

Introduction

1. Introduction	13
2. L'architecture MEAN pour une application web	15
2.1 Le principe des applications monopages (single page applications)	16
2.2 Le paradigme de conception modèle-vue-contrôleur	16
3. Angular au centre de l'architecture MEAN	19
4. Présentation du fil rouge : une application d'e-commerce	21

Le langage JavaScript

1. Introduction à JavaScript	25
1.1 Bref historique	25
1.2 Panorama de l'utilisation de JavaScript	26
1.3 Les bibliothèques et les frameworks applicatifs JavaScript	27
2. Où coder du code JavaScript ?	28
3. Les outils du navigateur et le débogage	29

4. Les éléments de programmation de base	30
4.1 Les variables	30
4.1.1 Les types internes	30
4.1.2 Le transtypage	31
4.2 Les structures de données usuelles	32
4.3 Application d'expressions régulières	34
4.4 Les blocs d'instructions	35
4.5 Les structures conditionnelles	35
4.5.1 La structure if ... else ...	35
4.5.2 La structure switch d'aiguillage multiple	36
4.6 Les structures itératives	36
4.6.1 Les structures itératives avec indices de boucle	36
4.6.2 Les structures itératives sans indices de boucle	37
5. La programmation fonctionnelle en JavaScript	39
5.1 Une fonction passée en paramètre (fonction de callback)	39
5.1.1 Exemple avec la méthode forEach()	39
5.1.2 Exemple avec la méthode map()	40
5.2 Une fonction retourne une fonction (factory)	41
6. La programmation objet avec JavaScript	42

6.1 Les principes de la programmation objet avec JavaScript	42
6.2 Les objets littéraux	43
6.3 L'héritage par chaînage de prototypes	44
6.3.1 La propriété __proto__ de l'objet héritant	44
6.3.2 La propriété prototype	45
6.4 La création d'un objet par l'appel d'une fonction constructrice	46
6.5 Exemples d'implémentations d'une méthode	47
6.6 La problématique de l'objet courant (this)	48
6.7 L'héritage	51
6.8 Le chaînage de méthodes	51
7. Les principaux apports de la norme ECMAScript 6	52
7.1 La norme ECMAScript	52
7.2 Le mot réservé let	52
7.3 L'interpolation de variables dans les chaînes	53
7.4 Les paramètres par défaut	53
7.5 Une manipulation plus confortable des listes	53
7.5.1 La structure for (... of ...) ...	53
7.5.2 La méthode includes()	54
7.6 L'opérateur « fat arrow » (=>)	54
7.7 Les classes	55

8. La programmation réactive, les observables et les promises	55
8.1 Premier exemple : un observable sur un bouton	57
8.2 Deuxième exemple : un observable sur un entier	58
8.3 Troisième exemple : un observable sur un timer	59
8.4 Les promises	59

Extensions JavaScript pour les classes

1. Présentation des extensions à JavaScript	61
2. Le langage TypeScript	62
2.1 Le transpiler tsc	62
2.2 Typage statique des variables	63
2.2.1 Les types de base	63
2.2.2 Typage de variables non scalaires	63
2.2.3 Le type enum	64
2.2.4 Le type générique any	64
2.2.5 Typage des fonctions	64
2.3 Les classes	65
2.3.1 L'héritage	66
2.3.2 Les interfaces	67

2.3.3 La généricité	67
3. Le langage Dart	68
3.1 Installation et test de Dart	68
3.2 Génération du code JavaScript avec Dart	69
3.3 Les classes	69
3.3.1 L'héritage	70
3.3.2 Les interfaces	71
La plateforme Node.js	
1. Présentation de Node.js	73
2. Installation et test de Node.js	74
2.1 Création du fichier de test	74
2.2 Installation et test de Node.js sous Ubuntu	75
2.3 Installation et test de Node.js sous Windows	75
2.4 Installation et test de Node.js sous Mac OS	76
3. La modularité de Node.js	76
3.1 Les modules et les packages	76
3.1.1 Le gestionnaire de modules de Node.js : npm	77

