

Les éléments à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
Saisissez la référence ENI de l'ouvrage **HSRI43PYT** dans la zone de recherche
et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

1. Introduction	23
2. Contenu de l'ouvrage	23
3. Progressivité de l'ouvrage	24
4. À destination des enseignants et élèves	25
5. À destination des chercheurs ou doctorants	26
6. À destination de ceux qui viennent d'un autre langage	27

Partie 1 : Les atouts de Python

Chapitre 1-1

Clés théoriques

1. Petite histoire des langages informatiques	29
1.1 Informatique théorique	29
1.2 Chronologie de l'informatique	30
1.2.1 Évolutions des problématiques liées à l'informatique	30
1.2.2 Chronologie des langages informatiques	31
2. Typologie des langages de programmation	35
2.1 Paradigmes	35
2.1.1 Définition	35
2.1.2 Paradigme impératif et dérivés	36
2.1.3 Paradigme objet et dérivés	37
2.1.4 Programmation orientée aspect	37
2.1.5 Paradigme fonctionnel	38
2.1.6 Paradigme logique	38
2.1.7 Programmation concurrente	38
2.1.8 Synthèse	39
2.2 Interopérabilité	39
2.3 Niveau de programmation	41
2.3.1 Machine	41
2.3.2 Bas niveau	41
2.3.3 Haut niveau	42

2 _____ Python 3

Les fondamentaux du langage

2.4	Typage	43
2.4.1	Faible vs fort	43
2.4.2	Statique vs dynamique	43
2.5	Grammaire	44
2.5.1	Langages formels	44
2.5.2	Syntaxe	44
3.	Python et le reste du monde	45
3.1	Positionnement stratégique du langage Python	45
3.1.1	Segments de marchés	45
3.1.2	Niveau de complexité	45
3.1.3	Forces du langage	45
3.1.4	Points faibles	46
3.2	Intégration avec d'autres langages	47
3.2.1	Extensions C	47
3.2.2	Intégration de programmes écrits en C	47
3.2.3	Intégration de programmes Python dans du C	47
3.2.4	Intégration de programmes écrits en Java	47
3.2.5	Intégration de programmes Python dans Java	47
3.2.6	Autres intégrations	47

Chapitre 1-2

Présentation de Python

1.	Philosophie	49
1.1	Python en quelques lignes	49
1.1.1	D'où vient le nom « Python » ?	49
1.1.2	Présentation technique	50
1.1.3	Présentation conceptuelle	50
1.2	Comparaison avec d'autres langages	50
1.2.1	Shell	50
1.2.2	Perl	51
1.2.3	C, C++	51
1.2.4	Java	52
1.2.5	PHP	54
1.3	Grands principes	55
1.3.1	Le zen de Python	55
1.3.2	Le développeur n'est pas stupide	56
1.3.3	Documentation	56
1.3.4	Python est livré piles incluses	56
1.3.5	Duck Typing	57

1.3.6	Notion de code pythonique	57
2.	Histoire de Python.	57
2.1	La genèse	57
2.2	Extension du périmètre fonctionnel	58
2.3	Évolution de la licence	62
2.4	Avenir	62
3.	Gouvernance	63
3.1	Développement	63
3.1.1	Branches	63
3.1.2	Communauté	64
3.2	Mode de gouvernance	65
3.2.1	Créateur du langage	65
3.2.2	PEP	65
3.2.3	Prise de décisions	65
3.2.4	Contribuer à Python	66
4.	Que contient Python ?	66
4.1	Une grammaire et une syntaxe	66
4.2	Plusieurs implémentations	67
4.3	Une bibliothèque standard	67
4.4	Des bibliothèques tierces	67
4.5	Des frameworks	67
5.	Phases d'exécution d'un programme Python	68
5.1	Chargement de la machine virtuelle	68
5.2	Compilation	68
5.3	Interprétation	69

Chapitre 1-3 Pourquoi choisir Python

1.	Qualités du langage	71
1.1	Ticket d'entrée	71
1.2	Qualités intrinsèques	73
1.3	Couverture fonctionnelle	74
1.4	Domaines d'excellence	74
1.5	Garanties	75
2.	Diffusion	76
2.1	Entreprises	76
2.2	Le monde de la recherche	78
2.3	Le monde de l'éducation	78

