



TRP.

407

try ember

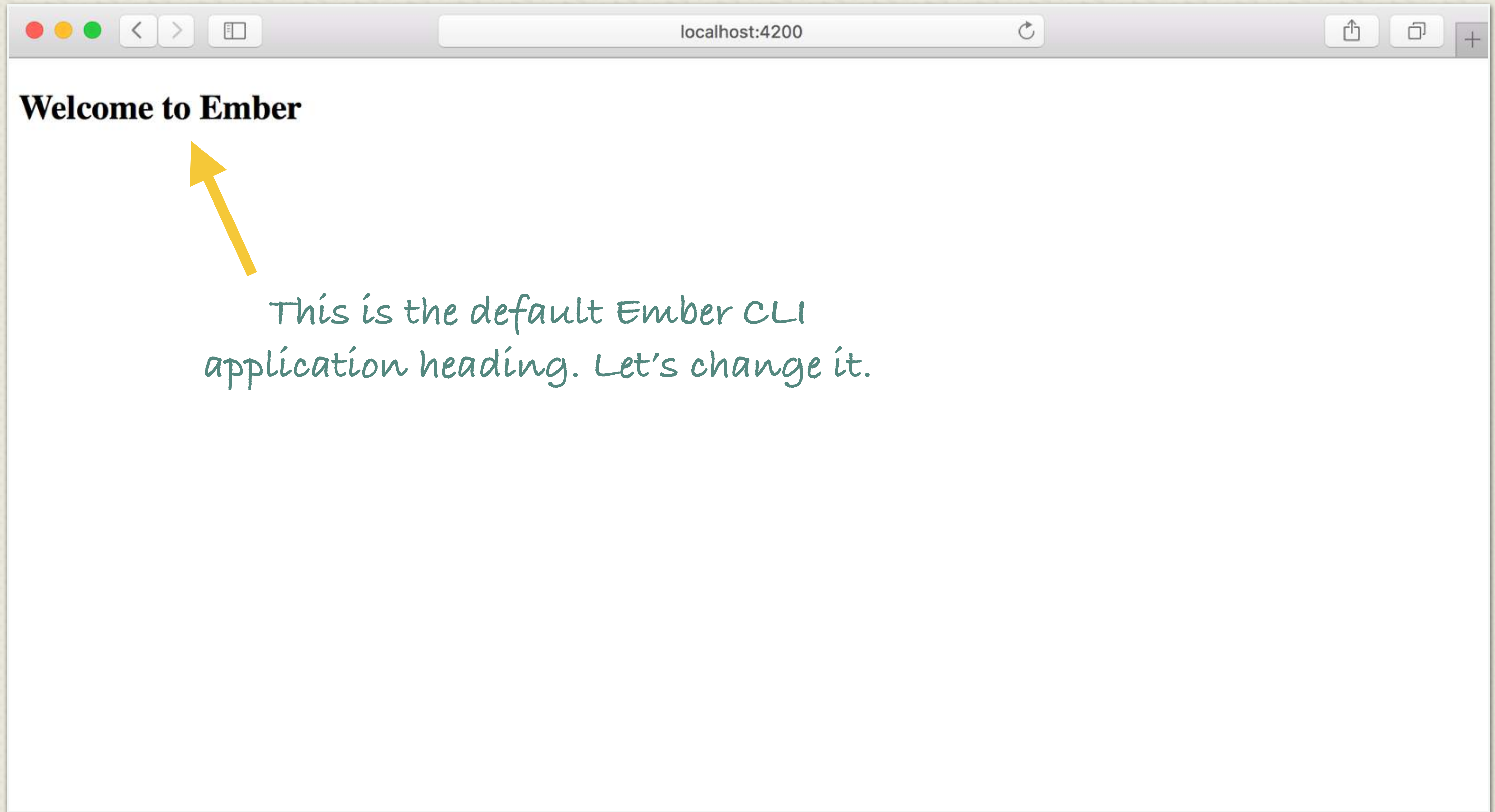
Level 2.1

Routing and Templating

.....
Templates and the Router



Getting Started

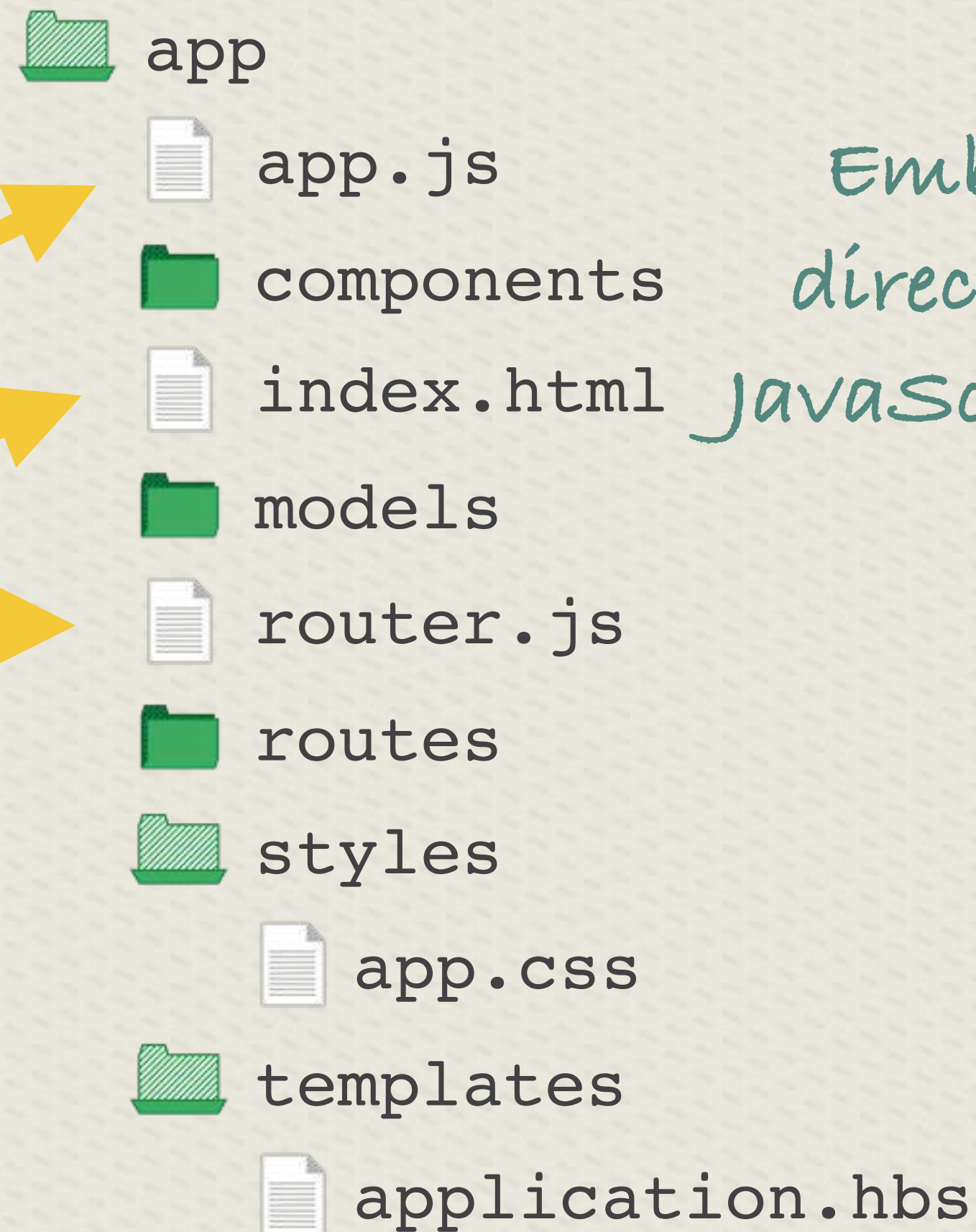


Working in App

Most of your work will be done within the “ember new”-generated app directory.

Console

```
$ ember new ...  
installing app  
