Table des matières

À propos des auteurs xvii
Avant-propos xix

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

2. Principes et éléments de syntaxe 5
9. De quoi est composé Lua ? 5
10. Les commentaires 6
11. Les commentaires sur plusieurs lignes 6
12. Les commentaires d'affichage 6
13. Les instructions 7
14. Les mots clés du langage 7
15. Afficher des résultats 7
16. Comment la mémoire est-elle allouée ou libérée ? 8
17. Comment connaître la version de Lua utilisée ? 8

3. Variables et types 9
18. Quel est le type d'une variable ? 9
19. Quels sont les types de données disponibles ? 9
20. Le type *nil* 9
21. Le type *boolean* 10
22. Le type numérique 10
23. Le type *string* 10
24. Le type *function* 11
25. Le type *userdata* 11
26. Qu'est-ce qu'un *thread* Lua ? 11

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

27. Le type *table* 11
28. Un type peut-il être converti ?..... 11
29. Connaître le type d'une donnée 12
30. Quelle est la portée d'une variable ?..... 12

4. Expressions et opérateurs 13
31. Quels sont les opérateurs arithmétiques ?..... 13
32. Quels sont les opérateurs relationnels ?..... 13
33. Quels sont les opérateurs logiques disponibles ?..... 14
34. Autres opérateurs à connaître 15

5. Portée des variables, blocs et chunks..... 16
35. Quand utiliser *local* ?..... 18
36. Où sont stockées les variables globales ?..... 18

6. Structures de contrôle 20
37. Comment déclarer un bloc de code ?..... 20
38. Comment exprimer des conditions ?..... 20
39. Les expressions ternaires existent-elles comme en C/C++ ? 21
40. Comment faire un *switch/case* ?..... 21
41. Comment effectuer des boucles ?..... 21
42. Utiliser les boucles de type *while* 22
43. Utiliser les boucles de type *repeat* 22
44. Utiliser les boucles de type *for* 22
45. Créer une boucle *for* numérique 23
46. Créer une boucle *for* générique 24
47. Comment sortir d'une boucle ?..... 25
48. Peut-on faire un *try/catch* comme en Java ou C++ ? 25
49. Existe-il une instruction *continue* ?..... 26
50. Effectuer un saut inconditionnel avec *goto* 26

7. Fonctions 27
51. Nombre inconsistant d'arguments et de valeurs 28
52. Nommer des arguments pour les sélectionner lors de l'appel..... 28
53. Déclarer une liste variable d'arguments 29
54. Renvoyer plusieurs résultats 30
55. Que sont les fonctions anonymes ?..... 30
56. Que sont les *closures* ?..... 30

8. Tables 32
57. Construire une table et y accéder 32
58. Initialiser une table 33

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

59. Créer et utiliser un enregistrement 34
60. Comment obtenir la taille d'une table ?..... 34
61. Pourquoi une indexation numérique à partir de 1 ?..... 35
62. Table de fonctions 35
63. Instancier des objets avec les tables 36
64. Parcourir une table avec un *for* générique 37
65. Copier une table 38
66. Que sont les *weak tables* ?..... 38
67. Utiliser des tables pour faire des matrices 39
68. Le module *table* 39
69. La fonction *table.concat* 40
70. Insérer une valeur dans une séquence 40
71. Supprimer une valeur dans une séquence 40
72. Créer un tableau depuis une liste de valeurs 41
73. Extraire une liste de valeurs d'une séquence 41
74. Trier une séquence 41

9. Les fonctions internes 43
75. *assert()* 43
76. *collectgarbage()* 44
77. *dofile()* 46
78. *error()* 46
79. *setmetatable()/getmetatable()* 46
80. *ipairs()/pairs()* ?..... 46
81. *next()* 47
82. *loadfile()/load()* 47
83. *pcall()/xpcall()* 47
84. *print()* 48
85. *rawequal()/rawget()/rawlen()/rawset()* 48
86. *require()* 48
87. *select()* 48
88. *tonumber()* 49
89. *tostring()* 49
90. *type()* 49

10. La gestion des erreurs 50
91. Quelles sont les erreurs générées ?..... 50
92. Remonter des erreurs 50

11. Les coroutines 51
93. Créer une coroutine 51

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

94. Connaître l'état des coroutines 53
95. Que fait la fonction *coroutine.wrap()* ?..... 54

Lua, librairies 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 librairie 60
5. Gérer un fichier de configuration 60
6. Contrôler le code chargé par les fonctions *load()*/*loadfile()* 61

