EmberJS

or How I learned to stop worrying and love JavaScript frameworks

< Murcia Frontend / >

About me - José David Cano

- Bachelor's in Computer Science (UM)
- 6+ years of experience on Full-Stack web development, working on the backend using PHP.
- Currently working as a Senior Frontend Developer and Technical Lead on Neuromobile.

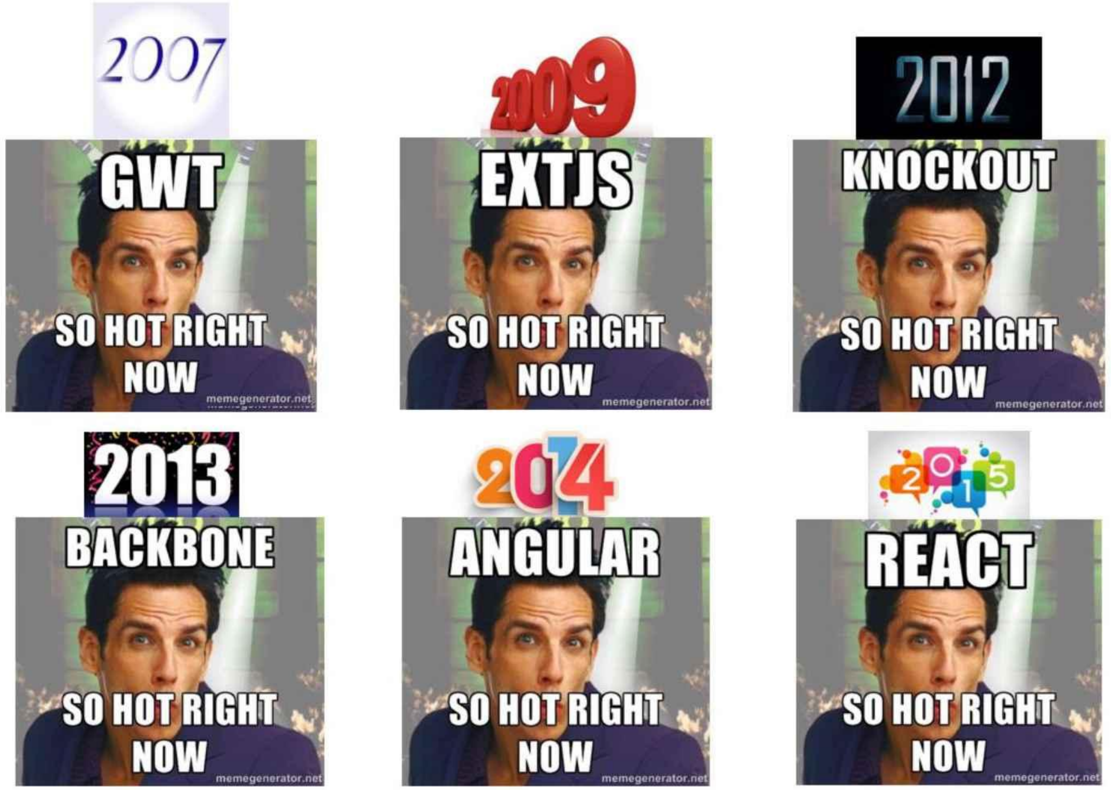
@gorzas

- I'm also a DJ who loves videogames. Up for some Overwatch or HearthStone hit me up at Gorzas#2117.

< Murcia Frontend / >

JavaScript framework mayhem

< Murcia Frontend / >



http://blog.bitovi.com/longevity-or-lack-thereof-in-javascript-frameworks/

< Murcia Frontend / >

The JavaScript framework mayhem

- Lots of frameworks, everyone borns to solve a different problem
- Short life expectancy
- Choose one is like “marry” one: it’s hard to change to another framework after it has been implemented on your WebApp
- JavaScript fatigue
- HDD (Hype-Driven Development)

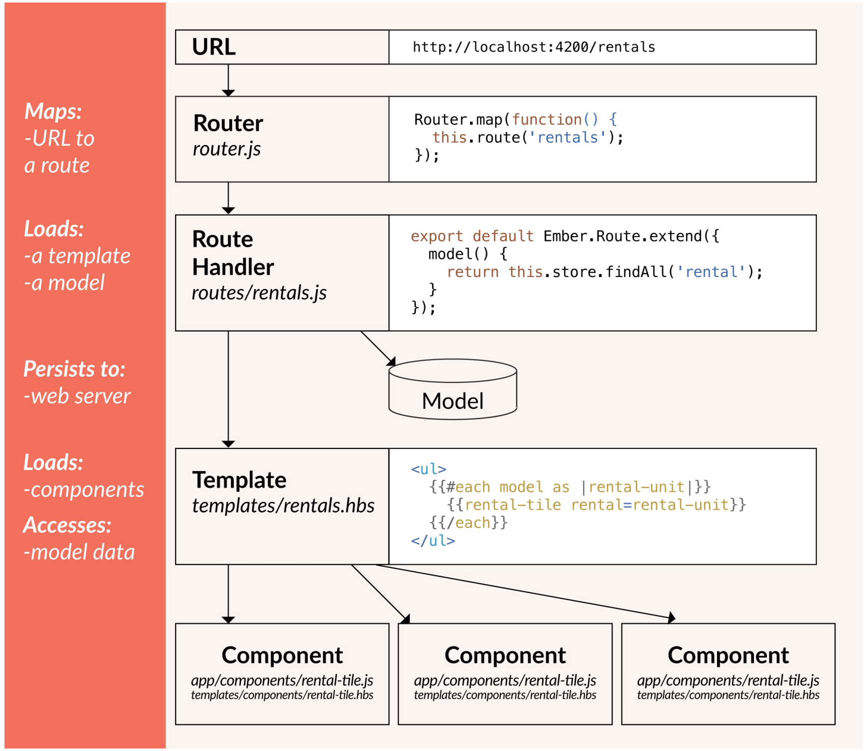
< Murcia Frontend / >

Why Ember?

