# SimplonBoulogne AngularJS

## MVW & Requêtes serveur

# Sommaire

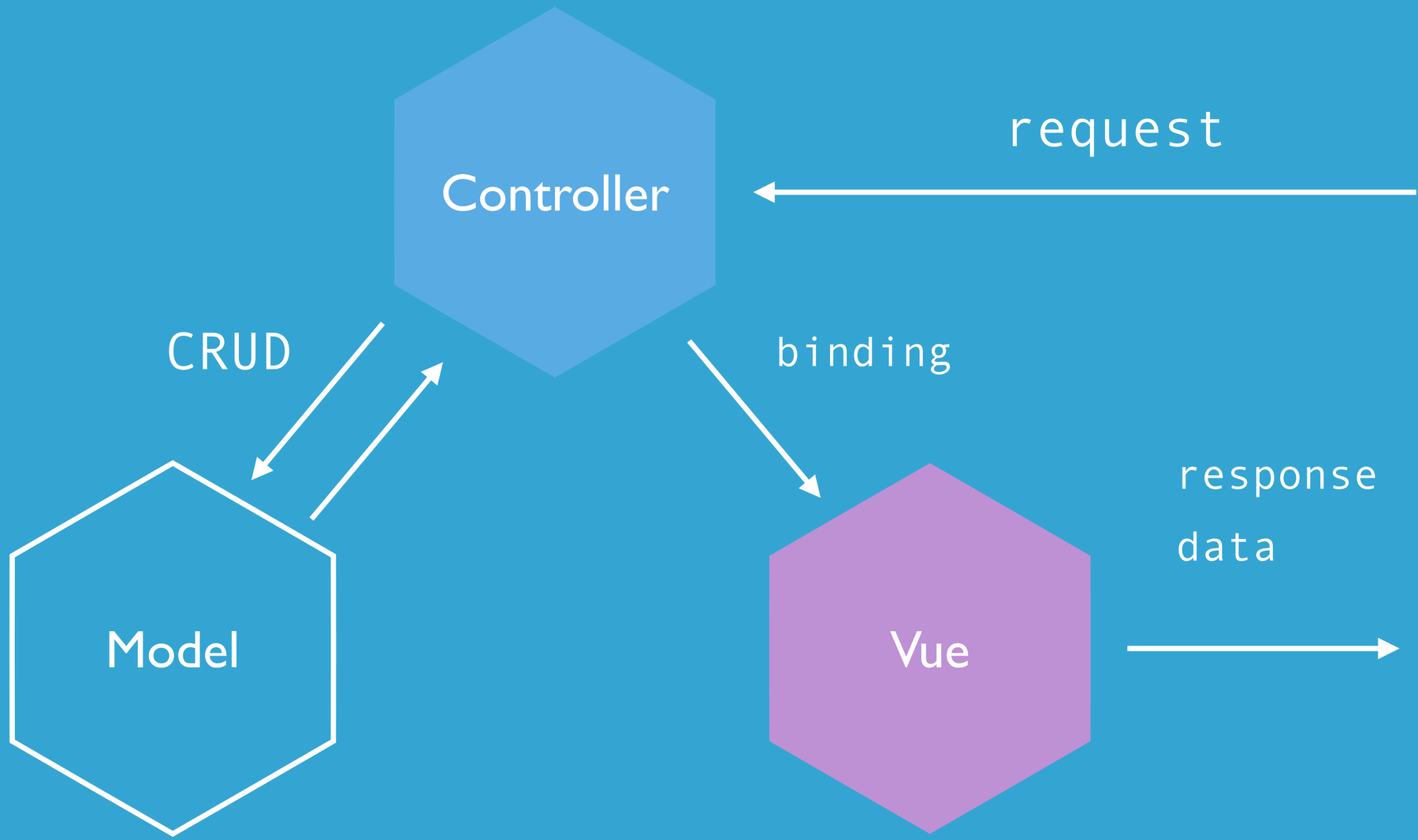
- Quelques rappels
- Créer une directive
- Les filtres
- L’objet $http - requête serveur

# Rappels

AngularJS se décompose sous une architecture MVC, certains parlent également d'architecture MVW (Model-View-Whatever) ou MVVM (Model-Vue-VueModel)

