

Les éléments à télécharger sont disponibles à l'adresse suivante :

<http://www.editions-eni.fr>

Saisissez la référence ENI de l'ouvrage **LF2RASPYT** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

Chapitre 1

Raspberry Pi 3, premier contact

1. Introduction au Raspberry Pi	7
2. Démarrage de Raspbian	11
2.1 Historique	11
3. Comprendre l'écosystème Python : quelle version utiliser ?	16
4. Installer des bibliothèques Python	17
4.1 La méthode aptitude	18
4.2 La méthode pip	19
4.3 pip ou aptitude ?	21
5. IDLE : l'éditeur de code en Python pour Python	22
6. Conclusion	27

Chapitre 2

Python : bases et concepts avancés

1. Hello World	29
2. Les types de base : int, float, str et bool	30
2.1 L'entier : int	30
2.2 Le flottant : float	31
2.3 La chaîne de caractères : str	31
2.4 Le booléen	35

2 Python et Raspberry Pi - Apprenez à développer sur votre nano-ordinateur

3. Les structures de données : list, dict, tuple	35
3.1 La liste : list.....	35
3.2 Les tuples	36
3.3 La table de hachage ou dictionnaire : dict.....	37
3.4 L'ensemble : set.....	38
4. Les instructions, conditions et boucles	38
4.1 La condition if	39
4.2 La condition else	39
4.3 La boucle for	40
4.4 L'instruction break	40
4.5 L'instruction continue	41
4.6 La boucle while.....	41
5. Les opérateurs.....	42
5.1 Opérateurs arithmétiques	42
5.2 Opérateurs logiques.....	45
5.3 Opérateur d'appartenance.....	45
5.4 Opérateur d'affectation.....	46
6. Les classes : définition avec le mot-clé class	47
6.1 Premiers pas	48
6.2 Exposer les attributs d'une classe.....	49
6.3 Composition de classes.....	50
6.4 Réutiliser du code	51
7. Les fonctions : les mots-clés def et lambda	53
7.1 Définir une fonction	53
7.2 La fonction anonyme	56
8. La syntaxe en compréhension	57
9. Itérateur et générateur : les mots-clés iter et yield	60
10. La gestion des exceptions avec les mots-clés try, except, raise et finally	64
11. L'import des modules avec le mot-clé import	68
12. La gestion de contexte avec les mots-clés with et as	70
13. Conclusion.....	72

Chapitre 3

Administration du Raspberry Pi en Python

1. Introduction	73
2. Naviguer dans le système de fichiers avec les modules os et pwd	74
2.1 Manipuler et interroger le système de fichiers	77
2.2 Explorer le système de fichiers du Raspberry Pi	78
3. Interagir avec l'interpréteur Python via le module sys	81
4. Lancer des commandes shell avec le module subprocess	83
5. Chercher des fichiers avec le module glob	85
6. Comparer des fichiers ou répertoires avec le module filecmp	87
7. Capturer des signaux UNIX avec le module signal	89
8. Écriture de scripts avec le module argparse	92
9. Conclusion	95

Chapitre 4

Le Raspberry Pi en console avec urwid

1. Introduction	97
2. urwid, les fondamentaux	97
3. Projet #1 : une horloge en console	100
4. Projet #2 : un navigateur de fichiers en console	102
5. Conclusion	106

Chapitre 5

Programmation d'interfaces graphiques avec tkinter

1. Les fondamentaux	107
2. Projet #1 : Hello world avec tkinter	110
3. Projet #2 : une visionneuse d'images	117
4. Projet #3 : un éditeur de texte	125
5. Conclusion	136

4 Python et Raspberry Pi - Apprenez à développer sur votre nano-ordinateur

Chapitre 6

À l'assaut du Web avec le Raspberry Pi

1. Webscraping facile avec les modules urllib et HTMLParser	137
2. Développer un serveur HTTP avec le module http.server	142
3. Exécuter des scripts avec le module cgi	148
4. Envoyer des e-mails avec le module smtplib	155
5. Écrire une API légère avec Flask	158
6. Conclusion	166