- Framework MVC & SPA
- Convention over configuration
- Testing environment
- Great community and life expectancy (LinkedIn, Twitch, Discourse, Apple Music…)
- Based on future standards (WebComponents, JSON API…)
- Six-week release cycle with backwards compatibility
- Legacy browser support (IE9)

Core concepts

- Router & Routes
- Models
- Templates
- Components



```
http://localhost:4200/rentals
```

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

```
export default Ember.Route.extend({
  model() {
    return this.store.findAll('rental');
  }
});
```

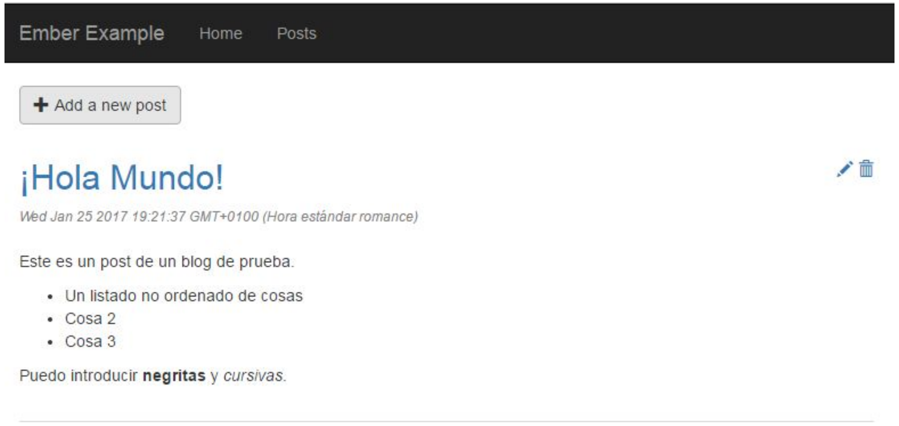
```
<ul>
  {{#each model as |rental-unit|}}
    {{rental-tile rental=rental-unit}}
  {{/each}}
</ul>
```

Let’s get started!

< Murcia Frontend />

An example

- A basic CMS made with ember



Ember Example Home Posts

Add a new post

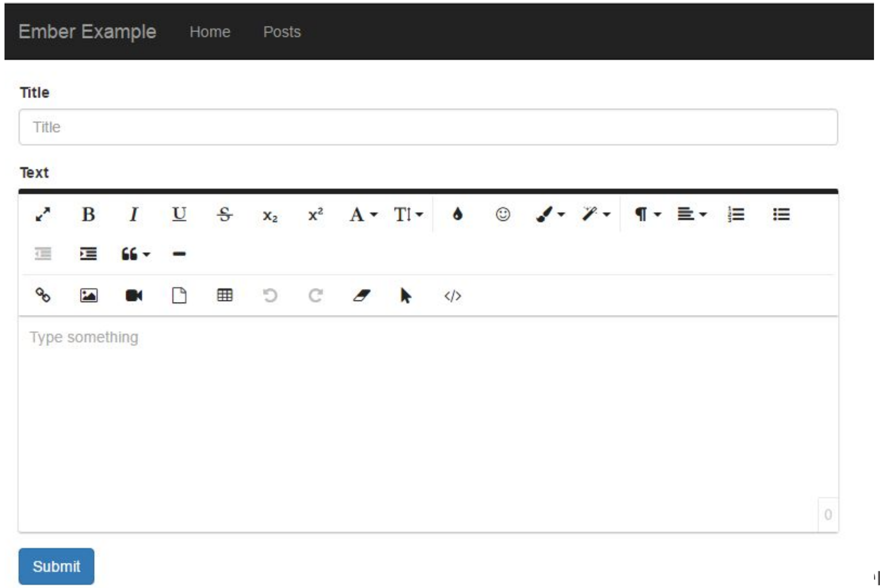
¡Hola Mundo!

Wed Jan 25 2017 19:21:37 GMT+0100 (Hora estándar romance)

Este es un post de un blog de prueba.

- Un listado no ordenado de cosas
- Cosa 2
- Cosa 3

Puedo introducir **negritas** y *cursivas*.



Ember Example Home Posts

Title

Title

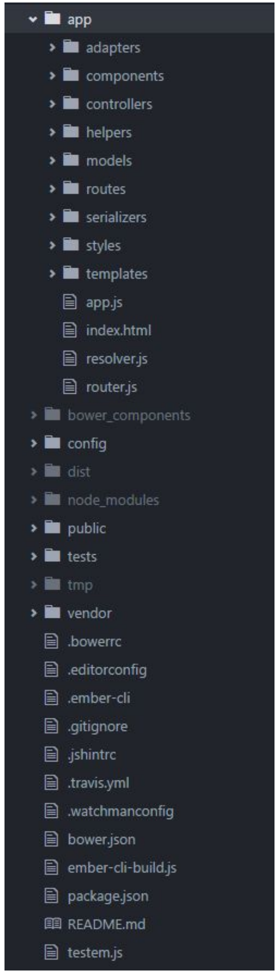
Text

Type something

Submit

First steps

1. Install ember-cli
   a. `npm install -g ember-cli`
2. Create new application
   a. `ember new ember-example`
3. Launch the app
   a. `ember server --watcher`
   b. `ember test (--server)`



```
app
  adapters
  components
  controllers
  helpers
  models
  routes
  serializers
  styles
  templates
  app.js
  index.html
  resolver.js
  router.js
bower_components
config
dist
node_modules
public
tests
tmp
vendor
.bowerrc
.editorconfig
.ember-cli
.gitignore
.jshintrc
.travis.yml
.watchmanconfig
bower.json
ember-cli-build.js
package.json
README.md
testem.js
```

