# TD n°2

| | |
|---|---|
|  | Tous les exercices utilisent Bootstrap pour la partie CSS.<br>`ember install ember-bootstrap` |
| | Pour passer à la v4 de bootstrap (sass requis) :<br>`npm install --save-dev ember-cli-sass`<br>`ember generate ember-bootstrap --bootstrap-version=4` |
| | pour les fonts :<br>`ember install ember-font-awesome` |

- Bases emberJS
- Computed properties
- Actions

Créer le projet **td2**

## -- Exercice : choix multiples

### -- Objectifs

1. Création de Computed properties
2. Manipulation de Tableaux d'objets
3. Création de helper et de component

### -- Fonctionnalités

1. Lister les produits disponibles
2. Ajouter un ou plusieurs produits disponibles aux produits sélectionnés
3. Ajouter tous les produits disponibles aux produits sélectionnés
4. Retirer un ou plusieurs produits sélectionnés
5. Retirer tous les produits sélectionnés
6. Valider le choix de produits sélectionnés
7. Revenir au choix des produits

### -- Helper/component

Création d'un composant permettant d'afficher une liste HTML composée d'un tableau d'éléments :

Créer le helper **array-contains**

```
ember g helper array-contains
```

```
import Ember from 'ember';

export function arrayContains(params) {
  const [items, value] = params;
  return items.indexOf(value) > -1;
}

export default Ember.Helper.helper(arrayContains);
```

Créer le composant **multi-select** :

```
ember g component multi-select
```

Composant modifié le 13/02/2018 18h00

```
import Component from '@ember/component';
import Ember from 'ember';

export default Component.extend({
  tagName: 'select',
  classNames: ['form-control'],
  selectedIds:[],
  attributeBindings: ['multiple','size'],

  multiple: true,
  size: 10,
  selectedElements: Ember.computed('selectedIds.[]', function() {
    return this.get('selectedIds').map((id) => {
      return this.get('elements').findBy('id', id);
    });
  }),
  change(event){
    const selectedIds = Ember.$(event.target).val();
    this.set('selectedIds', selectedIds || []);
  },
  doubleClick(event){
      const selected = Ember.$(event.target).val();
      this.sendAction('dblClick',selected,this.get('id'));
  }
});
```

Template :

```
  {{#each elements as |elm|}}
    <option value={{elm.id}} selected={{array-contains selectedIds
elm.id}}>{{elm.title}}</option>
  {{/each}}
```

## -- Classes/Routes

| | |
|---|---|
| **Classe** | **Listes** |
| | dispoItems<br>Tableau des produits disponibles |
| | includedItems<br>Tableau des produits à inclure |
| | selectedDispoItemsIds<br>Tableau des ids des éléments sélectionnés dans la liste des produits disponibles |
| | selectedIncludedItemsIds<br>Tableau des ids des éléments sélectionnés dans la liste des produits à inclure |
| | selectedDispoItems*<br>Tableau des éléments sélectionnés dans la liste des produits disponibles (* sur selectedDispoItemsIds.[]) |
| | selectedIncludedItems*<br>Tableau des ids des éléments sélectionnés dans la liste des produits à inclure (* sur selectedIncludedItemsIds.[]) |
| **actions** | addToIncluded()<br>Ajoute les produits disponibles en surbrillance (**selectedDispoItems**) à la liste des produits à inclure (**includedItems**) |
| | addAllToIncluded()<br>Ajoute tous les produits disponibles (**dispoItems**) à la liste des produits à inclure (**includedItems**) |
| | removeFromIncluded()<br>Retire les produits sélectionnés en surbrillance (**selectedIncludedItems**) pour les remettre dans la liste des produits disponibles (**items**) |
| | RemoveAllFromIncluded()<br>Retire tous les produits à inclure (**includedItems**) pour les remettre dans la liste des produits disponibles (**items**) |

Remarque : Il est possible de factoriser les actions en ajoutant uniquement une action dont le prototype est le suivant :

```
addTo:function(source,destination,items){...}
```

Il faudra dans ce cas passer les paramètres nécessaires à l'action **addTo** dans le template.

