

Plumbing the Depths of Handlebars



Ryan Lewis

ryanhlewis.com



[ryanmurakami](#)





PLURALSIGHT

JavaScript Templating with Handlebars

By Ryan Lewis

Handlebars is the most widely used JavaScript templating library, supporting both client-side and server-side applications. In this course, Handlebars is covered from basic concepts all the way to production-ready tactics.

Start free trial now

Course info

Level Beginner

Rating ★★★★★ (60)

Duration 2h 19m

Released 29 Sep 2015

Course authors



Ryan Lewis

Ryan Lewis is a Software Engineer who specializes in ambitious single page web applications. He teaches Java and JavaScript to aspiring web developers and technology professionals. In his free time, Ryan enjoys spending time with his family, playing video games, and releasing underground Japanese music on his record label, MeatCube.

Start free trial now

Share course



Table of contents

Description

Exercise files

Transcript

Discussion

Knowledge check

Expand all

▶	Handlebars: An Introduction	21m 50s	▼
▶	Building Blocks of Handlebars	46m 25s	▼
▶	Harnessing the Power of Helpers	37m 12s	▼
▶	Beyond the Basics	33m 51s	▼

1. Where are Helpers?

```
Handlebars.registerHelper('isCorrect',  
  function() { return false; });
```

```
Handlebars.helpers.isCorrect = function() {  
  return false;  
}
```

```
Handlebars.registerPartial('awesome-templ',  
  '{{#if isCorrect}} Awesome {{/if}}');
```

```
Handlebars.partials['awesome-templ'] =  
  '{{#if isCorrect}} Awesome {{/if}}';
```

//some template is rendered with partial

```
Handlebars.partials['awesome-templ'] =  
  function() { //render };
```

2. Helpers first


```
Handlebars.registerHelper('isCorrect',  
  function() { return false; });
```

```
{ name: 'Ness', city: 'Onett', isCorrect: true }
```

```
{{#if isCorrect}}  
  I'm right!  
{{/if}}
```



You can prefix properties
with `this` to avoid
collisions

are you kidding me!?

Postfix all your helpers
with “helper”

3. Conditional block syntax

```
{{#if isCorrect}}  
Dogs are the best!  
{{/if}}
```

that's right!

```
{{#isCorrect}}  
Dogs are the best!  
{{/isCorrect}}
```



4. Handlebars implementations

Handlebars.Java

github.com/jknack/handlebars.java

Handlebars.Net

github.com/rexm/Handlebars.Net

pybars

github.com/wbond/pybars3

handlebars.php

github.com/XaminProject/handlebars.php

5. node  Handlebars

hapi.js

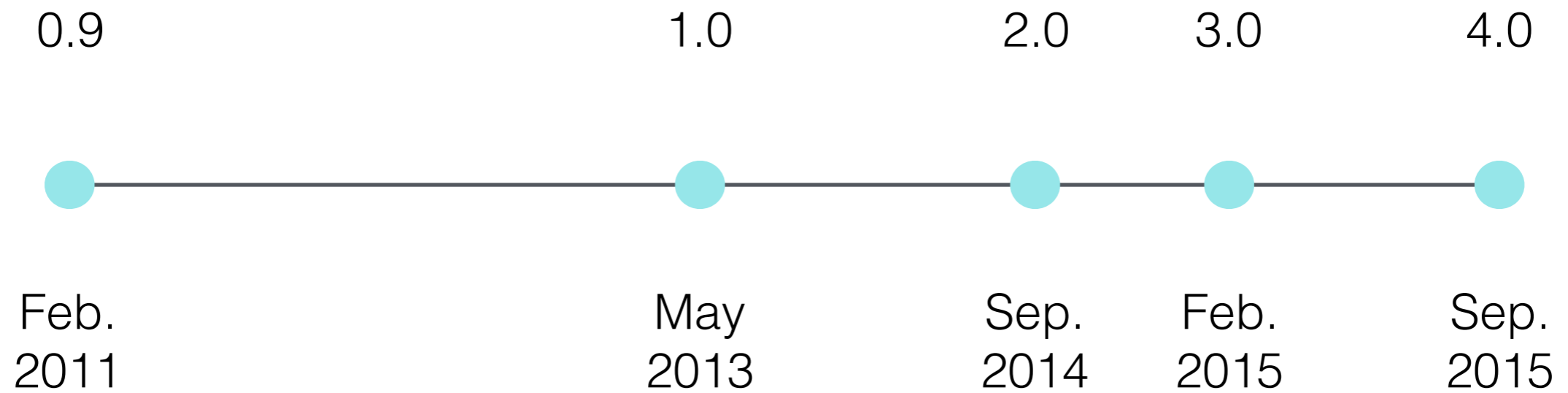
```
server.views({
  engines: { html: require('handlebars') },
  path: __dirname + '/templates',
  partialsPath: __dirname + '/templates/partials'
});
```

