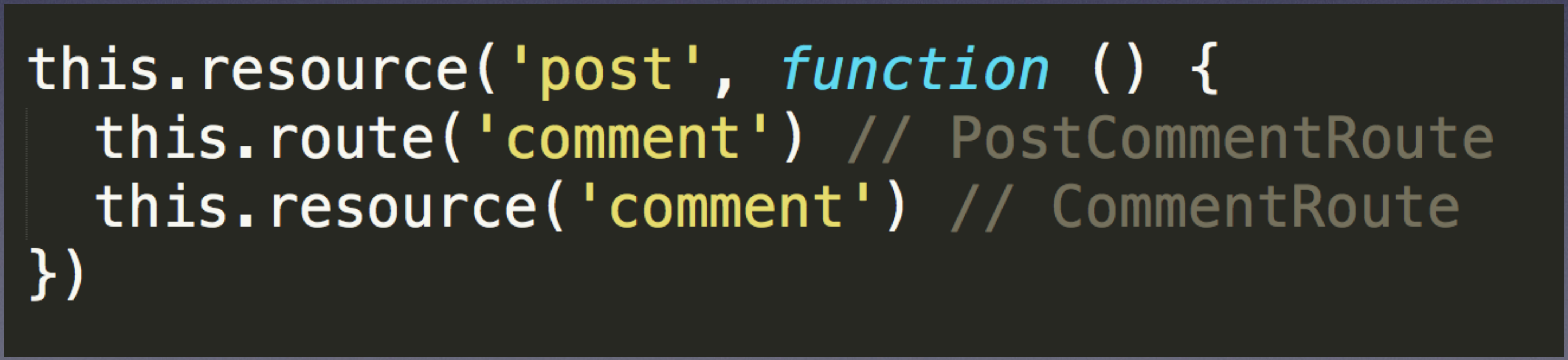
Avoiding Common Pitfalls

The mistakes everyone makes with Ember.js

@alexspeller

Routing

- Resource vs Route
  - < 1.7 - routes cannot have sub routes
  - >= 1.7 - only difference is namespace inheritance
  - Guideline: route == verb, resource == noun



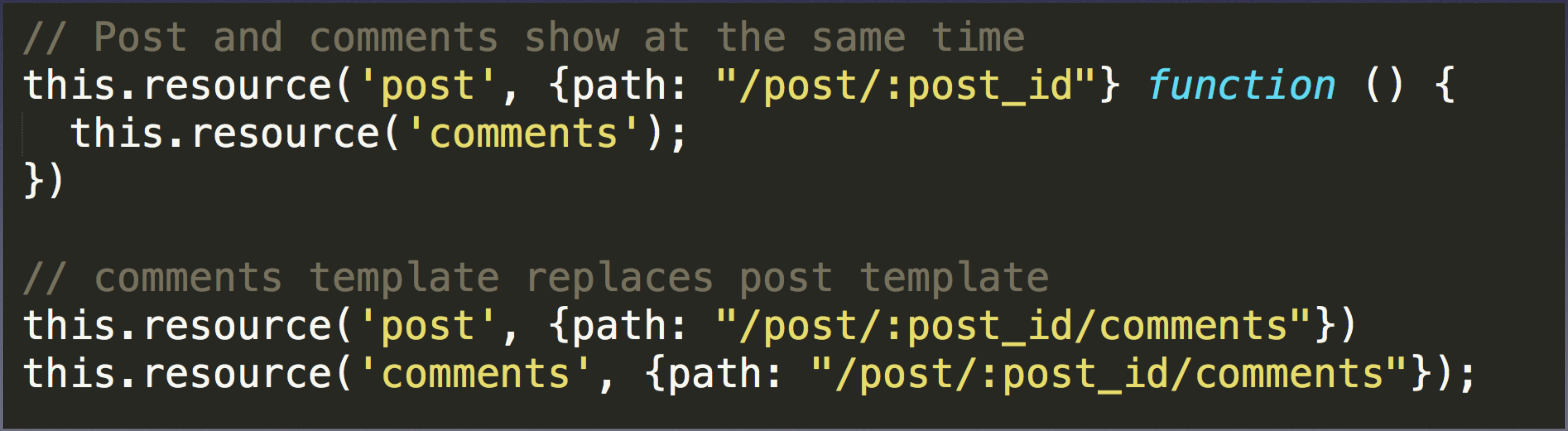
```
this.resource('post', function () {
  this.route('comment') // PostCommentRoute
  this.resource('comment') // CommentRoute
})
```

Routing

- Validation vs Setup phase
- First *all* beforeModel / model / afterModel hooks are called
- Then *all* activate / setupController / renderTemplate hooks are called
- Calling transition.abort() or this.transitionTo() in the validation phase cancels the transition without affecting controller state

Routing

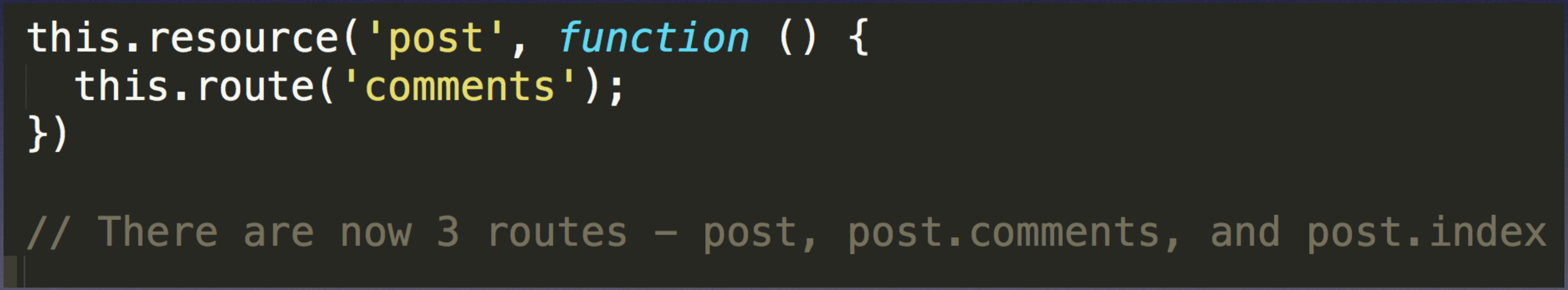
- Route nesting === Template Nesting
- If you want a non-nested template, give the path option



```
// Post and comments show at the same time
this.resource('post', {path: "/post/:post_id"} function () {
  this.resource('comments');
})

// comments template replaces post template
this.resource('post', {path: "/post/:post_id/comments"})
this.resource('comments', {path: "/post/:post_id/comments"});
```

Routing

- Automatic index route is generated when using resources



```
this.resource('post', function () {
  this.route('comments');
})

// There are now 3 routes - post, post.comments, and post.index
```

Routing

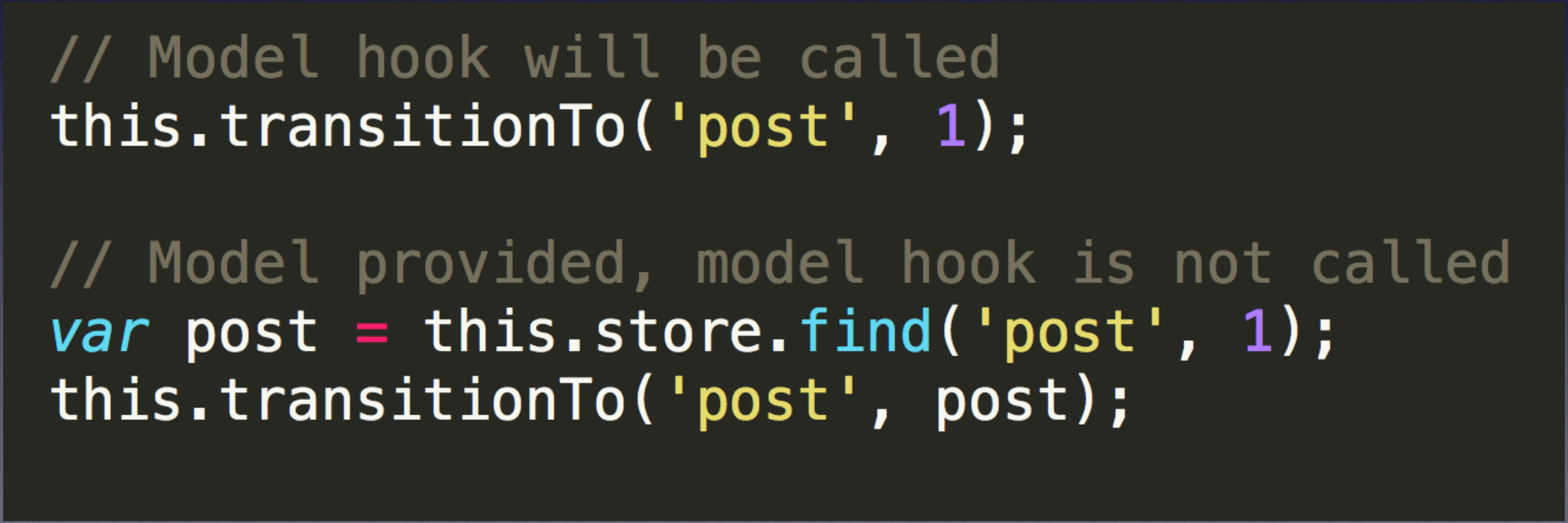
- Index is not just for collections

```
this.resource('post', {path: "/post/:post_id"} function () {
  this.resource('comments');
})
```

```
{{!-- post.hbs --}}
{{outlet}}

{{!-- post/index.hbs --}}
<h1>Post {{title}}</h1>

{{!-- post/comments.hbs --}}
<h2>Comments</h2>
{{#each}} {{body}} {{/each}}
```