Chapitre 7

Multimédia et audio sur le Raspberry Pi

1. Dessiner avec Pillow	167
1.1 Créer et manipuler des images	167
1.2 Dessiner des figures géométriques	173
2. Contrôler les entrées et sorties audio avec pyalsaudio	175
3. Projet #1 : un enregistreur/lecteur audio	182
4. Conclusion	187

Chapitre 8

Persistance de données sur le Raspberry Pi

1. Introduction	189
2. Sérialisation et désérialisation avec les modules pickle et shelve	189
3. Traiter des fichiers CSV avec le module csv	193
3.1 Création et lecture d'un fichier CSV	193
3.2 Créer son propre dialecte CSV	195
4. Manipuler des données XML avec le module xml.etree.ElementTree	196
4.1 Créer et sérialiser un fichier XML	197
4.2 Interroger un fichier XML	198
4.3 Ajouter et supprimer des nœuds	199
5. Travailler avec le format d'échange de données JSON via le module json	201

6. Gestion d'une base de données SQL légère avec le module sqlite3.....	203
7. Conclusion.....	208

Chapitre 9

Documenter et tester ses scripts en Python

1. Introduction.....	209
2. Consulter de la documentation avec pydoc3.....	210
3. Documenter et tester son code en une seule fois avec le module doctest.....	217
4. Écriture de tests unitaires avec le module unittest.....	221
5. Benchmarker son code avec le module timeit.....	226
6. Déboguer ses programmes avec le module pdb.....	230
6.1 Déboguer pas à pas.....	231
6.2 Déboguer à un endroit précis du programme.....	234
6.3 Procéder à l'autopsie de son programme.....	234
7. Conclusion.....	235

Chapitre 10

Raspberry Pi et GPIO

1. Les GPIO, comment ça marche ?.....	237
2. Connecter un écran LCD 16x2 au Raspberry Pi.....	239
3. Projet #1 : communiquer avec l'écran LCD.....	245
4. Projet #2 : créer un tube FIFO dédié à l'écran LCD.....	248
5. Projet #3 : écrire des messages depuis une interface en ligne de commande.....	253
6. Projet #4 : piloter l'écran LCD depuis une interface graphique tkinter.....	256
7. Conclusion.....	259

Index.....	261
------------	-----

Les éléments à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
 Saisissez la référence ENI de l'ouvrage **LFPYRASPFL** dans la zone de recherche
 et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Préface

Chapitre 1

Présentation

1. Avant-propos	15
2. Motivations	16
3. Présentation du projet	17
4. Objectifs de l'ouvrage	19
5. Prérequis	20
6. Matériel utilisé	21
6.1 Raspberry Pi.....	21
6.2 Feather ESP8266 Huzzah	22
6.3 Module relais	23
6.4 DHT11 - Humidité	24
6.5 AM2315 - Température et humidité	25
6.6 DS18B20 - Température.....	25
6.7 BME280 et BMP280 - Pression, humidité, température.....	26
6.8 TSL2561 - Luminosité	27
6.9 ADS1115 - Lecture analogique	27
6.10 TMP36 - Température	28
6.11 Photorésistance - luminosité	28
6.12 PIR - Détection de mouvement.....	29
6.13 Contact magnétique.....	29
6.14 Capteur à effet Hall numérique.....	30

2 Python, Raspberry Pi et Flask - Données télémétriques et tableaux de bord web

7. Code source	31
7.1 Téléchargement.....	31
7.2 GitHub	31
8. Configuration	33
8.1 Installation du Raspberry Pi	33
8.2 Utilitaires : des outils pour travailler	48
8.2.1 Connexion SSH.....	48
8.2.2 Éditeur de texte Nano	49
8.2.3 Transfert de fichiers via SSH (sftp)	51
8.2.4 Système de fichiers SSH.....	53
8.2.5 Bureau à distance	56
9. Type de données collectées.....	58

