than just using mustache.js was to allow users to define global helpers. Handlebars supports defining both expression and block helpers.

Custom Expression Helpers

To register an expression helper, use the `Handlebars.registerHelper` method. It takes the name of the helper and the helper function as arguments. Handlebars.js takes whatever is returned from the helper function and writes it out to the template, so be sure to always return a string from your custom helpers.

To write an expression helper function to output a formatted phone number, you could define the following helper:

```
Handlebars.registerHelper("formatPhoneNumber", function(phoneNumber)
  phoneNumber = phoneNumber.toString();
  return "(" + phoneNumber.substr(0,3) + ") " + phoneNumber.substr(3
});
```

You would use the `formatPhoneNumber` helper in a template like this:

```
{{formatPhoneNumber phoneNumber}}
```

Custom Block Helpers

Custom block helpers are also registered with the `Handlebars.registerHelper` method. When a helper is used with a block, Handlebars will pass the contents of the block compiled into a function to the helper. If an `{{else}}` expression is found in the block Handlebars will also pass the contents of the `else` block to the helper as well.

Here's an example block helper that iterates through an array, letting the contents know whether it's an even or odd row. The helper takes the array to iterate over, the css class name for even rows, and the css class name for odd rows as arguments. You'll also notice the compiled template function `fn` for the contents of the block and the compiled else block function, `elseFn` are arguments to the helper function. The helper simply adds a property named `stripeClass` to each item in the array as we iterate over it so that we can output that class name within the block. If the array given is falsy or empty the helper just returns the contents of the else block.

```
Handlebars.registerHelper("stripes", function(array, even, odd, fn,
  if (array && array.length > 0) {
    var buffer = "";
    for (var i = 0, j = array.length; i < j; i++) {
      var item = array[i];

      // we'll just put the appropriate stripe class name onto the
      item.stripeClass = (i % 2 == 0 ? even : odd);

      // show the inside of the block
      buffer += fn(item);
    }

    // return the finished buffer
    return buffer;
  }
  else {
    return elseFn();
  }
});
```

You would use the stripes helper in your template like this:

```
{{#stripes myArray "even" "odd"}}
  <div class="{{stripeClass}}">
    ... code for the row ...
  </div>
{{else}}
  <em>There aren't any people.</em>
{{/stripes}}
```

See It In Action

I actually wrote up a quick sample project that uses all of the techniques I've describe here. You can check that code out on GitHub or download a zip file of the source.

There's More!

There's a lot going on with Handlebars.js helpers, so we've got at least one more article worth of content to cover. Next week I'll show you how to do some neat tricks with the internals of how Handlebars.js blocks work. Please feel free to email me at alan@carsonified.com if you have any questions and I can cover those as well.

Follow @thinkvitamin on Twitter Please check out Think Vitamin Membership

Other Posts You Might Find Interesting

- Sorry - No Related Posts Found