**dispoItems** : à intégrer au model hook

```
[
        {
            "id": "1",
            "url":
"https://tutorialzine.com/2018/01/what-every-developer-should-know-about-cryptocurr
ency",
            "title": "What Every Developer Should Know About Cryptocurrency",
            "image":
"https://tutorialzine.com/media/2018/01/everything-you-should-know-cryptocurrency.p
ng"
        },
        {
            "id": "2",
            "url":
"https://tutorialzine.com/2018/01/15-interesting-javascript-and-css-libraries-for-j
anuary-2018",
            "title": "15 Interesting JavaScript and CSS Libraries for January
2018",
```

```
            "image":
"https://tutorialzine.com/media/2018/01/15-interesting-js-and-css-libraries-2018.pn
g"
        },
        {
            "id": "3",
            "url":
"https://tutorialzine.com/2017/12/the-best-javascript-and-css-libraries-for-2017",
            "title": "The Best JavaScript and CSS Libraries for 2017",
            "image":
"https://tutorialzine.com/media/2017/12/the-best-javascript-and-css-libraries-for-2
017.png"
        },
        {
            "id": "4",
            "url": "https://tutorialzine.com/2017/10/getting-started-with-graphql",
            "title": "Getting Started With GraphQL",
            "image":
"https://tutorialzine.com/media/2017/10/getting-started-graphql.png"
        },
        {
            "id": "5",
            "url":
"https://tutorialzine.com/2017/11/9-free-icon-packs-for-web-developers",
            "title": "9 Free Icon Packs For Web Developers",
            "image": "https://tutorialzine.com/media/2017/10/9-free-icon-packs.png"
        },
        {
            "id": "6",
            "url":
"https://tutorialzine.com/2018/01/10-free-programming-books-you-should-read-in-2018
",
            "title": "10 Free Programming Books You Should Read in 2018",
            "image":
"https://tutorialzine.com/media/2018/01/10-free-programming-books-you-should-read.p
ng"
        },
        {
            "id": "7",
            "url":
"https://tutorialzine.com/2016/12/the-languages-frameworks-tools-you-should-learn-i
n-2017",
            "title": "The Languages and Frameworks You Should Learn in 2017",
            "image":
"https://tutorialzine.com/media/2016/12/the-languages-and-frameworks-you-should-lea
rn-in-2017.png"
        }
    ]
```

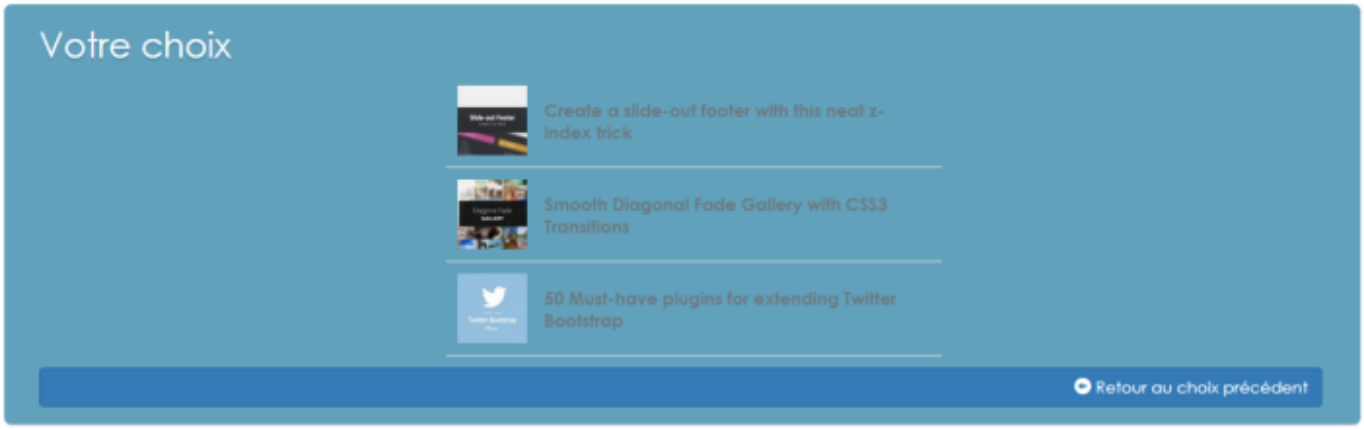

## -- Vues

### -- Interface

**ex1/step1 :**




**ex1/step2 :**




**-- Logique applicative / comportement de l'interface**

- Les 4 boutons d'ajout ou de retrait des services à inclure s'activent ou se désactive (disabled) en fonction du contenu des listes et/ou des éléments sélectionnés
  - bouton ▶| : ajoute le(s) élément(s) sélectionnés de la liste de gauche à la liste de droite
  - bouton ▶▶| : ajoute tous les éléments sélectionnés de la liste de gauche à la liste de droite
  - bouton |◀ : retire le(s) élément(s) sélectionnés de la liste de droite et les place dans la liste de gauche
  - bouton |◀◀ : retire tous les éléments sélectionnés de la liste de droite et les place dans la liste de gauche
- Un double clic sur l'un des éléments des 2 listes ajoute ou retire l'élément et le passe dans la liste associée
- Le nombre de produits inclus se met à jour en fonction du nombre d'éléments dans la liste de droite
- Le passage à l'étape suivante n'est possible que si la liste de droite n'est pas vide
  - le bouton “Passer à l'étape suivante” affiche la vue suivante (en passant step à 2)
  - la vue affiche la liste des éléments ajoutés dans les éléments à inclure

From:
http://slamwiki2.kobject.net/ - **SlamWiki 2.1**

Permanent link:
**http://slamwiki2.kobject.net/richclient/emberjs/td2?rev=1518569705**

Last update: **2019/08/31 14:38**