Plan de formation AngularJS

#01 - Introduction aux concepts AngularJS

Vue d'ensemble de tout les concepts AngularJS

AngularJS : Kezako

- Framework client
- Améliore le natif (HTML, JS)
- Indépendant
- Conception simple

Les modules

- Separation of concerns
- Paquet de fonctionnalités
- Indépendency injection

Les templates

- HTML + AngularJS Expressions
- Directive

Intéractions utilisateurs

- Fluidité de l'interface
- Click, mouse over, etc.
- Style & animations

Formulaires

- HTML5 Compliant
- Gestion avancée des validations / erreurs

Testing

- Testable à 100% (TU)
- Important : refactoring, securité, automatisation

Les données

- Dialogue server (XHR, JSONP)
- API : Custom, REST, JSON API, etc.

#02 - Environnement de développement

NodeJS

- Installation & Kezako

NPM

- Gestion de paquet
- Paquet JavaScript (module)
- semver.org
- Arbre de dépendance

Bower
- Gestion de paquet browser
- semver.org
- Dépendance à plat

Chrome Dev Tools
- Addy Osmani + Explication

Batarang
- Chrome extension
- Accès rapide au scope

$$watchers
- Chrome extensions
- Compteur de $watch présent sur la page

Angular-hint
- Inclut dans batarang
- Astuces de code AngularJS

IDE
- Sublim Text (w/ ton of plugins)
- Webstorm
- Netbeans

Librairie JS
- Lodash
- jQuery
- angular-ui
- angular-translate

Structure et concepts
- Architecture MV*
- Scope : dirty-checking, DOM (view) ⇔ Whatever (*)
- Contrôlleurs
- Vues
- Routing : angular-route, ui-route, angular-router
- Composants
- Data-binding
- Directives
- Filtre
- Providers
- L'injection dépendance
- Communication serveur : $http, $resource
- Testing : Karma, PhantomJS, Protractor

#02 - Scopes, templates & filters

Le scope
- Rappel : Héritage par prototypage en JavaScript
- Héritage de $scope
    - Normal ou isolé

    - $rootScope
- syntaxe controllerAs

Le cycle de vie AngularJS

- Présentation schéma event loop $apply(fn) ⇒ fn() ⇒ ($eval ⇔ $watch)
- Two-way data binding ($apply & $digest)
- Quand et comment utiliser $apply ?
- Performance (jsperf), les humains sont : Lent & limité

Les templates

- Les directives natives
- Les formulaires simples ⇒ HTML5 Validation API

Les filtres

- Pure functions
- Filtres natifs
- Création filtre perso

#03 - Le routing & les directives simple

Le routing

Routing angularJS

- $routeProvider
- ngView
- $route
- $routeParams

UI Router

- Multi-vues
- Vues enfantes
- Directive pour les liens
- Décorateurs, etc.

Router AngularJS 2.0

- Routing sous forme de conposamt
- angular.module('...', "Le gnou")

Les directives

- Kezako
- Function JavaScript qui s'occupe du DOM
- simple ⇦> compliquée
- Attribut, classe or element
- Options : template, templateUrl, restrict, scope, transclude, require, controller, controllerAs, bindToController, link

#04 - Les promises, les providers et le serveur

Les promises

- Asynchrone / Sequentielle
- Syntaxe seq (versus callbacks)
- resolce, reject, error, notify
- throw, error, success, then
- $q en angularJS

Les providers AngularJS

- constant
- value
- service
- factory
- decorator
- provider

La communication serveur

- SPA : AJAX : Kezako
- Authentification
- Requête HTTP : $http
    - XMLHttpRequest / JSONP
    - GET, POST, HEAD, DELETE, PUT
    - Promises
- Requête API type RESTful : $resource
    - get, save, query, remove, delete
    - Utilisation avec AngularJS

#05 - Correction TP + Pratique

TP : SPA

- Directives
- Service (requêtage fichier JSON de mock ou API)
- Templates avancés (ng-repeat, ng-if, etc)
- Filter
- Routing
- Architecture modulaire (approche de separation of concerns)

Example d'architecture

projet/
- index.html
- **vendors**/
--- angular.js
--- jquery.js
--- etc.
- **app**/
--- app.js ⇒ *angular.module('myApp', ['myApp.home', 'myApp.user'])*

--- **components**/
----- **home**/ ⇒ module “app.home”
------- home.js (controller) ⇒ *angular.module('myApp.home', [])*
------- home.html (template)
------- homeFilter.js
------- etc.
----- **user**/ ⇒ module “app.user”
------- user.js (controller) ⇒ *angular.module('myApp.user', [])*
------- user.html (template)
------- userService.js (récupération données serveur)
------- userDirective.js

#06 - Firebase

- Présentation Firebase
- La force : IHM
- Organisation des data (chemin d’accès aux données)
- L’authentification
    - via cookie (explication process)
    - via JWT : JSON Web Token (explication process)
    - OAUTH
    - Utilisation Firebase
- Hors-ligne : Example et explication de la queue de message + Synchro
- Firebase & Angular : Angularfire
    - Data-binding, authentification
    - $firebaseArray, $firebaseObject, $firebaseAuth
    - Live coding
        - Connection via Github
        - Récupération des repository GH
        - Création d’une salle de chat par repository

TP : Firebase

Reprendre le TP précédent et remplacer l’utilisation du JSON de mock par Firebase.
Utiliser à minima $firebaseObject.

#07 - Les animations

- Présentation des évolutions dans 1.4
- Module angular-animate + ngAnimate
- Liste des directives par défault : ngRepeat, ngView, ngInclude, etc.
- Animation CSS
    - Synchronisation avec JS
    - Séquencer animation (stagger)
    - Présentation des classes CSS utilisable
- Animation JS
    - module.animation()
    - même event que pour le CSS
    - Synchronisation avec CSS
- Service $animate (non étudié en détail)

- Service $animateCss (non étudié en détail)
- Anchoring : ng-animate-ref
    - Explication : lien entre deux éléments DOM non présent en même temps sur la page HTML
- Demo avec code source de tout les concepts

TP : Animations

Reprendre le TP précédent et rajouter des transitions CSS simple.
Utiliser le ng-animate-ref avec le routing en place.

#08 - Formulaire & directives avancée

Formulaires

- Comment créer un formulaire <form/> ou <div ng-form>
- Le nom du formulaire ⇔ Disponible sur le scope
- Utilisation du novalidate
- Les classes CSS du formulaire : ng-pristine, ng-dirty, ng-validate, etc.
- L'objet Formulaire publié sur le scope
    - ngFormController
    - ngModelController

Directives avancées

- Option require ⇒ Explication + example
- Récupération du controller d'une directive A dans la directive B

require ngModel

- $formatters (m2v)
- $parsers
- $validators
- $asyncValidators
- Allez plus loin ⇒ doc angularjs API ngModelController

ngModelOptions

- Configuration : updateOn, debounce, getterSetter, allowInvalid, timezone

ngMessages

- module extenre: angular-messages & ngMessages
- Directive gestion d'affichage
    - Affiche le premier cas (sauf si ng-message-multiple)

TP : Formulaire & Directive

Reprendre le TP précédent et rajouter un formulaire de création & Authentification.
Utiliser :

- ngMessages,
- $parsers/$formatters
- $validators
- $asyncValidators
- Style CSS pour les erreurs