Chapitre 2

Le broker MQTT

1. Présentation et concepts	59
1.1 Le broker MQTT, élément central du réseau MQTT.....	61
1.2 Les éléments de MQTT	63
1.3 Le broker MQTT	63
1.4 Les topics	64
1.5 Les publishers.....	64
1.6 Les subscribers.....	65
1.7 Le ClientId.....	66
2. Les topics en détail	66
2.1 Contenu du message.....	68
2.1.1 Le message selon MQTT	68
2.1.2 En marge du standard.....	68
2.2 Création de topic et bonnes pratiques	69
2.3 Les topics système	71
3. Souscription et expression de filtrage	72
3.1 Expression de filtrage sans joker	73
3.2 Le joker de niveau.....	73
3.3 Le joker multiniveau.....	73

4. Les qualités de service MQTT.....	74
4.1 Les niveaux de qualités.....	75
4.1.1 QoS 0 : une fois max.....	75
4.1.2 QoS 1 : au moins une fois.....	75
4.1.3 QoS 2 : exactement une fois.....	76
4.2 Rétrogradation de QoS.....	77
4.3 Quel QoS utiliser et quand ?.....	78
5. La rétention de messages.....	79
6. Les clients persistants.....	80
7. Quel broker MQTT ?.....	81
8. Installation de Mosquitto.....	85
8.1 Mise à jour.....	85
8.2 Installation.....	86
9. Test avec Mosquitto.org.....	87
9.1 La souscription.....	88
9.2 La publication.....	88
9.3 Tester le broker MQTT du Raspberry Pi.....	90
10. Topics du projet.....	91
11. QoS du projet.....	93
12. Sécurité.....	94
13. Configurer le login du broker MQTT.....	94
13.1 Modifier la configuration.....	94
13.2 Tester la configuration.....	96
14. MQTT en Python.....	97
14.1 test-mqtt-client-sub.py.....	98
14.2 test-mqtt-client-pub.py.....	101
14.3 Documentation complémentaire.....	103
15. MQTT en MicroPython.....	104

4 Python, Raspberry Pi et Flask - Données télémétriques et tableaux de bord web

Chapitre 3

ESP8266 sous MicroPython

1. Présentation de l'ESP8266	105
1.1 Les possibilités offertes par l'ESP8266	106
1.2 Les plateformes ESP8266 populaires	107
1.3 Programmer un ESP8266	110
1.4 Feather Huzzah ESP8266 en détail	111
1.5 Brochage du Feather Huzzah ESP8266	112
1.5.1 Alimentation	113
1.5.2 Port série	114
1.5.3 Broches d'entrée/sortie	115
1.5.4 Entrée analogique	117
1.5.5 Les autres broches	118
2. Charger le firmware MicroPython	118
2.1 Identifier le firmware MicroPython	119
2.2 Préparatifs	120
2.3 Reflasher l'ESP8266	122
3. Prise de contrôle	125
3.1 Communiquer avec MicroPython	125
3.2 Communiquer avec un ESP8266 sous MicroPython	126
3.3 REPL : l'invite de commandes MicroPython	126
3.4 RShell	129
3.5 Ampy	137
4. WebREPL	143
4.1 Le démon WebREPL	144
4.1.1 Activer WebREPL sur l'ESP8266	144
4.1.2 Le mot de passe WebREPL	146
4.2 Client WebREPL	146
5. Nom d'hôte et adresse MAC	147
6. Le mode point d'accès (AP)	148
7. Le mode station (STA)	151
7.1 Mode STA et scan réseau	152
7.2 Réseau Wi-Fi visible ou masqué	153
7.3 Connexion en mode STA	153
7.4 WebREPL en mode STA	155

