

THE CORNERSTONES OF EMBERJS

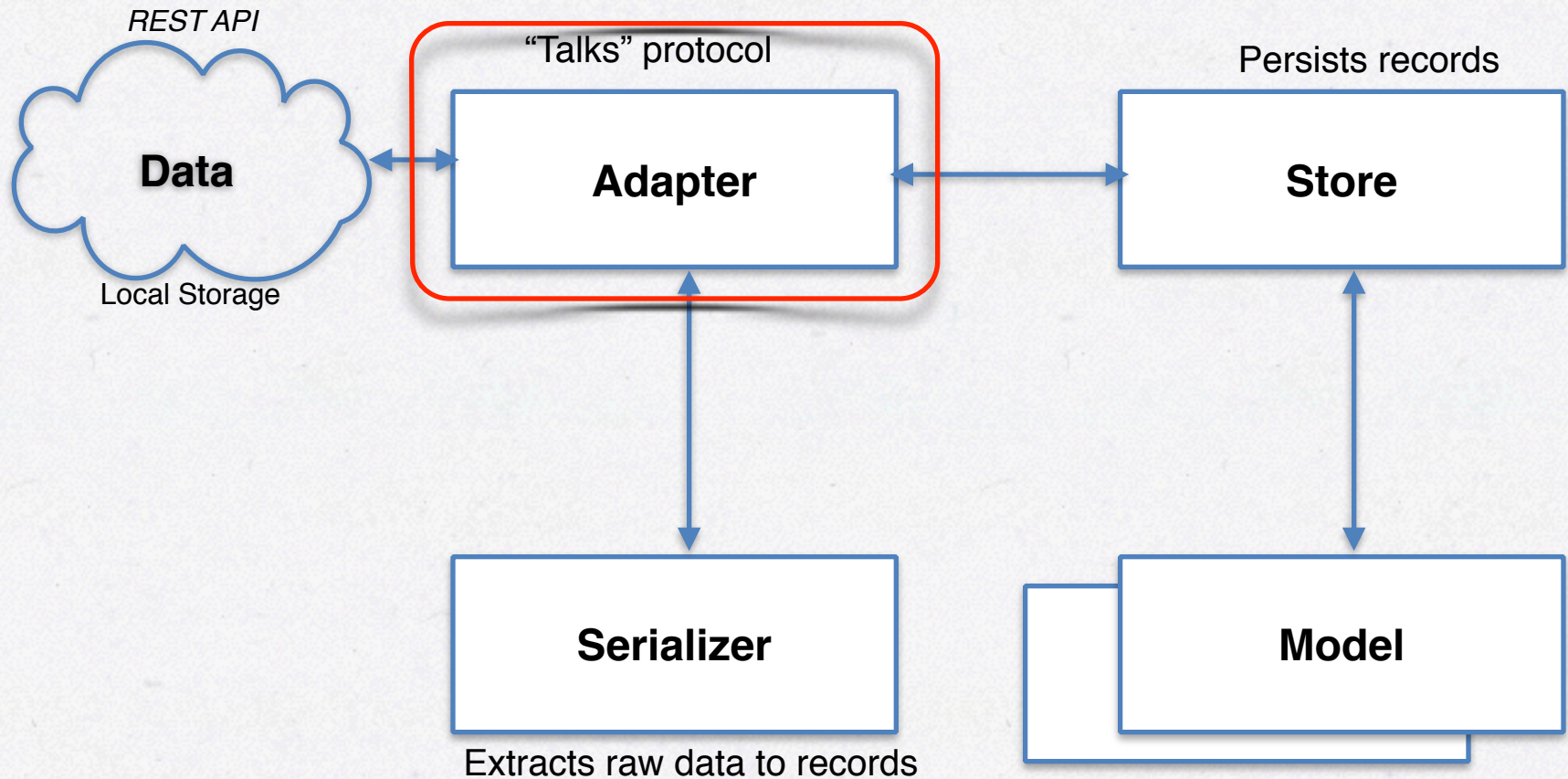
CADEC 2015

| CALLISTAENTERPRISE.SE

EMBER DATA

- Data persistence library
 - —> Only for Ember.js
- Separated code-source from Ember
- Assumes “sane” REST API defaults
- Manages models and their relationships
- Local cache

EMBER DATA - ARCHITECTURE



EMBER DATA - ADAPTER

- Receives requests from a store and translates them to appropriate actions against persistence layer
- Persistence layer examples:
 - HTTP API
 - browser local storage
- Provided adapters:
 - FixtureAdapter
 - RESTAdapter

