

Table des matières

Introduction	13
Partie 1. Les fondements : Core JavaScript	23
Introduction de la partie 1	25
Chapitre 1. Variables : déclaration, définition et types	27
1.1. Déclaration de variable et de fonction	28
1.1.1. Quatre mots-clés de déclaration	28
1.1.2. Portée lexicale et définition d'une variable selon : var, let, const .	30
1.1.3. Commentaires sur les possibilités ouvertes par ES6	32
1.1.4. Conclusion sur les déclarations de variable en JavaScript	32
1.1.5. Récapitulation du mécanisme des deux passes	34
1.2. Initialisation et définition des variables	35
1.3. Comportement du moteur JavaScript	35
1.3.1. Recommandations	36
1.4. Nommage des variables et des fonctions	36
1.5. Types en JavaScript	37
1.5.1. Usage de la valeur native 'undefined' et du type 'undefined' . . .	38
Chapitre 2. Contrôles : booléens, tests et boucles	39
2.1. Valeurs de vérité, opérateurs booléens, de comparaison et relationnels	39
2.1.1. Opérateurs booléens : notés « ! » (not), « && » (and) et « » (or)	39

2.1.2. Opérateurs relationnels : supérieur, inférieur strict (>), (<), ou égal (>=), (<=).	40
2.1.3. Opérateurs de comparaison (égalité ou inégalité)	40
2.2. Test d'instruction conditionnelle et condition d'arrêt dans les boucles	41
2.2.1. Les instructions conditionnelles : « if..else, if..else if..else ».	41
2.2.2. Opérateur ternaire pour expressions conditionnelles	42
2.2.3. Instruction « switch ».	42
2.2.4. La boucle classique : l'instruction « for ».	43
2.2.5. Les répétitions sous condition : instructions « while » et « do..while »	43
2.2.6. Conversion des valeurs 'undefined' et 'null' dans les tests	44
2.2.7. Évaluation « court-circuit » : usage pour les définitions incertaines	45
2.3. Gestion des exceptions et émission d'exceptions (<i>Exception handling</i>)	45

Chapitre 3. Données : nombres et chaînes de caractères 49

3.1. Manipulation des nombres	50
3.1.1. Variables de type « number ».	50
3.1.2. Opérateurs arithmétiques	50
3.1.3. Conversion de type entre « number » et « boolean »	54
3.2. Manipulation des chaînes de caractères (type « string »)	54
3.2.1. Notation littérale d'une chaîne de caractères	54
3.2.2. La syntaxe « back-tick » introduite par ES6	55
3.2.3. Opérateur de concaténation de chaînes de caractères	55
3.2.4. Résolution du polymorphisme de l'opérateur +	55
3.2.5. Comportement des opérateurs relationnels et d'égalité	56
3.3. Les méthodes de l'objet String.prototype	59
3.3.1. Nécessité des prétraitements avant comparaison.	59
3.3.2. Manipulation interne de la chaîne et comparaisons partielles	60
3.4. Les expressions régulières	62
3.4.1. Syntaxe d'une expression régulière	64
3.4.2. Combinaison des RegExp avec les méthodes String.prototype	65

Chapitre 4. Objets et prototypes en JavaScript 67

4.1. Plan de ce chapitre.	67
4.2. Les objets : concepts ou entités définies ?	68
4.3. Le littéral objet en JavaScript	69
4.3.1. Syntaxe de littéral 'Object'	69
4.3.2. Remarques importantes sur la syntaxe du littéral objet JavaScript	70

4.3.3. Premier usage du littéral objet : définir une variable de type objet	71
4.3.4. Deuxième usage du littéral objet : notation de données JSON . .	71
4.3.5. Accès aux propriétés individuelles d'un objet	72
4.3.6. Notation des propriétés et méthodes dans la norme ES6	73
4.4. Les méthodes de Object et Object.prototype	74
4.4.1. Les méthodes de Object, de Object.prototype et de JSON	74
4.4.2. Créer un objet, spécifier ses propriétés	76
4.4.3. Lister les propriétés d'un objet, afficher un objet, analyser un littéral	77
4.5. Principe de l'approche « prototypale » en JavaScript	79
4.5.1. À la base des objets du JavaScript : la relation « a pour prototype »	80
4.5.2. Rôle et mise en œuvre des prototypes	82
4.5.3. Schéma de construction d'objet par l'approche « littérale »	83
4.5.4. Schéma de construction d'objet par l'approche « prototypale » avec Object.create	85
4.5.5. La combinaison Assign/Create	86
4.5.6. Schéma de construction d'objet par l'approche « classique » avec l'opérateur 'new'	87
4.6. Compléments sur l'approche « prototypale » (combinaison Assign/Create)	89
4.6.1. Simulations alternatives d'une hiérarchie de classes en JavaScript .	90
4.6.2. Quelques points à retenir	94