7.5	Désactivation du point d'accès	156
7.6	Rechercher l'adresse IP d'un ESP8266	157
8.	Séquence de démarrage MicroPython	158
8.1	Fichier boot.py	158
8.2	Fichier main.py	158
8.3	Un fichier boot.py pour ESP8266	159
8.3.1	Script trop optimiste et conséquences	160
8.3.2	RunApp - Activation de l'application	161
8.3.3	Un script de boot avancé	162
9.	Programmer	164
9.1	Création d'une bibliothèque	165
9.2	Les bibliothèques MicroPython	167
9.2.1	Bibliothèques standards et microbibliothèques	167
9.2.2	Bibliothèques spécifiques à MicroPython	169
9.2.3	Bibliothèque spécifique à l'ESP8266	169
9.2.4	Autres bibliothèques MicroPython	169
9.2.5	Mécanisme de chargement d'une bibliothèque	170
9.3	Charger et exécuter un script à la volée	170
9.4	RunApp : exécution conditionnelle de main.py	172
9.5	Entrées/sorties sur ESP8266	174
9.5.1	Entrée numérique	174
9.5.2	Entrée numérique (pull-up interne)	176
9.5.3	Entrée numérique et déparasitage logiciel	178
9.5.4	Sortie numérique	180
9.5.5	Entrée analogique	182
9.5.6	Ajout d'entrée/sortie avec MCP23017	185
9.5.7	Lecture analogique avec l'ADS1115	189
9.6	Senseur et interface sur ESP8266	192
9.6.1	Senseur PIR - senseur de proximité	192
9.6.2	Contact magnétique	193
9.6.3	DHT11 - humidité	195
9.6.4	Senseur à effet Hall	197
9.6.5	TSL2561 - luminosité	199
9.6.6	BME280 - température, humidité et pression barométrique	203
9.6.7	Module relais	205

6 Python, Raspberry Pi et Flask - Données télémétriques et tableaux de bord web

10. MQTT sous ESP8266	208
10.1 Publication MQTT sous MicroPython	209
10.2 Souscription MQTT sous MicroPython	212
11. Asyncio sur ESP8266	217
11.1 Asyncio en quelques mots	217
11.2 Asyncio par l'exemple	218
11.3 Fonction run_every pour Asyncio	222
11.4 Plus d'informations sur Asyncio	224

Chapitre 4

Les objets ESP8266

1. Informations pratiques	225
1.1 Prérequis et configurations	225
1.2 LED de statut	226
1.3 Les topics MQTT	227
1.4 Télécharger et préparer le code des objets IoT	228
2. Fonctionnement général d'un objet IoT	230
2.1 Principales sections	233
2.2 Paramètres d'un objet IoT	234
2.3 RunApp et la LED d'activité	234
2.4 La fonction led_error()	234
2.5 Les tâches et fonctions asynchrones des objets IoT	236
3. Objet 1 : Météo cabane de jardin	238
3.1 Schéma de raccordement	239
3.2 Téléverser les scripts	240
3.3 Fonctionnement du script	241
3.4 Tester l'objet	246
4. Objet 2 : Surveillance salon	246
4.1 Téléverser les scripts	247
4.2 Fonctionnement du script	248
4.3 La fonction capture_1h()	253
4.4 Capteur PIR - variables et utilisation	253
4.5 Capteur PIR - la fonction pir_activated	254
4.6 Capteur PIR - la fonction pir_alert	255

4.7	Senseur PIR – la fonction <code>pir_update</code>	255
4.8	Problèmes de concurrence	256
4.9	Tester l'objet	257
5.	Objet 3 : Surveillance de la véranda	257
5.1	Téléverser les scripts	258
5.2	Fonctionnement du script	259
5.3	La fonction <code>capture_1h()</code>	263
5.4	La fonction <code>check_contact()</code>	264
5.5	La fonction <code>check_ldr()</code>	265
5.6	Tester l'objet	268
6.	Objet 4 : Chauffage	268
6.1	Téléverser les scripts	270
6.2	Fonctionnement du script	271
6.3	La fonction <code>capture_1h()</code>	277
6.4	La fonction <code>capture_10m()</code>	278
6.5	La fonction <code>check_mqtt_sub()</code>	278
6.6	La fonction <code>sub_cb()</code>	279
6.7	La fonction <code>chaud_exec_cmd()</code>	280
6.8	Tester l'objet	281
7.	Dépannage d'un objet IoT	282