Routing

- The model hook will not be always be called



```
// Model hook will be called
this.transitionTo('post', 1);

// Model provided, model hook is not called
var post = this.store.find('post', 1);
this.transitionTo('post', post);
```

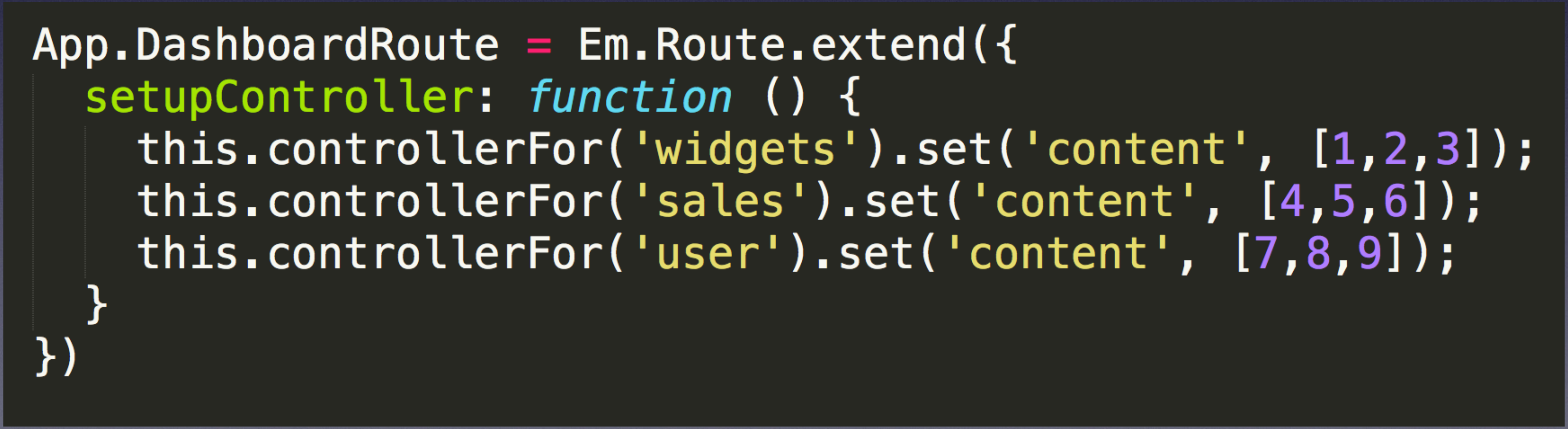
Routing

- You can transition without providing all params

```
// in your router
this.resource('post', {path: "/posts/:post_id"});

// somewhere else
this.transitionTo('post');
```

Routing

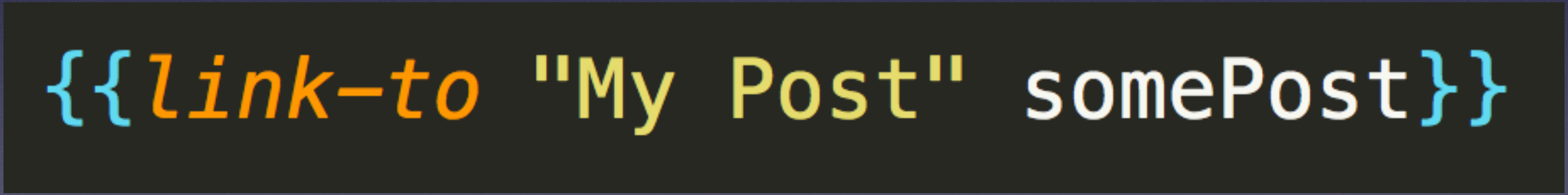
- Routes and controllers are not necessarily 1:1



```
App.DashboardRoute = Em.Route.extend({
  setupController: function () {
    this.controllerFor('widgets').set('content', [1,2,3]);
    this.controllerFor('sales').set('content', [4,5,6]);
    this.controllerFor('user').set('content', [7,8,9]);
  }
})
```

Routing

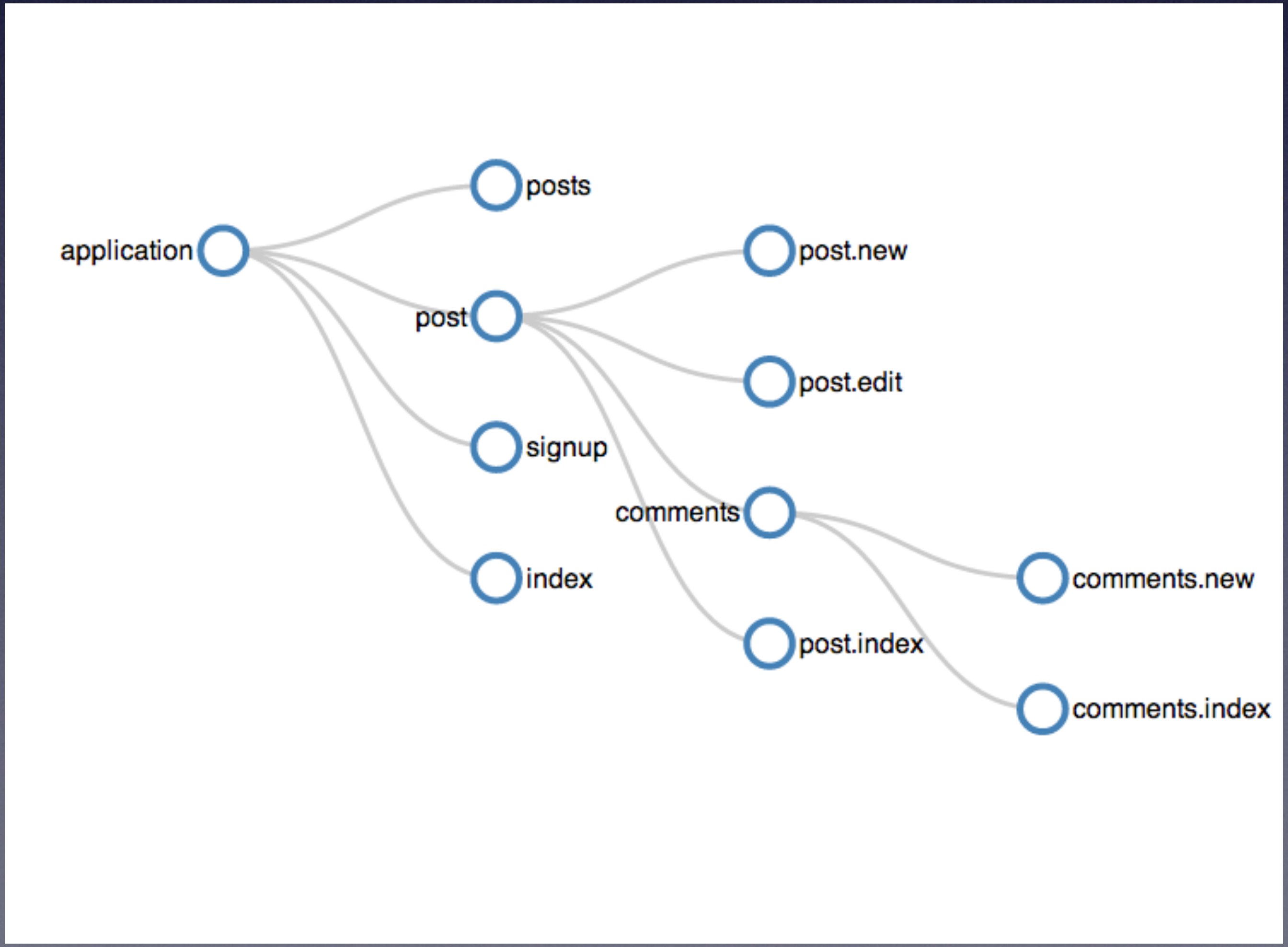
- serialize is called for link-to



```
{{link-to "My Post" somePost}}
```