< Murcia Frontend / >

Templating

- Uses Handlebars templating library, mixing HTML static with dynamic content
- Basics:
    - Render a property: `{{value}}`
    - Conditionals:
        - `{{if value “show”}}`
        - `{{#if value}}<strong>{{value}}</strong>{{/if}}`
        - `{{unless value “hide”}}`
    - Loops:
        - `{{#each posts as |post|}}<li>{{post}}</li>{{/each}}`

< Murcia Frontend / >

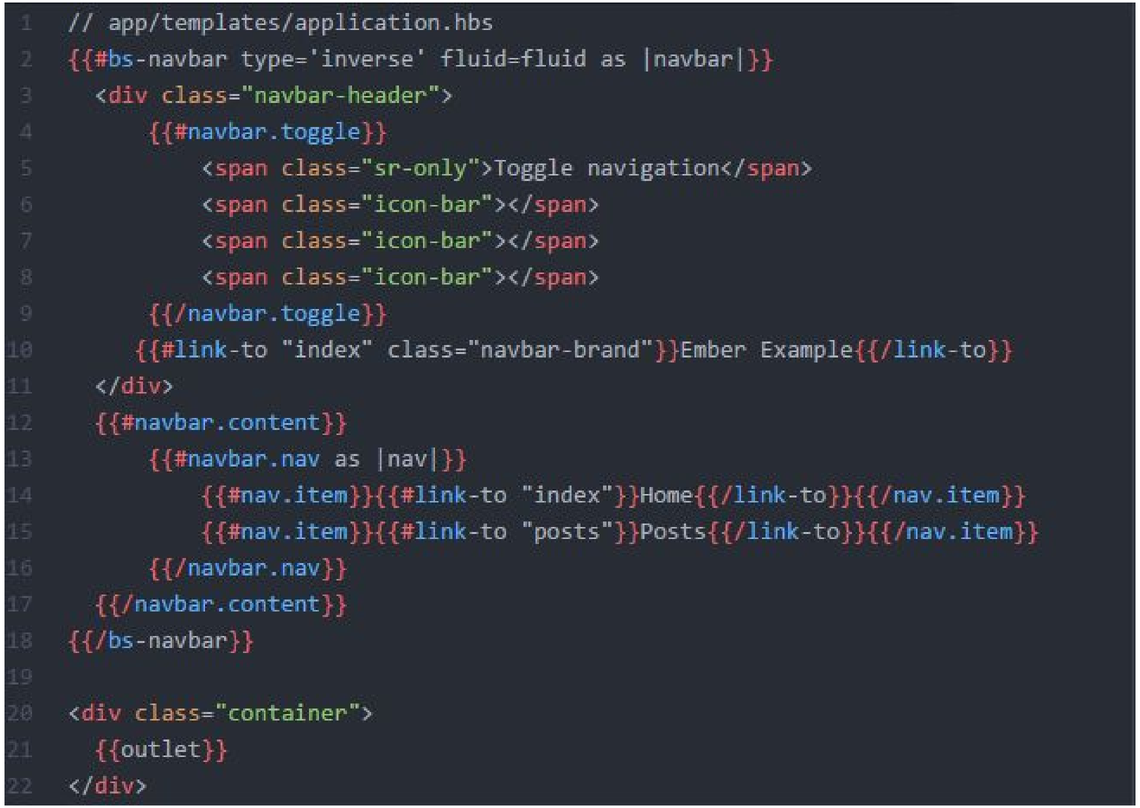
Templating - application.hbs

```
ember g template application
```

- Main template of the app, it’s always on the screen.
- Route template will be rendered in an `{{outlet}}` helper.

< Murcia Frontend / >

Templating - application.hbs



```
// app/templates/application.hbs
{{#bs-navbar type='inverse' fluid=fluid as |navbar|}}
  <div class="navbar-header">
      {{#navbar.toggle}}
          <span class="sr-only">Toggle navigation</span>
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
          <span class="icon-bar"></span>
      {{/navbar.toggle}}
     {{#link-to "index" class="navbar-brand"}}Ember Example{{/link-to}}
  </div>
  {{#navbar.content}}
      {{#navbar.nav as |nav|}}
          {{#nav.item}}{{#link-to "index"}}Home{{/link-to}}{{/nav.item}}
          {{#nav.item}}{{#link-to "posts"}}Posts{{/link-to}}{{/nav.item}}
      {{/navbar.nav}}
  {{/navbar.content}}
{{/bs-navbar}}

<div class="container">
  {{outlet}}
</div>
```

