Découvrir AngularJS

Les fondamentaux

Présentation

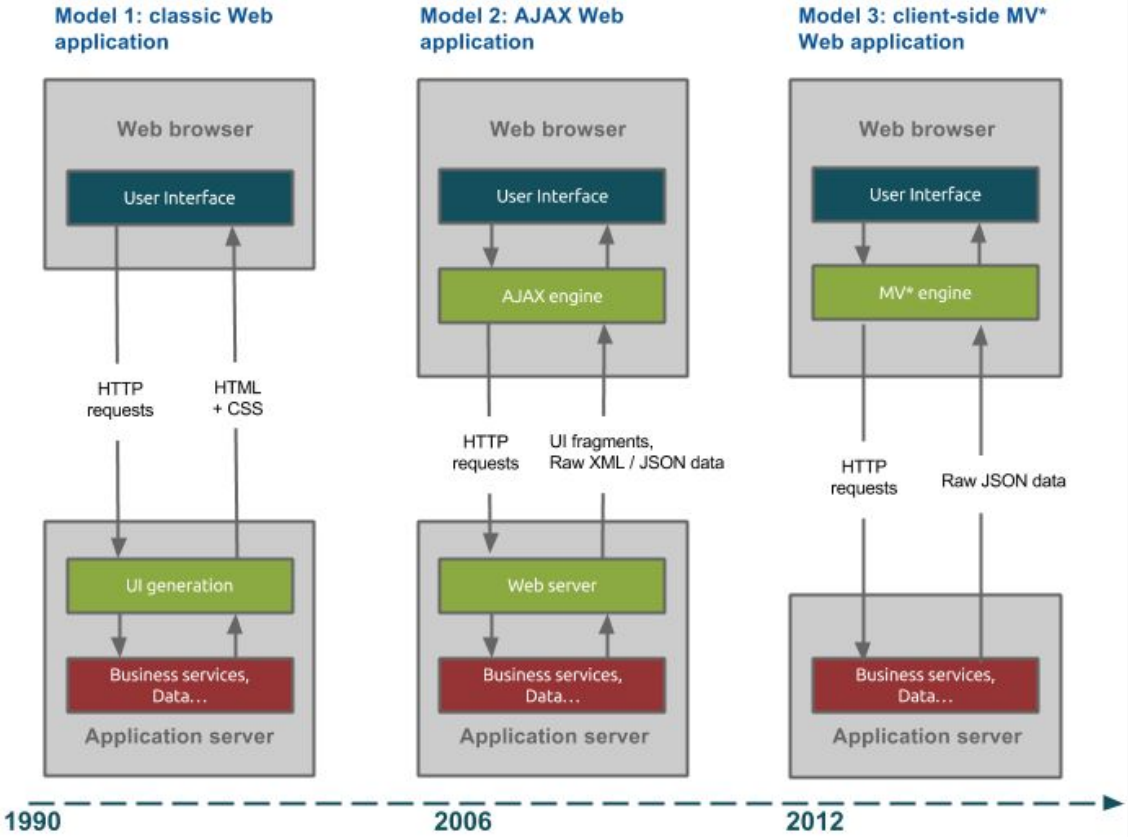
- Framework JavaScript publié en 2011
- Libre et open source
- Développé par Google
- Développement web (côté client)
- Version 2.0 considérablement différente (courant 2016)

Pourquoi AngularJS ?

- Permet d’organiser plus facilement le code JavaScript
- Permet de créer des sites internet respectant le pattern MVC
- Inclut et enrichit les fonctionnalités de jQuery
- Développé pour faciliter le test

Pourquoi AngularJS ?

Fonctionnement



Architecture - Controller principal

- app.js

```
angular.module('mySuperApp', [
   'module1',
   'module2'
 ])
 .config(function (factoryUsed, serviceUsed) {
     ...
 });
```

Ajout des modules utilisés

Éléments de configuration tels que les routes (cf routeProvider)

Architecture - Controller de vue

- controller.js

```
angular.module('mySuperApp')
 .controller('Ctrl', function ($scope, serviceUsed) {
     ...
 });
```

Création du controller Ctrl

Ajout de Ctrl à mySuperApp

Ajout des services utilisés dans le controller ($scope notamment)

Architecture - Service

- service.js

```
angular.module('mySuperApp')
 .service('myService', function (serviceUsed) {
   this.superFunction = function () {
     return ‘foo’
   };
});
```

Les services s’occupent d’aller chercher les données ou d’effectuer des opérations

Création de fonctions avec this.nomDeLaFonction

Usage dans un controller : nomDuService.nomDeLaFonction();

Architecture - Directive

- directive.js

```
angular.module('mySuperApp')
.directive('myDirective', function() {
  return {
    template: 'foo'
  };
});
```

```
-><div my-directive></div>
->foo affiché à l'écran
```

Beaucoup de possibilités !