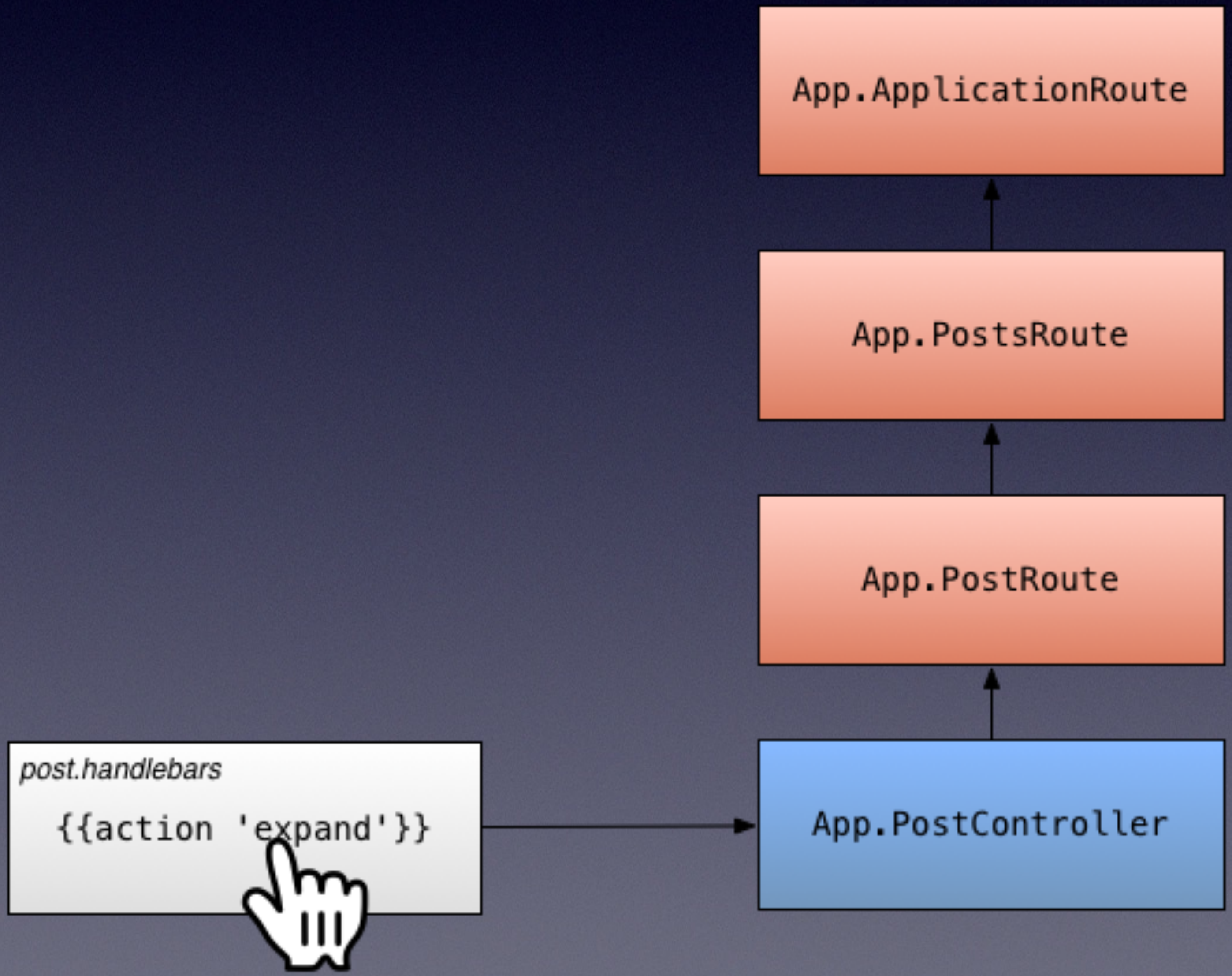
Routing

- Hackathon result
- http://alexspeller.com/ember-diagonal/



Actions

- Actions bubble to all active routes



Templates

- Precompilation

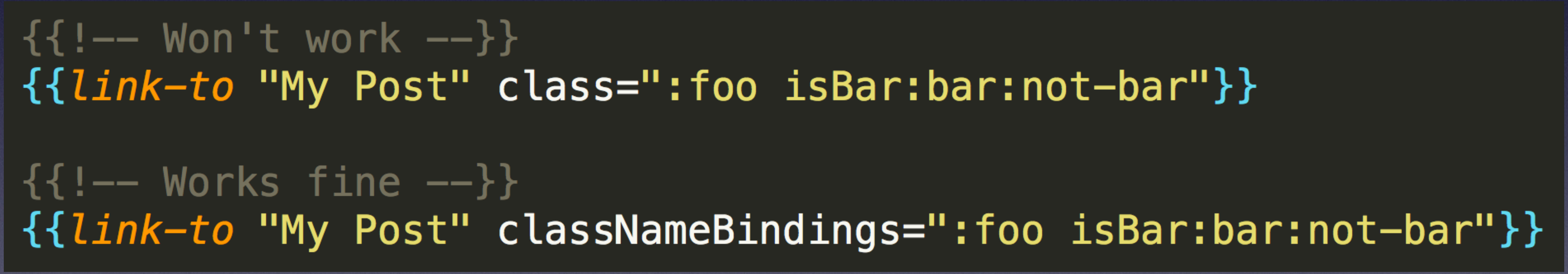
Templates

- {{bind-attr}} special behaviour for class

```
<div {{bind-attr class=":foo isBar:bar:not-bar"}}>
```

Templates

- classNameBindings on helpers e.g. {{link-to}}



```
{{!-- Won't work --}}
{{link-to "My Post" class=":foo isBar:bar:not-bar"}}

{{!-- Works fine --}}
{{link-to "My Post" classNameBindings=":foo isBar:bar:not-bar"}}
```

Templates

- Difference between {{render}}, {{partial}} and {{view}}
- http://emberjs.com/guides/templates/rendering-with-helpers/
- Render cannot take arguments - is not contextual
- Render does not have anything to do with the route of the same name, does not need an associated route

Templates

- {{#each}} helper has a lot of subtle different uses
- http://ember.guru/2014/hidden-features-of-the-each-aka-loopedy-loop-helper

Components

- Components are *intentionally* isolated
- You should pass in all the state you need
- Communicating between components is hard
- http://madhatted.com/2013/8/31/emberfest-presentation-complex-architectures-in-ember

Components - sendAction

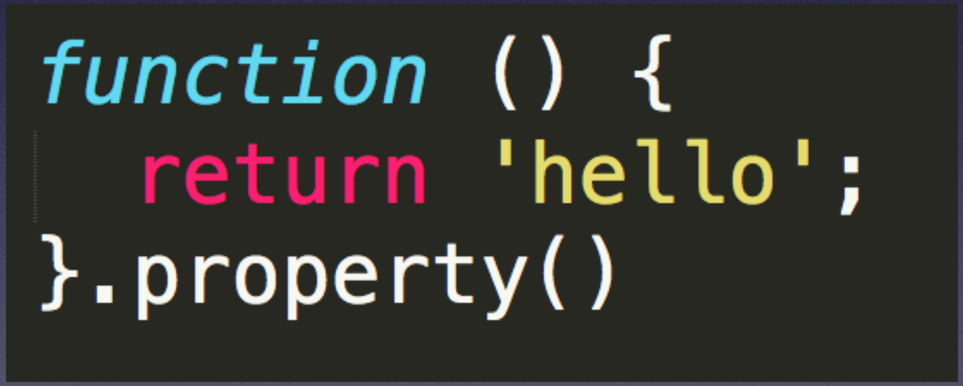
- Actions must be passed in the template *as strings*

```
{{post-editor action="savePost" cancel="cancelPost"}}
```

```
App.PostEditorComponent = Em.Component.extend({
  doStuff: function () {
    this.send('savePost'); // does nothing
    this.send('action'); // does nothing
    this.sendAction('savePost'); // does nothing
    this.sendAction(); // sends savePost to controller

    // probably causes error
    this.sendAction(argumentToSavePost);

    // sends savePost to controller with argument
    this.sendAction('action', argumentToSavePost);

    // sends cancelPost to controller
    this.sendAction('cancel');
  }
})
```

