

# Table des matières

---

Préface .....	xvii
Avant-propos .....	xix

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

### 1. Informations générales ..... 3

1. Installer Lua .....	3
2. Sur quels systèmes puis-je installer Lua ?.....	3
3. Quel éditeur pour développer en Lua ?.....	3
4. Comment Lua se situe-t-il par rapport aux autres langages en termes de performances ?.....	4
5. Quelles versions faut-il utiliser ?.....	4
6. Quels outils pour documenter son code ?.....	4
7. Quels sont les sites de référence pour développer en Lua ?.....	4
8. Comment passer de Lua 5.1 à Lua 5.2 ?.....	5
9. Comment passer de Lua 5.2 à Lua 5.3 ?.....	5

### 2. Principes et éléments de syntaxe ..... 7

10. De quoi est composé Lua ?.....	7
11. Les commentaires .....	8
12. Les commentaires sur plusieurs lignes .....	8
13. Les commentaires d'affichage .....	8
14. Les instructions .....	9
15. Les mots clés du langage .....	9
16. Afficher des résultats .....	9
17. Comment la mémoire est-elle allouée ou libérée ?.....	10
18. Comment connaître la version de Lua utilisée ?.....	10

### 3. Variables et types ..... 11

19. Quel est le type d'une variable ?.....	11
20. Quels sont les types de données disponibles ?.....	11
21. Le type <i>nil</i> .....	11
22. Le type <i>boolean</i> .....	12
23. Le type numérique .....	12
24. Le type <i>string</i> .....	12
25. Le type <i>function</i> .....	13
26. Le type <i>userdata</i> .....	13

27. Qu'est-ce qu'un <i>thread</i> Lua ?.....	13
28. Le type <i>table</i> .....	13
29. Un type peut-il être converti ?.....	13
30. Connaître le type d'une donnée .....	14
31. Quelle est la portée d'une variable ?.....	14
<b>4. Expressions et opérateurs .....</b>	<b>15</b>
32. Quels sont les opérateurs arithmétiques ?.....	15
33. Quels sont les opérateurs logiques binaires ?.....	15
34. Quels sont les opérateurs relationnels ?.....	16
35. Quels sont les opérateurs logiques disponibles ?.....	16
36. Autres opérateurs à connaître .....	17
<b>5. Portée des variables, blocs et chunks.....</b>	<b>19</b>
37. Quand utiliser <i>local</i> ?.....	21
38. Où sont stockées les variables globales ?.....	21
<b>6. Structures de contrôle .....</b>	<b>23</b>
39. Comment déclarer un bloc de code ?.....	23
40. Comment exprimer des conditions ?.....	23
41. Les expressions ternaires existent-elles comme en C/C++ ? .....	24
42. Comment faire un <i>switch/case</i> ?.....	24
43. Comment effectuer des boucles ?.....	24
44. Utiliser les boucles de type <i>while</i> .....	25
45. Utiliser les boucles de type <i>repeat</i> .....	25
46. Utiliser les boucles de type <i>for</i> .....	25
47. Créer une boucle <i>for</i> numérique .....	26
48. Créer une boucle <i>for</i> générique .....	27
49. Comment sortir d'une boucle ?.....	28
50. Peut-on faire un <i>try/catch</i> comme en Java ou C++ ? .....	28
51. Existe-il une instruction <i>continue</i> ?.....	29
52. Effectuer un saut inconditionnel avec <i>goto</i> .....	29
<b>7. Fonctions .....</b>	<b>31</b>
53. Nombre inconsistant d'arguments et de valeurs .....	32
54. Nommer des arguments pour les sélectionner lors de l'appel.....	32
55. Déclarer une liste variable d'arguments .....	33
56. Renvoyer plusieurs résultats .....	34
57. Que sont les fonctions anonymes ?.....	34
58. Que sont les <i>closures</i> ?.....	34
<b>8. Tables .....</b>	<b>37</b>