create app/app.js  
create app/index.html  
create app/router.js  
...
```



Ember CLI generated several directories with many HTML, JavaScript, and CSS files for us.

This listing is truncated for brevity.

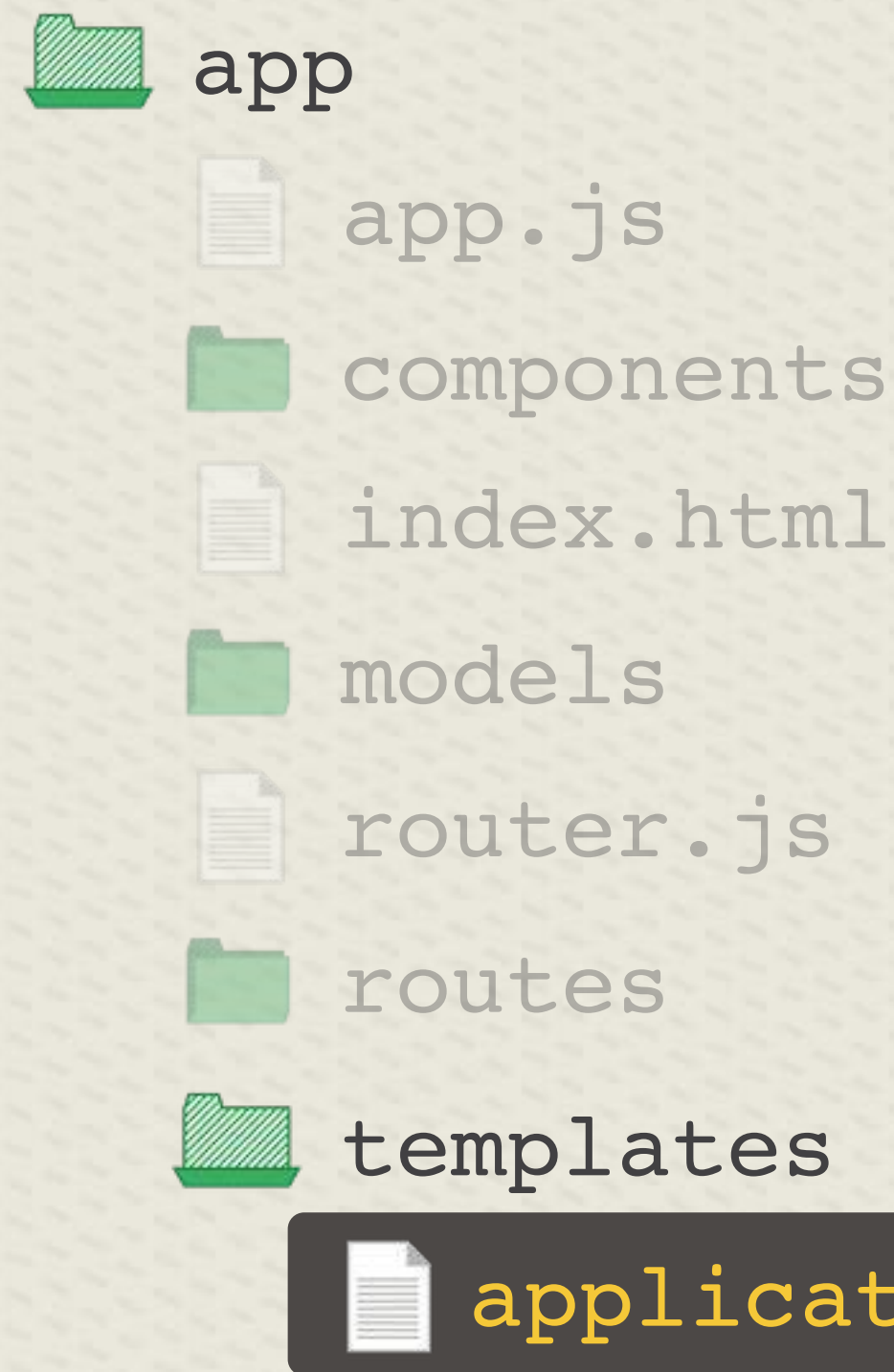


Introducing Templates



Templates tell Ember what HTML to generate and display in the web browser.