< Murcia Frontend / >

Routes

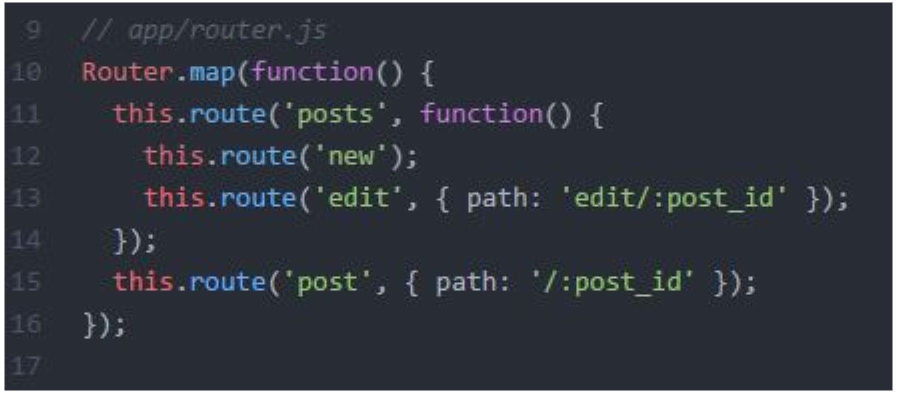
- Creating a new route:

```
ember g route posts
```

- Loads data to be available on the template
- Creates:
    - app/routes/posts.js
    - app/templates/posts.hbs
    - tests/unit/routes/posts.js

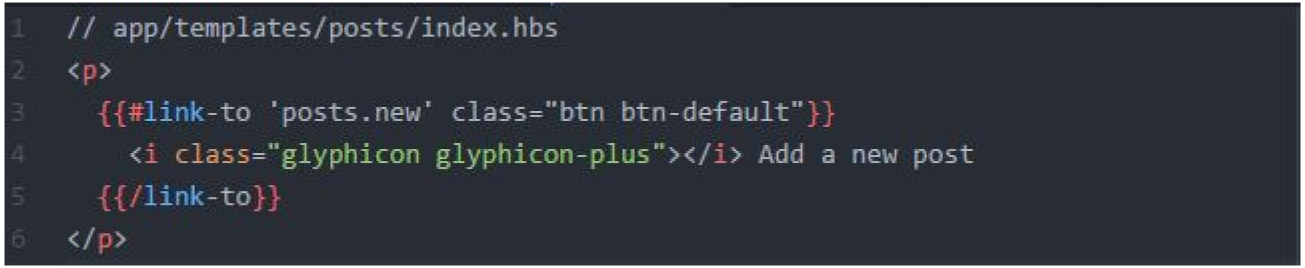
< Murcia Frontend / >

Routes



```
// app/router.js
Router.map(function() {
  this.route('posts', function() {
    this.route('new');
    this.route('edit', { path: 'edit/:post_id' });
  });
  this.route('post', { path: '/:post_id' });
});
```

link-to helper



```
// app/templates/posts/index.hbs
<p>
  {{#link-to 'posts.new' class="btn btn-default"}}
    <i class="glyphicon glyphicon-plus"></i> Add a new post
  {{/link-to}}
</p>
```

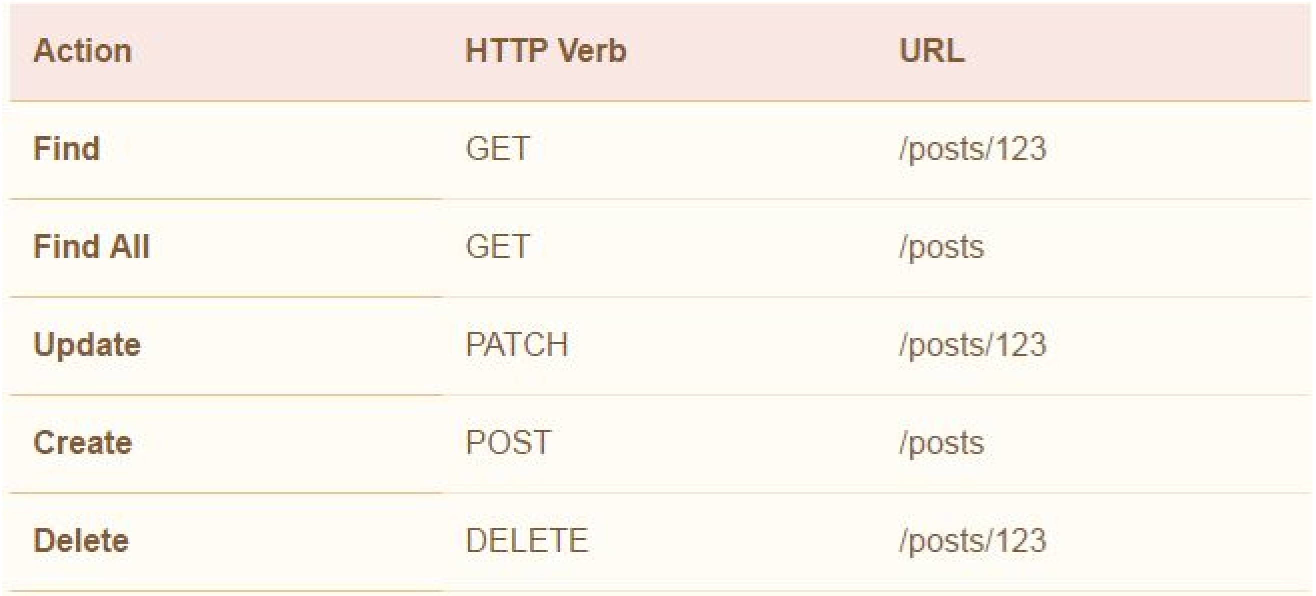
- Redirects to route /posts/new
- Allows parameters: `{{#link-to ‘posts’ post}}`

