Router

Le router EmberJs associe chaque URL connue à un ou plusieurs route handlers.

Un route handler peut :

- Afficher un template.
- Charger un model pour le mettre à disposition d'un template.
- Effectuer une redirection vers une nouvelle route (si par ex l'utilisateur n'a pas l'autorisation d'accès à la route sollicitée).
- Gérer les actions permettant la modification du model ou réalisant une redirection.

Création de routes

Pour générer une route avec ember-cli :

```
ember generate route route-name
```

La commande generate route :

- crée le fichier js correspondant au route handler app\routes\route-name.js
- crée le template app\templates\route-name.hbs
- met à jour le router en ajoutant la route au map
- crée un fichier de test

Exemple de fichier router :

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

Utilisation dans un fichier template avec le helper **{{link-to}}** :

```
{{#link-to "index"}}<img class="logo">{{/link-to}}

<nav>
  {{#link-to "about"}}About{{/link-to}}
  {{#link-to "favorites"}}Favorites{{/link-to}}
</nav>
```

Routes imbriquées

Une route imbriquée permet d'afficher le résultat d'un template dans un template principal.

La route posts/new correspond à l'affichage du formulaire new post, appelé depuis la liste des posts.

Elle peut-être générée par la commande :

```
ember generate route posts/new
```

```
Router.map(function() {
  this.route('posts', function() {
    this.route('new');
  });
});
```

Le template posts.hbs devra dans ce cas faire appel au helper **{{outlet}}** pour insérer le contenu du template **new** :

```
<h1>Posts</h1>
<!-- Display posts and other content -->
{{outlet}}
```

Routes index

A chaque url, ember associe automatiquement une route index implicite :

La configuration suivante du router :

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

est équivalente à :

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

Ember ajoute automatiquement un index à chaque appel de function.

Ainsi, pour une route imbriquée :

```
Router.map(function() {
  this.route('posts', function() {
    this.route('favorites');
  });
});
```

l'équivalent pour ember est :

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

A l'adresse **/posts**, la route active sera **posts.index**, le template **posts/index** sera affiché grâce au helper {{outlet}} dans le template **posts**.

A l'adresse **/posts/favorites**, Ember remplacera {{outlet}} dans le template **posts** avec le contenu du template **posts/favorites**.

Segments dynamiques

Une url avec segments dynamiques possède une ou plusieurs parties variables, introduites par un **:** et suivies par un **identifier**.

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

Nommage des identifiers

Pour faciliter le chargement des models, il est conseillé de respecter la convention de nommage suivante : un identifier devra respecter la norme **:model-name_id**

```
Router.map(function() {
  this.route('photo', { path: '/photo/:photo_id' }, function() {
    this.route('comment', { path: '/comment/:comment_id' });
  });
});
```

Route Model

Une route est responsable du chargement du ou des models qui seront affichés dans le template associé.

Soit le fichier de routage suivant :

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

Le chargement de models est réalisé dans la fonction model() (model hook) du router handler associé :

```
import Route from '@ember/routing/route';

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