59. Construire une table et y accéder .....	37
60. Initialiser une table .....	38
61. Créer et utiliser un enregistrement .....	39
62. Comment obtenir la taille d'une table ?.....	39
63. Pourquoi une indexation numérique à partir de 1 ?.....	40
64. Table de fonctions .....	40
65. Instancier des objets avec les tables .....	41
66. Parcourir une table avec un <code>for</code> générique .....	42
67. Copier une table .....	43
68. Que sont les <i>weak tables</i> ?.....	43
69. Utiliser des tables pour faire des matrices .....	44
70. Le module <code>table</code> .....	44
71. La fonction <code>table.concat</code> .....	45
72. Insérer une valeur dans une séquence .....	45
73. Supprimer une valeur dans une séquence .....	45
74. Créer un tableau depuis une liste de valeurs .....	46
75. Extraire une liste de valeurs d'une séquence .....	46
76. Trier une séquence .....	46
77. Déplacer des éléments d'une table vers une autre .....	47
<b>9. Les fonctions internes .....</b>	<b>49</b>
78. <code>assert()</code> .....	49
79. <code>collectgarbage()</code> .....	50
80. <code>dofile()</code> .....	52
81. <code>error()</code> .....	52
82. <code>setmetatable()/getmetatable()</code> .....	52
83. <code>ipairs()/pairs()</code> ?.....	52
84. <code>next()</code> .....	53
85. <code>loadfile()/load()</code> .....	53
86. <code>pcall()/xpcall()</code> .....	53
87. <code>print()</code> .....	54
88. <code>rawequal()/rawget()/rawlen()/rawset()</code> .....	54
89. <code>require()</code> .....	54
90. <code>select()</code> .....	54
91. <code>tonumber()</code> .....	55
92. <code>tostring()</code> .....	55
93. <code>type()</code> .....	55
<b>10. La gestion des erreurs .....</b>	<b>57</b>
94. Quelles sont les erreurs générées ?.....	57

95. Remonter des erreurs .....	57
<b>11. Les coroutines .....</b>	<b>59</b>
96. Créer une coroutine .....	59
97. Connaître l'état des coroutines .....	61
98. Que fait la fonction <code>coroutine.wrap()</code> ?.....	62
<b>Lua, bibliothèques et modules .....</b>	<b>65</b>
<b>12. Appeler et exécuter du code externe.....</b>	<b>67</b>
1. Charger et exécuter un script .....	67
2. Charger un script à partir d'une chaîne de caractères .....	68
3. Charger un script à partir d'un fichier .....	69
4. Créer une bibliothèque .....	70
5. Gérer un fichier de configuration .....	70
6. Contrôler le code chargé par les fonctions <code>load()/loadfile()</code> .....	71
<b>13. Créer ses bibliothèques .....</b>	<b>73</b>
7. Le module <code>package</code> .....	73
8. Appeler une bibliothèque avec <code>require()</code> .....	73
9. Comment les modules sont-ils recherchés ?.....	73
10. Changer les chemins de recherche .....	75
11. La variable <code>package.config</code> .....	75
12. Écrire un module Lua destiné à être chargé par <code>require()</code> .....	76
13. Écrire un module C destiné à être chargé par <code>require()</code> .....	78
14. Accéder depuis Lua à une bibliothèque C ou C++.....	78
15. Déclarer le <code>loader</code> d'une bibliothèque .....	79
16. À quoi sert <code>package.loadlib()</code> ?.....	79
<b>14. Les métatables .....</b>	<b>81</b>
17. Quels opérateurs peut-on modifier avec les métatables ?.....	81
18. Peut-on ajouter de nouveaux opérateurs aux métatables ?.....	82
19. À quoi cela peut-il servir ?.....	82
20. Utiliser la métaméthode multiplication ( <code>__mul</code> ) .....	85
21. Utiliser la métaméthode division ( <code>__div</code> ) .....	86
22. Utiliser la métaméthode modulo ( <code>__mod</code> ) .....	86
23. Utiliser la métaméthode de négation ( <code>__unm</code> ) .....	87
24. Utiliser la mise à la puissance ( <code>__pow</code> ) .....	87
25. Utiliser la métaméthode de concaténation .....	88
26. Utiliser les métaméthodes de comparaison .....	88
27. Utiliser la métaméthode de ramasse-miettes .....	89