EMBER DATA - ADAPTER - FIXTURE

- FixtureAdapter
 - records from memory
 - used for development and testing
 - provides all CRUD operations
 - possible to implement entire app with fixtures
- Docs : <http://emberjs.com/api/data/classes/DS.FixtureAdapter.html>

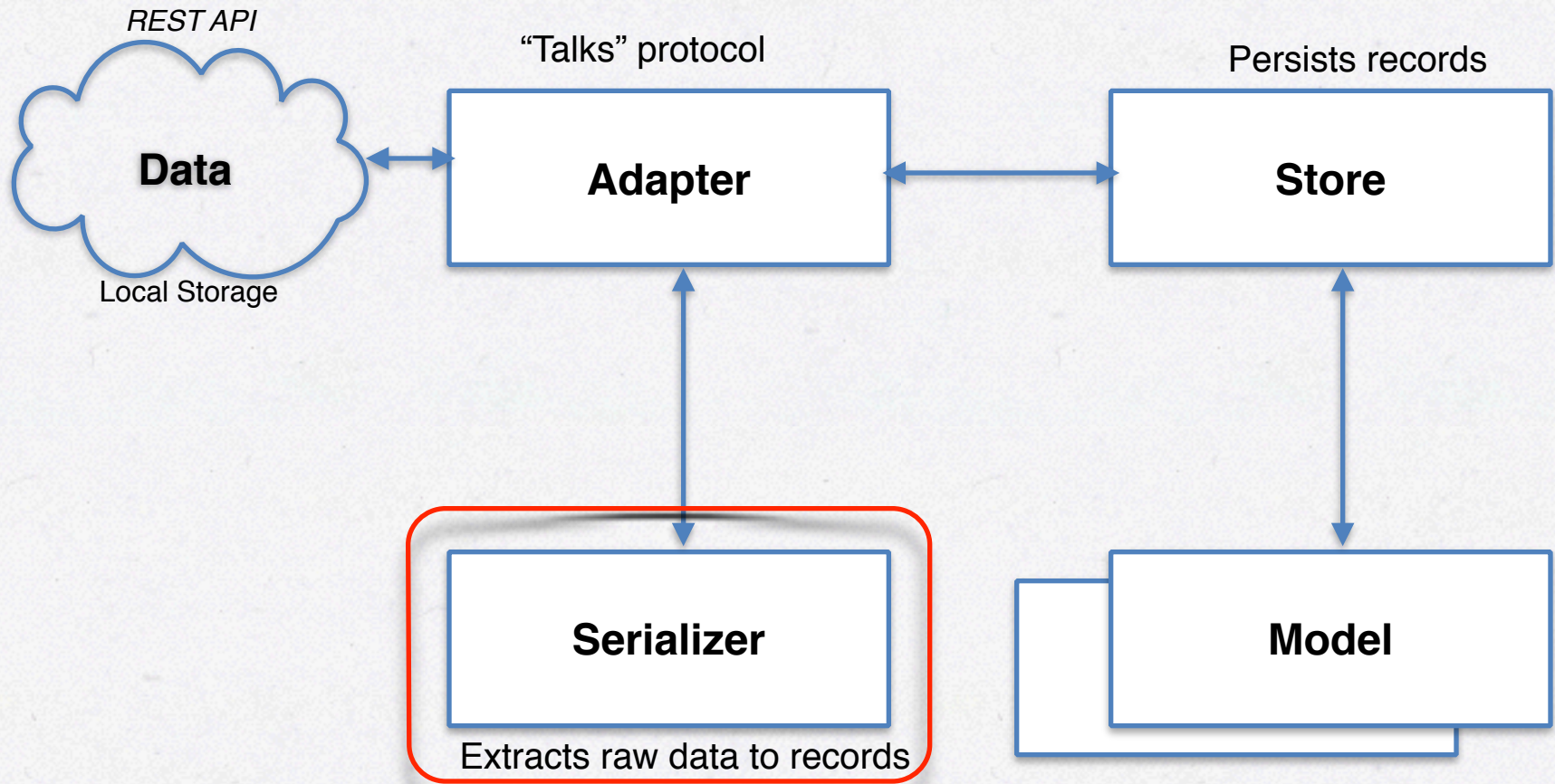
EMBER DATA - ADAPTER - REST

- RESTAdapter
 - allows store to communicate with HTTP server backend
 - expects conventional JSON structure
 - Example JSON for *GET* request *posts/1*:

```
{
  "post": {
    "id": 1,
    "title": "Ember Rocks",
    "body": "Ember is real MVC framework"
  }
}
```

