2025/10/19 13:42 1/7 TD n°1

TD n°1



Tous les exercices utilisent Bootstrap pour la partie CSS. ember install ember-bootstrap

- ECMAScript6 (2015)
- Bases emberJS

- 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.

- Création du projet ember

Créer le projet ember td1

- Exercice: application Note

- Objectifs

- 1. Créer un projet
- 2. Créer des routes
- 3. Créer/manipuler les templates
- 4. Créer des classes et instancier des objets
- 5. Mettre en oeuvre le Data-binding avec computed properties

- Fonctionnalités

- 1. Saisir une note (message textuel) et afficher le nombre de caractères restants (le message est limité à 100 caractères saisis)
- 2. Enregistrer (pseudo enregistrement)
- 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

Fichier app/routes/ex1.js

Last update: 2019/08/31 14:21

ember g route ex1

	Note (EmberObject.extend({}))
	content contenu de la note (par défaut : 'Entrez votre texte')
	MAX nombre maximum de caractères dans la note (par défaut: 100)
Classe	info message temporaire affiché dans la zone info (par défaut: '')
	size retourne le nombre de caractères dispo dans la note (* sur content)
	style retourne le style css à appliquer sur la zone de message (* sur size)
	alertVisible Détermine si l'alert info est visible (* sur info)
Route	ex1
Noute	model()model hook, doit retourner une instance de Note

Fichier app/controllers/ex1.js

ember g controller ex1

Controller	er ex1	
	save() Pseudo enregistrement (Affiche enregistrement + contenu de la note)	
	clear() vide noteContent	

(*) computed property

- Vue

- helpers handlebars utilisés

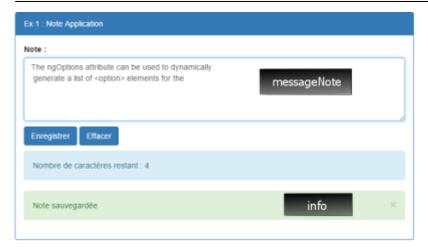
- {{expression}} voir expressions
- {{textarea ...}} voir textarea
- {{action...}} voir actions

- Interface

http://slamwiki2.kobject.net/

Printed on 2025/10/19 13:42

2025/10/19 13:42 3/7 TD n°1



- Logique applicative / comportement de l'interface

- sur saisie dans la zone **messageNote** (textarea), le nombre de caractères restants est indiqué dans la zone **status**
 - o 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) :
 - o la zone **info** disparaît

- Test en ligne

Tester l'appli Note

- Exercice : Choix de services

- Objectifs

- 1. Créer des routes avec model
- 2. Utiliser/créer 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

Fichier app/route/ex2.js

ember g route ex2

```
| Services (EmberObject.extend({}))
| services | tableau des services (par défaut: [])
| Classe | CountActive | Retourne le nombre de services actifs (* sur services.@each.active) | sumActive | Retourne le montant total correspondant aux services actifs (* sur services.@each.active)
| Route | ex2 | model()model hook, doit retourner une instance de Services initialisée avec le tableau ci-dessous
```

Fichier app/controllers/ex2.js

```
ember g controller ex2
```

Controller	ex2
Actions	toggleActive(service) Change le statut du service

(*) computed properties

A utiliser

- filterBy pour filtrer les services en fonction de leur propriété active
- forEach pour le parcours des services

Services : à passer en paramètre de l'instance de la classe Services

```
Services.create({services: ... });
```

```
[
    {
        "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
    }
]
```

2025/10/19 13:42 5/7 TD n°1

- Vue

- helpers utilisés

- Expressions {{expression}}
- {{#each}}{{/each}}
- {{#if}}{{/if}}
- {{input}}

- helpers à créer

Nom	Paramètres	Rôle
plural	count, zero, one, other	Formate une expression correctement en fonction de count
format-currency	value, symbol	Formate value en monétaire (2 déc + symbole € par défaut)
format-percent	value	Formate value en pourcentage (x100 + symbole %)

- Interface



-- 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

Last update: 2019/08/31 14:21

- 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 en JS, elle sera a ajouter au model hook

```
{
    "B22":0.05,
    "AZ":0.01,
    "UB0AT":0.02
}
```



- Test en ligne

Tester l'appli Services

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

Permanent link:

http://slamwiki2.kobject.net/richclient/emberjs/td1

Last update: 2019/08/31 14:21



2025/10/19 13:42 7/7 TD n°1