Chapitre 5. Tableaux 97

5.1. Manipulation des tableaux : création et accès aux éléments	98
5.1.1. Création d'un tableau par littéral tableau	98
5.1.2. Type d'un tableau	98
5.1.3. Propriété <i>length</i>	99
5.1.4. Accéder aux valeurs d'un tableau : les indices	100
5.2. Méthodes statiques de l'objet Array	101
5.3. Méthodes du prototype Array.prototype	101
5.3.1. La famille « Mutators »	101
5.3.2. La famille « Accessors ».	104
5.3.3. La famille « Iteration ».	105
5.4. Itérer sur les éléments d'un tableau	106
5.4.1. Les inconvénients des boucles classiques	106
5.4.2. Itération sans boucle, à l'aide des méthodes de Array.prototype .	106
5.4.3. Usage fonctionnel des méthodes de tableaux	109

5.5. Tableau de tableaux (tableau à deux dimensions)	111
5.5.1. Simulation de tableau bidimensionnel en JavaScript	111
5.5.2. Bibliothèques qui proposent un « Array.prototype augmenté »	112
5.5.3. Propriétés d'un tableau.	113
5.5.4. Les « array-like », objets de type « Iterable »	113

Chapitre 6. Fonctions 115

6.1. Syntaxe générale d'une fonction en JavaScript	116
6.1.1. Nom	117
6.1.2. Paramètres	117
6.1.3. Retour	117
6.1.4. Bloc de code et « portée »	118
6.2. Création des fonctions	118
6.2.1. Déclaration de fonction	118
6.2.2. Expression de fonction.	119
6.3. Invocation de fonction avec l'opérateur (.)	119
6.4. Choix entre déclaration de fonction <i>versus</i> expression de fonction	120
6.5. Arguments	121
6.5.1. Les arguments sont toujours passés par valeur.	121
6.5.2. L'objet interne « arguments »	122
6.6. Portée des noms de variables et de fonctions : « scope »	123
6.6.1. Vocabulaire : portée et espace de noms (« scope », « namespace »)	123
6.6.2. Récapitulation et mises en garde	127
6.6.3. Note sur le mécanisme interne	128
6.6.4. Vaccination contre le 'var' : série d'exercices	129
6.7. Clôture de fonctions : « closure »	130
6.7.1. Pérennisation de la valeur d'une variable libre dans un contexte donné	131
6.7.2. Création d'une liste de fonctions dépendantes d'une liste de données (tableau)	132
6.7.3. « Curryfication » : décomposition d'une fonction en fonctions à un seul paramètre.	134
6.7.4. Composition de fonctions à partir d'un tableau	135
6.8. Fonctions auto exécutables : IIFE (<i>Immediately-Invoked Function Expression</i>).	136
6.8.1. Création d'un « namespace » ou d'une bibliothèque nommée avec une IIFE	138
6.9. Fonctions	138

6.10. Les méthodes de <code>Function.prototype</code>	139
6.10.1. <code>Function.prototype.call()</code> and <code>.apply()</code> : retour sur le ‘this’	140
6.10.2. <code>Function.prototype.bind()</code>	141
6.10.3. Les fonctions natives (« built-in functions »)	142
Chapitre 7. Du signe au motif	143
7.1. Mots réservés	144
7.2. Le pronom ‘this’	145
7.2.1. Ambiguïtés liées au pronom ‘this’	146
7.3. Comment passer le pronom d’un contexte à un autre ?	147
7.4. Opérateur <i>new</i>	148
7.5. Ponctuation	148
7.6. Patrons de conception (<i>design patterns</i>)	150
7.6.1. Idiomes de programmation	151
7.6.2. Patron de création : « Assign/Create Combo »	152
7.6.3. Patron structurel : <i>singleton</i> ou <i>namespace pattern</i>	154
7.6.4. Patron structurel : <i>decorator pattern</i>	155
7.6.5. Patron de comportement : <i>observer pattern</i>	156
7.7. Méta-programmation avec ES6	158
7.7.1. La « Reflection » par les « Symbols »	158
7.7.2. Outils de mesure de la performance	158
Partie 2. JavaScript Client Web (Client-Side JavaScript)	159
Introduction de la partie 2.	161
Chapitre 8. JavaScript dans l’écosystème de la page web	163
8.1. Écosystème de la page web : la séquence HTML	163
8.1.1. Structure et sémantique/mise en page et présentation	163
8.1.2. Rappel sur les balises HTML5	164
8.2. Fonctionnement de la page web : le moteur de rendu	166
8.2.1. Arbre DOM créé par le moteur de rendu, sélection des nœuds par le CSS	167
8.2.2. Les règles CSS : relation avec les accès JavaScript	168
8.3. Fonctionnement de la page web : le moteur de script	169
8.4. Interface avec le <i>Document Object Model</i>	171
8.4.1. Interface DOM : la sélection des éléments	171