- Docs: <http://emberjs.com/api/data/classes/DS.RESTAdapter.html>

EMBER DATA - ARCHITECTURE



EMBER DATA - SERIALIZER

- Turns raw JSON payload into record object
- Used by Adapter
- When sending to server **serialize**
- When receiving from server **normalize**
- Provided serializers:
 - RESTSerializer
 - <http://emberjs.com/api/data/classes/DS.RESTSerializer.html>

EMBER DATA - SERIALIZER

- Example *RESTSerializer*

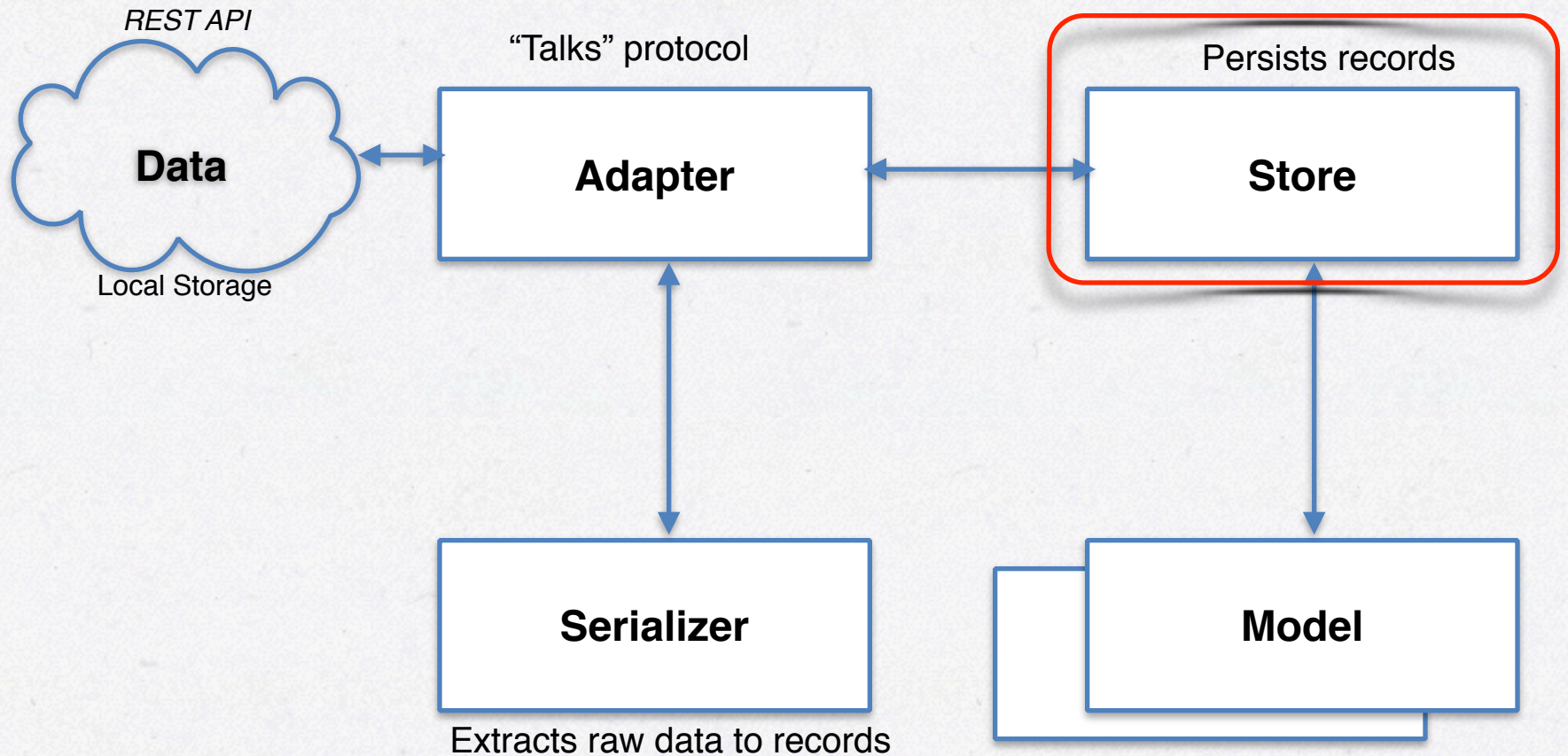
```
import DS from 'ember-data';

export default DS.RESTSerializer.extend({
  serializeHasMany: function(record, json, relationship) {
    var key = relationship.key;
    var payloadKey = this.keyForRelationship ? this.keyForRelationship(key, 'hasMany') : key;

    var relationshipType = record.constructor.determineRelationshipType(relationship);

    if (['manyToNone', 'manyToMany', 'manyToOne'].contains(relationshipType)) {
      json[payloadKey] = record.get(key).mapBy('id');
    }
  }
});
```