Models

```
ember g model post
```

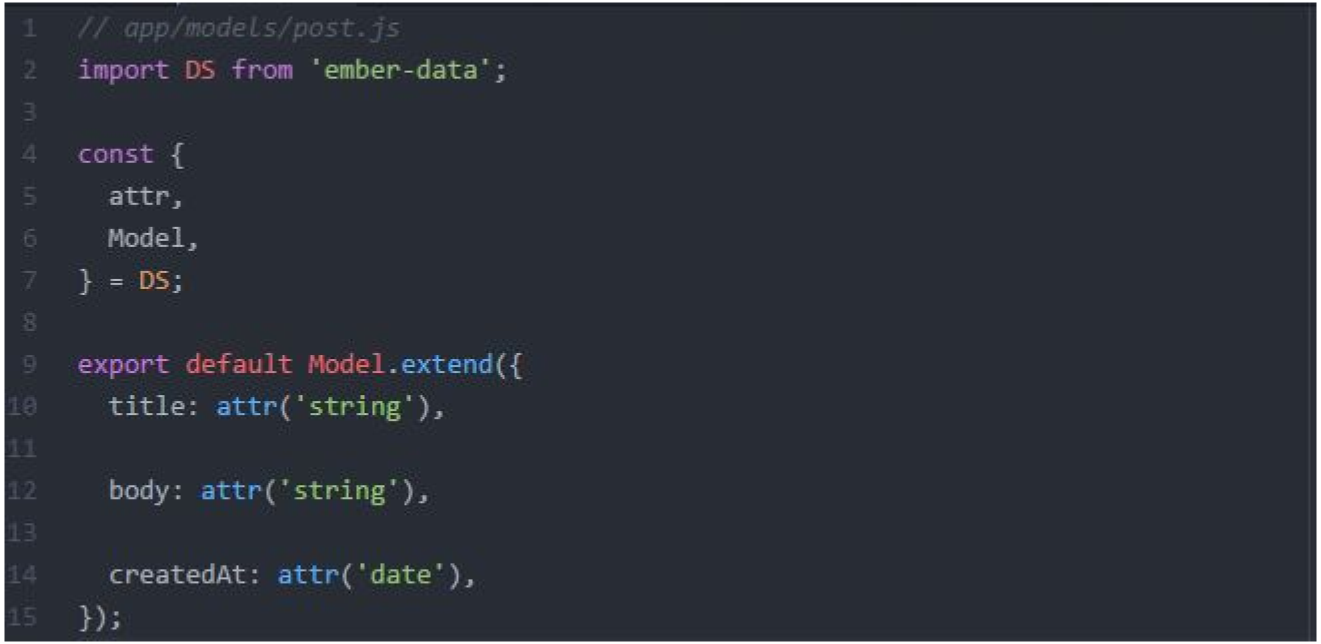
- Added with Ember-Data library (included in ember-cli)
- Represents underlying data of the app
- Transforms: translates server data to UI data and viceversa (default types: boolean, date, string and number)
- Methods belongsTo and hasMany to define relationships between models
- By default, it uses JSON API

Models



Action	HTTP Verb	URL
Find	GET	/posts/123
Find All	GET	/posts
Update	PATCH	/posts/123
Create	POST	/posts
Delete	DELETE	/posts/123

Models



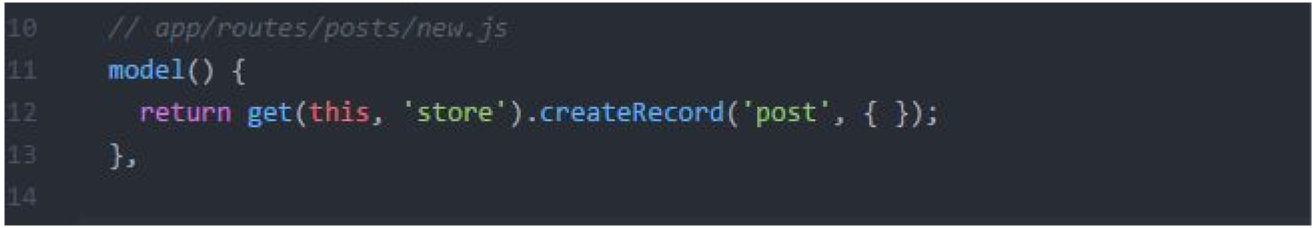
```
// app/models/post.js
import DS from 'ember-data';

const {
  attr,
  Model,
} = DS;

export default Model.extend({
  title: attr('string'),

  body: attr('string'),

  createdAt: attr('date'),
});
```

Components

```
ember g component form-blog-post
```

- Must have at least one dash in the name
- Creates:
  - app/component/form-blog-post.js
  - app/templates/components/form-blog-post.hbs
  - tests/integration/components/form-blog-post-test.js
- Data down, actions up

Routes - model

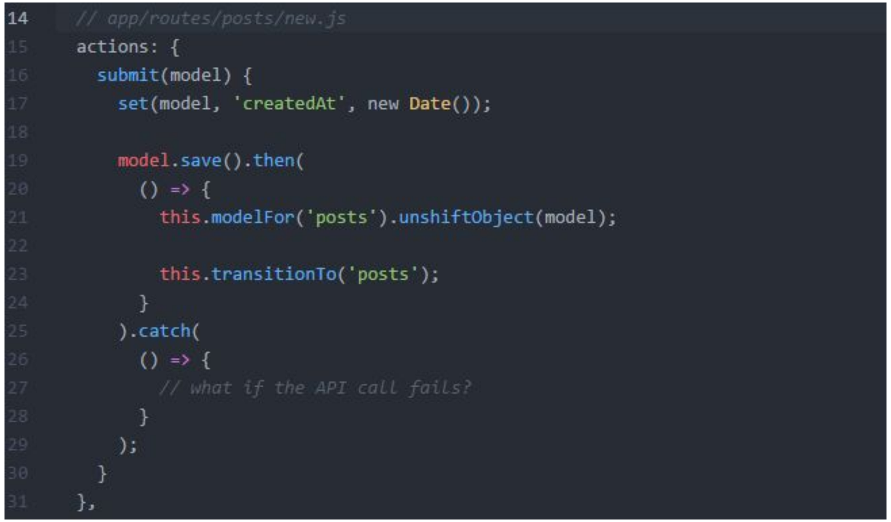


```
// app/routes/posts/new.js
model() {
  return get(this, 'store').createRecord('post', { });
},
```