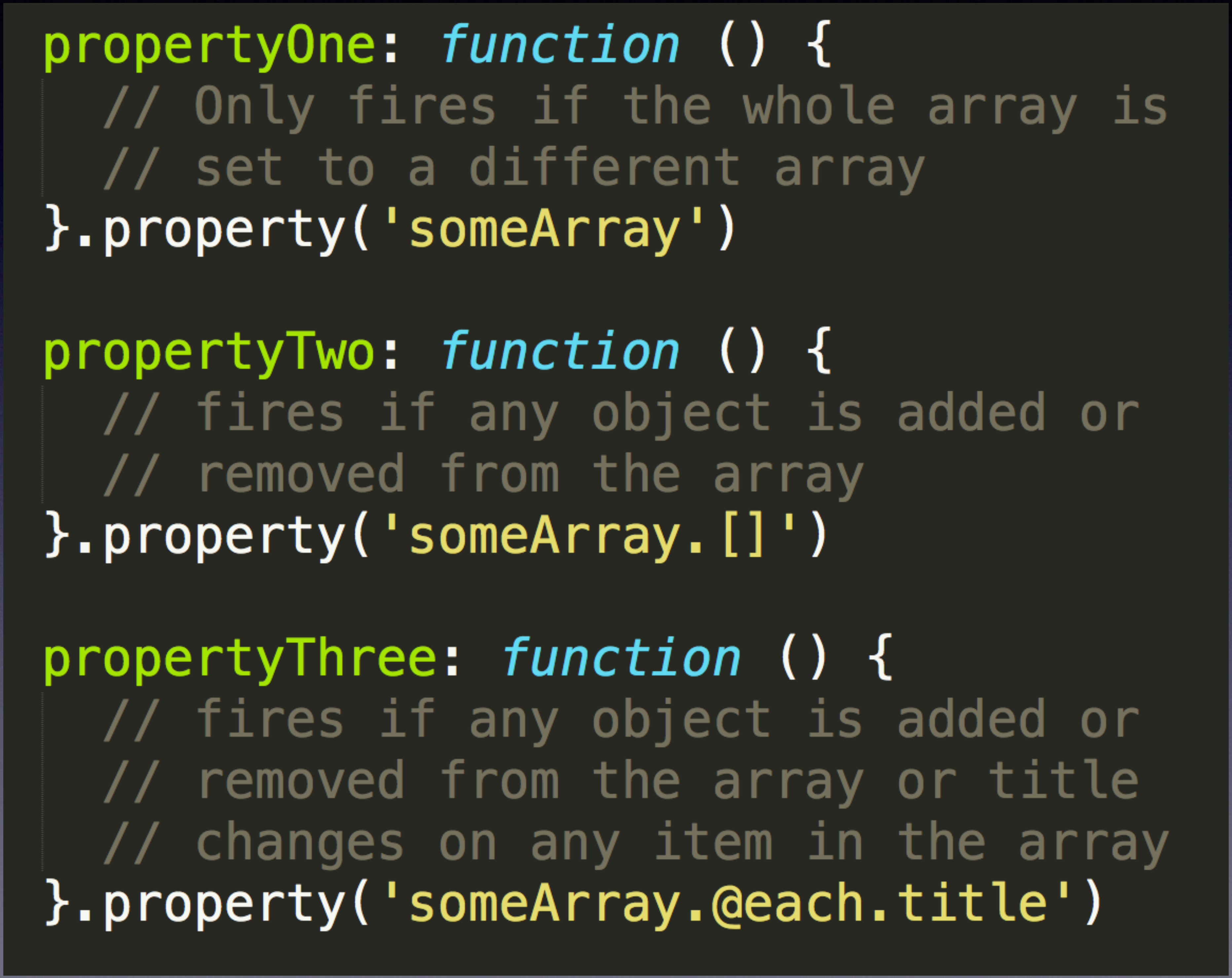
Computed Properties

- Forgetting the property helper



```
function () {
  return 'hello';
}.property()
```

Array dependent keys - [], @each



```
propertyOne: function () {
  // Only fires if the whole array is
  // set to a different array
}.property('someArray')

propertyTwo: function () {
  // fires if any object is added or
  // removed from the array
}.property('someArray.[]')

propertyThree: function () {
  // fires if any object is added or
  // removed from the array or title
  // changes on any item in the array
}.property('someArray.@each.title')
```

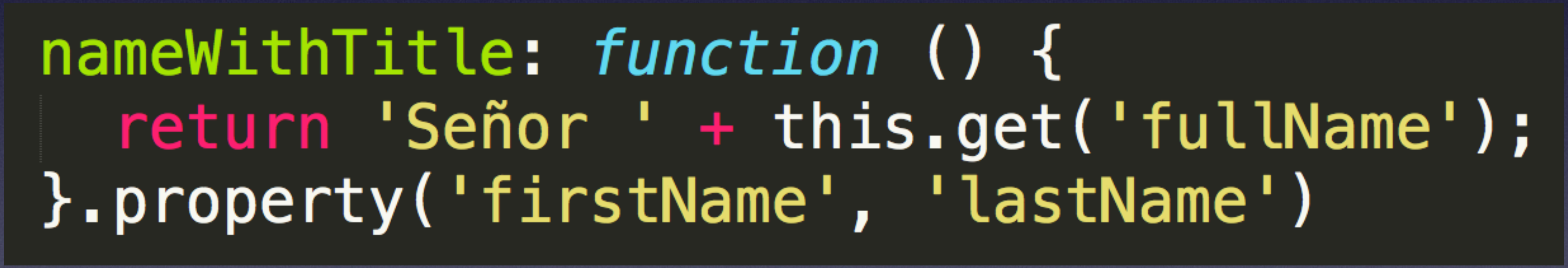
Computed Properties

- Overwriting Computed properties

```
ApplicationController = Em.Controller.extend({
  isOn: Em.computed.not('isOff');
});

controller.set('isOn', false);
// you have now overwritten the isOn property forever
```

Computed Properties

- Wrong dependencies



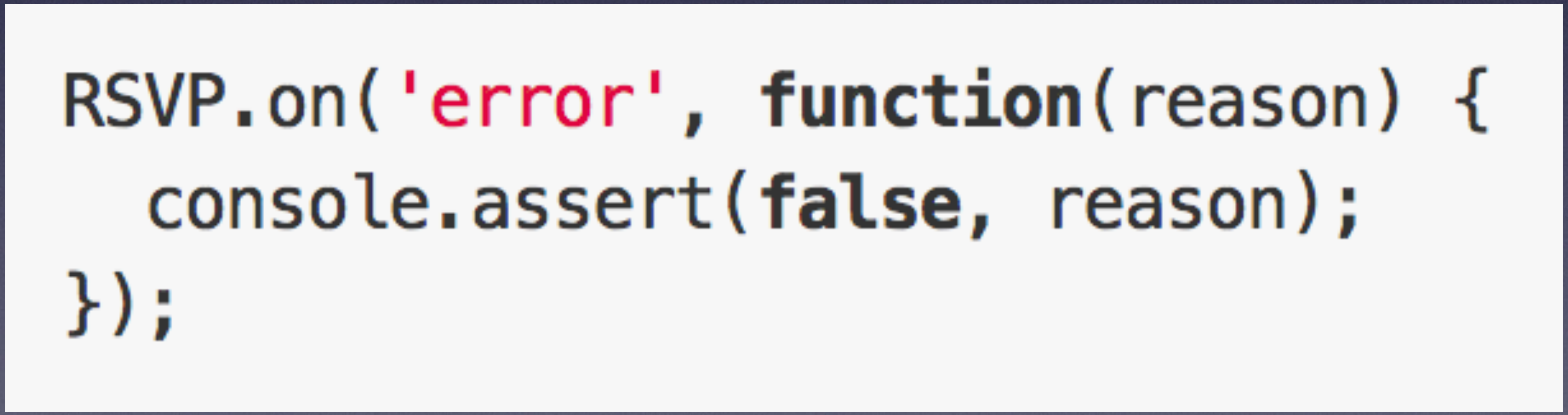
```
nameWithTitle: function () {
  return 'Señor ' + this.get('fullName');
}.property('firstName', 'lastName')
```

Observers

- Not called for unconsumed computed properties
- Synchronous
- http://emberjs.com/guides/object-model/observers/ is (now) good at explaining all the issues

