Table des matières

À propos des auteurs	xviii
Préface	xix
Avant-propos	xx

Bases du langage, spécificités de Lua..... 1

1. Informations générales	2
1. Installer Lua	2
2. Sur quels systèmes puis-je installer Lua ?.....	2
3. Quel éditeur pour développer en Lua ?.....	2
4. Comment Lua se situe-t-il par rapport aux autres langages en termes de performances ?.....	3
5. Quelles versions faut-il utiliser ?.....	3
6. Quels outils pour documenter son code ?.....	3
7. Quels sont les sites de référence pour développer en Lua ?.....	3
8. Comment passer de Lua 5.1 à Lua 5.2 ?.....	4
9. Comment passer de Lua 5.2 à Lua 5.3 ?.....	4
2. Principes et éléments de syntaxe	5
10. De quoi est composé Lua ?.....	5
11. Les commentaires	6
12. Les commentaires sur plusieurs lignes	6
13. Les commentaires d'affichage	6
14. Les instructions	7
15. Les mots clés du langage	7
16. Afficher des résultats	7
17. Comment la mémoire est-elle allouée ou libérée ?.....	8
18. Comment connaître la version de Lua utilisée ?.....	8
3. Variables et types	9
19. Quel est le type d'une variable ?.....	9
20. Quels sont les types de données disponibles ?.....	9
21. Le type <i>nil</i>	9
22. Le type <i>boolean</i>	10
23. Le type numérique	10
24. Le type <i>string</i>	10
25. Le type <i>function</i>	11

26. Le type <i>userdata</i>	11
27. Qu'est-ce qu'un <i>thread</i> Lua ?.....	11
28. Le type <i>table</i>	11
29. Un type peut-il être converti ?.....	11
30. Connaître le type d'une donnée	12
31. Quelle est la portée d'une variable ?.....	12
4. Expressions et opérateurs	13
32. Quels sont les opérateurs arithmétiques ?.....	13
33. Quels sont les opérateurs logiques binaires ?.....	13
34. Quels sont les opérateurs relationnels ?.....	14
35. Quels sont les opérateurs logiques disponibles ?.....	14
36. Autres opérateurs à connaître	15
5. Portée des variables, blocs et chunks.....	16
37. Quand utiliser <i>local</i> ?.....	18
38. Où sont stockées les variables globales ?.....	18
6. Structures de contrôle	20
39. Comment déclarer un bloc de code ?.....	20
40. Comment exprimer des conditions ?.....	20
41. Les expressions ternaires existent-elles comme en C/C++ ?	21
42. Comment faire un <i>switch/case</i> ?.....	21
43. Comment effectuer des boucles ?.....	21
44. Utiliser les boucles de type <i>while</i>	22
45. Utiliser les boucles de type <i>repeat</i>	22
46. Utiliser les boucles de type <i>for</i>	22
47. Créer une boucle <i>for</i> numérique	23
48. Créer une boucle <i>for</i> générique	24
49. Comment sortir d'une boucle ?.....	25
50. Peut-on faire un <i>try/catch</i> comme en Java ou C++ ?	25
51. Existe-il une instruction <i>continue</i> ?.....	26
52. Effectuer un saut inconditionnel avec <i>goto</i>	26
7. Fonctions	27
53. Nombre inconsistant d'arguments et de valeurs	28
54. Nommer des arguments pour les sélectionner lors de l'appel.....	28
55. Déclarer une liste variable d'arguments	29
56. Renvoyer plusieurs résultats	30
57. Que sont les fonctions anonymes ?.....	30
58. Que sont les <i>closures</i> ?.....	30