3.1.2 Spécification des dépendances : le fichier package.json	77
3.2 Création d'un premier serveur Node.js de test	78
3.3 Création et réutilisation d'un module	80
3.4 Création d'un serveur renvoyant des données	82
3.5 Le module express	83
3.5.1 Gestion de routes REST avec le module express	83
3.5.2 Gestion des templates avec le module express	86
3.5.3 Spécification des dépendances dans un fichier package.json	88
3.5.4 Installation du module express	88
3.6 Le module fs (FileSystem)	88
3.7 Test d'un serveur Node.js	90
3.7.1 Création d'un fichier de données JSON	90
3.7.2 La problématique du contrôle d'accès HTTP	91
3.7.3 Renvoi au client d'un fichier JSON	92
3.7.4 Paramétrage des routes	93
3.7.5 Gestion des paramètres	98
4. Sécurisation d'un serveur Node.js (protocole HTTPS)	99
4.1 Création de l'autorité de certification	100
4.2 Création du certificat	100
4.3 Création du serveur	101

5. Bilan des acquis de ce chapitre	102
Le SGBD NoSQL MongoDB	
1. Introduction	103
2. Pourquoi utiliser une base de données NoSQL ?	104
3. Présentation de MongoDB	105
3.1 Les collections et les documents	105
3.2 Les index	106
4. Mise en œuvre de MongoDB	106
4.1 Installation de MongoDB	106
4.1.1 Installation de MongoDB sous Linux	106
4.1.2 Installation de MongoDB sous Windows ou sous Mac OS	107
4.1.3 Utilisation de MongoDB en lignes de commande	107
4.2 Affichage de la liste des bases de données	108
4.3 Création d'une base de données	108
4.4 Affichage de la liste des collections	108
4.5 Création d'une collection	108
4.5.1 Insertion des documents dans une collection	109

4.5.2 Importation de documents à partir d'un fichier	109
4.5.3 Exportation des documents d'une collection dans un fichier JSON	110
4.6 Interrogation d'une collection	111
4.6.1 Interrogation via un objet « filtre »	111
4.6.2 Les opérateurs de comparaison, les opérateurs ensemblistes et logiques	112
4.6.3 L'opérateur \$ exists	113
4.6.4 L'opérateur \$ in	113
4.6.5 L'opérateur \$ nin	113
4.6.6 L'opérateur \$ or	114
4.6.7 L'opérateur \$ not	114
4.6.8 L'opérateur \$ nor	114
4.7 Application d'expressions régulières	115
4.8 Les projections et la méthode distinct()	115
4.8.1 Les projections	115
4.8.2 La méthode distinct()	116
4.9 Référencement des documents et jointures	116
4.9.1 Les objets imbriqués (nested objects)	117
4.9.2 Les objets référencés	118
4.9.3 Les jointures	119
4.10 Mise à jour et suppression d'un document	123
4.10.1 Mise à jour d'un document	123

4.10.2	Suppression d'un document	124
4.11	Suppression d'une collection	124
5.	Utilisation de MongoDB via Node.js	124
5.1	Installation du module MongoDB pour Node.js	124
5.2	Connexion au serveur MongoDB	126
5.3	Insertion de données à partir d'un serveur Node.js	127
5.4	Interrogation de données à partir d'un serveur Node.js	127
5.4.1	Exploitation du résultat de la méthode find()	128
5.4.2	Utilisation de la méthode toArray()	129
5.5	Synchronisation des requêtes	131
5.5.1	Utilisation des fonctions de callback	132
5.5.2	Utilisation du module async	134
5.5.3	La méthode async.series()	135
5.5.4	La méthode async.waterfall()	136
6.	Interrogation de MongoDB via les routes gérées par express	138
6.1	La structure d'un serveur Node.js interrogeant MongoDB	138
6.2	La problématique du cross-origin resource sharing	139
6.3	Exemples de gestion de routes	139
6.3.1	Gestion d'une route pour lister les marques	140
6.3.2	Gestion d'une route pour filtrer les produits	