8.4.2. Interface DOM : lecture/écriture/création d'un élément	173
8.4.3. Propriétés et méthodes des objets HTML DOM <i>Document</i> et <i>Element</i>	174
8.5. Les événements dans le JavaScript client	176
8.5.1. La boucle d'événements du navigateur : <i>browser event loop</i> . . .	176
8.5.2. Gestion des événements	177
8.6. Interaction avec le DOM : exemples combinant éléments et événements	179
8.6.1. Usage des événements : attendre la fin du chargement du DOM .	179
8.6.2. Définir une fonction de création de listes HTML	180
8.6.3. Usage des événements : modifier les attributs ou les noms de classe d'un élément	181
8.6.4. Créer des événements et les émettre : <i>dispatchEvent</i> , <i>CustomEvent()</i>	181

Chapitre 9. Outils graphiques et multimédias du client 183

9.1. Dessiner dans la page web	183
9.1.1. Les éléments <code><figure></code> et <code><canvas></code>	184
9.1.2. Tracé de courbe 2D	184
9.2. Graphique 3D	187
9.3. Langage graphique SVG	187
9.4. Gestion du temps dans l'animation	189
9.4.1. Les méthodes <i>setTimeout</i> , <i>setInterval</i> , <i>requestAnimationFrame</i> .	189
9.4.2. Considérations sur la performance, fonctions « génératrices » . .	190
9.5. La persistance des données entre sessions de JavaScript client	192
9.5.1. Les « <i>http cookies</i> ».	192
9.5.2. Les « <i>local storages</i> ».	193
9.6. Note sur les « JavaScript frameworks » (jQuery, d3, etc.)	194
9.6.1. Présentation de jQuery	194
9.6.2. Exemple d'évolution : requête Ajax	194
9.6.3. Exemple d'évolution : modifier le style d'un élément du DOM .	195
9.6.4. Recommandation	195

Chapitre 10. Technologie AJAX (requêtes asynchrones) 197

10.1. Architecture d'échange de données entre le client et le serveur	197
10.2. Usage de <i>XMLHttpRequest</i>	198
10.3. Note sur HTTP	199
10.4. Les « promesses » du JavaScript	199
10.4.1. Exemple : <i>XMLHttpRequest</i> transformé en promesse	200

10.4.2. Enchaînement de promesses	201
10.4.3. Note sur Fetch	202
10.4.4. Note sur « Same Origin Policy » (SOP)	203
10.5. Le format d'échange JSON.	203
10.5.1. Un usage très utile du JSON : convertir les données d'un logiciel tableur	204
10.5.2. Exporter des données tabulées en format JSON	206
10.5.3. Notation de littéral objet JavaScript et notation JSON	208
10.5.4. Gestion des erreurs avec la lecture JSON	209
10.6. JSONP (<i>JavaScript Object Notation with Padding</i>)	210
10.7. Autre outil asynchrone du JavaScript : les « Workers »	211
Partie 3. Applications	213
Introduction de la partie 3.	215
Chapitre 11. Données chronologiques	217
11.1. Accéder à un fichier JSON <i>via</i> une interface Ajax	217
11.1.1. Description sommaire de l'API Quandl	217
11.1.2. Accès aux données	218
11.1.3. Traitement des données.	219
11.1.4. Explication	219
11.1.5. Données additionnelles.	220
11.2. Usage de bibliothèques graphiques Open Source.	221
11.2.1. Tracer plusieurs courbes par rapport à un même axe	221
11.2.2. Tracé dynamique d'une courbe (rafraîchissement des données).	223
Chapitre 12. Données relationnelles	225
12.1. Agrégation de données JSON tabulées	225
12.1.1. Données électorales : découpage administratif et découpage en affiliations politiques	226
12.1.2. Agrégation de données le long d'une dimension : votes par circonscription	229
12.1.3. Agrégation de données le long d'une dimension : affiliations par candidat	231
12.2. Jointure de données : multiples fichiers JSON	233
12.2.1. Intérêt de la flexibilité de l'approche prototypale	234

12.3. Post-traitement : analyse	235
12.4. Intérêt des promesses pour les jointures multiples	236
12.4.1. Mise en promesses des étapes de l'application	236
12.4.2. Résultats obtenus avec l'application électorale.	237
12.5. Application graphique : usage d'une librairie spécialisée (Google Charts)	237

Chapitre 13. Données cartographiques. 239

13.1. Application cartographique : usage d'une librairie spécialisée	239
13.1.1. Préparation	240
13.1.2. Création de marqueurs	241
13.2. Cartographie avec le « Géoportail ».	243
13.3. Cartographie basée sur des éléments <svg>	244
13.3.1. Description de l'application	244
13.3.2. Procédure d'intégration du SVG : par copie directe	245
13.3.3. Procédure d'intégration du SVG : élément par élément.	246
13.3.4. Procédure de jointure SVG : infos par pays.	246
13.3.5. Post-traitement : combinaison des informations	247

Chapitre 14. Données servies par JSONP 249

14.1. Serveur de flux RSS : Yahoo Query Language	249
14.2. Serveur de données tabulées partagées : Google Spreadsheets	251
14.2.1. Code (<i>client side</i>) du HTML et de la fonction de <i>callback</i>	251
14.2.2. Code (<i>server side</i>) dans l'environnement GoogleScript.	252
14.3. Serveur d'images : Flickr API	252

Bibliographie 255

Index 259