4 _____ Python 3

Les fondamentaux du langage

2.4	Communauté	79
3.	Références	80
3.1	Poids lourds de l'industrie informatique	80
3.1.1	Google	80
3.1.2	Mozilla	81
3.1.3	Microsoft	81
3.1.4	Canonical	81
3.1.5	Cisco	82
3.2	Entreprises innovantes	82
3.2.1	Services de stockage en ligne	82
3.2.2	Informatique dématérialisée	82
3.2.3	Forge	83
3.2.4	Réseaux sociaux	83
3.3	Éditeurs de contenus	83
3.3.1	Disney Animation Studio	83
3.3.2	YouTube	83
3.3.3	Box ADSL	83
3.3.4	Spotify	83
3.4	Éditeurs de logiciels	83
4.	Retours d'expérience	84
4.1	Internet des objets	84
4.2	Système et développement web	85
4.3	Enseignement	86
4.4	Embarqué	86
4.5	Développement web	87
4.6	ERP	87

Chapitre 1-4

Installer son environnement de travail

1.	Introduction	89
2.	Installer Python	89
2.1	Pour Windows	89
2.2	Pour Mac	92
2.3	Pour GNU/Linux et BSD	93
2.4	Par la compilation	94
2.5	Pour un smartphone	94
3.	Installer une bibliothèque tierce	95
3.1	À partir de Python 3.4	95
3.2	Pour une version inférieure à Python 3.4	97

3.3	Pour Linux	97
4.	Créer un environnement virtuel	97
4.1	À quoi sert un environnement virtuel ?	97
4.2	Pour Python 3.3 ou version supérieure	98
4.3	Pour toute version de Python	98
4.4	Pour Linux	100
5.	Gestion des dépendances	101
6.	Installer Anaconda	102
6.1	Pour Windows	102
6.2	Pour Linux	103
6.3	Pour Mac	103
6.4	Mettre à jour Anaconda	103
6.5	Installer une bibliothèque externe	103
6.6	Environnements virtuels	104
7.	Docker	104
8.	La console Python	105
8.1	Démarrer la console Python	105
8.2	BPython	105
8.3	IPython	106
8.4	IPython Notebook	106
9.	Installer un IDE	107
9.1	Liste d'IDE	107
9.2	Présentation de PyCharm	108
9.3	Configuration de PyCharm	108
10.	VSCode	112

Partie 2 : Guide Python

Chapitre 2-1

Les premiers pas

1.	Avant de commencer	113
1.1	Quelques notions importantes	113
1.1.1	Comment fonctionne un ordinateur ?	113
1.1.2	Qu'est-ce qu'un programme informatique ?	114
1.1.3	Qu'est-ce qu'un code source ?	114
1.2	Quelques conventions utilisées dans ce livre	114
1.2.1	Code Python	114
1.2.2	Terminal	115

6 **Python 3**

Les fondamentaux du langage

1.2.3	Mise en forme	115
1.3	Quelle est la meilleure méthode pour apprendre ?	116
2.	Premier programme	116
2.1	Hello world !	116
2.2	Affectation	118
2.3	Valeur booléenne	119
2.4	Type	120
2.5	Exceptions	121
2.6	Bloc conditionnel	123
2.7	Conditions avancées	125
2.8	Bloc itératif	125
3.	Premier jeu : Devine le nombre	127
3.1	Description du jeu	127
3.2	Aides	127
3.2.1	Gestion du hasard	127
3.2.2	Étapes de développement	128
3.3	Pour aller plus loin	128

Chapitre 2-2

Fonctions et modules

1.	Les fonctions	129
1.1	Pourquoi utiliser des fonctions ?	129
1.2	Introduction aux fonctions	131
1.2.1	Comment déclarer une fonction	131
1.2.2	Gestion d'un paramètre	132
1.2.3	Comment rendre une fonction plus générique	134
1.2.4	Paramètres par défaut	136
1.3	Problématiques de couplage et duplication de code	137
1.3.1	Niveau de ses fonctions	137
1.3.2	Notion de complexité	139
1.3.3	Bonnes pratiques	141
2.	Les modules	142
2.1	Introduction	142
2.1.1	Qu'est-ce qu'un module ?	142
2.1.2	Comment crée-t-on un module Python ?	143
2.1.3	Organiser son code	143
2.2	Gérer le code de ses modules	143
2.2.1	Exécuter un module, importer un module	143
2.2.2	Gérer une arborescence de modules	144

3.	Terminer le jeu.	145
3.1	Créer des niveaux	146
3.2	Déterminer un nombre de coups maximal	146
3.3	Enregistrer les meilleurs scores.	146
3.4	Intelligence artificielle	146