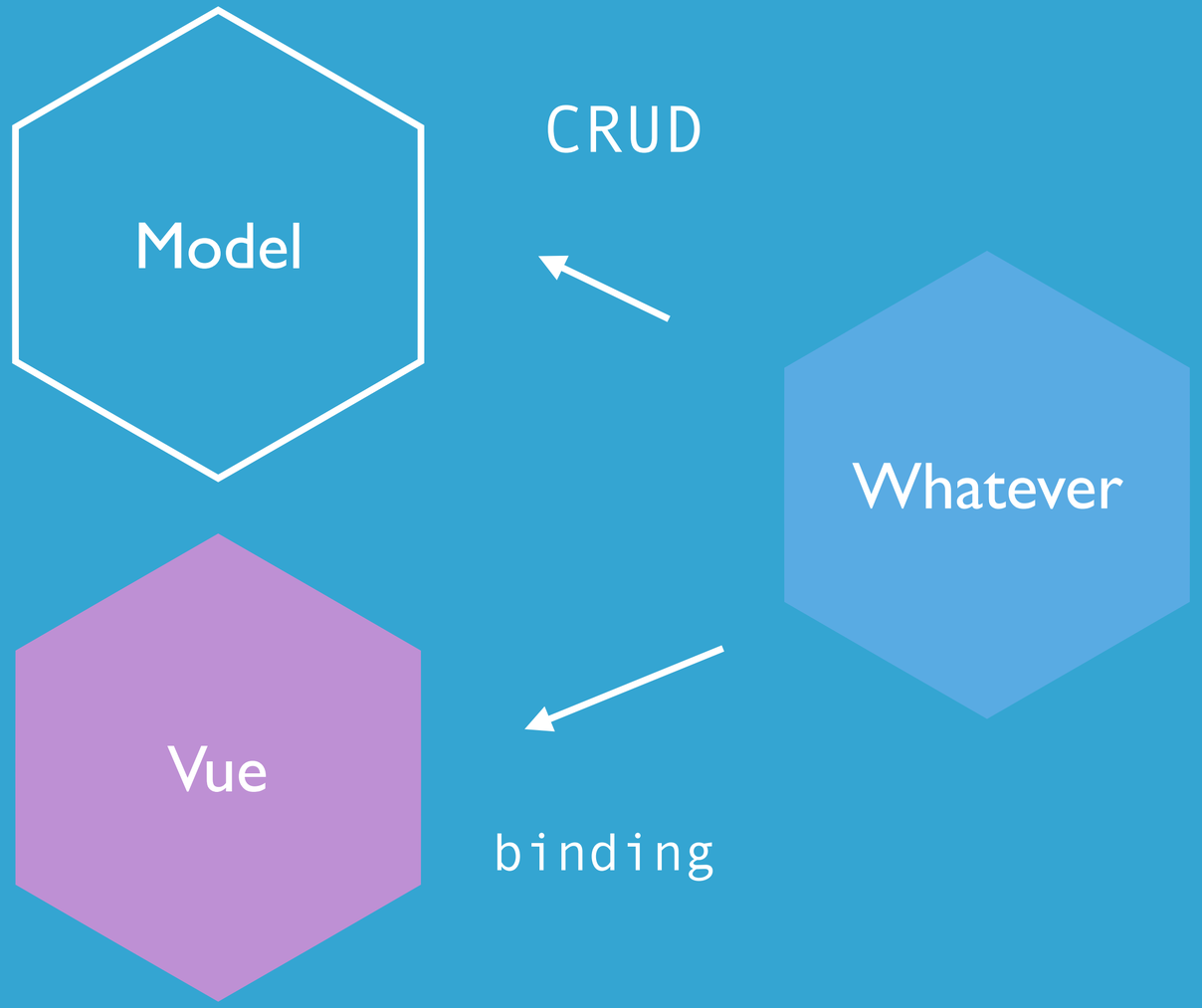
Il offre aussi la possibilité de réaliser des applications WEB mono-pages (SPA pour Single Page Applications)

# Rappel : Modèle MVC



# Rappel :

# Modèle MVW




- Contrôleur
- Directives
- Tests unitaires
- ...