- Gets data from the storage (usually the server) and makes it available in the template

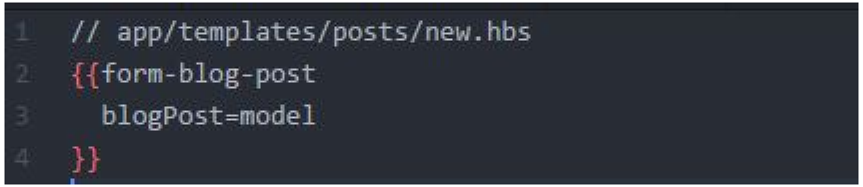
Route/controller - actions

- Callback for events on templates and inner components
- Can define them in routes and controllers
- For enable actions on routes you'll need the library ember-route-action-helper



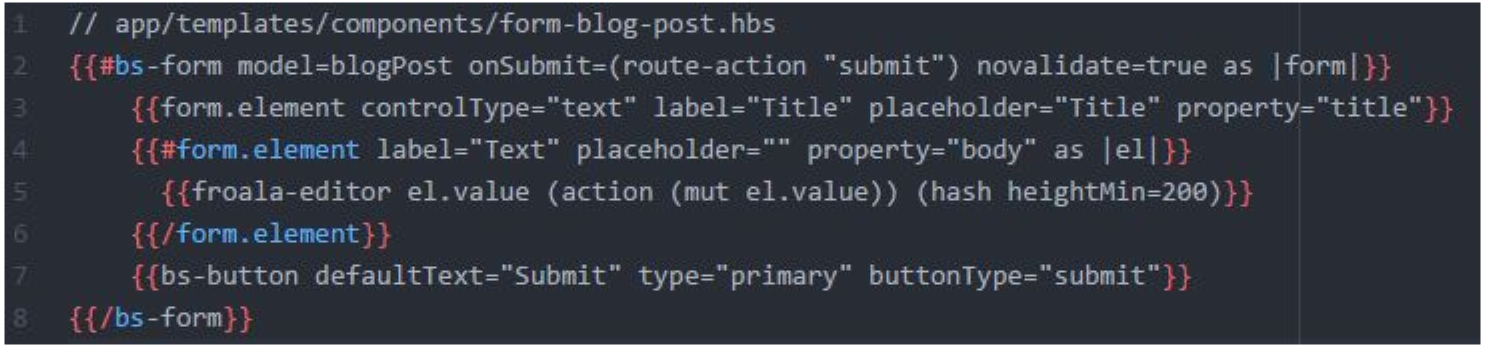
```
// app/routes/posts/new.js
actions: {
  submit(model) {
    set(model, 'createdAt', new Date());

    model.save().then(
      () => {
        this.modelFor('posts').unshiftObject(model);

        this.transitionTo('posts');
      }
    ).catch(
      () => {
        // what if the API call fails?
      }
    );
  }
},
```

Route - posts/new.hbs



```
// app/templates/posts/new.hbs
{{form-blog-post
  blogPost=model
}}
```

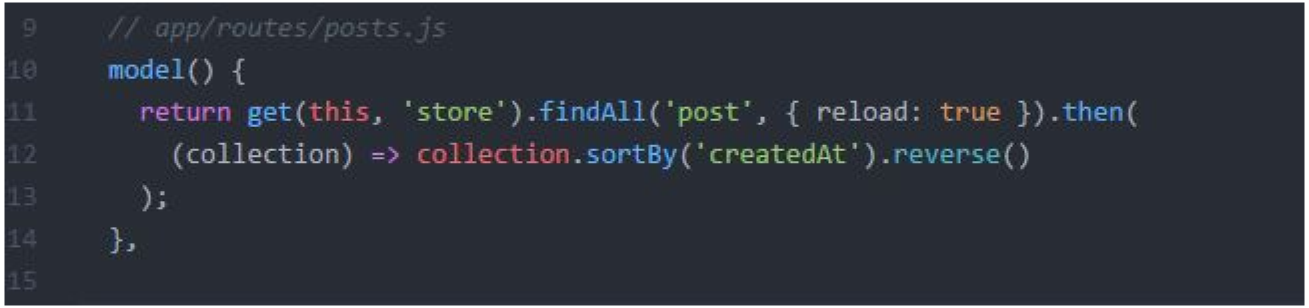
Components - form-blog-post.hbs



```
// app/templates/components/form-blog-post.hbs
{{#bs-form model=blogPost onSubmit=(route-action "submit") novalidate=true as |form|}}
    {{form.element controlType="text" label="Title" placeholder="Title" property="title"}}
    {{#form.element label="Text" placeholder="" property="body" as |el|}}
      {{froala-editor el.value (action (mut el.value)) (hash heightMin=200)}}
    {{/form.element}}
    {{bs-button defaultText="Submit" type="primary" buttonType="submit"}}
{{/bs-form}}
```