Ne sera pas utilisé lors du tp

Plus d'infos : https://docs.angularjs.org/guide/directive

Outils

Génération de projet	Gestionnaire de dépendances	Task Runner	Validation syntaxique
			
`$ yo Angular`	`$ bower install`	`$ grunt serve`	`$ jshint myfile.js` `$ grunt jshint`

Data Binding

- Tout ce qui est entre doubles accolades **{{}}** sera **interprété**
- Les variables et fonctions définies dans **$scope** sont utilisables ici
- Changements des données dans le contrôleur répercutés sur la vue

➔ **One-way data binding**

Data Binding - Exemple

- index.html

```
<!doctype html>
<html lang="fr" ng-app>
    ...
<body>
<div ng-controller="MainCtrl">
    <h1>{{title}}</h1>
    <input type="text" ng-model="name">
    <p>Bonjour {{name}}</p>
</div>
...
</body>
</html>
```

- app.js

```
var MainCtrl = function ($scope) {
    $scope.title = "Ma première page"
    $scope.name = "";
};
```

Data Binding

- Changement des données dans le controller répercutés sur la vue.
- Mise à jour des données dans la vue répercutée dans le controller

➔ **Two-way data binding**

Data Binding - Exemple

- index.html

```
<!doctype html>
<html lang="fr" ng-app>
    ...
<body>
<div ng-controller="MainCtrl">
    <h1>{{title}}</h1>
    <input type="text" ng-model="name">
    <p>Bonjour {{name}}</p>
</div>
...
</body>
</html>
```

- app.js

```
var MainCtrl = function ($scope) {
    $scope.title = "Ma première page"
    $scope.name = "";
};
```

ng-model="name" met à jour la variable **$scope.name** du controller

Modules

- Définir les composants d’une application
  - controllers
  - services
  - directives
  - ...
- Meilleure lisibilité du code
- Réutilisabilité du module

Module ngRoute

- Permet la correspondance URL - page précise
  - template (affichage)
  - controller (comportement)
  - variables (données)
- Module ngRoute
  - 2 services
    - **$route** - lecture URL
    - **$routeParams** - gestion arguments passés via l'URL
  - 1 provider
    - **$routeProvider** - initialisation routage
  - 1 directive
    - **ngView** - insertion template

Module ngRoute - exemple

- app.js

```
angular
    .module('myApp', ['ngRoute'])
    .config(function ($routeProvider) {
        $routeProvider
        .when('/', {
              templateUrl: 'views/main.html',
              controller: 'MainCtrl',
              controllerAs: 'main'
        })
        .when('/contact', {
            templateUrl: 'views/contact.html',
            controller: 'contactCtrl'
        });
    }
]);
```

Services

- Permet de partager du code dans toute l'application grâce à l'injection de dépendances

- Service $http
  - facilite la communication avec un serveur web
  - méthode `get(url)` → renvoie une *promise*
  - promise → objet javascript représentant une valeur pas encore disponible
    → possède fonction then qui prend 2 callback (*success* ou *error*)

Service $http - exemple

- dataservice.js

```
angular.module('myApp')
    .service('dataService', function ($http) {
        this.getData = function () {
            return $http.get('url')
                .success(function (response) {
                    return response;
                })
        };
    });
```

Features

- Filters
  - But : convertir le contenu dans un format spécifique
  - `{{ data | filter }}`

- Directives
  - AngularJS inclut plusieurs directives prédéfinies
    - ng-repeat

```
<li ng-repeat="obj in tabObj"> Texte </li>
```

Répétition de la section Texte pour chaque élément `obj` présent dans le tableau `tabObj`

    - ng-show / ng-hide

```
<button ng-show="obj.prop1"> Add to Cart </button>
```

Affichage / masquage du bouton si la valeur de l'expression `obj.prop1` est vraie

Features

- ng-model

```
<select ng-model="obj.prop1"><option value="1"> Option1 </option>
```

Binding entre la valeur de l'élément du formulaire et la propriété `prop1` de `obj`

- ng-class

```
 <li ng-class="{active:val === 1}">
```

Changement d'état de la classe (vers l'état actif) si l'expression à évaluer `val===1` est vraie

- ng-submit

```
<form name="form1" ng-submit="$scope.doSomething()">
```

Appel de la fonction `doSomething` lorsque le formulaire `form1` est soumis

The End

Des questions ?