Promises

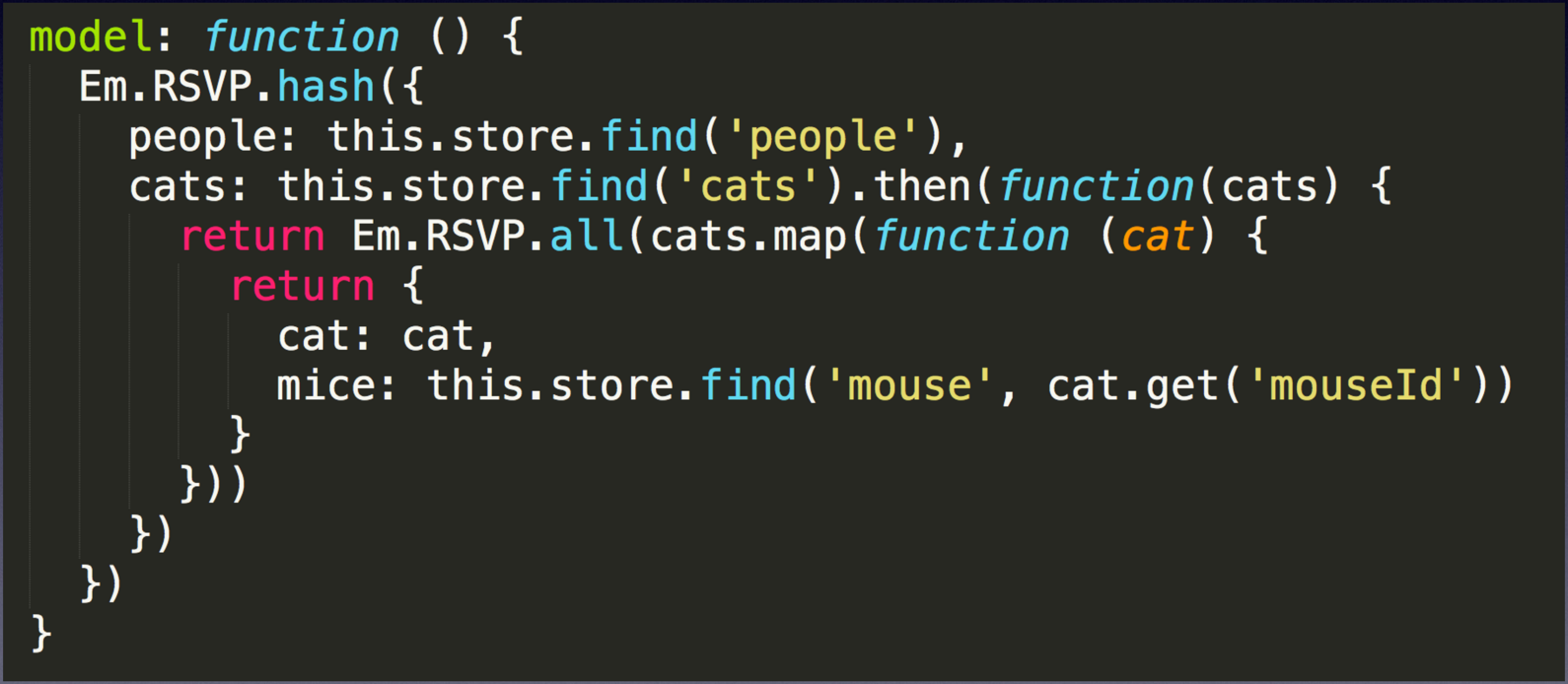
- Errors getting swallowed



```
RSVP.on('error', function(reason) {
  console.assert(false, reason);
});
```

Promises

- Aggregating Promises Em.RSVP.all etc



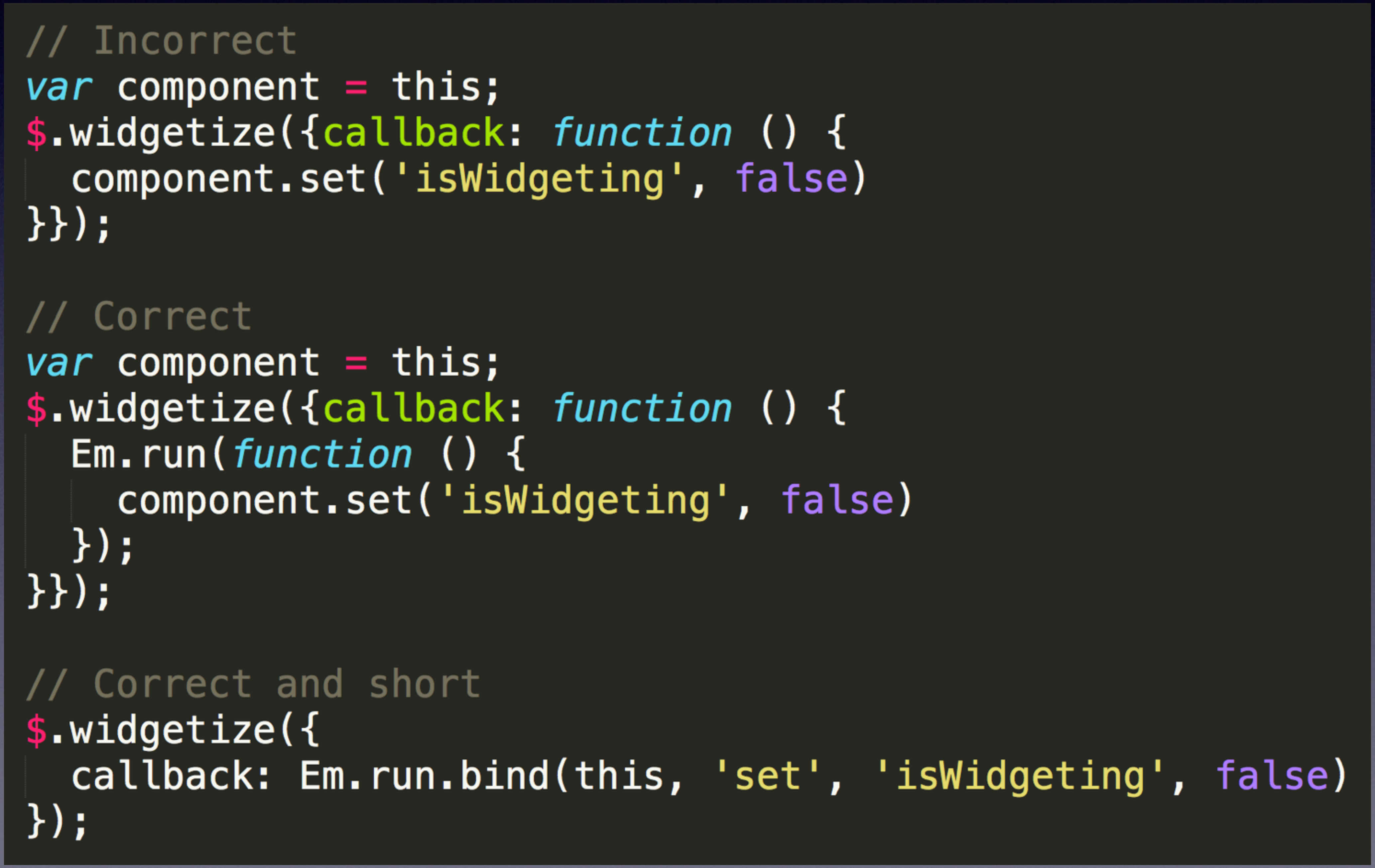
```
model: function () {
  Em.RSVP.hash({
    people: this.store.find('people'),
    cats: this.store.find('cats').then(function(cats) {
      return Em.RSVP.all(cats.map(function (cat) {
        return {
          cat: cat,
          mice: this.store.find('mouse', cat.get('mouseId'))
        }
      }))
    })
  })
}
```

Promises

- Stefan Penner talk:
- https://www.youtube.com/watch?v=eHomHs3PrP8

The Run Loop

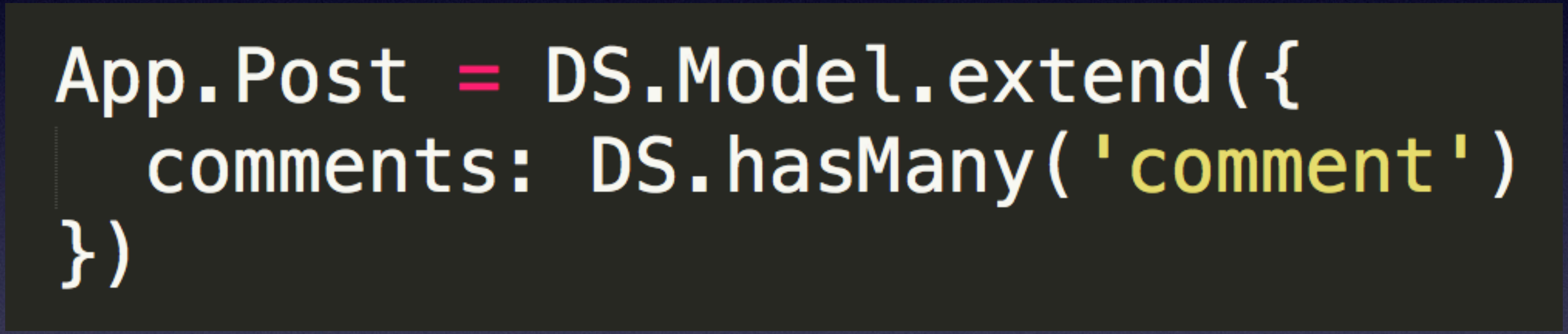
- Not using Em.run when dealing with 3rd Party Callbacks