This is the “application” template, named based on the file name.



app/templates/application.hbs

```
<h2 id="title">Welcome to Ember</h2>

{{outlet}}
```



Ember uses the Handlebars templating library.

By convention, all Ember applications use an application template.



Changing the Heading



app/templates/application.hbs

```
<h2 id="title">Welcome to Ember</h2>

{{outlet}}
```

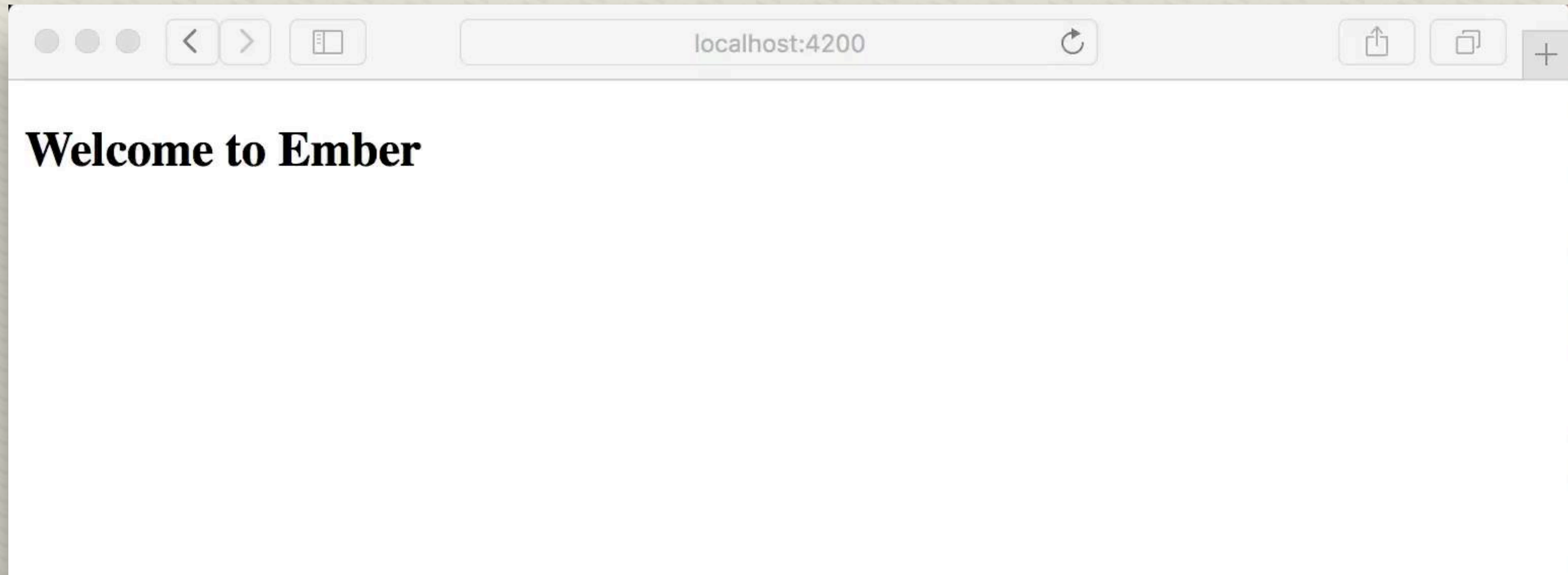


Changing the Heading

```
app/templates/application.hbs
```

```
<h2 id="title">Woodland Wanderer Whatchamacallits</h2>  
{{outlet}}
```

The browser automatically reloads whenever application files change using live reload.



Introducing Handlebars Expressions

Handlebars expressions mark locations of logic or dynamic content.

```
app/templates/application.hbs
```

```
<h2 id="title">...</h2>
```

```
{{outlet}}
```



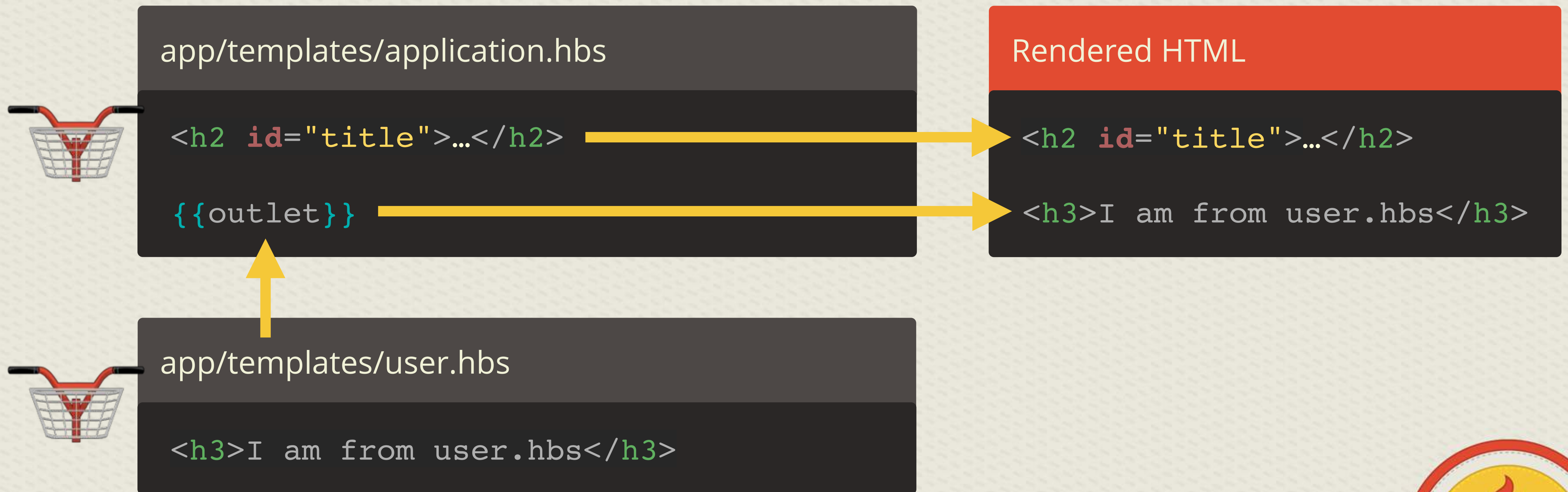
This is an “outlet” expression. It acts as a placeholder and tells Ember where to place other templates on the page.