< Murcia Frontend / >

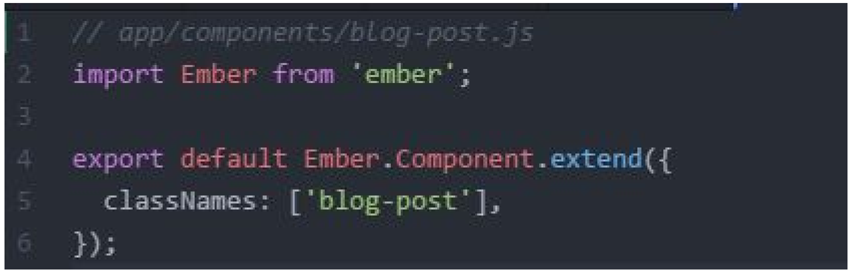
Showing a list of posts



```
// app/routes/posts.js
model() {
  return get(this, 'store').findAll('post', { reload: true }).then(
    (collection) => collection.sortBy('createdAt').reverse()
  );
},
```

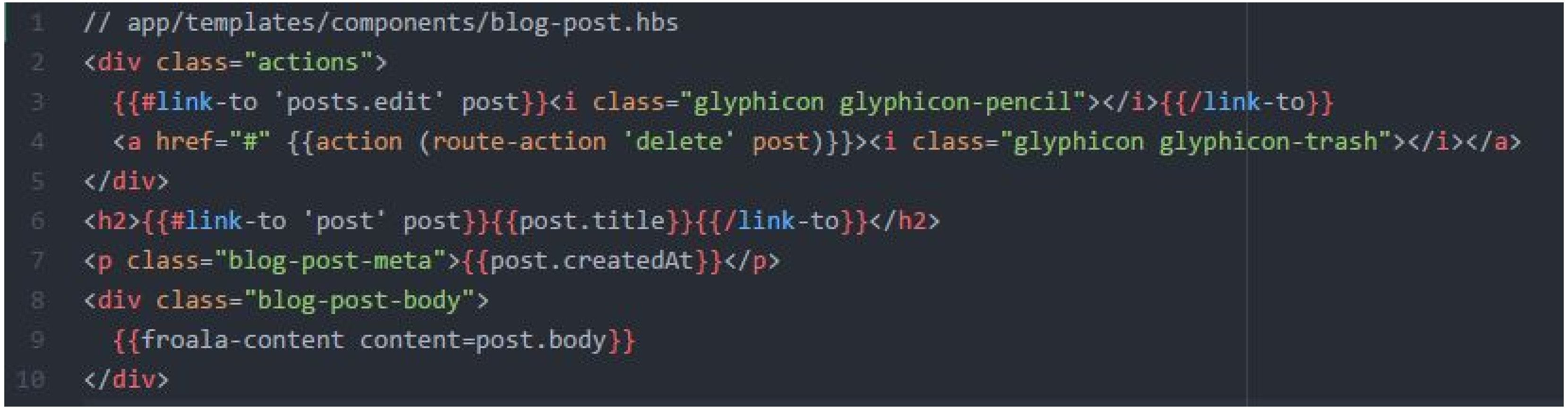
Components - blog-post

- Good practice: add className with the component name to encapsulate styles



```
// app/components/blog-post.js
import Ember from 'ember';

export default Ember.Component.extend({
  classNames: ['blog-post'],
});
```

Components - blog-post

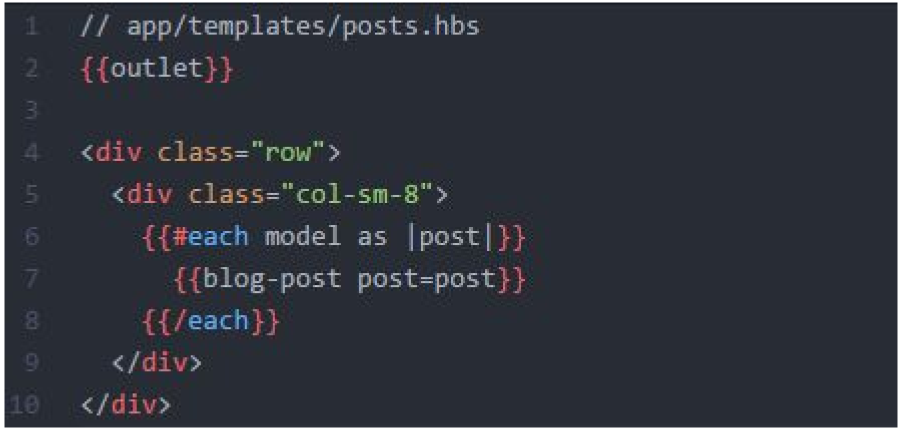


```
// app/templates/components/blog-post.hbs
<div class="actions">
  {{#link-to 'posts.edit' post}}<i class="glyphicon glyphicon-pencil"></i>{{/link-to}}
  <a href="#" {{action (route-action 'delete' post)}}><i class="glyphicon glyphicon-trash"></i></a>
</div>
<h2>{{#link-to 'post' post}}{{post.title}}{{/link-to}}</h2>
<p class="blog-post-meta">{{post.createdAt}}</p>
<div class="blog-post-body">
  {{froala-content content=post.body}}
</div>
```