6.3.3 Recherche d'un produit à partir de son identifiant interne	141
	142
7. Le fil rouge du côté serveur	144
7.1 Création de la collection	144
7.2 Mise en place de deux recherches sur les produits	146
7.2.1 La superstructure du serveur	146
7.2.2 Gestion de la route qui filtre les documents sur différents critères	148
7.2.3 Gestion de la route qui renvoie un document via son identifiant interne	149
7.2.4 Exemples de requête sur le serveur	149
8. Bilan des acquis de ce chapitre	150
Introduction au framework applicatif Angular	
1. Présentation d'Angular	151
1.1 Une évolution radicale d'AngularJS	151
1.2 La modularité de l'application : les modules et les composants	152
1.2.1 Les modules	153
1.2.2 Les composants et les services	154
1.3 Manipulation des composants comme des balises	155
1.4 Utilisation d'une extension à JavaScript (TypeScript ou Dart)	156

2. Angular par rapport au framework MVC (voire MVVM)	156
3. Mise en place d'une application Angular	158
3.1 Présentation d'Angular CLI	159
3.2 Installation d'Angular CLI	160
3.3 Création d'un projet Angular avec Angular CLI	160
3.4 Structure des dossiers d'un projet Angular CLI	162
3.5 Un premier composant créé par Angular CLI	163
3.6 Le root module créé par Angular CLI	166
3.7 Mise à jour d'Angular via Angular CLI	167
4. Génération du code JavaScript à partir de TypeScript	168
5. Les décorateurs	170
6. Création d'un nouveau composant qui affiche un message	171
6.1 Création du composant	171
6.1.1 Le template HTML	173
6.1.2 La feuille de style	173
6.2 Interfaçage du composant dans le composant racine	173
6.3 Spécification des composants dans le module	173
6.4 Activation du module	174

6.5 La page web frontale	175
7. Le cycle de vie d'un composant	176
7.1 Le hook ngOnChanges()	177
7.2 Le hook ngOnInit()	178
7.3 Le hook ngDoCheck()	178
7.4 Le hook ngOnDestroy()	179
8. Bilan des acquis de ce chapitre	180
Angular : les templates, bindings et directives	
1. Les templates	183
1.1 Imbrication des templates	185
1.2 Les templates insérés (embedded templates)	191
1.3 Les templates externalisés	192
2. Data bindings entre le composant et le template	192
2.1 Accès aux éléments du DOM	193
2.2 Interpolation d'une variable dans un template	193
2.3 Property binding	195
2.4 Event binding	197

2.5 Two-way data binding	199
3. Les directives	201
3.1 Les directives structurelles	202
3.1.1 La directive *ngFor	203
3.1.2 La directive *ngIf	205
3.2 Les directives attributs	206
3.3 Émission d'information d'un composant vers son père (@Output())	211
4. Les pipes	214
5. Exemple de synthèse : un formulaire d'authentification	218
6. Le fil rouge : création d'un composant qui affiche les produits	221
6.1 Le composant	222
6.1.1 La classe implémentant le composant	223
6.1.2 Le template HTML	224
6.1.3 La feuille de style	226
6.2 Le module spécifiant le composant	226
6.3 Activation du module	227
6.4 La page web frontale	227
6.5 Lancement de l'application	228

7. Bilan des acquis de ce chapitre	228
Angular et la connexion à Node.js : les services	
1. Introduction	231
2. Injection de dépendances	232
3. Utilisation des services pour le transfert de données	233
3.1 Récupération de données formatées en JSON	233
3.2 Envoi de données JSON au serveur	234
3.3 Envoi de données via la querystring	236
4. Mise en œuvre des services dans le fil rouge	237
4.1 Déclaration des routes du côté serveur	237
4.2 Gestion des produits	239
4.2.1 Affichage des sélecteurs	239
4.2.2 Affichage des produits suivant des critères de recherche	243
4.2.3 Affichage des produits associés à des mots-clés	246
4.2.4 Accès à un produit par son identifiant	249
4.3 Gestion du panier	250
4.3.1 Affichage des identifiants des produits du panier	250

