Installing Ember

Must have Node and NPM installed

1. npm install -g ember-cli
2. navigate to the dir you want ember new my-app
3. cd my-app
4. ember server
5. open in text editor
6. open app dir -> index.html
7. note the {{ … }} tags
   1. ember uses handlebars for templating <http://handlebarsjs.com/>
8. ember isn’t a traditional MVC. You have templates that talk to your model. These know the other exists which is why ember will auto update the template when the model is modified.

* **Expressions**, like {{firstName}}, which take information from the template's model and put it into HTML.
* **Outlets**, which are placeholders for other templates. As users move around your app, different templates can be plugged into the outlet by the router. You can put outlets into your template using the {{outlet}} helper.
* **Components**, custom HTML elements that you can use to clean up repetitive templates or create reusable controls.

<http://guides.emberjs.com/v1.11.0/concepts/core-concepts/>

1. the router is basically your controller.

The **router** translates a URL into a series of nested templates, each backed by a model. As the templates or models being shown to the user change, Ember automatically keeps the URL in the browser's address bar up-to-date.

This means that, at any point, users are able to share the URL of your app. When someone clicks the link, they reliably see the same content as the original user.

A route is an object that tells the template which model it should display.

1. as angular has custom directives, ember has Components, custom HTML elements that you can use to clean up repetitive templates or create reusable controls.
2. Install ember inspector in chrome, click allow access to file URLs
3. <http://guides.emberjs.com/v1.10.0/getting-started/>
4. todo app time
5. -----

<http://code.tutsplus.com/tutorials/getting-into-ember-js--net-30709>

dl ember starter kit

use a json api to get ember-data

naming conventions:

|  |  |
| --- | --- |
| 1  2  3 | App.Router.map( function() {     this.resource( 'employees' );  }); |

I would then name my components, like this:

* **Route object:** *App.EmployeesRoute*
* **Controller:** *App.EmployeesController*
* **Model:** *App.Employee*
* **View:** *App.EmployeesView*
* **Template:** *employees*