Handlebars expressions are wrapped with curly braces (or brackets) and sometimes called “mustaches.”



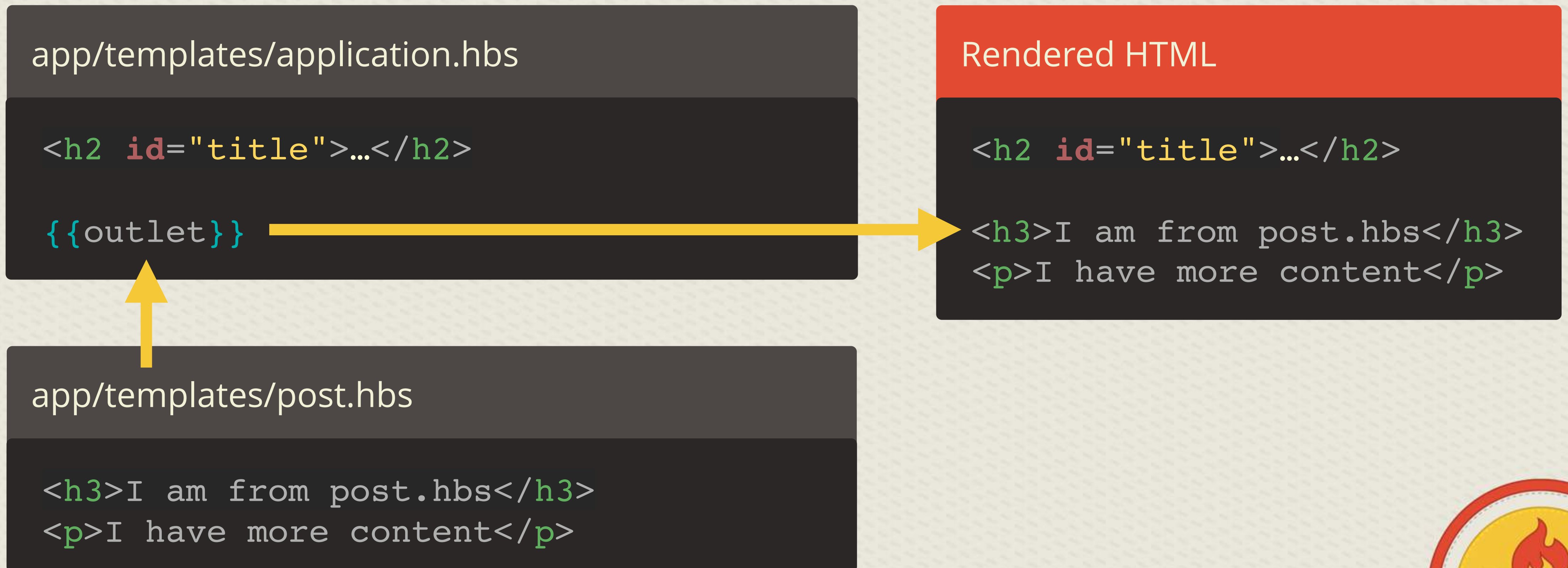
Layering Templates with {{outlet}}

The application template is the outermost layer into which other templates get rendered.



Changing {{outlet}} Content

When the template content changes, the outlet content is updated as well.