4.3.2 Affichage de tous les produits du panier	251
4.3.3 Ajout d'un produit au panier	253
4.3.4 Suppression d'un produit du panier	256
4.3.5 Réinitialisation du panier	258
5. La bibliothèque NgRx et les stores	258
6. Bilan des acquis de ce chapitre	259

Angular et la gestion des routes internes

1. Principe général du routage	261
1.1 Pourquoi mettre en place un routage ?	261
1.2 Les routes, le routeur, les tables de routage	263
1.3 Les vues activées par les routes	265
1.4 Exemple de routage	267
1.5 Définition d'un arbre de vues	270
1.6 Utilisation des outlets nommées	275
2. La syntaxe des routes	280
2.1 Les deux syntaxes d'une route : chaîne ou link parameters array	281
2.2 Les routes absolues et les routes relatives	282

2.3 Paramétrage des routes	282
2.4 Association d'une route à une auxiliary outlet	283
3. Sélection des routes	284
3.1 La directive routerLink	284
3.2 La méthode navigate()	286
3.3 Exemple de route	287
4. Gestion des routes du contrôleur vers le composant cible	287
4.1 Configuration de la table de routage	288
4.2 Les propriétés d'une route	289
4.3 Prise en charge d'une route par plusieurs modules/tables de routage	291
4.4 Contrôle des routes : les guards	293
4.5 Invocation d'un composant	301
4.6 Capture d'une route lors de l'invocation d'un composant	303
5. Gestion de routes dans le fil rouge	306
5.1 Le module de routage associé au root module	307
5.2 Le module de routage associé au feature module research	308
5.3 Le module de routage associé au feature module cart	309
6. Bilan des acquis de ce chapitre	310

Angular et la visualisation d'informations

1. Introduction	313
2. Création de charts avec D3.js et dc.js	314
2.1 Installation de D3.js	315
2.2 Le langage SVG	315
2.3 Génération d'éléments graphiques associés aux objets d'une collection	318
2.4 Sélection et modification d'éléments du DOM	319
2.4.1 Ajout d'éléments graphiques	320
2.4.2 Remplacement d'éléments graphiques	321
2.5 Mise en œuvre d'écouteurs d'événements	322
2.6 Intégration de D3.js dans Angular	323
2.6.1 Un serveur virtuel de données commerciales	323
2.6.2 Le service accédant aux données du serveur	325
2.6.3 Le template du composant qui affiche un histogramme	325
2.6.4 Implémentation du composant qui affiche un histogramme	326
2.7 Les bibliothèques dc.js et Crossfilter	330
2.7.1 Installation de dc.js et de Crossfilter	331
2.7.2 Implémentation du composant affichant l'histogramme	331

3. Intégration de cartes Google Map dans un projet Angular	333
3.1 Installation des prérequis techniques	334
3.1.1 Installation des types TypeScript	334
3.1.2 Installation de la bibliothèque PrimeNG	335
3.2 Présentation d'une carte Google Map statique	335
3.3 Un composant PrimeNG pour gérer une carte Google Map	337
3.4 Gestion d'un chart associé à une Google Map	341
4. Mise en œuvre de composants graphiques	346
4.1 Autre exemple de composant PrimeNG (Calendar)	346
4.2 La bibliothèque de composants Material	348
4.3 La bibliothèque ngx-bootstrap	350
5. Bilan des acquis de ce chapitre	355
Test et déploiement	
1. Test	357
2. Déploiement	361
2.1 Déploiement avec Apache	361
2.2 Déploiement avec Node.js	361

2.3 Déploiement avec http-server (raccourci sur Node.js)	362
	363
3. Pour aller plus loin	364
Index	365