express.js

```
var app = express();
app.engine('handlebars',
  require('express-handlebars')({defaultLayout:
  'main'}));
app.set('view engine', 'handlebars');
```

6. Handlebars is alive!

Handlebars Release Timeline



Version 5.0 coming Feb. 2016?

7. *this* in helpers

```
Handlebars.registerHelper('helpHelper',  
function() { return this.needsHelp; });
```

```
{ needsHelp: 'shore do!' }
```



At your service!

```
{{helpHelper}}
```

```
shore do!
```

8. `this` is mutable in helpers

```
Handlebars.registerHelper('helpHelper',  
  function() {  
    this.needsHelp = 'corse not!';  
  });
```

```
{ needsHelp: 'shore do!' }
```

```
{{helpHelper}} {{needsHelp}}
```

```
corse not!
```

9. Executing function properties


```
{  
  name: 'Paula',  
  getIt: function() {  
    return 'got it!';  
  }  
}
```

```
{{getIt}}
```

```
got it!
```

10. Re-scoping partials

```
Handlebars.partial('myPartial, '{{dog}}');
```

```
{{> myPartial dog="panda"}}
```

panda



11. *@data* variables

`@root`

root context

`@first`

true if first iteration

`@index`

zero-based iteration index

`@key`

key name

12. Inline partials in 4.0

```
{{#*inline 'fullName'}}
    {{firstName}} {{lastName}}
{{/inline}}
```

```
<h2>Hello {{> fullName}},</h2>
<h3>Friends:</h3>
<ul>
    {{#each friends}}
        <li>{{> fullName}}</li>
    {{/each}}
</ul>
```

13. Decorate your templates

Register...

```
Handlebars.registerDecorator('debug',  
  function(program, props, container, context) {  
    console.log('Debug: ' + JSON.stringify(context.data));  
  });
```

...then use

```
{{* debug}}  
<h1>{{dog.name}}</h1>  
<p>{{dog.desc}}</p>
```

Output

```
Debug: {"root":{"pet":{"name":"Garfunkel","desc":"Best  
dog","isDog":true}}}
```

14. Control the whitespace

```
{{#blocks~}}  
{{~/blocks}}
```

```
{{~properties~}}
```

```
{{~helpers with=arguments~}}
```

```
{{~*decorators~}}
```

```
{{~!-- comments ---~}}
```



15. Pre-compilation



grunt-contrib-handlebars

```
handlebars: {
  compile: {
    options: {
      namespace: "JST"
    },
    files: {
      "result.js": "source.hbs"
    }
  }
}
```



gulp-handlebars

```
module.exports = function() {
  var partials = gulp.src(['./templates/_*.hbs'])
    .pipe(handlebars())
    .pipe(wrap('Handlebars.registerPartial(<%=
processPartialName(file.relative) %>,
Handlebars.template(<%= contents %>));', {}, {
  imports: {
    processPartialName: function(fileName) {
      return JSON.stringify(path.basename(fileName,
'.js').substr(1));
    }
  }
})));

var templates = gulp.src('./templates/[^_]*.hbs')
  .pipe(handlebars())
  .pipe(wrap('Handlebars.template(<%= contents %>'))
  .pipe(declare({
    namespace: 'App.templates',
    noRedeclare: true
  })));

return merge(partials, templates)
  .pipe(concat('templates.js'))
  .pipe(gulp.dest('./build/js/'));
};
```

Questions?



I've got stickers!



Thank you!

Ryan Lewis



ryanhlewis.com



[ryanmurakami](#)