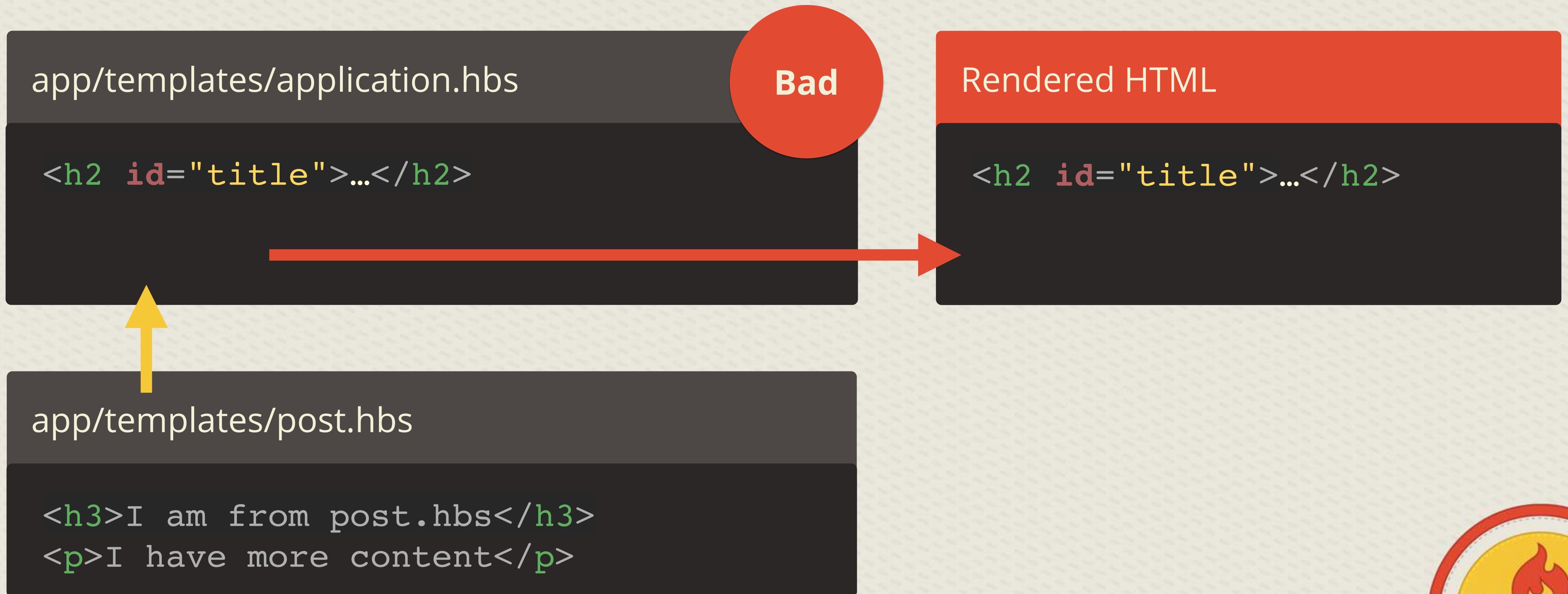


The application template is always displayed and often where headers, footers, and common navigation live.



Forgetting an {{outlet}}

Without an outlet in the application template, there is nowhere for other templates to go.



Auto-generating Templates

Ember creates templates in memory automatically whenever you do not provide one.

app/templates/application.hbs

```
<h2 id="title">Woodland Wanderer  
Whatchamacallits</h2>  
{{outlet}}
```

Rendered HTML

```
<h2 id="title">Woodland Wanderer  
Whatchamacallits</h2>
```

Auto-generated "index" Template

ember

Auto-generated Templates are empty.

Ember generates sane defaults across the application automatically.



Customizing the Index Template

The empty auto-generated index template can be overridden by using an index.hbs file.



```
app/templates/index.hbs
```

```
Hello from Index
```

An “index” template uses an
“index.hbs” file name.

Woodland Wanderer Whatchamacallits

Hello from Index

Adding New Templates

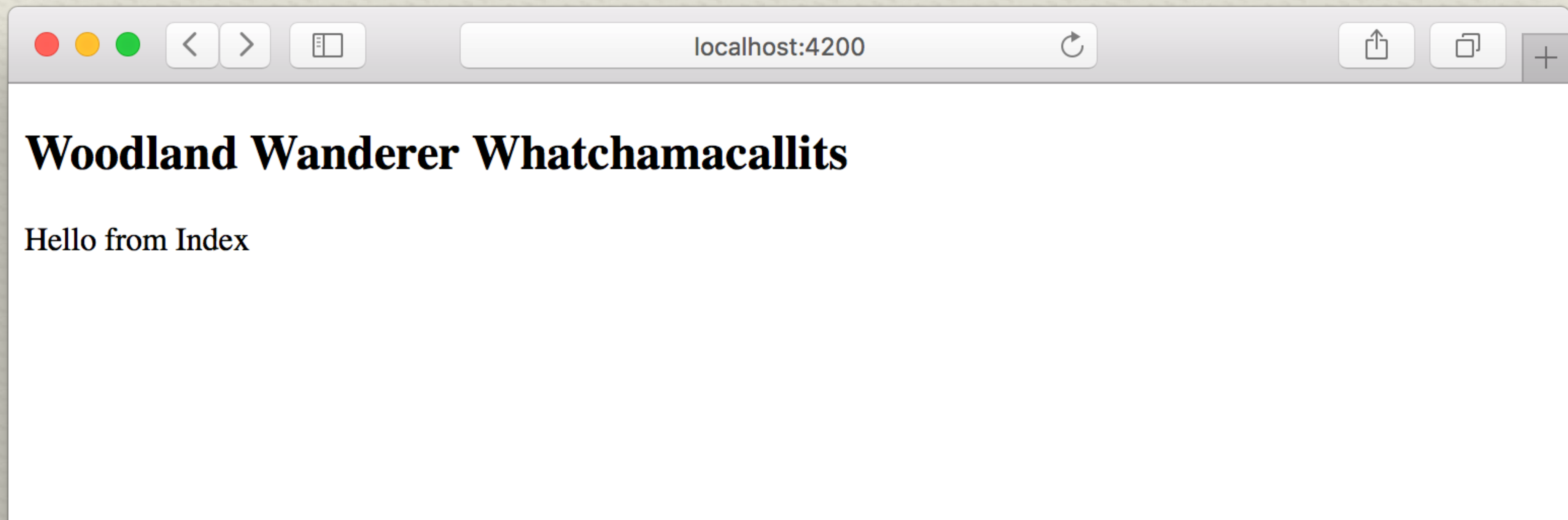
Templates may be added to the application, but Ember won't know about them unless we tell it.



```
app/templates/orders.hbs
```

```
Hello from Orders
```

*The index template worked because it's an Ember convention to be there.
The new orders template is not.*



Introducing the Router



The router manages your application state and maps it to the path of the URL.

States in This Application

State	URL
View the menu	/
List and create orders	/orders
View a receipt for Order #123	/orders/123



Customizing the Router

This is the default Ember CLI-generated router contents.



app/router.js

```
import Ember from 'ember';
import config from './config/environment';

const Router = Ember.Router.extend({
  location: config.locationType
});

Router.map(function() {
});

export default Router;
```