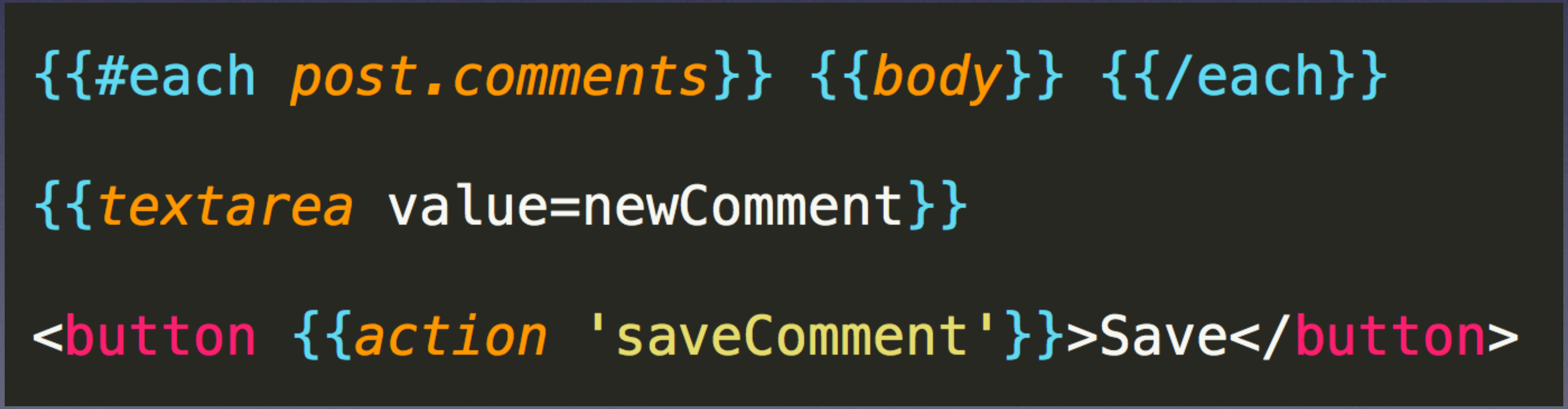
```
// Incorrect
var component = this;
$.widgetize({callback: function () {
  component.set('isWidgeting', false)
}});

// Correct
var component = this;
$.widgetize({callback: function () {
  Em.run(function () {
    component.set('isWidgeting', false)
  });
}});

// Correct and short
$.widgetize({
  callback: Em.run.bind(this, 'set', 'isWidgeting', false)
});
```

Ember Data

- Non-live relations https://github.com/emberjs/data/issues/1308



```
App.Post = DS.Model.extend({
  comments: DS.hasMany('comment')
})
```

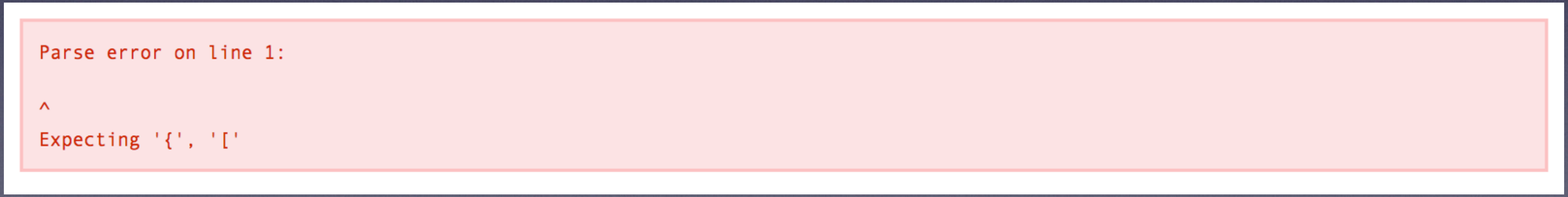


```
{{#each post.comments}} {{body}} {{/each}}

{{textarea value=newComment}}

<button {{action 'saveComment'}}>Save</button>
```

store.filter is a workaround for this

Ember Data

- Invalid JSON
- “” is not valid JSON:
- Silently fails :(



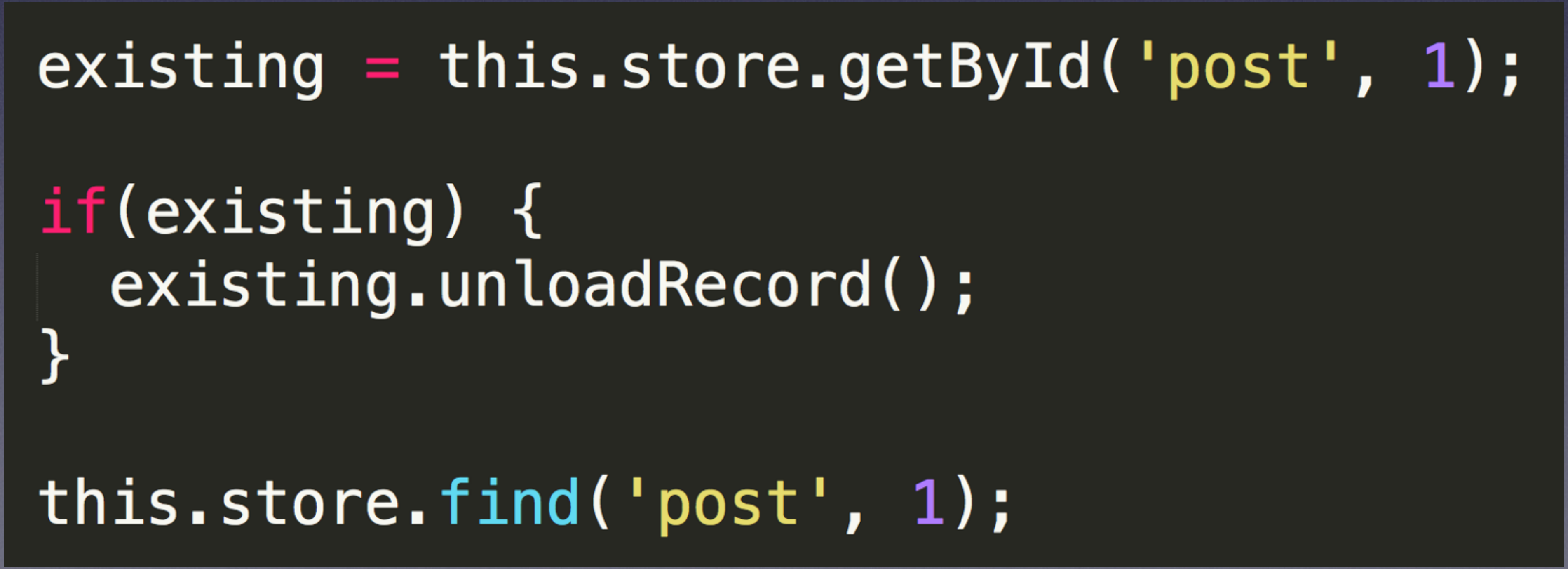
```
Parse error on line 1:

^
Expecting '{', '['
```

Ember Data

- find() vs find({})
- find('posts') returns a live array of all records in a store, even new records
- find('posts', {}) will perform the same query but the result is not live

Ember Data

- in-memory caching issues
- store.find('post', 1) will cache after first request
- post.reload() now possible
- If it may not be loaded it can be quite ugly:

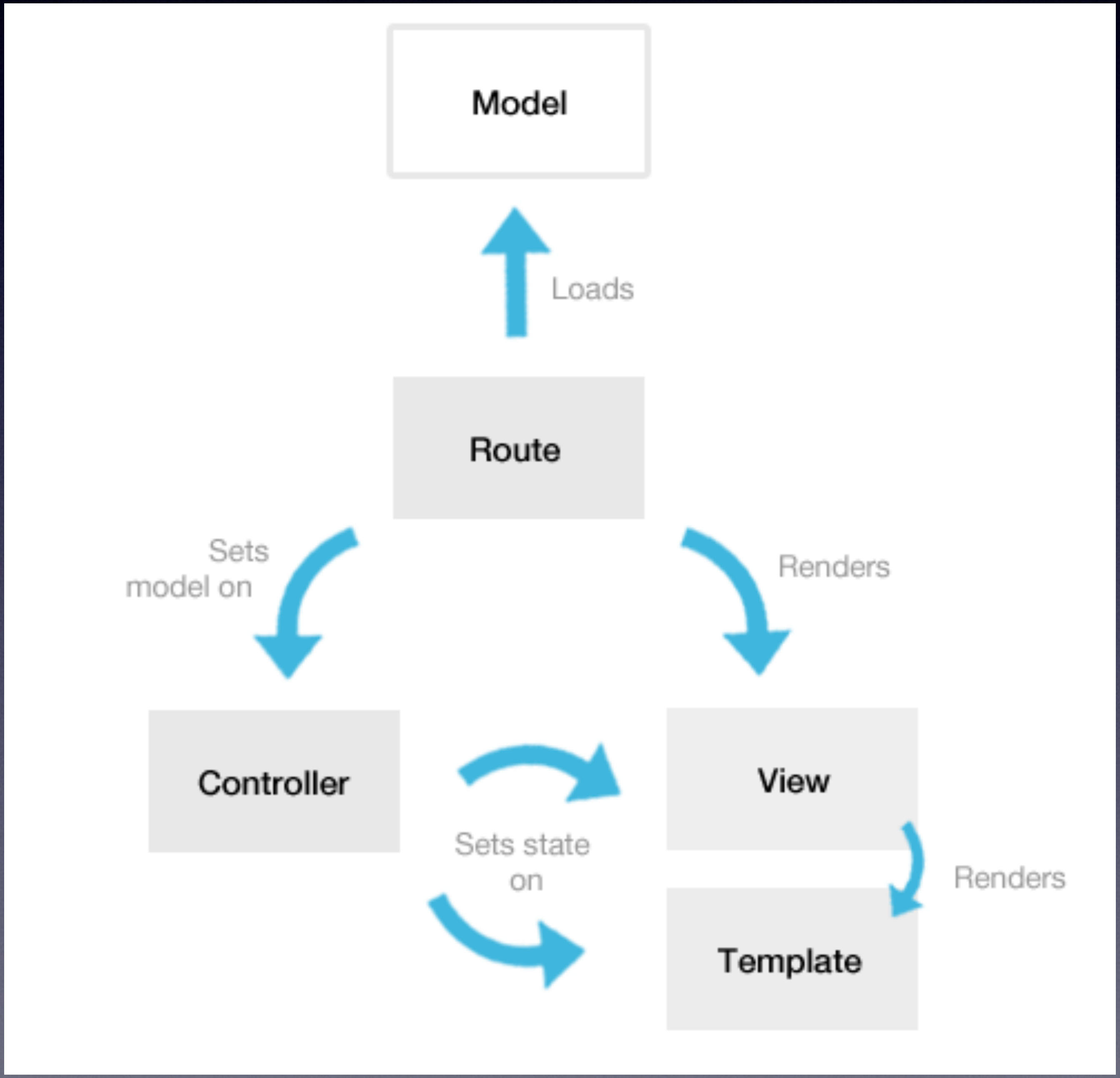


```
existing = this.store.getById('post', 1);

if(existing) {
  existing.unloadRecord();
}

this.store.find('post', 1);
```

