

TD n°1

	Tous les exercices utilisent Bootstrap pour la partie CSS.
---	--

-- Création du repository git

1. Créer un dossier **ember-tds** ;
2. Publier **ember-tds** sur github en tant que nouveau repository ;
3. Ajouter **jcheron** à la liste des **colaborators** de ce projet ;
4. Publier (commit and push) régulièrement sur github.

-- Exercice : application Note

-- Objectifs

1. Créer un projet
2. Créer des routes
3. Créer/manipuler les templates
4. Mettre en oeuvre le Data-binding

-- Fonctionnalités

1. Saisir une note (message textuel) et afficher le nombre de caractères restants (le message est limité à 100 caractères saisis)
2. Enregistrer (côté client en JS)
3. Effacer le contenu
4. Afficher les messages d'info (sauvegarde, modification...)
5. Gérer les changements de classe CSS sur l'affichage d'info

-- Application

Contrôleur	ex1 (app/controllers/ex1.js)
	MAX nombre maximum de caractères dans la note (100)
	noteContent contenu de la note
	message message affiché
	noteCount nombre de caractères dans la note (*)
	messageEmpty vrai si le message est vide, faux sinon (*)
	save() Pseudo enregistrement (Affiche enregistrement + contenu de la note)
	clear() vide noteContent
	closeAlert() ferme l'alerte affichant le message

(*) computed property

-- Vue

-- helpers handlebars utilisés

- {{expression}} voir [expressions](#)
- {{textarea ...}} voir [textarea](#)
- {{action...}} voir [actions](#)

-- Interface



-- Logique applicative / comportement de l'interface

- sur saisie dans la zone **messageNote** (textarea), le nombre de caractères restants est indiqué dans la zone **status**
 - la zone **info** apparaît et indique “note modifiée” dès que **messageNote** est modifié
 - Le style de la zone **status** passe à **<fc orange>alert-warning</fc>** si le nombre de caractères restant est inférieur à 50, et à **<fc red>alert-danger</fc>** s'il est inférieur à 20

- Sur enregistrement (à condition que le message ne soit pas vide) :
 - la zone **info** affiche "note sauvegardée" et sont style passe à **<fc #008000>alert-success</fc>**
- Sur effacement (à condition que le message ne soit pas vide) :
 - la zone **info** disparaît

-- Test en ligne

Tester l'appli **Note**

-- Exercice : Choix de services

-- Objectifs

1. Créer des routes avec model
2. Utiliser des helpers handlebar
3. Mettre en oeuvre le Data-binding
4. Utiliser les tableaux

-- Fonctionnalités

1. Sélectionner/désélectionner des services
2. Calculer le montant dû
3. Afficher le nombre de services sélectionnés

-- Application/Contrôleurs

Application	ServicesApp (app/servicesApp.js)
Contrôleur	ServicesController (app/servicesController.js)
	services
	Tableau des services disponibles défini en JSON
	total()
	Calcul le total des services actifs
	toggleActive()
	Change le statut d'un service

Services : à intégrer dans le hook model()

```
[
  {
    "name": "Web Development",
    "price": 300,
    "active": true
  }, {
    "name": "Design",
    "price": 400,
    "active": false
  }, {
    "name": "Integration",
    "price": 250,
```

```
    "active":false
  },{
    "name": "Formation",
    "price": 220,
    "active":false
  }
]
```

Ressources :

- [Fichier de zone Fr pour angular](#)

-- Vue

-- helpers utilisées

- Expressions {{expression}}

-- Interface

Services	
Choisissez les éléments à inclure 2 services sélectionnés	
Web Development	300,00 €
Design	400,00 €
Integration	250,00 €
Formation	220,00 €
Total:	700,00 €

-- Logique applicative / comportement de l'interface

- Le service **Web development** est sélectionné par défaut
- La classe css d'un service sélectionné est égale à **active**
- La sélection/dé-sélection met à jour l'affichage du nombre de services sélectionnés, ainsi que le **total**

-- Ajout code promo

- La saisie d'un code promo permet d'appliquer un taux de réduction au montant total (si la case est cochée et le code valide)
- La liste des codes est fournie au format JSON, dans un fichier, qu'il faudra charger via le service [\\$http](#) (à injecter dans le contrôleur)

```
{
  "B22":0.05,
  "AZ":0.01,
  "UBOAT":0.02
}
```

The screenshot shows a mobile application interface titled 'Services'. It features a list of services with their respective prices. The services are: Web Development (300,00 €), Design (400,00 €), Integration (250,00 €), and Formation (220,00 €). The total price is 920,00 €. Below the list, there is a checkbox labeled 'UBOAT' which is checked. The 'Remise' (discount) is 18,40 €, and the 'Total avec remise' (total with discount) is 901,60 €.

Service	Prix
Web Development	300,00 €
Design	400,00 €
Integration	250,00 €
Formation	220,00 €
Total:	920,00 €

☒ UBOAT

Remise : 18,40 €
Total avec remise : 901,60 €

-- Test en ligne

Tester l'appli [Services](#)

From:
<http://slamwiki2.kobject.net/> - SlamWiki 2.1

Permanent link:
<http://slamwiki2.kobject.net/richclient/emberjs/td1?rev=1516317045>

Last update: 2019/08/31 14:38