EMBER DATA - ARCHITECTURE



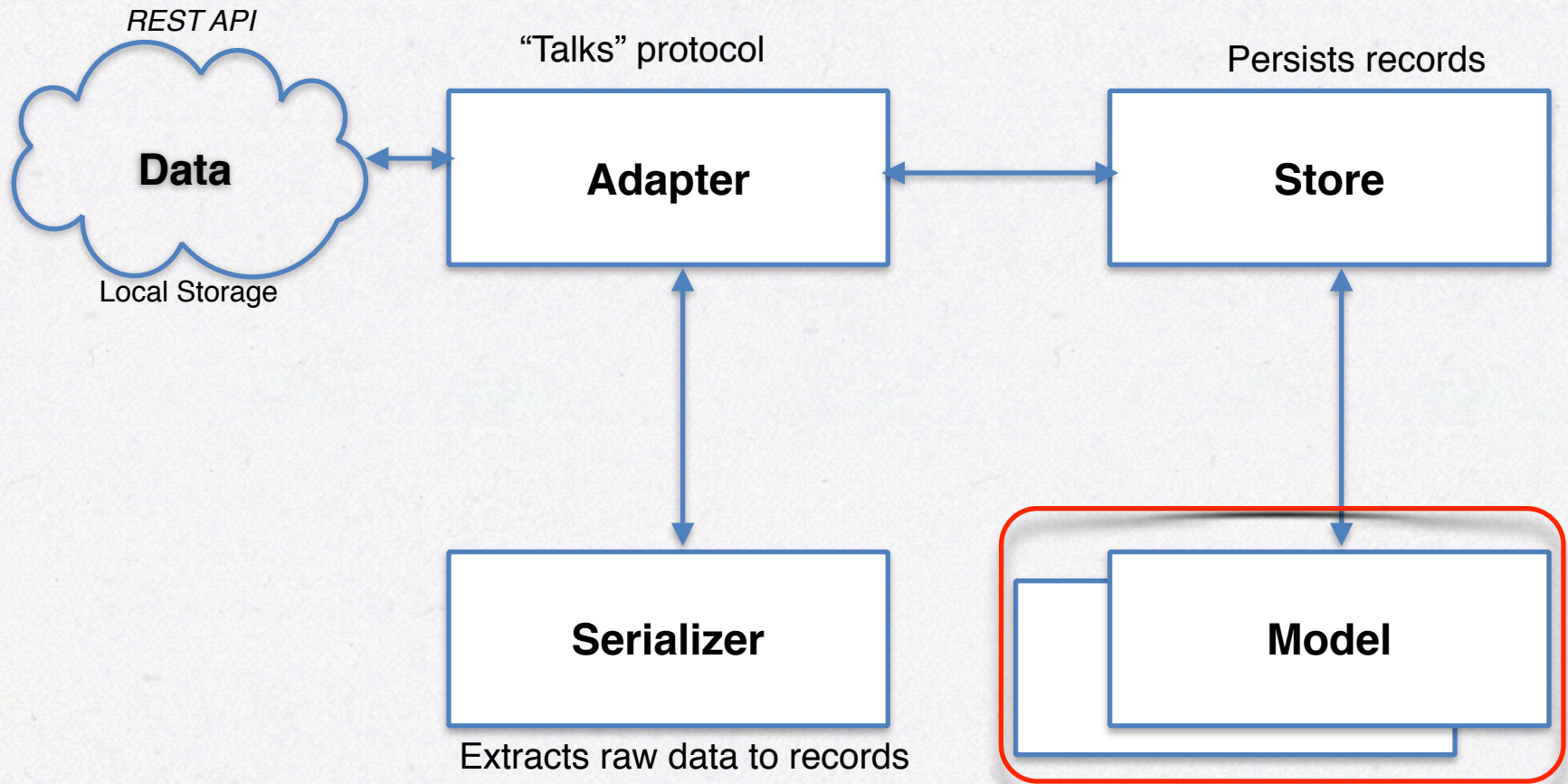
EMBER DATA - STORE

- Singleton instance
- Namespace === *DS*
- Central repository of records
- Functions as local cache
- Responsible for querying and saving of records
- Finders and save return promises
- Example finding a record *post* with id *1*:

```
this.store.find('post', 1).then(function(){  
  // do something useful  
});
```