13. Créer ses librairies 62
7. Le module *package* 62
8. Appeler une librairie avec *require()* 62
9. Comment les modules sont-ils recherchés ?..... 62
10. Changer les chemins de recherche 64
11. La variable *package.config* 64
12. Écrire un module Lua destiné à être chargé par *require()* 65
13. Écrire un module C destiné à être chargé par *require()* 67
14. Accéder depuis Lua à une librairie C ou C++..... 67
15. Déclarer le *loader* d'une librairie 68
16. À quoi sert *package.loadlib()* ?..... 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 (*__mul*) 74
21. Utiliser la métaméthode division (*__div*) 75
22. Utiliser la métaméthode modulo (*__mod*) 75
23. Utiliser la métaméthode de négation (*__unm*) 76
24. Utiliser la mise à la puissance (*__pow*) 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 *__index* 79
29. Simuler une classe avec des méthodes 79
30. Créer un index avec *__newindex* 80

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

31. Retourner la taille d'une valeur avec *__len* 81
32. Capturer un appel de fonction avec *__call* 81
33. Modifier les fonctions *pairs()* et *ipairs()* 81
34. Éviter l'appel à une métaméthode 81
35. Peut-on parler de classe comme en C++ ? 82
36. Peut-on interdire la modification des métatables ?..... 82
37. Comment supprimer une métatable ?..... 82

Manipuler ses données et ses fichiers..... 83

15. Les chaînes de caractères 84
1. Déclarer des chaînes 85
2. Déclarer des chaînes sur plusieurs lignes 86
3. Concaténer des chaînes 86
4. Obtenir la taille d'une chaîne 87
5. Conversions automatiques 87
6. Le module intégré *string* 87
7. Formater des chaînes 88
8. Extraire des portions de chaînes 88
9. Convertir en minuscules ou majuscules 89
10. Récupérer les codes de caractères d'une chaîne 89
11. Créer une chaîne à partir de codes de caractères 90
12. Répéter une chaîne 90
13. Inverser une chaîne 90
14. Transformer une fonction en chaîne 91
15. Et l'Unicode ?..... 91

16. Recherche de motifs dans des chaînes (*pattern matching*) 92
16. Manipuler des chaînes à l'aide de motifs 92
17. Les classes de caractères dans les motifs 93
18. Construire des motifs complets 95
19. Extraire des portions de chaînes 96
20. Rechercher un motif, avec indices 96
21. Rechercher simplement un motif 97
22. Itérer sur la recherche d'un motif 97
23. Remplacer un motif 98

17. La librairie LPeg 101
24. Utiliser LPeg 101
25. Vérifier un motif 102

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

26. Déclarer un motif 103
lpeg.P 103
lpeg.S et *lpeg.R* 103
27. Utiliser des jeux de caractères prédéfinis 104
28. Préciser le nombre d'occurrences 105
29. Combiner des motifs 106
30. Rechercher un motif n'importe où dans une chaîne 107
31. Exemple de traitement avec LPeg d'une expression régulière complexe 108
32. Extraire des séquences 111
33. Capturer un motif simple 111
34. Capturer la position du caractère suivant le motif 112
35. Capturer un motif n'importe où dans une chaîne 112
36. Insérer des valeurs constantes 113
37. Retourner un argument de *lpeg.match()* 114
38. Appliquer une fonction à la capture 114
39. Grouper les valeurs capturées 115
40. Récupérer les valeurs du dernier groupe nommé 115
41. Récupérer les captures dans une table 116
42. Combiner des captures 117
43. Que signifie l'expression *motif/valeur* ?..... 119
44. Remplacer la capture par une valeur 121
45. Capturer en temps réel 122
46. Créer une grammaire 124
47. Comment mettre au point les motifs ?..... 127
48. Le module RE 133

18. Calculs mathématiques 134
49. Tronquer ou arrondir des nombres 134
50. Fonctions trigonométriques 135
51. Fonctions de conversions angulaires 135
52. Fonctions logarithmiques et exponentielles 135
53. Conversion de fractions normalisées 136
54. Mettre à la puissance et calculer des racines carrées 136
55. Obtenir des minimum et maximum 136
56. Les constantes mathématiques 137
57. Calculer la valeur absolue 137
58. Générer des nombres aléatoires 137
59. Extraire une partie entière et fractionnaire 137
60. Calcul de modulo réel 138

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

19. Calculs logiques 139
61. Les opérations logiques usuelles 139
62. Test logique 140
63. Champs de bits 140
64. Décalages logiques 141
65. Rotations logiques 141
66. Décalage arithmétique 142