28. Accéder à un indice avec <code>__index</code> .....	90
29. Simuler une classe avec des méthodes .....	90
30. Créer un index avec <code>__newindex</code> .....	91
31. Retourner la taille d'une valeur avec <code>__len</code> .....	92
32. Capturer un appel de fonction avec <code>__call</code> .....	92
33. Modifier les fonctions <code>pairs()</code> et <code>ipairs()</code> .....	92
34. Éviter l'appel à une métaméthode .....	93
35. Peut-on parler de classe comme en C++ ? .....	93
36. Peut-on interdire la modification des métatables ? .....	93
37. Comment supprimer une métatable ? .....	94

## Manipuler ses données et ses fichiers..... 95

<b>15. Les chaînes de caractères .....</b>	<b>97</b>
1. Déclarer des chaînes .....	98
2. Déclarer des chaînes sur plusieurs lignes .....	99
3. Concaténer des chaînes .....	99
4. Obtenir la taille d'une chaîne .....	100
5. Conversions automatiques .....	100
6. Le module intégré <code>string</code> .....	100
7. Formater des chaînes .....	101
8. Extraire des portions de chaînes .....	101
9. Convertir en minuscules ou majuscules .....	102
10. Récupérer les codes de caractères d'une chaîne .....	102
11. Créer une chaîne à partir de codes de caractères .....	103
12. Répéter une chaîne .....	103
13. Inverser une chaîne .....	103
14. Transformer une fonction en chaîne .....	104
15. Et l'Unicode ? .....	104
<b>16. Lua et l'Unicode .....</b>	<b>105</b>
16. Créer une chaîne UTF-8 à partir de codes .....	105
17. Parcours des chaînes UTF-8 .....	105
18. Obtenir les points de code d'une chaîne .....	106
19. Longueur d'une chaîne UTF-8 .....	106
20. Obtenir la position d'un caractère Unicode .....	106
21. Extraire un pattern UTF-8 .....	107
<b>17. Sérialisation et désérialisation de données.....</b>	<b>109</b>
22. Sérialiser des données .....	109

23. Sérialiser des données .....	111
24. Désérialiser des données .....	111
<b>18. Recherche de motifs dans des chaînes (pattern matching) .....</b>	<b>113</b>
25. Manipuler des chaînes à l'aide de motifs .....	113
26. Les classes de caractères dans les motifs .....	114
27. Construire des motifs complets .....	116
28. Extraire des portions de chaînes .....	117
29. Rechercher un motif, avec indices .....	117
30. Rechercher simplement un motif .....	118
31. Itérer sur la recherche d'un motif .....	118
32. Remplacer un motif .....	119
<b>19. La librairie LPeg .....</b>	<b>123</b>
33. Utiliser LPeg .....	123
34. Vérifier un motif .....	124
35. Déclarer un motif .....	125
<i>lpeg.P</i> .....	125
<i>lpeg.S</i> et <i>lpeg.R</i> .....	125
36. Utiliser des jeux de caractères prédéfinis .....	126
37. Préciser le nombre d'occurrences .....	127
38. Combiner des motifs .....	128
39. Rechercher un motif n'importe où dans une chaîne .....	129
40. Exemple de traitement avec LPeg d'une expression régulière complexe .....	130
41. Extraire des séquences .....	133
42. Capturer un motif simple .....	133
43. Capturer la position du caractère suivant le motif .....	134
44. Capturer un motif n'importe où dans une chaîne .....	134
45. Insérer des valeurs constantes .....	135
46. Retourner un argument de <i>lpeg.match()</i> .....	136
47. Appliquer une fonction à la capture .....	136
48. Grouper les valeurs capturées .....	137
49. Récupérer les valeurs du dernier groupe nommé .....	137
50. Récupérer les captures dans une table .....	138
51. Combiner des captures .....	139
52. Que signifie l'expression <i>motif/valeur ?</i> .....	141
53. Remplacer la capture par une valeur .....	143
54. Capturer en temps réel .....	144
55. Créer une grammaire .....	146