8. Tables	32
59. Construire une table et y accéder	32
60. Initialiser une table	33
61. Créer et utiliser un enregistrement	34
62. Comment obtenir la taille d'une table ?	34
63. Pourquoi une indexation numérique à partir de 1 ?	35
64. Table de fonctions	35
65. Instancier des objets avec les tables	36
66. Parcourir une table avec un <i>for</i> générique	37
67. Copier une table	38
68. Que sont les <i>weak tables</i> ?	38
69. Utiliser des tables pour faire des matrices	39
70. Le module <i>table</i>	39
71. La fonction <i>table.concat</i>	40
72. Insérer une valeur dans une séquence	40
73. Supprimer une valeur dans une séquence	40
74. Créer un tableau depuis une liste de valeurs	41
75. Extraire une liste de valeurs d'une séquence	41
76. Trier une séquence	41
77. Déplacer des éléments d'une table vers une autre	42
9. Les fonctions internes	43
78. <i>assert()</i>	43
79. <i>collectgarbage()</i>	44
80. <i>dofile()</i>	46
81. <i>error()</i>	46
82. <i>setmetatable()/getmetatable()</i>	46
83. <i>ipairs()/pairs()</i> ?	46
84. <i>next()</i>	47
85. <i>loadfile()/load()</i>	47
86. <i>pcall()/xpcall()</i>	47
87. <i>print()</i>	48
88. <i>rawequal()/rawget()/rawlen()/rawset()</i>	48
89. <i>require()</i>	48
90. <i>select()</i>	48
91. <i>tonumber()</i>	49
92. <i>tostring()</i>	49
93. <i>type()</i>	49
10. La gestion des erreurs	50

94. Quelles sont les erreurs générées ?	50
95. Remonter des erreurs	50
11. Les coroutines	51
96. Créer une coroutine	51
97. Connaître l'état des coroutines	53
98. Que fait la fonction <code>coroutine.wrap()</code> ?	54
Lua, bibliothèques et modules	56
12. Appeler et exécuter du code externe	57
1. Charger et exécuter un script	57
2. Charger un script à partir d'une chaîne de caractères	58
3. Charger un script à partir d'un fichier	59
4. Créer une bibliothèque	60
5. Gérer un fichier de configuration	60
6. Contrôler le code chargé par les fonctions <code>load()/loadfile()</code>	61
13. Créer ses bibliothèques	62
7. Le module <code>package</code>	62
8. Appeler une bibliothèque avec <code>require()</code>	62
9. Comment les modules sont-ils recherchés ?	62
10. Changer les chemins de recherche	64
11. La variable <code>package.config</code>	64
12. Écrire un module Lua destiné à être chargé par <code>require()</code>	65
13. Écrire un module C destiné à être chargé par <code>require()</code>	67
14. Accéder depuis Lua à une bibliothèque C ou C++	67
15. Déclarer le <code>loader</code> d'une bibliothèque	68
16. À quoi sert <code>package.loadlib()</code> ?	68
14. Les métatables	70
17. Quels opérateurs peut-on modifier avec les métatables ?	70
18. Peut-on ajouter de nouveaux opérateurs aux métatables ?	71
19. À quoi cela peut-il servir ?	71
20. Utiliser la métaméthode multiplication (<code>__mul</code>)	74
21. Utiliser la métaméthode division (<code>__div</code>)	75
22. Utiliser la métaméthode modulo (<code>__mod</code>)	75
23. Utiliser la métaméthode de négation (<code>__unm</code>)	76
24. Utiliser la mise à la puissance (<code>__pow</code>)	76
25. Utiliser la métaméthode de concaténation	77
26. Utiliser les métaméthodes de comparaison	77

27. Utiliser la métaméthode de ramasse-miettes	78
28. Accéder à un indice avec <code>__index</code>	79
29. Simuler une classe avec des méthodes	79
30. Créer un index avec <code>__newindex</code>	80
31. Retourner la taille d'une valeur avec <code>__len</code>	81
32. Capturer un appel de fonction avec <code>__call</code>	81
33. Modifier les fonctions <code>pairs()</code> et <code>ipairs()</code>	81
34. Éviter l'appel à une métaméthode	82
35. Peut-on parler de classe comme en C++ ?	82
36. Peut-on interdire la modification des métatables ?	82
37. Comment supprimer une métatable ?	83