20. Gestion des fichiers 143
67. Généralités sur les entrées/sorties 143
68. Comment lire un fichier ?..... 144
69. Lire un petit fichier texte 144
70. Lire un petit fichier binaire 144
71. Peut-on faire une lecture plus fine ?..... 145
72. Lire un fichier texte ligne par ligne 146
73. Écrire un petit fichier 147
74. Mettre à jour un fichier 147
75. Ajouter des lignes à un fichier 148
76. Écrire un gros fichier 148
77. Le modèle simple de *io* 149

21. Le module LFS et ses utilisations 151
78. Lister les fichiers d'un répertoire 151
79. Lister les fichiers et les sous-répertoires d'un répertoire 152
80. Manipuler des répertoires 154
81. Manipuler des fichiers 154
82. Verrouiller répertoires et fichiers 155

S'interfacer avec le monde extérieur..... 156

22. Les fonctions d'interfaçage avec l'OS..... 157
1. Récupérer la date système avec *os.date()* 157
2. Récupérer la date système avec *os.time()* 158
3. Récupérer le temps CPU consommé 158
4. Calculer un écart de dates 158
5. Exécuter des commandes externes 158
6. Forcer la sortie d'un programme 159
7. Récupérer une variable d'environnement 159
8. Supprimer un fichier 160
9. Renommer un fichier 160

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

10. Changer les paramètres de localisation 160
11. Créer un fichier temporaire 161

23. Lua et *POSIX* 162
12. Quelles sont les fonctions disponibles ?..... 162
13. Quelles sont les fonctions *curses* disponibles ?..... 164
14. Comment utiliser *curses* ?..... 164

24. Les bases de données 166
15. Lua comme format de base de données 167
16. Exploiter des données CSV / TSV 169
17. Importer des fichiers JSON 171
18. Utiliser des fichiers YAML 172
19. Traiter du XML 172
20. Communiquer avec des bases SQL 173
21. LuaSQLite 3 176
22. Communiquer avec des bases NoSQL 177

25. Le réseau 178
23. *LuaSocket* 178
24. Créer un serveur TCP/IP 179
25. Créer un client TCP/IP 179
26. Envoyer et recevoir des données UDP 180
27. Effectuer des recherches DNS 181
28. Effectuer les opérations d'un client FTP 181
29. Le module *ltn12* 182
30. Récupérer le contenu d'une page web 182
31. Autres fonctions de *LuaSocket* 183

26. Les interfaces utilisateur graphiques 184
32. Qu'entend-on par interface utilisateur graphique ?..... 184
33. Quel système de GUI utiliser ?..... 185
34. Quelles liaisons avec Lua sont disponibles ?..... 186
35. Fonder sa GUI sur des composants natifs 186
36. Recourir à une boîte à outils de composants 187
37. Installer une librairie de type IUP, lqt ou tekUI 187
38. Écrire *Hello World!* avec IUP, lqt ou tekUI 188
39. Ajouter de l'interactivité avec IUP, lqt ou tekUI 190

27. Lua dans les jeux vidéo 192
40. Utiliser Lua dans le gameplay 192
41. Utiliser Lua pour les paramétrages du jeu 196

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

42. Utiliser Lua pour l'intelligence artificielle 197

S'interfacer avec le C 200

28. Les bases de l'API C 201
1. Changer la fonction d'allocation 202
2. Charger les modules de la librairie standard 203
3. Charger des scripts *via* l'API C 203
4. Opérations mathématiques depuis le C 204
5. Convertir une fonction en *chunk* 204

29. Manipulation de la pile d'appel 205
6. Modifier la pile 205
7. Comment savoir ce que contient la pile ? 206
8. Récupérer des variables globales 210
9. Comparer des données dans la pile 210
10. Concaténer des données dans la pile 211
11. Appeler une fonction 211

30. Manipulation des tables 214
12. Créer, modifier et lire une table 215
13. Parcourir une table avec l'API C 216
14. Utiliser les métatables depuis l'API C 217
15. Le registre Lua 218

31. Les fonctions C et les fermetures 220
16. Utiliser des fonctions C en Lua 220
17. Manipuler des fonctions C 221
18. Associer des valeurs à une fonction C 221
19. Manipuler les *upvalues* d'une fermeture 223