56. Comment mettre au point les motifs ?	149
57. Le module RE	155
<b>20. Calculs mathématiques</b>	<b>157</b>
58. Conversions entre flottants et entiers	157
59. Tronquer ou arrondir des nombres	158
60. Fonctions trigonométriques	159
61. Fonctions de conversions angulaires	159
62. Fonctions logarithmiques et exponentielles	159
63. Conversion de fractions normalisées	160
64. Mettre à la puissance et calculer des racines carrées	160
65. Obtenir des minimum et maximum	160
66. Les constantes mathématiques	161
67. Calculer la valeur absolue	161
68. Générer des nombres aléatoires	161
69. Extraire une partie entière et fractionnaire	161
70. Calcul de modulo réel	162
<b>21. Calculs logiques</b>	<b>163</b>
71. Les opérations logiques usuelles	163
72. Test logique	164
73. Champs de bits	164
74. Décalages logiques	165
75. Rotations logiques	165
76. Décalage arithmétique	166
<b>22. Gestion des fichiers</b>	<b>167</b>
77. Généralités sur les entrées/sorties	167
78. Comment lire un fichier ?	168
79. Lire un petit fichier texte	168
80. Lire un petit fichier binaire	168
81. Peut-on faire une lecture plus fine ?	169
82. Lire un fichier texte ligne par ligne	170
83. Écrire un petit fichier	171
84. Mettre à jour un fichier	171
85. Ajouter des lignes à un fichier	172
86. Écrire un gros fichier	172
87. Le modèle simple de <i>io</i>	173
<b>23. Le module LFS et ses utilisations</b>	<b>175</b>
88. Lister les fichiers d'un répertoire	175

89. Lister les fichiers et les sous-répertoires d'un répertoire .....	176
90. Manipuler des répertoires .....	178
91. Manipuler des fichiers .....	178
92. Verrouiller répertoires et fichiers .....	179
<b>S'interfacer avec le monde extérieur.....</b>	<b>181</b>
<b>24. Les fonctions d'interfaçage avec l'OS.....</b>	<b>183</b>
1. Récupérer la date système avec <i>os.date()</i> .....	183
2. Récupérer la date système avec <i>os.time()</i> .....	184
3. Récupérer le temps CPU consommé .....	184
4. Calculer un écart de dates .....	184
5. Exécuter des commandes externes .....	184
6. Forcer la sortie d'un programme .....	185
7. Récupérer une variable d'environnement .....	185
8. Supprimer un fichier .....	186
9. Renommer un fichier .....	186
10. Changer les paramètres de localisation .....	186
11. Créer un fichier temporaire .....	187
<b>25. Lua et POSIX .....</b>	<b>189</b>
12. Quelles sont les fonctions disponibles ?.....	189
13. Quelles sont les fonctions <i>curses</i> disponibles ?.....	191
14. Comment utiliser <i>curses</i> ?.....	191
<b>26. Les bases de données .....</b>	<b>193</b>
15. Lua comme format de base de données .....	194
16. Exploiter des données CSV / TSV .....	196
17. Importer des fichiers JSON .....	198
18. Utiliser des fichiers YAML .....	199
19. Traiter du XML .....	199
20. Communiquer avec des bases SQL .....	200
21. LuaSQLite 3 .....	203
22. Communiquer avec des bases NoSQL .....	204
<b>27. Le réseau .....</b>	<b>207</b>
23. <i>LuaSocket</i> .....	207
24. Créer un serveur TCP/IP .....	208
25. Créer un client TCP/IP .....	208
26. Envoyer et recevoir des données UDP .....	209
27. Effectuer des recherches DNS .....	210