Chapitre 2-3

Les principaux types

1.	Chaînes de caractères	147
1.1	Syntaxe	147
1.2	Formatage d'une chaîne	148
1.3	Notion de casse.	148
1.4	Notion de longueur.	149
1.5	Appartenance	150
1.6	Notion d'occurrence.	150
1.7	Remplacement	151
1.8	Notion de caractère.	151
1.9	Typologie des caractères.	152
1.10	Séquencer une chaîne de caractères.	153
2.	Listes.	154
2.1	Syntaxe	154
2.2	Indices	154
2.3	Valeurs.	155
2.4	Hasard	157
2.5	Techniques d'itération	157
2.6	Tri	160
3.	Dictionnaires	162
3.1	Présentation des dictionnaires	162
3.2	Parcourir un dictionnaire	162
3.3	Exemple.	163

Chapitre 2-4 **Les classes**

1. Syntaxe.....	165
2. Notion d'instance courante.....	166
3. Opérateurs	169
4. Héritage	170
4.1 Spécialisation.....	171
4.2 Programmation par composants	172

Partie 3 : Les fondamentaux du langage

Chapitre 3-1 **Algorithmique de base**

1. Délimiteurs.....	175
1.1 Instruction.....	175
1.2 Une ligne de code = une instruction.....	175
1.3 Commentaire	176
1.4 Une instruction sur plusieurs lignes	176
1.5 Mots-clés	176
1.6 Mots réservés.....	177
1.7 Indentation	178
1.8 Symboles	179
1.9 Opérateurs.....	182
1.10 Utilisation du caractère souligné	186
1.11 PEP-8	187
1.12 PEP-7	187
1.13 PEP-257	187
2. Instructions	187
2.1 Définitions.....	187
2.1.1 Variable.....	187
2.1.2 Fonction	189
2.1.3 Fonctions lambda.....	190
2.1.4 Classe	191
2.1.5 Instruction vide	192
2.1.6 Suppression	192
2.1.7 Renvoyer le résultat de la fonction	193

2.2	Instructions conditionnelles	194
2.2.1	Définition	194
2.2.2	Condition	194
2.2.3	Instruction if	194
2.2.4	Instruction elif	195
2.2.5	Instruction else	195
2.3	Instruction de correspondance	197
2.4	Utilisation d'une expression d'affectation	200
2.4.1	Instruction switch	200
2.4.2	Interruptions	201
2.4.3	Approfondissement des conditions	201
2.4.4	Performances	202
2.5	Itérations	203
2.5.1	Instruction for	203
2.5.2	Instruction while	203
2.5.3	Quelle différence entre for et while ?	204
2.5.4	Instruction break	204
2.5.5	Instruction return	206
2.5.6	Instruction continue	206
2.5.7	Instruction else	206
2.5.8	Générateurs	207
2.6	Constructions fonctionnelles	210
2.6.1	Construction conditionnelle	210
2.6.2	Générateurs	210
2.6.3	Compréhensions de listes	210
2.6.4	Compréhensions d'ensembles	211
2.6.5	Compréhensions de dictionnaires	211
2.7	Compréhensions et expressions d'affectation	211
2.8	Gestion des exceptions	211
2.8.1	Présentation rapide des exceptions	211
2.8.2	Lever une exception	212
2.8.3	Pourquoi lever une exception ?	212
2.8.4	Assertions	213
2.8.5	Capturer une exception	214
2.8.6	Effectuer un traitement de l'exception	215
2.8.7	Gérer la sortie du bloc de capture	217
2.8.8	Gérer le non-déclenchement d'exceptions	217
2.8.9	Gestionnaire de contexte	219
2.8.10	Programmation asynchrone	220

2.9	Divers	221
2.9.1	Gérer des imports	221
2.9.2	Traverser les espaces de nommage	222
2.9.3	Fonctions print, help, eval et exec	224

Chapitre 3-2

Déclarations

1.	Variable	227
1.1	Qu'est-ce qu'une variable ?	227
1.1.1	Contenu	227
1.1.2	Contenant	227
1.1.3	Modes de modification d'une variable	229
1.2	Typage dynamique	232
1.2.1	Affectation : rappels	232
1.2.2	Primitive type et nature du type	232
1.2.3	Caractéristiques du typage Python	233
1.3	Visibilité	236
1.3.1	Espace global	236
1.3.2	Notion de bloc	236
2.	Fonction	240
2.1	Déclaration	240
2.2	Paramètres	241
2.2.1	Signature d'une fonction	241
2.2.2	Notion d'argument ou de paramètre	242
2.2.3	Valeur par défaut	243
2.2.4	Valeur par défaut mutable	244
2.2.5	Paramètres nommés	245
2.2.6	Déclaration de paramètres extensibles	246
2.2.7	Passage de paramètres étoilés	247
2.2.8	Signature universelle	248
2.2.9	Obliger un paramètre à être nommé (keyword-only)	249
2.3	Obliger un paramètre à être positionnel (positional-only)	251
2.3.1	Annotations/type hint/typage statique	251
3.	Classe	255
3.1	Déclaration	255
3.1.1	Signature	255
3.1.2	Attribut	255
3.1.3	Méthode	256
3.1.4	Bloc local	256