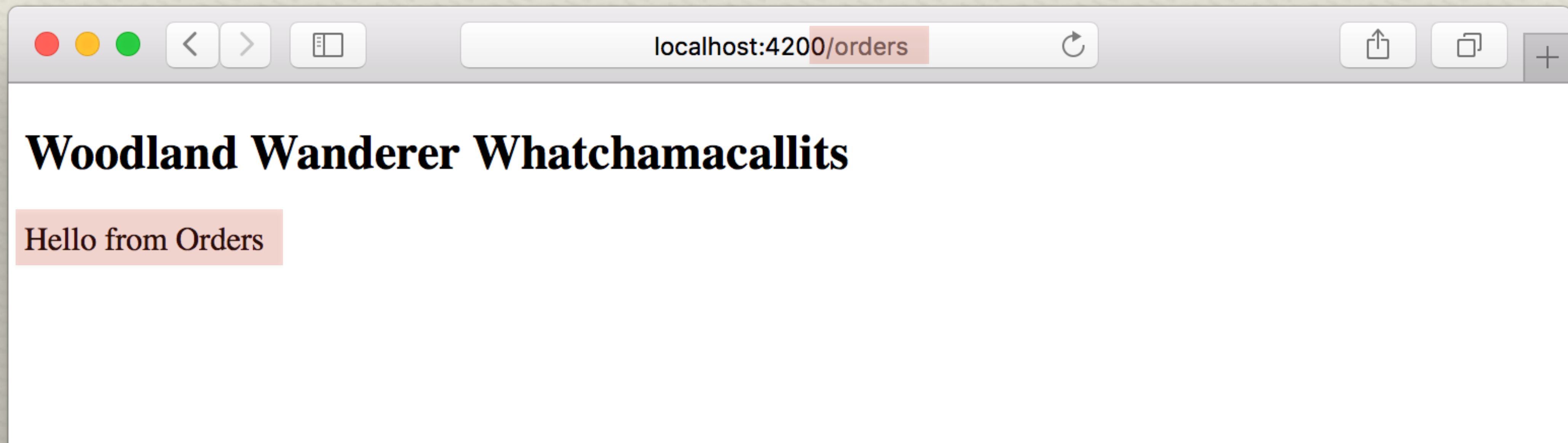
Application endpoints are mapped in here.

Adding a New Mapping to the Router

An orders mapping needs to be added to let Ember know about the new endpoint.

```
Router.map(function() {  
  this.route('orders', { path: '/orders' });  
});
```


"I should see the orders template when I navigate to /orders."



Revealing the Index

If the index were explicitly defined in router.js, it would look like this:

```
Router.map(function() {  
  this.route('orders', { path: '/orders' });  
  this.route('index', { path: '/' });  
});
```



"I should see the index template when I navigate to /."

*Ember automatically maps the index for you, so
it's usually omitted from the router file.*

Cleaning Up With the Defaults

If the path matches the name, we can omit the path.

```
Router.map(function() {  
  this.route('orders', { path: '/orders' });  
});
```



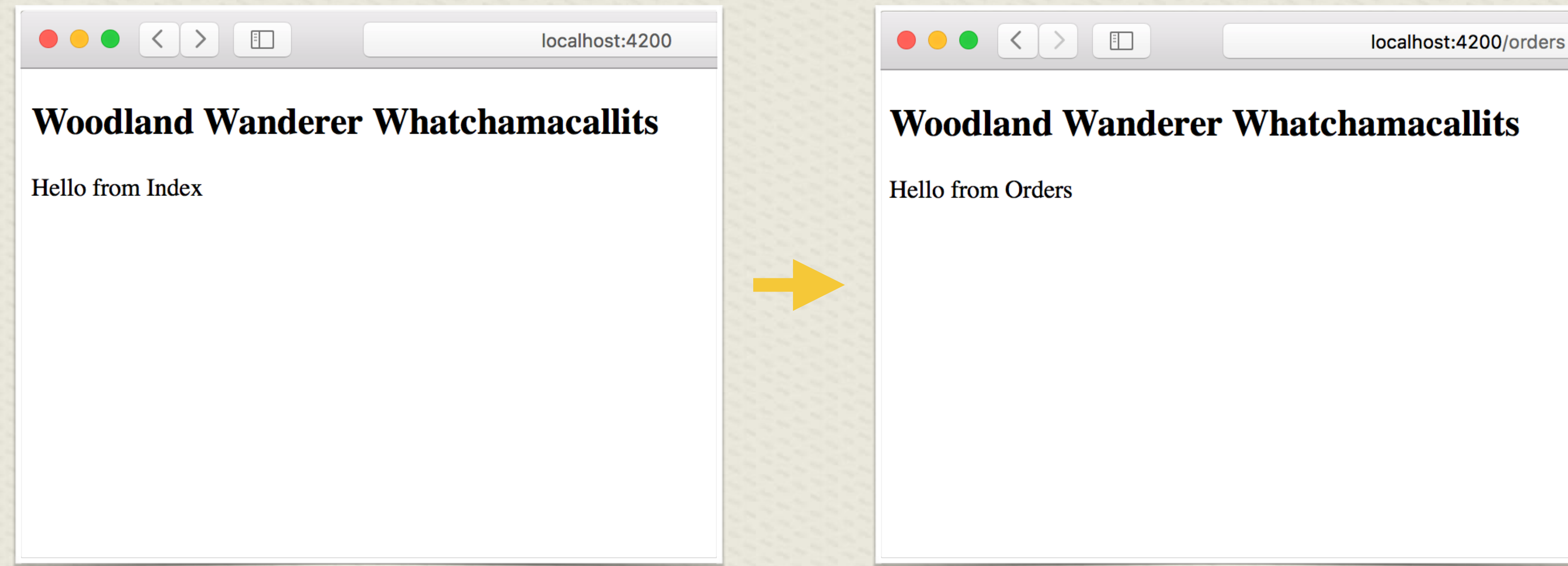
These are equivalent routers.

```
Router.map(function() {  
  this.route('orders');  
});
```


Navigating Between Endpoints

Users won't often navigate by manipulating the browser location bar.

How do we link from index to orders?