Manipuler ses données et ses fichiers..... 84

15. Les chaînes de caractères	85
1. Déclarer des chaînes	86
2. Déclarer des chaînes sur plusieurs lignes	87
3. Concaténer des chaînes	87
4. Obtenir la taille d'une chaîne	88
5. Conversions automatiques	88
6. Le module intégré <code>string</code>	88
7. Formater des chaînes	89
8. Extraire des portions de chaînes	89
9. Convertir en minuscules ou majuscules	90
10. Récupérer les codes de caractères d'une chaîne	90
11. Créer une chaîne à partir de codes de caractères	91
12. Répéter une chaîne	91
13. Inverser une chaîne	91
14. Transformer une fonction en chaîne	92
15. Et l'Unicode ?	92
16. Lua et l'Unicode	93
16. Créer une chaîne UTF-8 à partir de codes	93
17. Parcours des chaînes UTF-8	93
18. Obtenir les points de code d'une chaîne	94
19. Longueur d'une chaîne UTF-8	94
20. Obtenir la position d'un caractère Unicode	94
21. Extraire un pattern UTF-8	95
17. Sérialisation et désérialisation de données.....	96

22. Srialiser des données	96
23. Srialiser des données	98
24. Désrialiser des données	98
18. Recherche de motifs dans des chaînes (pattern matching)	99
25. Manipuler des chaînes à l'aide de motifs	99
26. Les classes de caractères dans les motifs	100
27. Construire des motifs complets	102
28. Extraire des portions de chaînes	103
29. Rechercher un motif, avec indices	103
30. Rechercher simplement un motif	104
31. Itérer sur la recherche d'un motif	104
32. Remplacer un motif	105
19. La librairie LPeg	108
33. Utiliser LPeg	108
34. Vérifier un motif	109
35. Déclarer un motif	110
<code>lpeg.P</code>	110
<code>lpeg.S</code> et <code>lpeg.R</code>	110
36. Utiliser des jeux de caractères prédéfinis	111
37. Préciser le nombre d'occurrences	112
38. Combiner des motifs	113
39. Rechercher un motif n'importe où dans une chaîne	114
40. Exemple de traitement avec LPeg d'une expression régulière complexe	115
41. Extraire des séquences	118
42. Capturer un motif simple	118
43. Capturer la position du caractère suivant le motif	119
44. Capturer un motif n'importe où dans une chaîne	119
45. Insérer des valeurs constantes	120
46. Retourner un argument de <code>lpeg.match()</code>	121
47. Appliquer une fonction à la capture	121
48. Grouper les valeurs capturées	122
49. Récupérer les valeurs du dernier groupe nommé	122
50. Récupérer les captures dans une table	123
51. Combiner des captures	124
52. Que signifie l'expression <i>motif/valeur</i> ?.....	126
53. Remplacer la capture par une valeur	128
54. Capturer en temps réel	129

55. Créer une grammaire	131
56. Comment mettre au point les motifs ?.....	134
57. Le module RE	140
20. Calculs mathématiques	141
58. Conversions entre flottants et entiers	141
59. Tronquer ou arrondir des nombres	142
60. Fonctions trigonométriques	143
61. Fonctions de conversions angulaires	143
62. Fonctions logarithmiques et exponentielles	143
63. Conversion de fractions normalisées	144
64. Mettre à la puissance et calculer des racines carrées	144
65. Obtenir des minimum et maximum	144
66. Les constantes mathématiques	145
67. Calculer la valeur absolue	145
68. Générer des nombres aléatoires	145
69. Extraire une partie entière et fractionnaire	145
70. Calcul de modulo réel	146
21. Calculs logiques	147
71. Les opérations logiques usuelles	147
72. Test logique	148
73. Champs de bits	148
74. Décalages logiques	149
75. Rotations logiques	149
76. Décalage arithmétique	150
22. Gestion des fichiers	151
77. Généralités sur les entrées/sorties	151
78. Comment lire un fichier ?.....	152
79. Lire un petit fichier texte	152
80. Lire un petit fichier binaire	152
81. Peut-on faire une lecture plus fine ?.....	153
82. Lire un fichier texte ligne par ligne	154
83. Écrire un petit fichier	155
84. Mettre à jour un fichier	155
85. Ajouter des lignes à un fichier	156
86. Écrire un gros fichier	156
87. Le modèle simple de io	157
23. Le module LFS et ses utilisations	159

88. Lister les fichiers d'un répertoire	159
89. Lister les fichiers et les sous-répertoires d'un répertoire	160
90. Manipuler des répertoires	162
91. Manipuler des fichiers	162
92. Verrouiller répertoires et fichiers	163

