

Learn

# Handlebars.js Part 2: Partials and Helpers



Treehouse

*writes on April 1, 2011*



Last week we looked at the basics of [using Handlebars.js](#). We learned how to write a basic template to loop over a list of people. This week we'll build on that by learning about using Handlebars' built-in helpers, writing and using partials, and writing custom helpers. Let's get started!

# 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.

```
1 <script id="each-template" type="text/x-handlebars-template">
2   {{#each people}}
3     ... output person's info here...
4   {{/each}}
5 </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:

```
1 <script id="each-template" type="text/x-handlebars-template">
2   {{#if people}}
3     ... output person's info here...
4   {{/if}}
5 </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:

```
1 <script id="each-template" type="text/x-handlebars-template">
2   {{#unless people.length}}
3     There aren't any people.
4   {{/unless}}
5 </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:

```
1 <script id="each-template" type="text/x-handlebars-template">
2   {{#if people.length}}
```

```
3     ... output person's info here...
4     {{else}}
5     There aren't any people.
6     {{/if}}
7 </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:

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

```
6
7 <script id="person-partial" type="text/x-handlebars-template">
8   <div class="person">
9     <h2>{{first_name}} {{last_name}}</h2>
10    <div class="phone">{{phone}}</div>
11    <div class="email"><a href="mailto:{{email}}">{{email}}</a></div>
12    <div class="since">User since {{member_since}}</div>
13  </div>
14 </script>
15
16 <script type="text/javascript">
17   $(document).ready(function() {
18     var template = Handlebars.compile($("#people-template").html());
19     Handlebars.registerPartial("person", $("#person-partial").html());
20
21     template(yourData);
22   }
23 </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:

```
1 Handlebars.registerHelper("formatPhoneNumber", function(phoneNumber)
2   phoneNumber = phoneNumber.toString();
3   return "(" + phoneNumber.substr(0,3) + ") " + phoneNumber.substr(3,
4   });
```

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

```
1 {{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.

```
1 Handlebars.registerHelper("stripes", function(array, even, odd, fn, e
2   if (array && array.length > 0) {
3     var buffer = "";
4     for (var i = 0, j = array.length; i < j; i++) {
5       var item = array[i];
6
7       // we'll just put the appropriate stripe class name onto the it
8       item.stripeClass = (i % 2 == 0 ? even : odd);
9
```

```

10      // show the inside of the block
11      buffer += fn(item);
12  }
13
14      // return the finished buffer
15      return buffer;
16  }
17  else {
18      return elseFn();
19  }
20  });

```

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

```

1  {{#stripes myArray "even" "odd"}}
2    <div class="{{stripeClass}}">
3      ... code for the row ...
4    </div>
5  {{else}}
6    <em>There aren't any people.</em>
7  {{/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](mailto:alan@carsonified.com) if you have any questions and I can cover those as well.

code

html and xhtml

javascript

---

## 15 Responses to “Handlebars.js Part 2: Partial and Helpers”

emi on [December 4, 2015 at 1:48 am](#) said:

Nice Artical

### Reply



 on November 16, 2015 at 9:23 am said:



2015 10% !

 <http://www.ooobrand.com/bags/bv/index.html>

## Reply

 on November 16, 2015 at 9:23 am said:

9

<http://www.ooowatch.com/tokei/vuitton/index.html>

Reply

 on November 14, 2015 at 12:39 pm said:

IWC 2015 10% !  
 100% (C) !  
 :

[illegible]

**Reply**

3047,96,

on November 14, 2015 at 12:38 pm said:

[url=http://www.brandiwc.com/brand-47-copy-0.html]

30   47,                  

Reply

Here is big library of handlebars helpers <http://assemble.io/docs/helpers/index.html> so you can see examples of how they work. The library was created for <http://assemble.io> but the helpers can be used in any project.

Reply

Awesome.. This article solves my problem in register helper..  
Thanks..

Reply

Thanks the zip examples were very helpful.

I notice that the registerHelper in Handlebars 2011 now takes “options”

instead of “fn”. You now use fn as options.fn.

```
Handlebars.registerHelper("stripes", function(array, even, odd, fn){
```

```
...
```

```
buffer += options.fn;
```

```
...
```

```
}
```

Reply

---

**Horia Dragomir** on **April 7, 2011 at 10:51 pm** said:

This seems to have a more robust backbone than the unofficial official jquery template system (\$.tmpl) but I'll wait to check out the documentation first. 😊

Reply

---

**Marco Sempellegri** on **April 3, 2011 at 11:38 am** said:

Awesome, thanks!

Reply

---

**Vic Rau** on **April 2, 2011 at 8:25 am** said:

Great article, but just 2 questions.

1. When would you use handlebars.js? Website? Widgets? Webapps? ...
2. What about SEO?

regards

Vic

Reply

**Anonymous** on [April 4, 2011 at 1:54 am](#) said:

Hey Vic – great questions!

1. I've primarily used handlebars.js on really dynamic apps – think GMail type apps. If you're already doing everything with XHR it's a great fit because it's way easier to write Handlebars templates than to generate html or dom elements yourself.

2. You'll want to really think through the consequences of using handlebars.js. Content that you're generating on the client side isn't going to be indexed, but a lot of times that's not all that important anyway. Would you use it for a public page on a marketing site? Probably not. Would you use it for a part of your app that's already password protected? Go for it!

Reply

**Joey Lomanto** on [April 6, 2011 at 4:09 pm](#) said:

Really great question and answer about the



implications with indexing.

Alan, this may be a ridiculous question, the JavaScript code blocks where you define the type as text/x-handlebars-template. Do these have to reside embedded in the actual page or can they be separate JS files?

Reply

**Elving Rodriguez** on **April 7, 2011 at 2:39 pm** said:

I was wondering how to do this too. How can we move this embedded templates into separate files and just include them as scripts so we can have unobtrusive templates?

Reply

**Scott Carroll** on **May 23, 2013 at 3:20 pm** said:

You can load the template asynchronously.

```
function  
getTemplateAjax(templateName,
```

```
$target, context) {  
  var source, template;  
  
  jQuery.ajax({  
    url:  
    '/path/to/template/'+templateName+'.mustache',  
    cache: true,  
    success: function (data) {  
      source = data;  
      template =  
      Handlebars.compile(source);  
      $target.html(template(context));  
    },  
    error: function () {  
      console.log("Aw, snap."); }  
  });  
}  
  
// simple use case.  
getTemplateAjax("TemplateName",  
jQuery("body"), data);
```

---

## Leave a Reply

Name \*

Email Address (will not be published) \*

Website

**Submit Comment**



# Want to learn more about Javascript?

Learn how to use JavaScript to  
add interactivity to websites.

[Learn more](#)

# Stay current

Sign up for our newsletter, and we'll send you news and tutorials on web design, coding, business, and more!

**Subscribe**



©2016 Treehouse Island, Inc.

[About](#) • [Careers](#) • [Blog](#) • [Affiliate Program](#) • [Terms](#) • [Privacy](#) • [Press Kit](#) • [Contact](#)