Using HTML to Navigate

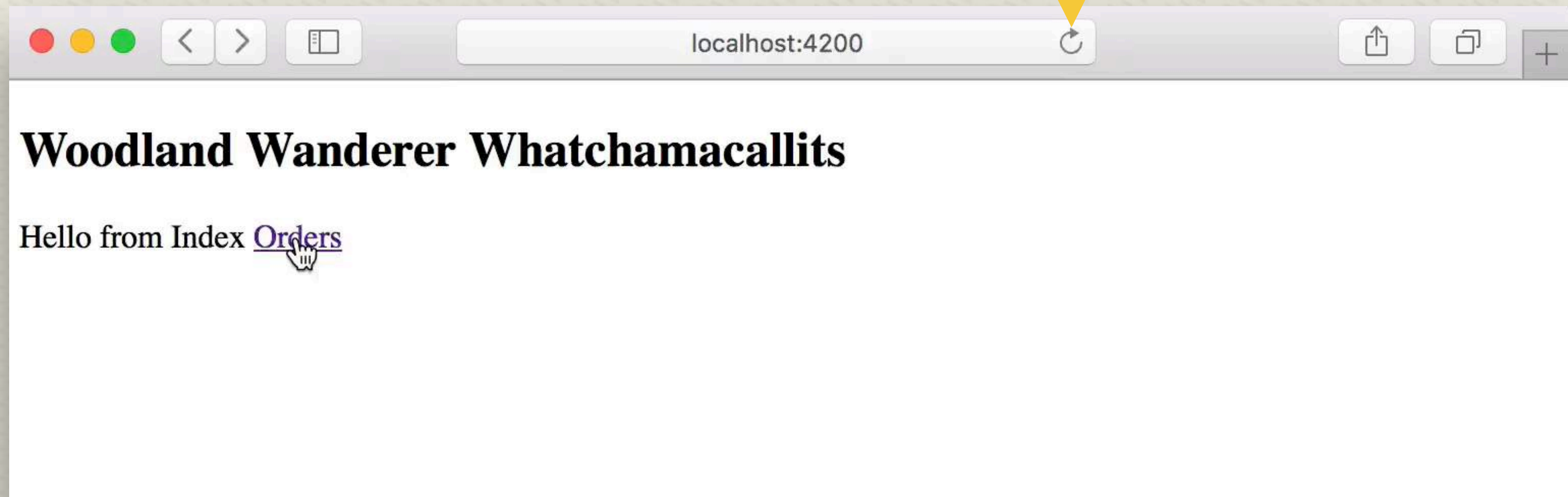
While you *can* use any HTML that you like, hard-coding links to endpoints should be avoided.

```
app/templates/index.hbs
```

Bad

```
Hello from Index  
<a href="/orders">Orders</a>
```

While this works, it causes a browser reload and loses much of the value of Ember.



Navigating With {{link-to}}

Use the Handlebars {{link-to}} expression to navigate while avoiding page reloads.

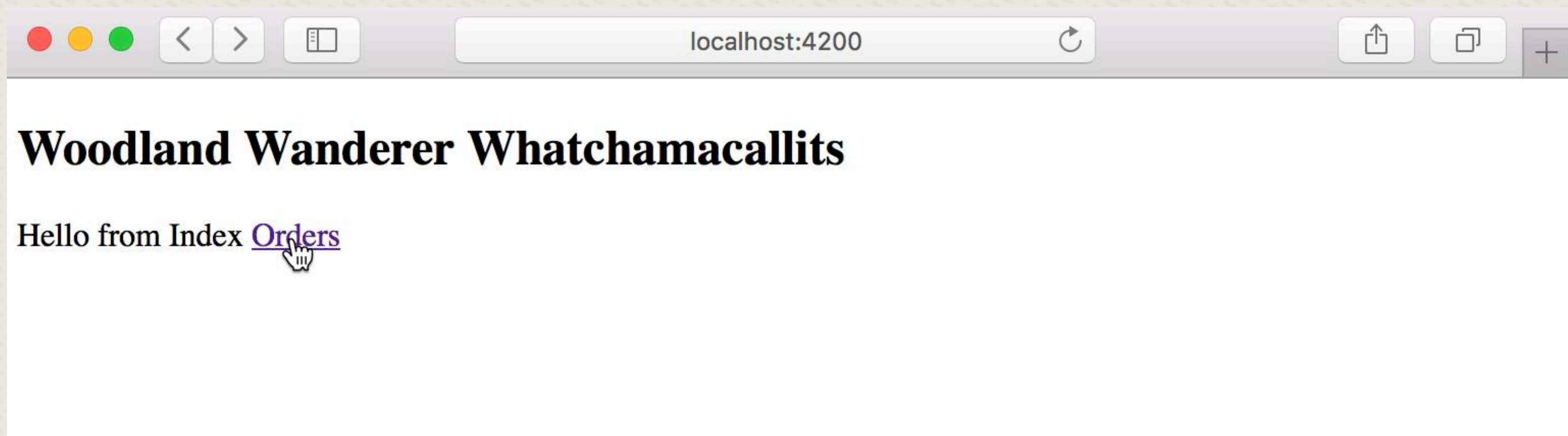
app/templates/index.hbs

Good

```
Hello from Index  
{{#link-to "orders"}}Orders{{/link-to}}
```

```
Router.map(function() {  
  this.route('orders');  
});
```

“orders” is the name mapped in the router, so it’s used here.



Navigating With {{link-to}}

Use the Handlebars {{link-to}} expression to navigate while avoiding page reloads.

app/templates/index.hbs

```
Hello from Index  
{{#link-to "orders"}}Orders{{/link-to}}
```

```
Router.map(function() {  
  this.route('orders');  
});
```



The href path is determined by the router.

```
Hello from Index  
<a id="ember3" href="/orders" class="ember-view">Orders</a>
```


Customizing {{link-to}}

The generated anchor tag may be customized by passing additional attributes to {{link-to}}.

app/templates/index.hbs

```
Hello from Index  
{{#link-to "orders" class="orders-link"}}Orders{{/link-to}}
```

The custom class is added to the generated class list.

```
Hello from Index  
<a id="ember3" href="/orders" class="orders-link ember-view">Orders</a>
```

With no ID attribute defined, Ember auto-generated one.