- Docs: <http://emberjs.com/api/data/classes/DS.Store.html>

EMBER DATA - ARCHITECTURE



EMBER DATA - MODEL

- properties and behaviour of the data
- data types of the properties
- relationships with other objects

```
import DS from "ember-data";

export default DS.Model.extend({
  title: DS.attr('string'),
  body: DS.attr('string'),
  user: DS.belongsTo('user', {async:true}),
  comments : DS.hasMany('comment', {async:true})
});
```

- Docs: <http://emberjs.com/guides/models/defining-models/>

EMBER DATA - MODEL - RELATIONSHIPS

DS.belongsTo for one-to-one

```
User = DS.Model.extend({  
  profile: DS.belongsTo('profile')  
});
```

```
Profile = DS.Model.extend({  
  user: DS.belongsTo('user')  
});
```

EMBER DATA - MODEL - RELATIONSHIPS

DS.belongsTo in combination with ***DS.hasMany*** for **one-to-many**

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

```
Comment = DS.Model.extend({  
  post: DS.belongsTo('post')  
});
```

EMBER DATA - MODEL - RELATIONSHIPS

DS.hasMany for many-to-many

```
Post = DS.Model.extend({  
  tags: DS.hasMany('tag')  
});
```

```
Tag = DS.Model.extend({  
  posts: DS.hasMany('post')  
});
```


EXERCISE 1

- ***GitHub link***

<https://github.com/callistaenterprise/cadec-2015-ember/wiki/Models-Fixtures-and-Adapters>

- ***Time 15 minutes***

ROUTER

- Example of a router mapping:

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

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

Router.map( function() {
  this.resource("posts", function() {
    this.resource("posts.post", {
      path: ":post_id"
    }, function() {
      this.resource("posts.post.comments", {
        path: "comments"
      }, function(){
        this.route('comment', { path: ':comment_id' });
      });
    });
  });
  this.route("new");
});
this.route("login");
this.route("logoff");
});
export default Router;
```

ROUTE

- One *route* per state
- Decides which model to be used
- Default routes:
 - *application* is the container for entire application
 - *index* is the route for the root '/'
- Transitioning
 - *beforeModel*
 - *afterModel*
- Path can be left out if same as template name

```
this.route('new');
```

URL	Route Name	Controller	Route	Template
/	index	IndexController	IndexRoute	index
N/A	posts	PostsController	PostsRoute	posts
/posts	posts.index	PostsController ↳ PostsIndexController	PostsRoute ↳ PostsIndexRoute	posts ↳ posts/index
/posts/new	posts.new	PostsController ↳ PostsNewController	PostsRoute ↳ PostsNewRoute	posts ↳ posts/new

- Docs: <http://emberjs.com/guides/routing/>

ROUTE

- Example *Post* route

```
import Ember from 'ember';

export default Ember.Route.extend({
  model: function(params) {
    return this.store.find('post', params.post_id);
  }
});
```

- Docs: <http://emberjs.com/guides/routing/>

CONTROLLER

- Can act as model decorator or as a proxy
- Template gets all its properties from a controller
- Storage of application properties
- Controller types:
 - ObjectController
 - ArrayController

- Example dependency between *controllers*:

```
export default Ember.Controller.extend({  
  needs: "posts/post"  
});
```

- Docs: <http://emberjs.com/guides/controllers/>

CONTROLLER

- Example *PostController*

```
import Ember from 'ember';

export default Ember.ObjectController.extend({
  isEditing : false,
  actions : {
    edit : function(){
      this.set('isEditing', true);
    },
    done : function(post){
      this.set('isEditing', false);
      post.save();
    },
    cancel : function(post){
      post.rollback();
      this.set('isEditing', false);
      this.transitionToRoute('posts.post', post.get('id'));
    }
  }
});
```

- Docs: <http://emberjs.com/guides/controllers/>

TEMPLATE

- Handlebars templates
 - regular HTML with embedded expressions `{{someProperty}}`
- Default template is `application.hbs`
 - header, footer, and any other decorative content here
- `{{outlet}}`: a placeholder that the router will fill in with the appropriate template, based on the current URL
- Handlebars Helpers
 - `{{log}}` - to output variables to browser's console
 - `{{debugger}}` - to set breakpoints in template
- Navigation between *routes* using `{{link-to}}` Handlebars helper
`<div>{{#link-to 'posts.new'}}New Post{{/link-to}}</div>`
- Docs: <http://emberjs.com/guides/templates/handlebars-basics/>

TEMPLATE

- Example application.hbs:

```
<div id="wrap">
  {{partial 'p-header'}}

  <!-- Page content -->
  <div class="container">
    {{outlet}}
  </div>

  {{partial 'p-header'}}
</div>
```

- Docs: <http://emberjs.com/guides/templates/handlebars-basics/>

TEMPLATE - HELPER

- Example date-time helper:

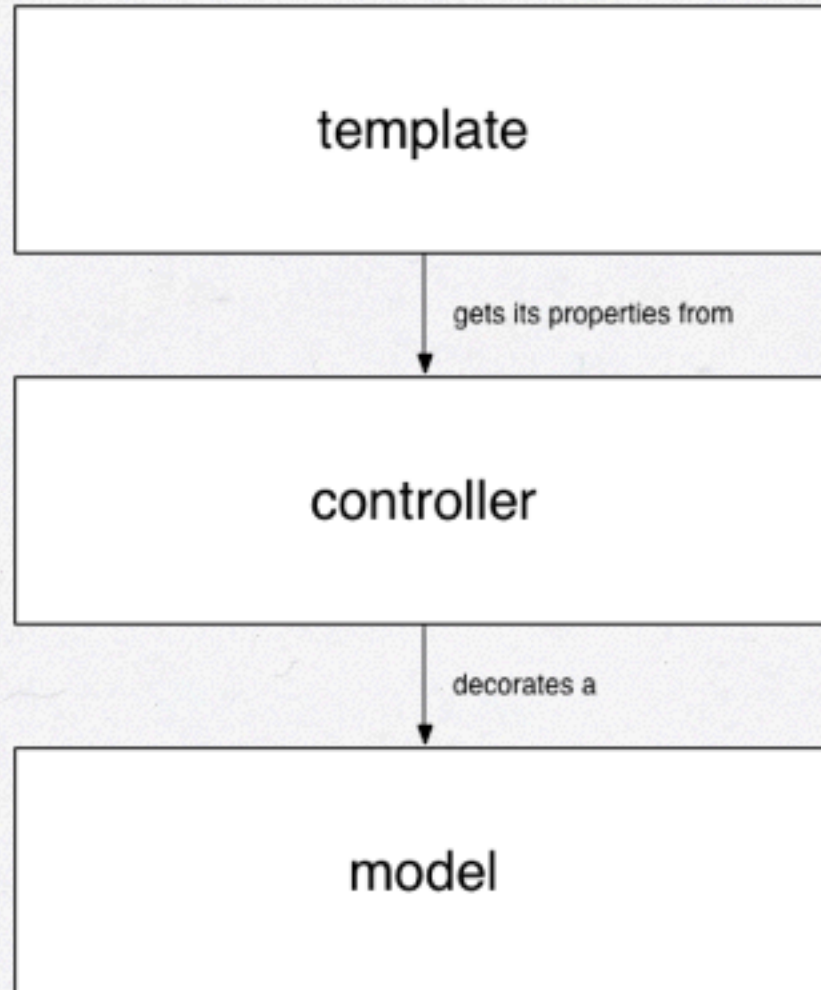
```
import Ember from 'ember';
/* global moment */

export function dateTime(date, format) {
  if (!date) {
    return "";
  }
  if(!format || !(typeof format === 'string' || format instanceof String)){
    return moment(date).format('YYYY-MM-DD');
  } else if(format === 'fromTo'){
    return moment(date).fromNow();
  } else {
    return moment(date).format(format);
  }
  return date;
}

export default Ember.Handlebars.makeBoundHelper(dateTime);
```

- Docs: <http://emberjs.com/guides/templates/handlebars-basics/>

COUPLING: TEMPLATE-CONTROLLER-MODEL



EXERCISE 2

- ***GitHub link***

<https://github.com/callistaenterprise/cadec-2015-ember/wiki/Routes-Templates>

- ***Time 15 minutes***

EXERCISE 3

- ***GitHub link***

<https://github.com/callistaenterprise/cadec-2015-ember/wiki/Routes-Templates-Controllers>

- ***Time 30 minutes***