# TD n°1

| | |
|---|---|
| ANGULARJS | Tous les exercices utilisent Bootstrap pour la partie CSS.<br>URL : https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css |

- Réalisation du TD n°0
- Bases javascript
- Concepts de base AngularJS

Créer un dossier différent pour chaque exercice, dans le même projet

## -- Exercice : application Note

### -- Objectifs

1. Créer un module et un contrôleur
2. Utiliser des directives Angular
3. 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/Contrôleurs

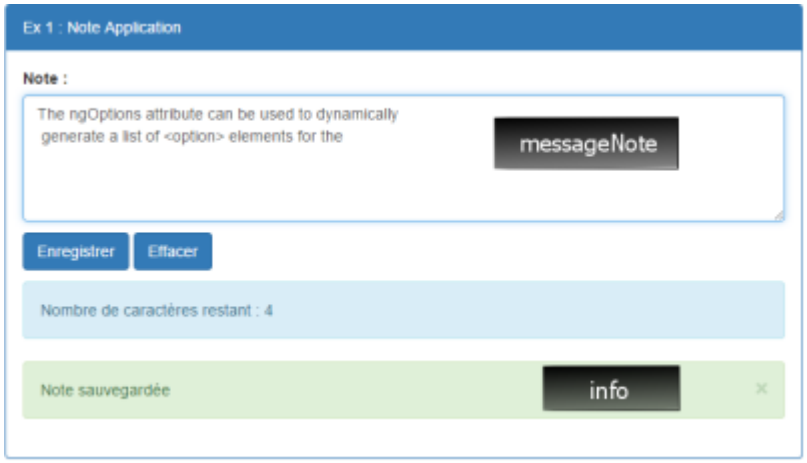
| | |
|---|---|
| **Application** | **NoteApp** (app/noteApp.js) |
| **Contrôleur** | **NoteController** (app/noteController.js) |
| | messageNote<br>variable stockant le contenu de la note |
| | info<br>message affiché (modifié, sauvegardé...) |
| | save()<br>Met à jour le modèle (messageNote) |
| | clear()<br>vide messageNote |
| | count()<br>Calcule le nombre de caractères restants (sur 100) |

### -- Vue

#### -- Directives utilisées

- ng-app
- ng-change
- ng-controller
- ng-click
- ng-model
- ng-bind
- ng-class

#### -- Interface




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

- sur saisie dans la zone **messageNote** (textarea), le nombre de caractères restants est indiqué
  - la zone **info** apparaît et indique “note modifiée” dès que **messageNote** est modifié, avec le style **<fc orange>alert-warning</fc>**
  - Le style de la zone **info** passe à **<fc red>alert-danger</fc>** si le nombre de caractères restant est inférieur à 10
- 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

## -- Exercice : Choix de services

### -- Objectifs

1. Créer un module et un contrôleur

2. Utiliser des directives Angular
3. Mettre en oeuvre le Data-binding

## -- 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<br>Tableau des services disponibles défini en JSON |
| | total()<br>Calcul le total des services actifs |
| | toggleActive()<br>Change le statut d'un service |

**Services** : à intégrer dans le contrôleur

```
[
    {
        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

### -- Directives utilisées

- ng-app

- ng-controller
- ng-repeat
- ng-class
- ng-pluralize
- Expressions {{expression}}

**Filtres** : currency

**-- 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**

**-- 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)

**-- Test en ligne**