< Murcia Frontend />

Showing a list of posts

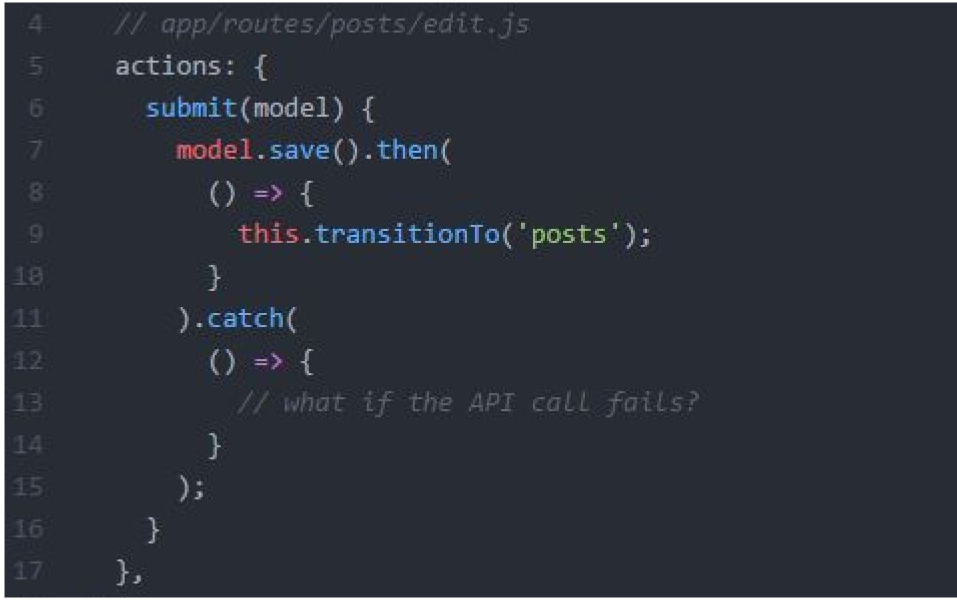
- {{outlet}} helper shows nested routes (posts.index, posts.new and posts.edit).



```
// app/templates/posts.hbs
{{outlet}}

<div class="row">
  <div class="col-sm-8">
    {{#each model as |post|}}
      {{blog-post post=post}}
    {{/each}}
  </div>
</div>
```

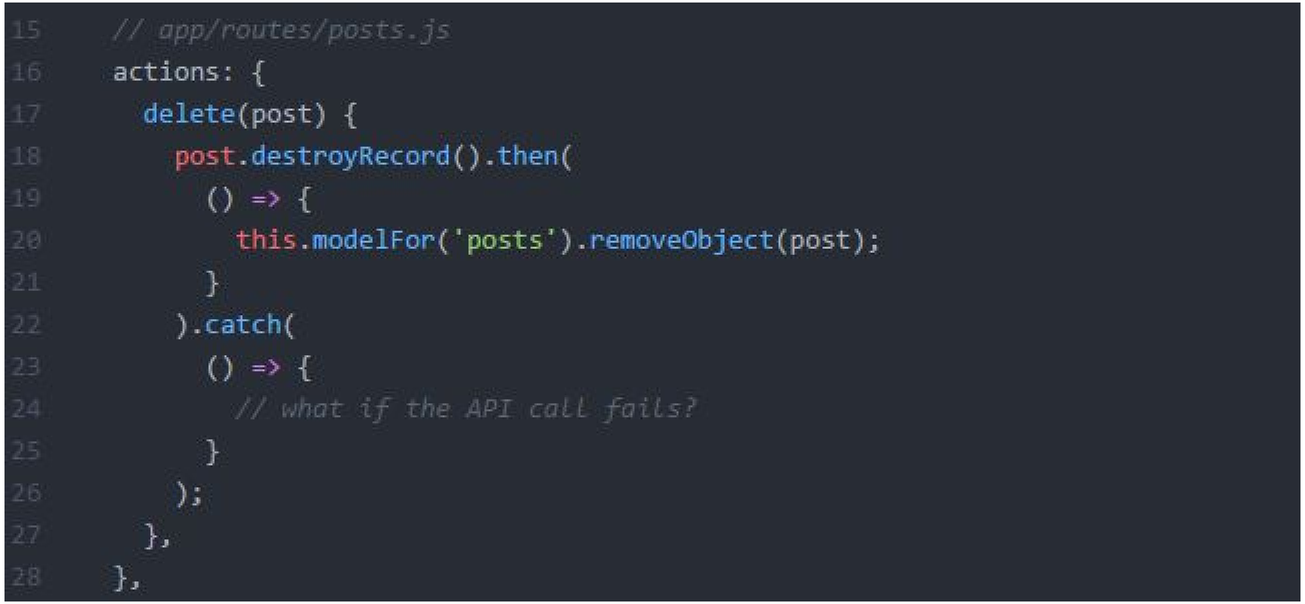
Update a post

- Route posts/edit/:post_id
- Loads the model automatically.
- Reuses component form-blog-post



```
// app/routes/posts/edit.js
actions: {
  submit(model) {
    model.save().then(
      () => {
        this.transitionTo('posts');
      }
    ).catch(
      () => {
        // what if the API call fails?
      }
    );
  }
},
```

Delete a post



```
// app/routes/posts.js
actions: {
  delete(post) {
    post.destroyRecord().then(
      () => {
        this.modelFor('posts').removeObject(post);
      }
    ).catch(
      () => {
        // what if the API call fails?
      }
    );
  },
},
```

< Murcia
Frontend / >

Other elements

- Services
- Custom handlebars helpers
- Transforms
- Adapters and serializers
- Ember inspector

Resources

- Ember-Cli - https://ember-cli.com/
- Guide - https://guides.emberjs.com/v2.11.0/
- Slack - https://ember-community-slackin.herokuapp.com/
- Styleguide - https://github.com/netguru/ember-styleguide
- Example code - https://github.com/Gorzas/ember-example

Thanks!

< Murcia Frontend />