28. Effectuer les opérations d'un client FTP .....	210
29. Le module <i>ltn12</i> .....	211
30. Récupérer le contenu d'une page web .....	211
31. Autres fonctions de <i>LuaSocket</i> .....	212
<b>28. Les interfaces utilisateur graphiques .....</b>	<b>213</b>
32. Qu'entend-on par interface utilisateur graphique ?.....	213
33. Quel système de GUI utiliser ?.....	214
34. Quelles liaisons avec Lua sont disponibles ?.....	215
35. Fonder sa GUI sur des composants natifs .....	215
36. Recourir à une boîte à outils de composants .....	216
37. Installer une librairie de type IUP, Iqt ou tekUI .....	216
38. Écrire <i>Hello World!</i> avec IUP, Iqt ou tekUI .....	217
39. Ajouter de l'interactivité avec IUP, Iqt ou tekUI .....	219
<b>29. Lua dans les jeux vidéo .....</b>	<b>221</b>
40. Utiliser Lua dans le gameplay .....	221
41. Utiliser Lua pour les paramètres du jeu .....	225
42. Utiliser Lua pour l'intelligence artificielle .....	226
<b>S'interfacer avec le C .....</b>	<b>229</b>
<b>30. Les bases de l'API C.....</b>	<b>231</b>
1. Changer la fonction d'allocation .....	232
2. Charger les modules de la librairie standard .....	233
3. Charger des scripts via l'API C.....	233
4. Opérations mathématiques depuis le C.....	234
5. Conversions de type depuis le C.....	234
6. Convertir une fonction en <i>chunk</i> .....	234
<b>31. Manipulation de la pile d'appel .....</b>	<b>237</b>
7. Modifier la pile .....	237
8. Comment savoir ce que contient la pile ?.....	239
9. Récupérer des variables globales .....	242
10. Comparer des données dans la pile .....	243
11. Concaténer des données dans la pile .....	243
12. Appeler une fonction .....	244
<b>32. Manipulation des tables .....</b>	<b>247</b>
13. Créer, modifier et lire une table .....	249
14. Parcourir une table avec l'API C.....	250

15. Utiliser les métatables depuis l'API C.....	250
16. Le registre Lua .....	251
<b>33. Les fonctions C et les fermetures .....</b>	<b>253</b>
17. Utiliser des fonctions C en Lua .....	253
18. Manipuler des fonctions C.....	254
19. Associer des valeurs à une fonction C.....	254
20. Manipuler les <i>upvalues</i> d'une fermeture .....	256
<b>34. Les <i>userdata</i> .....</b>	<b>257</b>
21. Qu'est-ce que les <i>userdata</i> ?.....	257
22. Qu'est-ce que les <i>lightuserdata</i> ?.....	257
23. Quand faut-il se servir des <i>userdata</i> ?.....	258
24. Créer une <i>userdata</i> .....	259
25. Échanger des <i>userdata</i> entre Lua et C .....	260
26. Associer une métatable à une <i>userdata</i> .....	260
27. Accès de type objet avec les <i>userdata</i> .....	262
28. Associer une table à une <i>userdata</i> .....	263
29. <i>userdata</i> et le ramasse-miettes ?.....	263
<b>35. Utilisation avancée .....</b>	<b>265</b>
30. Déboguer avec l'API C.....	265
31. Gérer les erreurs .....	265
32. Remonter une erreur .....	265
33. Gérer le ramasse-miettes .....	266
34. Créer et gérer des coroutines .....	266
35. Déplacer des données entre deux coroutines .....	268
36. Tester l'état de coroutines .....	269
<b>36. La librairie auxiliaire .....</b>	<b>271</b>
37. Quand utiliser la librairie auxiliaire ?.....	271
38. Créer un état Lua .....	273
39. Charger rapidement la librairie standard .....	273
40. Charger et exécuter un fichier Lua .....	273
41. Charger et exécuter une chaîne de caractères .....	274
42. Charger du code à partir d'un buffer .....	274
43. Vérifier et manipuler la pile d'appel .....	274
44. Facilité pour les énumérations C.....	276
45. Facilité pour les chaînes de caractères .....	276
46. Les tableaux et les métatables .....	276
47. Traiter les erreurs .....	277