Debugging

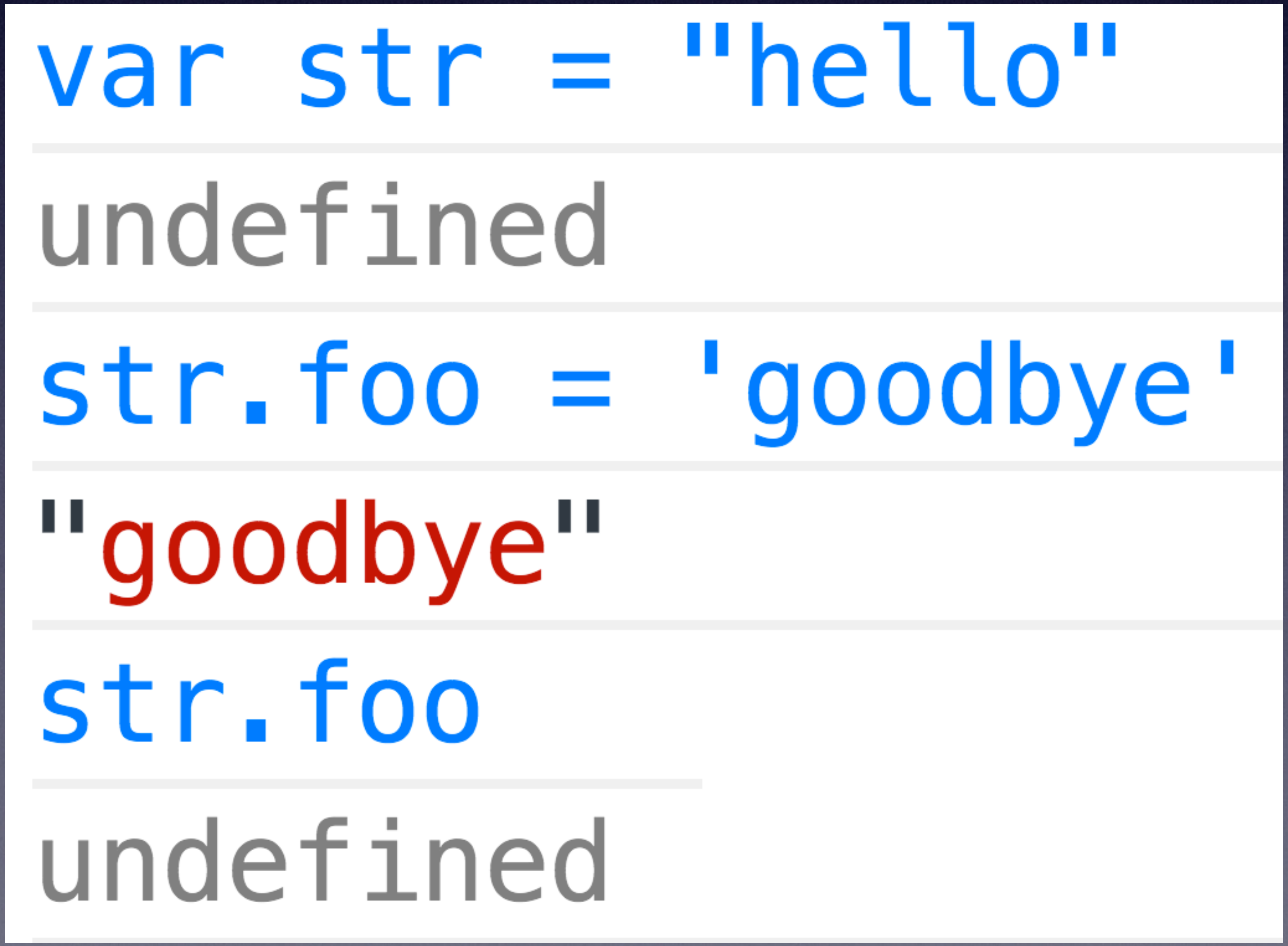
- Use the debugger! (and {{debugger}})
- Use console.log! (and {{log}})
- Use the Ember Inspector!
- LOG_TRANSITIONS and LOG_TRANSITIONS_INTERNAL
- http://eviltrout.com/2014/08/16/debugging-ember-js.html

Architecture



JavaScript

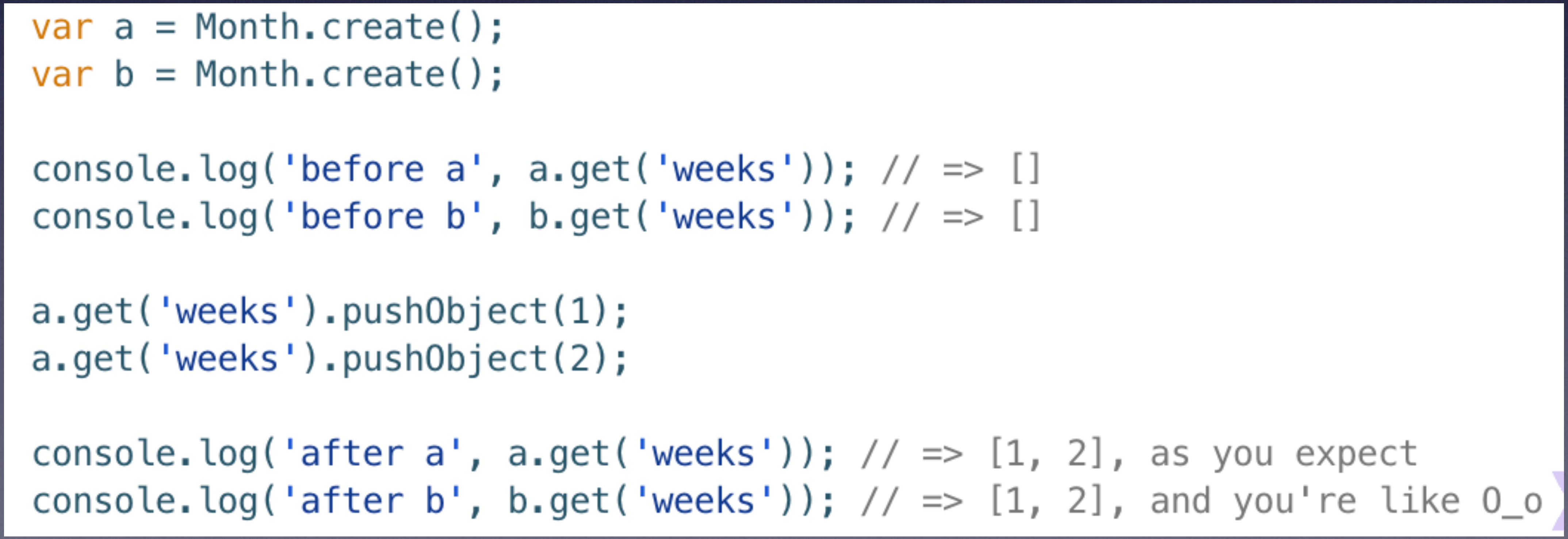
- Some things aren’t real objects, e.g. strings
- Can be confusing sometimes



```
var str = "hello"
undefined
str.foo = 'goodbye'
"goodbye"
str.foo
undefined
```