Chapitre 5

Persistance des données

1.	Introduction	285
1.1	Pourquoi utiliser une base de données ?	285
1.2	Quel moteur de base de données ?	286
1.3	Principe de fonctionnement de push-to-db	287
2.	SQLite 3	288
2.1	Présentation	288
2.2	Classe de stockage, type de données et affinité	289
2.2.1	Classe de stockage	290
2.2.2	Stockage des date et heure	291
2.2.3	Affinité de type pour les colonnes	291
2.2.4	Résolution de l'affinité de type	293

8 Python, Raspberry Pi et Flask - Données télémétriques et tableaux de bord web

2.3	Affinité, expressions, comparaison et tri	293
2.3.1	Affinité des expressions	293
2.3.2	Comparaison, tri et groupage	296
2.4	Clé primaire et auto-incrément	297
2.4.1	Définir une clé primaire	297
2.4.2	Table rowid et clé primaire	298
2.5	SQLite3 et accès concurrents	299
2.6	Installation	301
2.6.1	Installer SQLite 3	301
2.6.2	Installer le support Python	302
2.7	Premiers pas avec SQLite3	302
2.7.1	Documentation SQL pour SQLite	304
2.7.2	Commandes de l'interpréteur SQLite	304
2.8	SQLite et Python	307
2.8.1	Opération de lecture SQLite	307
2.8.2	Opération d'insertion SQLite	310
2.8.3	Row Factory de SQLite	312
3.	Approches techniques de push-to-db	313
3.1	Approche base de données de push-to-db	314
3.1.1	topicmsg - dernier message reçu	314
3.1.2	ts_xxx - historique de messages	315
3.2	Approche logicielle de push-to-db	317
3.2.1	Diagramme des classes (partie 1)	318
3.2.2	Fichier de configuration de push-to-db	324
3.2.3	Diagramme des classes (partie 2)	327
4.	Configuration de push-to-db	349
4.1	Les répertoires de stockage de push-to-db	351
4.2	Création des tables de push-to-db	352
4.3	push-to-db.ini	353
4.4	Le script d'installation de push-to-db	356
5.	Logger Python	359
5.1	Logger et fichier de configuration	359
5.2	Configuration du logger	359
5.3	Utilisation du logger	361
6.	Exécution du script push-to-db	362

7. Service systemd pour push-to-db	363
7.1 Quand démarrer le service ?	363
7.2 Créer le fichier Unit	364
7.3 Configurer, démarrer, contrôler	365
7.4 Documentation sur systemd	365
8. Améliorations	366

Chapitre 6

Développement web en Python

1. Présentation de Flask	367
1.1 Pourquoi Flask ?	368
1.2 La flexibilité de Flask	369
1.3 Les nombreuses extensions Flask	369
1.4 Flask plus en détails	371
1.4.1 Werkzeug	372
1.4.2 WSGI	373
1.4.3 Application Flask	374
1.4.4 Jinja	374
1.4.5 Base de données	375
1.5 Documentations	375
2. Anatomie d'un projet Flask	376
3. Installation et prise en main	377
3.1 L'utilitaire flask	379
3.2 Prise en main avancée	380
3.3 Déboguer avec Flask	384
3.4 Application Flask en production	388
4. Les fondamentaux de Flask	389
4.1 Routes et paramètres	389
4.2 Retourner une erreur	398
4.3 Utilisation de template	399
4.4 Création d'URL	402
4.5 Redirection	403
4.6 Requêtes GET et POST	406