S'interfacer avec le monde extérieur..... 164

24. Les fonctions d'interfaçage avec l'OS.....	165
1. Récupérer la date système avec <code>os.date()</code>	165
2. Récupérer la date système avec <code>os.time()</code>	166
3. Récupérer le temps CPU consommé	166
4. Calculer un écart de dates	166
5. Exécuter des commandes externes	166
6. Forcer la sortie d'un programme	167
7. Récupérer une variable d'environnement	167
8. Supprimer un fichier	168
9. Renommer un fichier	168
10. Changer les paramètres de localisation	168
11. Créer un fichier temporaire	169
25. Lua et POSIX	170
12. Quelles sont les fonctions disponibles ?.....	170
13. Quelles sont les fonctions <i>curses</i> disponibles ?.....	172
14. Comment utiliser <i>curses</i> ?.....	172
26. Les bases de données	174
15. Lua comme format de base de données	175
16. Exploiter des données CSV / TSV	177
17. Importer des fichiers JSON	179
18. Utiliser des fichiers YAML	180
19. Traiter du XML	180
20. Communiquer avec des bases SQL	181
21. LuaSQLite 3	184
22. Communiquer avec des bases NoSQL	185
27. Le réseau	187
23. <i>LuaSocket</i>	187
24. Créer un serveur TCP/IP	188
25. Créer un client TCP/IP	188
26. Envoyer et recevoir des données UDP	189

27. Effectuer des recherches DNS	190
28. Effectuer les opérations d'un client FTP	190
29. Le module <i>ltn12</i>	191
30. Récupérer le contenu d'une page web	191
31. Autres fonctions de <i>LuaSocket</i>	192
28. Les interfaces utilisateur graphiques	193
32. Qu'entend-on par interface utilisateur graphique ?.....	193
33. Quel système de GUI utiliser ?.....	194
34. Quelles liaisons avec Lua sont disponibles ?.....	195
35. Fonder sa GUI sur des composants natifs	195
36. Recourir à une boîte à outils de composants	196
37. Installer une librairie de type IUP, lqt ou tekUI	196
38. Écrire <i>Hello World!</i> avec IUP, lqt ou tekUI	197
39. Ajouter de l'interactivité avec IUP, lqt ou tekUI	199
29. Lua dans les jeux vidéo	201
40. Utiliser Lua dans le gameplay	201
41. Utiliser Lua pour les paramétrages du jeu	205
42. Utiliser Lua pour l'intelligence artificielle	206
S'interfacer avec le C	209
30. Les bases de l'API C.....	210
1. Changer la fonction d'allocation	211
2. Charger les modules de la librairie standard	212
3. Charger des scripts via l'API C.....	212
4. Opérations mathématiques depuis le C.....	213
5. Conversions de type depuis le C.....	213
6. Convertir une fonction en <i>chunk</i>	213
31. Manipulation de la pile d'appel	215
7. Modifier la pile	215
8. Comment savoir ce que contient la pile ?.....	217
9. Récupérer des variables globales	220
10. Comparer des données dans la pile	221
11. Concaténer des données dans la pile	221
12. Appeler une fonction	222
32. Manipulation des tables	224
13. Créer, modifier et lire une table	226

14. Parcourir une table avec l'API C.....	227
15. Utiliser les métatables depuis l'API C.....	227
16. Le registre Lua	228
33. Les fonctions C et les fermetures	230
17. Utiliser des fonctions C en Lua	230
18. Manipuler des fonctions C.....	231
19. Associer des valeurs à une fonction C.....	231
20. Manipuler les <i>upvalues</i> d'une fermeture	233
34. Les <i>userdata</i>	234
21. Qu'est-ce que les <i>userdata</i> ?.....	234
22. Qu'est-ce que les <i>lightuserdata</i> ?.....	234
23. Quand faut-il se servir des <i>userdata</i> ?.....	235
24. Créer une <i>userdata</i>	236
25. Échanger des <i>userdata</i> entre Lua et C	237
26. Associer une métatable à une <i>userdata</i>	237
27. Accès de type objet avec les <i>userdata</i>	239
28. Associer une table à une <i>userdata</i>	240
29. <i>userdata</i> et le ramasse-miettes ?.....	240
35. Utilisation avancée	241
30. Déboguer avec l'API C.....	241
31. Gérer les erreurs	241
32. Remonter une erreur	241
33. Gérer le ramasse-miettes	242
34. Créer et gérer des coroutines	242
35. Déplacer des données entre deux coroutines	244
36. Tester l'état de coroutines	245
36. La librairie auxiliaire	246
37. Quand utiliser la librairie auxiliaire ?.....	246
38. Créer un état Lua	248
39. Charger rapidement la librairie standard	248
40. Charger et exécuter un fichier Lua	248
41. Charger et exécuter une chaîne de caractères	249
42. Charger du code à partir d'un buffer	249
43. Vérifier et manipuler la pile d'appel	249
44. Facilité pour les énumérations C.....	251
45. Facilité pour les chaînes de caractères	251
46. Les tableaux et les métatables	251