JavaScript Mistakes

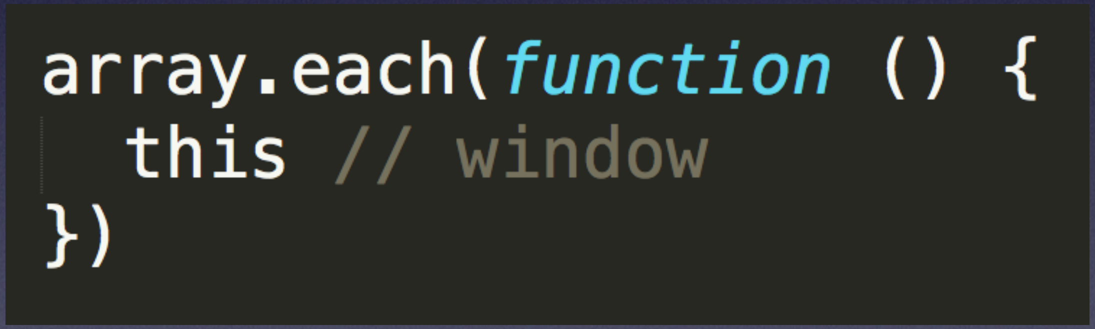
- "Ember Object Self Troll"
- http://reefpoints.dockyard.com/2014/04/17/ember-object-self-troll.html



```
var a = Month.create();
var b = Month.create();

console.log('before a', a.get('weeks')); // => []
console.log('before b', b.get('weeks')); // => []

a.get('weeks').pushObject(1);
a.get('weeks').pushObject(2);

console.log('after a', a.get('weeks')); // => [1, 2], as you expect
console.log('after b', b.get('weeks')); // => [1, 2], and you're like 0_o
```

JavaScript Mistakes

- Function scope and "this"



```
array.each(function () {
  this // window
})
```

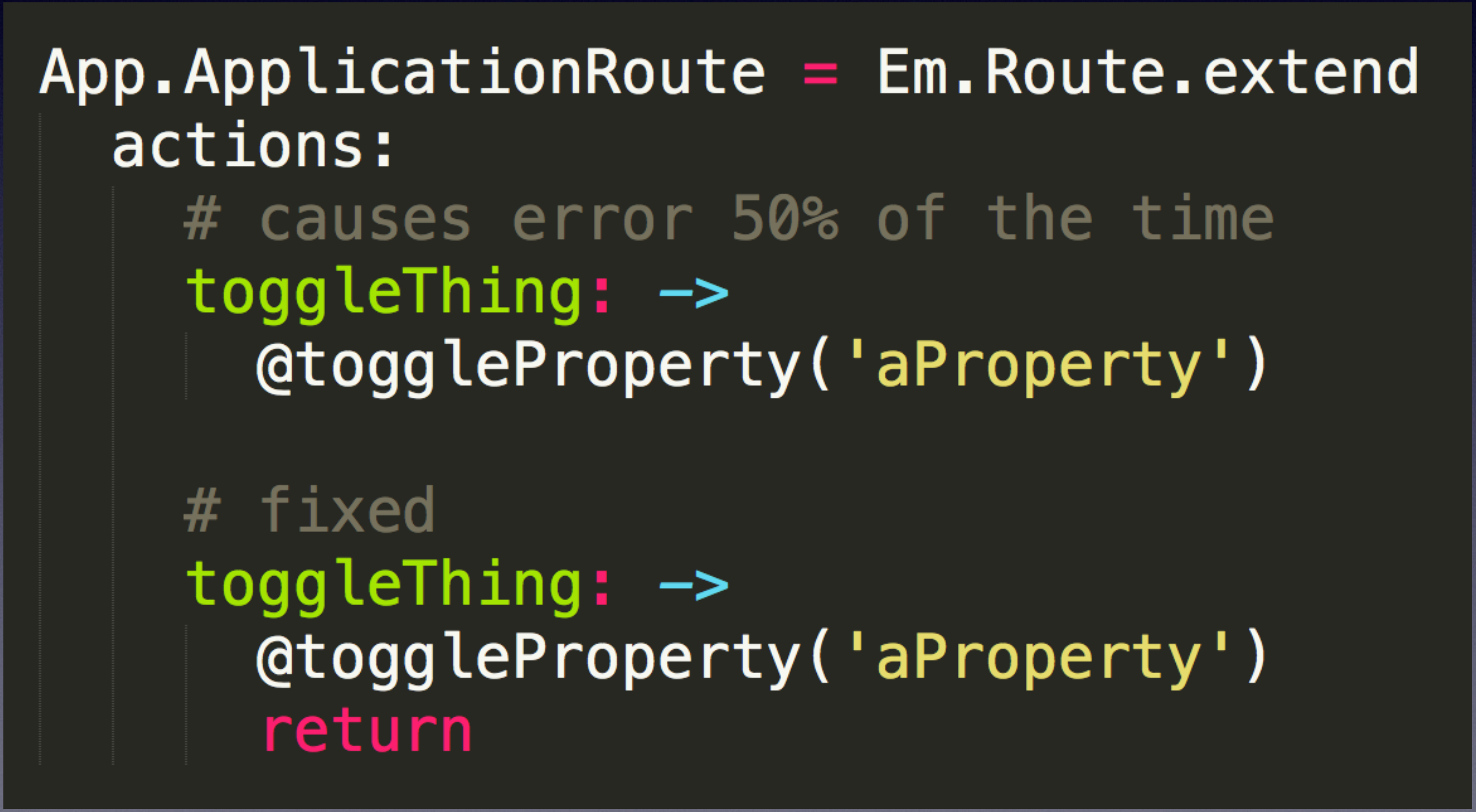
JavaScript Mistakes

- Native array methods e.g. push

```javascript
// will not update templates, bindings etc
array.push('newElement');

// you need this:
array.pushObject('newElement');
```

Coffeescript

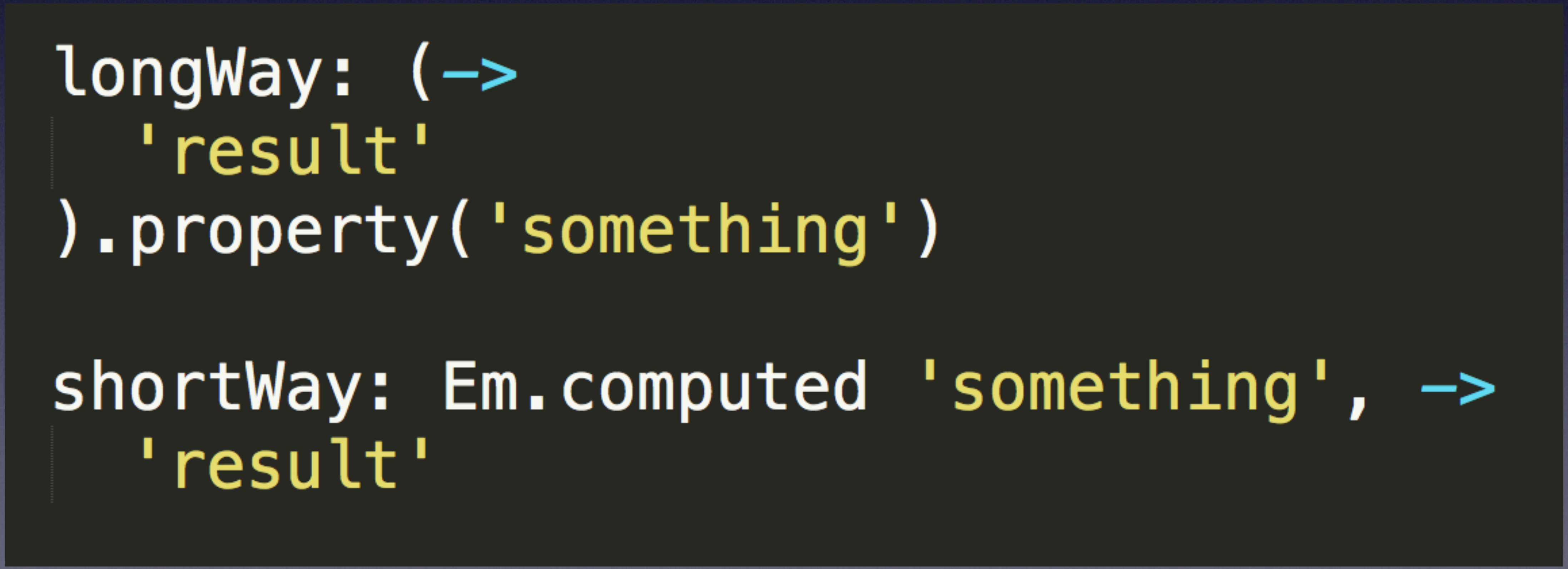
- Action bubbling



```
App.ApplicationRoute = Em.Route.extend
  actions:
    # causes error 50% of the time
    toggleThing: ->
      @toggleProperty('aProperty')

    # fixed
    toggleThing: ->
      @toggleProperty('aProperty')
      return
```

Coffeescript

- Em.computed makes for nicer properties



```coffeescript
longWay: (->
  'result'
).property('something')

shortWay: Em.computed 'something', ->
  'result'
```

https://gist.github.com/alexspeller/8317125

Mistakes asking for help

- Asking the wrong question - asking too broadly
- Not giving a reproducible example
  - Not using a jsbin / jsfiddle
- Noone will show you how to do it “wrong”
- Not saying “I don’t understand”

Thank You!

@alexspeller

emberkit.com