10 Python, Raspberry Pi et Flask - Données télémétriques et tableaux de bord web

4.7	Contexte applicatif	410
4.7.1	L'objet g	410
4.7.2	Connexion à la base de données	411
4.8	Les cookies	411
4.9	Les sessions	413
4.10	Journalisation	414
4.11	Mini-projet Fruits	417
4.11.1	Sources du mini-projet	418
4.11.2	La connexion SQLite 3	419
4.11.3	Organisation du mini-projet	420
4.11.4	Détails du mini-projet	420
4.12	Ressources et documentations	435
5.	Templates Jinja	435
5.1	Exécution d'un template	436
5.2	Tester un template	436
5.2.1	Créer une application Flask	436
5.2.2	Test avec serveur web Flask et string Python	437
5.2.3	Test en console et string Python	438
5.2.4	Utiliser le projet Jinja Live Parser	438
5.3	Évaluation des balises	440
5.3.1	{{ ... }} : évaluation d'expression	441
5.3.2	{% ... %} : instructions de contrôle de flux	441
5.3.3	{# ... #} : insertion de commentaire	441
5.3.4	# ... : ligne d'instruction	442
5.4	Variables et expressions	442
5.4.1	Variables spéciales	444
5.4.2	Séquence d'échappement	445
5.4.3	Assignation	446
5.5	Branchement	446
5.6	Itération	447
5.7	Les macros	451
5.8	Contrôle des espaces	453
5.9	Filtres Jinja	455
5.10	Inclusion de template	461
5.11	Importer des macros	464

5.12 Héritage de template	465
5.12.1 Les éléments de l'héritage	468
5.12.2 Heritage-app : l'héritage Jinja par la pratique	468
5.12.3 Template de base et block	469
5.13 Template enfant	471
5.13.1 Super bloc	472
5.13.2 Ressources	474
5.14 Message Flash	474

Chapitre 7

Le tableau de bord

1. Présentation	485
1.1 Préambule	485
1.2 Dépôt du projet Dashboard	487
1.3 Éléments principaux	488
1.4 Fonctionnalités du projet Dashboard	493
2. Structure HTML	505
2.1 Disposition de la page	505
2.2 Les blocs d'informations	510
2.3 La liste	512
3. Template Jinja	513
3.1 Le template de base	513
3.2 Utilisation du template de base	519
4. Configuration	522
4.1 Base de données dashboard.db	522
4.1.1 Schéma de la base de données	523
4.1.2 Répertoire de stockage	526
4.1.3 Création des tables de Dashboard	526
4.1.4 Copie de la base de données	527
4.2 Fichier de configuration de Dashboard	527
5. Détails de l'application Flask	530
5.1 Répertoires et fichiers	530
5.2 Les routes de Dashboard	536

12 Python, Raspberry Pi et Flask - Données télémétriques et tableaux de bord web

5.3	Accès aux données	537
5.3.1	La fonction <code>get_db(db_key)</code> multi bases de données	537
5.3.2	Les classes <code>DBHelper</code> de Dashboard	541
5.3.3	Exemple : liste des topics disponibles pour Dashboard	545
5.3.4	Exemple : extraction de l'historique dans Dashboard	546
5.3.5	Affichage d'un tableau de bord	549
5.4	Les filtres Jinja personnalisés	556
5.5	Affichage du tableau de bord	559
5.6	Les macros Jinja	564
5.6.1	La macro <code>make_block</code>	564
5.6.2	La macro <code>block_icon</code>	566
5.6.3	La macro <code>block_big_text</code>	568
5.6.4	La macro <code>select_color</code> (édition d'un bloc)	569
6.	Bloc switch (marche/arrêt)	574
6.1	Développements complémentaires	577
6.1.1	MQTT sources	577
6.1.2	Bloc et paramètres additionnels	580
6.2	Ajout du bloc SWITCH	583
6.2.1	<code>Block_config</code> du switch	583
6.2.2	Ajouter le nouveau type de bloc	584
6.3	Le switch et MQTT	587
6.3.1	Client MQTT JavaScript	589
6.3.2	MQTT en Javascript et WebSocket	590
6.3.3	Activer le support WebSocket sur Mosquitto	591
6.3.4	Tester le client MQTT JavaScript	591
6.3.5	Mille milliards de mille sabords !	593
6.3.6	La route <code>MqttProxyPublish</code>	595
6.3.7	Événement <code>on_switch_change</code>	596
6.4	Tester le bloc switch	598
7.	Améliorations	602

Conclusion

1. Introduction	603
2. Remerciements.....	604
3. Retour sur client MQTT JavaScript	604

Annexes

1. Installation rapide.....	607
1.1 Prérequis.....	607
1.2 Début de l'installation	607
1.3 Récupération des sources	608
1.4 Poursuivre l'installation.....	609

Index	613
-------------	-----