32. Les *userdata* 224
20. Qu'est-ce que les *userdata* ? 224
21. Qu'est-ce que les *lightuserdata* ? 224
22. Quand faut-il se servir des *userdata* ? 225
23. Créer une *userdata* 226
24. Échanger des *userdata* entre Lua et C 227
25. Associer une métatable à une *userdata* 227
26. Accès de type objet avec les *userdata* 229
27. Associer une table à une *userdata* 230
28. *userdata* et le ramasse-miettes ? 230

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

33. Utilisation avancée 231
29. Déboguer avec l'API C..... 231
30. Gérer les erreurs 231
31. Remonter une erreur 231
32. Gérer le ramasse-miettes 232
33. Créer et gérer des coroutines 232
34. Déplacer des données entre deux coroutines 234
35. Tester l'état de coroutines 235

34. La librairie auxiliaire 236
36. Quand utiliser la librairie auxiliaire ?..... 236
37. Créer un état Lua 238
38. Charger rapidement la librairie standard 238
39. Charger et exécuter un fichier Lua 238
40. Charger et exécuter une chaîne de caractères 239
41. Charger du code à partir d'un buffer 239
42. Vérifier et manipuler la pile d'appel 239
43. Facilité pour les énumérations C..... 241
44. Facilité pour les chaînes de caractères 241
45. Les tableaux et les métatables 241
46. Traiter les erreurs 242
47. Manipuler les buffers pour créer des chaînes de caractères 242
48. Vérifier dynamiquement la cohérence des versions..... 244
49. Facilité pour créer des librairies 245
50. Charger une librairie depuis le C..... 245
51. Obtenir une référence dans une table 245

Déboguer son code 246

35. Déboguer côté Lua 247
1. Quels outils pour déboguer du code Lua ?..... 247
2. Que peut-on faire avec le module *debug* ?..... 247
3. Débogage interactif 248
4. Obtenir des informations sur une fonction 248
5. Étudier ou modifier des variables locales 250
6. Étudier ou modifier des *upvalues* 251
7. Consulter et modifier la table associée à une *userdata* 252
8. Mettre en place une fonction d'hameçonnage 252
9. Afficher la pile d'appel 254

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

10. Manipuler les métatables 255
11. Récupérer le *registry* 255

36. Déboguer côté C 256
12. Comment déboguer ses scripts à partir du C ?..... 256
13. Obtenir des informations sur une fonction 256
14. Obtenir des informations sur la pile d'appel 257
15. Accéder à des variables locales 258
16. Accéder à des valeurs *upvalue* 258
17. Mettre en place une fonction d'hameçonnage 258

L'implémentation LuaJIT 260

37. L'implémentation LuaJIT 261
1. Quand LuaJIT est-il plus performant que Lua ?..... 261
2. Quelles sont les limitations de LuaJIT ?..... 262
3. Quelle compatibilité entre LuaJIT et les versions de Lua officielles ?..... 262
4. Quelles extensions apporte LuaJIT ?..... 262
5. Arrêter ou démarrer la compilation dynamique 263
6. Obtenir l'état de la compilation dynamique 263
7. Récupérer la version de LuaJIT 263
8. Récupérer des informations système 264

38. Le module *FFI* de LuaJIT 265
9. Quelle différence avec les mécanismes Lua ?..... 265
10. Principe d'utilisation de *FFI* 265
11. Déclarer les fonctions à importer 266
12. Comment les types sont-ils convertis ?..... 267
13. Comment résoudre les symboles déclarés avec *ffi.cdef()* ?..... 268
14. Peut-on déclarer des variables ou des types ?..... 269
15. Récupérer des informations sur la plate-forme 271
16. Peut-on utiliser des callbacks C ?..... 272
17. Peut-on associer des métatables au type *cdata* ?..... 274
18. Obtenir des informations sur les variables *cdata* 275
19. Récupérer la valeur de *errno* 276
20. Créer une *string* depuis un pointeur C 276
21. Copier le contenu de variables *cdata* ou *string* 276
22. Initialiser le contenu d'un type *cdata* 276

39. Le module *BitOp* de LuaJIT 277

© 2012 Éditions D-BookeR – Le guide Lua et ses applications

23. Conversion en chaînes hexadécimales 277
24. Opérations logiques usuelles 277
25. Décalages logiques 278
26. Rotations logiques 278
27. Décalage arithmétique 279
28. Changer l'ordre des octets 279
29. Retourner une valeur normalisée 279

Annexe 280
Lexique 281

Liste des illustrations 290
Liste des tableaux 291
Liste des exemples 292

Index 293

© 2012 Éditions D-BookeR – Le guide Lua et ses applications