47. Traiter les erreurs	252
48. Manipuler les buffers pour créer des chaînes de caractères	252
49. Vérifier dynamiquement la cohérence des versions	254
50. Facilité pour créer des librairies	255
51. Charger une librairie depuis le C	255
52. Obtenir une référence dans une table	255
Déboguer son code	256
37. Déboguer côté Lua	257
1. Quels outils pour déboguer du code Lua ?	257
2. Que peut-on faire avec le module <i>debug</i> ?	257
3. Débogage interactif	258
4. Obtenir des informations sur une fonction	258
5. Étudier ou modifier des variables locales	260
6. Étudier ou modifier des <i>upvalues</i>	261
7. Consulter et modifier la table associée à une <i>userdata</i>	262
8. Mettre en place une fonction d'hameçonnage	262
9. Afficher la pile d'appel	264
10. Manipuler les métatables	265
11. Récupérer le <i>registry</i>	265
38. Déboguer côté C	266
12. Comment déboguer ses scripts à partir du C ?	266
13. Obtenir des informations sur une fonction	266
14. Obtenir des informations sur la pile d'appel	267
15. Accéder à des variables locales	268
16. Accéder à des valeurs <i>upvalue</i>	268
17. Mettre en place une fonction d'hameçonnage	268
L'implémentation LuaJIT	270
39. L'implémentation LuaJIT	271
1. Quand LuaJIT est-il plus performant que Lua ?	271
2. Quelles sont les limitations de LuaJIT ?	272
3. Quelle compatibilité entre LuaJIT et les versions de Lua officielles ?	272
4. Quelles extensions apporte LuaJIT ?	273
5. Arrêter ou démarrer la compilation dynamique	273
6. Obtenir l'état de la compilation dynamique	274

7. Récupérer la version de LuaJIT	274
8. Récupérer des informations système	274
40. Le module FFI de LuaJIT	275
9. Quelle différence avec les mécanismes Lua ?.....	275
10. Principe d'utilisation de <i>FFI</i>	275
11. Déclarer les fonctions à importer	276
12. Comment les types sont-ils convertis ?.....	277
13. Comment résoudre les symboles déclarés avec <i>ffi.cdef()</i> ?.....	278
14. Peut-on déclarer des variables ou des types ?.....	279
15. Récupérer des informations sur la plate-forme	281
16. Peut-on utiliser des callbacks C ?.....	282
17. Peut-on associer des métatables au type <i>cdata</i> ?.....	284
18. Opérations sur les 64 bits sous LuaJIT	285
19. Obtenir des informations sur les variables <i>cdata</i>	286
20. Récupérer la valeur de <i>errno</i>	287
21. Créer une <i>string</i> depuis un pointeur C	287
22. Copier le contenu de variables <i>cdata</i> ou <i>string</i>	287
23. Initialiser le contenu d'un type <i>cdata</i>	287
41. Le module BitOp de LuaJIT	288
24. Conversion en chaînes hexadécimales	288
25. Opérations logiques usuelles	288
26. Décalages logiques	289
27. Rotations logiques	289
28. Décalage arithmétique	290
29. Changer l'ordre des octets	290
30. Retourner une valeur normalisée	290
31. Travailler avec des 64 bits	290
42. Le module de profiling de LuaJIT	292
32. À quoi ça sert ?.....	292
33. Comment invoquer le <i>profiler</i> ?.....	292
34. Cas d'utilisation	293
Annexe	296
Lexique	297
Liste des illustrations	307

Liste des tableaux	308
Liste des exemples	309
Index	310