Customizing the Generated Tag

Most Handlebars helpers accept a tagName property to change the generated HTML tag.

app/templates/index.hbs

```
Hello from Index  
{{#link-to "orders" tagName="div"}}Orders{{/link-to}}
```



The tagName changed the generated anchor element to a div.

```
Hello from Index  
<div id="ember3" class="ember-view">Orders</div>
```

Clicking the div still works because Ember is managing the navigation events via link-to.

Additional Helpers

Ember ships with many other Handlebars helpers for tag generation and logic.

debugger	Stop processing to enter the debugger.
each	Iterate over a collection of objects.
if	Conditionally render sections of content.
input	Create an HTML input element.
link-to	Create an HTML anchor element.
log	Console log for simple debugging.
textarea	Create an HTML textarea element.
unless	Conditionally render sections of content.



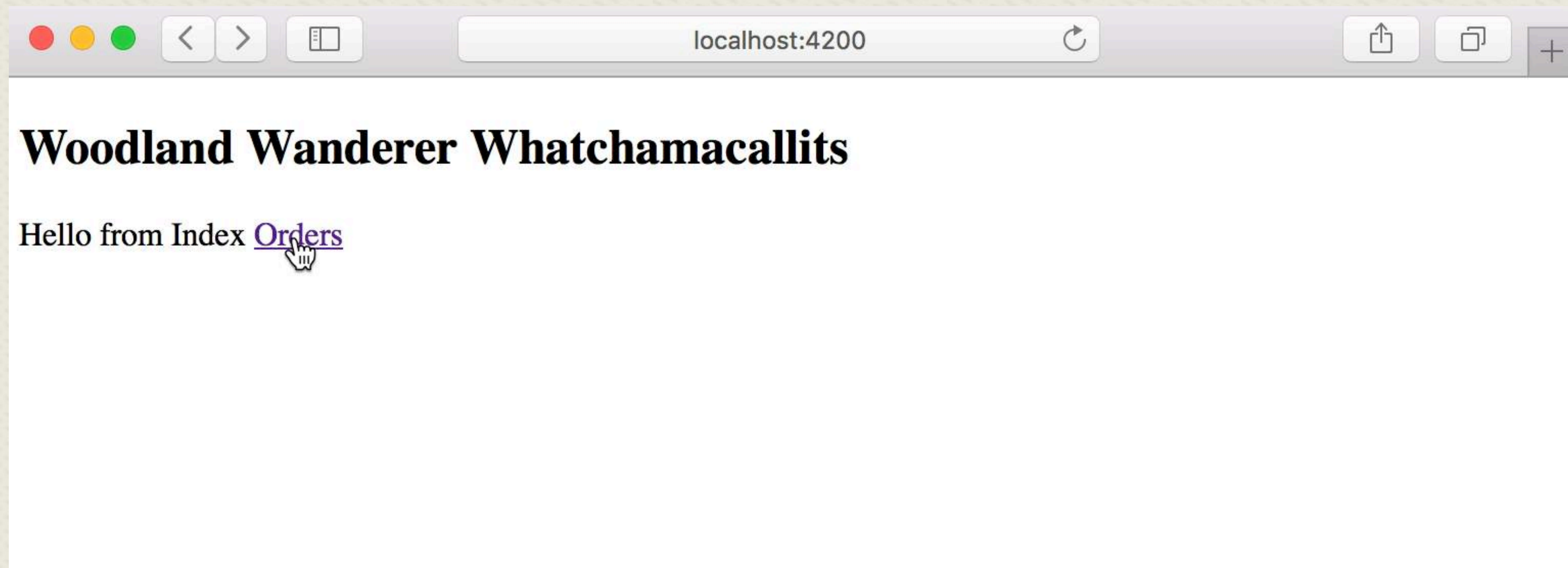
Changing the URL

If all page changes are happening on the client side, why bother updating the URL?

1. Ember uses the URL to keep track of where the user is in the application.

aka, "Application State"

2. Updating the URL means that site links are shareable and the back button works.



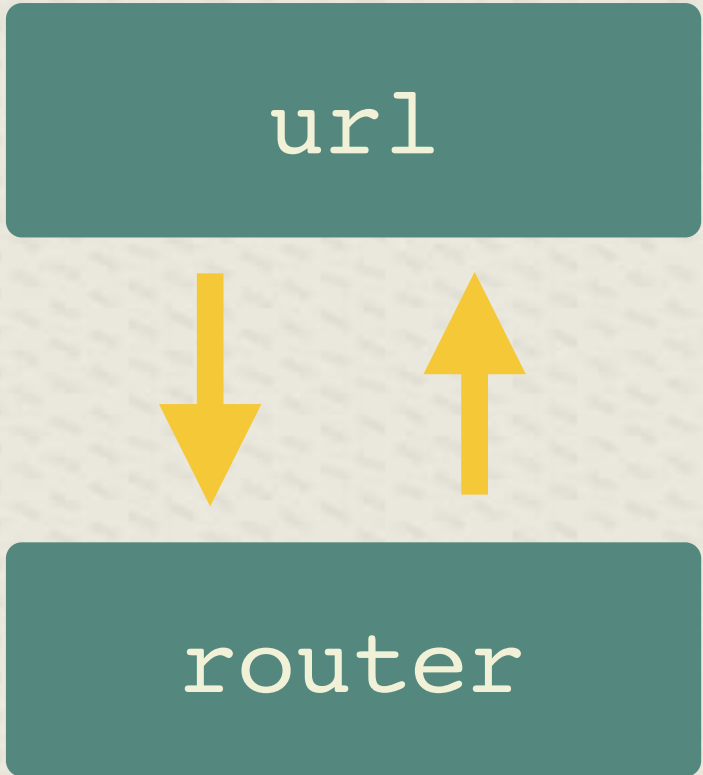
Routing With the URL

The router writes state to and reads state from the URL.



State	URL path
index	/
orders	/orders

Changes to the URL
affect the state of the
application.



User activity within the
app changes the state
and updates the URL.



Level 2.1

Routing and Templating

Templates and the Router