48. Manipuler les buffers pour créer des chaînes de caractères.....	277
49. Vérifier dynamiquement la cohérence des versions.....	279
50. Facilité pour créer des librairies .....	280
51. Charger une librairie depuis le C.....	280
52. Obtenir une référence dans une table .....	280
<b>Déboguer son code .....</b>	<b>281</b>
<b>37. Déboguer côté Lua .....</b>	<b>283</b>
1. Quels outils pour déboguer du code Lua ?.....	283
2. Que peut-on faire avec le module <i>debug</i> ?.....	283
3. Débogage interactif .....	284
4. Obtenir des informations sur une fonction .....	284
5. Étudier ou modifier des variables locales .....	286
6. Étudier ou modifier des <i>upvalues</i> .....	287
7. Consulter et modifier la table associée à une <i>userdata</i> .....	288
8. Mettre en place une fonction d'hameçonnage .....	288
9. Afficher la pile d'appel .....	290
10. Manipuler les métatables .....	291
11. Récupérer le <i>registry</i> .....	291
<b>38. Déboguer côté C .....</b>	<b>293</b>
12. Comment déboguer ses scripts à partir du C ?.....	293
13. Obtenir des informations sur une fonction .....	293
14. Obtenir des informations sur la pile d'appel .....	294
15. Accéder à des variables locales .....	295
16. Accéder à des valeurs <i>upvalue</i> .....	295
17. Mettre en place une fonction d'hameçonnage .....	295
<b>L'implémentation LuaJIT .....</b>	<b>297</b>
<b>39. L'implémentation LuaJIT .....</b>	<b>299</b>
1. Quand LuaJIT est-il plus performant que Lua ?.....	299
2. Quelles sont les limitations de LuaJIT ?.....	300
3. Quelle compatibilité entre LuaJIT et les versions de Lua officielles ?.....	300
4. Quelles extensions apporte LuaJIT ?.....	301
5. Arrêter ou démarrer la compilation dynamique .....	301
6. Obtenir l'état de la compilation dynamique .....	302
7. Récupérer la version de LuaJIT .....	302

8. Récupérer des informations système .....	302
<b>40. Le module FFI de LuaJIT .....</b>	<b>303</b>
9. Quelle différence avec les mécanismes Lua ?.....	303
10. Principe d'utilisation de <i>FFI</i> .....	303
11. Déclarer les fonctions à importer .....	304
12. Comment les types sont-ils convertis ?.....	305
13. Comment résoudre les symboles déclarés avec <i>ffi.cdef()</i> ?.....	306
14. Peut-on déclarer des variables ou des types ?.....	307
15. Récupérer des informations sur la plate-forme .....	309
16. Peut-on utiliser des callbacks C ?.....	310
17. Peut-on associer des métatables au type <i>cdata</i> ?.....	312
18. Opérations sur les 64 bits sous LuaJIT .....	313
19. Obtenir des informations sur les variables <i>cdata</i> .....	314
20. Récupérer la valeur de <i>errno</i> .....	315
21. Créer une <i>string</i> depuis un pointeur C .....	315
22. Copier le contenu de variables <i>cdata</i> ou <i>string</i> .....	315
23. Initialiser le contenu d'un type <i>cdata</i> .....	315
<b>41. Le module BitOp de LuaJIT .....</b>	<b>317</b>
24. Conversion en chaînes hexadécimales .....	317
25. Opérations logiques usuelles .....	317
26. Décalages logiques .....	318
27. Rotations logiques .....	318
28. Décalage arithmétique .....	319
29. Changer l'ordre des octets .....	319
30. Retourner une valeur normalisée .....	319
31. Travailler avec des 64 bits .....	319
<b>42. Le module de profiling de LuaJIT .....</b>	<b>321</b>
32. À quoi ça sert ?.....	321
33. Comment invoquer le <i>profiler</i> ?.....	321
34. Cas d'utilisation .....	322
<b>Annexe .....</b>	<b>325</b>
<b>Lexique .....</b>	<b>327</b>
Index .....	337
À propos des auteurs .....	353