# Rappels - $scope

La variable scope sert à attacher des variables pour les utiliser dans la vue qui utilise le controller.

```
<h1>Hello {{name}}!</h1>
<input type="text" ng-model="name" />
```

# Créer sa propre directive

Vous pouvez créer votre propre directive toujours pour améliorer la lisibilité de votre code et obtenir une architecture plus agréable.

```
<div class="list" ng-controller="mainCtrl">
…
</div>
```

templates/todos.html

# Créer sa propre directive

Pensez à placer vos directives définies dans le dossier « directives »

utilise le template créé

```
angular.module("todoListApp")
.directive('todos', function() {
  return {
    templateUrl: 'templates/todos.html',
    controller: 'mainCtrl',
    replace: true //remplace le tag <todos> par le template
  }
})
```

directives/todos.js

# Créer sa propre directive

Et voilà le travail, un code tout beau tout propre! On sait par exemple que cette seule ligne représente toute notre liste de tâche qui joue avec le contrôleur assigné dans sa directive.

```
<body ng-app="todoListApp">
  <h1>Todo app with Angularjs</h1>

  <todos></todos>

</body>
```

index.html

# Filtres

Permet de spécifier des critères spécifiques dans un élément. Bien évidemment les filtres se placent donc côté vue.

```
<div ng-repeat="text in texts">
  <span>{{ text | lowercase }}</span>
  <span>{{ text | uppercase }}</span>
</div>
```

# Filtres - Exemples

| | |
|---|---|
| filter | Fonctionne seulement avec un tableau, permet d'affiner le résultat |
| orderBy | permet de trier l'affichage des résultats d'un tableau |
| lowercase / uppercase | minuscules / MAJUSCULES |
| json | Formate un objet en un string JSON |
| currency | Formate un nombre en devise |
| date | Formate avec une date spécifique |

# Filtres - filter

Vous pouvez également filter un résultat directement côté client avec un ng-model !

```
<p><input type="text" ng-model="test"></p>

<ul>
  <li ng-repeat="name in names | filter : test">
    {{ name }}
  </li>
</ul>
```

# $http

$http est un objet XMLHttpRequest permettant de faire une requête à un serveur et d'obtenir une réponse. NB : Utilisé aussi pour l'utilisation d'API.

L'utilisation de fonctions callback sont nécessaires, elles sont appelés dès que nous obtenons une réponse du serveur (GET 200, ou encore une erreur 404 ou autre)

# $http

```
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope, $http) {
    $http.get("http://www.w3schools.com/angular/customers.php")
    .then(function(response) {
        //Fonction .then() pour le succès
        $scope.content = response.data;
    }, function(response) {
        //Une deuxième fonction pour attraper les erreurs
        $scope.content = "Something went wrong..";
    });
});
```

Pour afficher les erreurs serveurs, on ajoute une nouvelle fonction.

# Un p’tit récap’

- Un filtre se place dans …
- La définition d’une directive maison doit contenir … , … , …
- En fin de compte, $http est l’équivalent de …

# Challenge 2

Vous allez dans un premier temps créé votre propre directive <products> pour afficher tous les produits que vous aviez lors du premier challenge. Dans cette directive l'utilisateur peut faire une recherche par nom et avoir le résultat.

Pour les plus coriaces, nous allons créé une API externe, donc simuler une simple réponse en JSON émanant d'un fichier api.php qui récupère une liste de produits à partir d'une base de donnée PDO (libre pour les datas)

Ensuite vous allez dans votre contrôleur faire une requête serveur qui utilise l'objet $http pour récupérer la réponse JSON de votre fichier api.php et ensuite l'afficher correctement

# Ressources

- http://www.w3schools.com/angular/default.asp
- https://github.com/simplon-montreuil-promo4/angular-workshop
- https://www.youtube.com/watch?list=PLjwdMgw5TTLUDlJyx4yIPQjoI-w-7Zs1r&v=aBE0St5yI7U (Vidéo Grafikart Formation AngularJS 1.5)
- Et je vous invite à revoir les ressources de la première partie !