3.2	Instanciation	257
3.2.1	Syntaxe	257
3.2.2	Relation entre l'instance et la classe	257
4.	Module	258
4.1	À quoi sert un module ?	258
4.2	Déclaration	258
4.3	Instructions spécifiques	259
4.4	Comment appréhender le contenu d'un module ?	260
4.5	Compilation des modules	260

Chapitre 3-3 Modèle objet

1.	Tout est objet	263
1.1	Principes	263
1.1.1	Quel sens donner à « objet » ?	263
1.1.2	Adaptation de la théorie objet dans Python	264
1.1.3	Généralités	265
1.2	Classes	266
1.2.1	Introduction	266
1.2.2	Déclaration impérative d'une classe	266
1.2.3	Instance	267
1.2.4	Objet courant	268
1.2.5	Déclaration par prototype d'une classe	269
1.2.6	Tuples nommés	271
1.3	Méthodes	272
1.3.1	Déclaration	272
1.3.2	Appel de méthode	273
1.3.3	Méthodes et attributs spéciaux	276
1.3.4	Constructeur et initialisateur	279
1.3.5	Gestion automatisée des attributs	280
1.3.6	Intérêt du paradigme objet	281
1.3.7	Relation entre objets	281
1.4	Héritage	282
1.4.1	Polymorphisme par sous-typage	282
1.4.2	Surcharge de méthode	283
1.4.3	Surcharge des opérateurs	285
1.4.4	Polymorphisme paramétrique	285
1.4.5	Héritage multiple	287

2.	Autres outils de la programmation objet	289
2.1	Principes	289
2.2	Interfaces	290
2.3	Attributs	292
2.4	Propriétés	294
2.5	Emplacements	297
2.6	Métaclasses	298
2.7	Classes abstraites	300
2.8	La Zope Component Architecture	303
2.8.1	Présentation	303
2.8.2	Installation	303
2.8.3	Définir une interface et un composant	304
2.8.4	Autres fonctionnalités	305
2.8.5	Avantages de la ZCA	305
3.	Fonctions spéciales et primitives associées	305
3.1	Personnalisation	305
3.1.1	Classes	305
3.1.2	Instances	307
3.1.3	Comparaison	308
3.1.4	Évaluation booléenne	308
3.1.5	Relations d'héritage ou de classe à instance	309
3.2	Classes particulières	309
3.2.1	Itérateurs	309
3.2.2	Conteneurs	312
3.2.3	Instances assimilables à des fonctions	312
3.2.4	Ressources à protéger	313
3.2.5	Types	314
3.2.6	Classes de données	314

Chapitre 3-4

Nombres, booléens et algorithmes appliqués

1.	Nombres	315
1.1	Types	315
1.1.1	Entiers	315
1.1.2	Réels	316
1.1.3	Socle commun aux nombres entiers et réels	317
1.1.4	Méthodes dédiées aux nombres entiers	318
1.1.5	Méthodes dédiées aux nombres réels	319
1.1.6	Complexes	319

1.2	La console Python, la calculatrice par excellence	320
1.2.1	Opérateurs mathématiques binaires	320
1.2.2	Opérateurs binaires particuliers	321
1.2.3	Opérateurs mathématiques unaires	322
1.2.4	Arrondis	323
1.2.5	Opérateurs de comparaison	326
1.2.6	Opérations mathématiques n-aires	327
1.2.7	Fonctions mathématiques usuelles	328
1.3	Représentations d'un nombre	333
1.3.1	Représentation décimale	333
1.3.2	Représentation par un exposant	333
1.3.3	Représentation par une fraction	333
1.3.4	Représentation hexadécimale	334
1.3.5	Représentation octale	335
1.3.6	Représentation binaire	336
1.3.7	Opérations binaires	336
1.3.8	Longueur de la représentation mémoire d'un entier	338
1.4	Conversions	340
1.4.1	Conversion entre entiers et réels	340
1.4.2	Conversion entre réels et complexes	340
1.4.3	Conversion vers un booléen	341
1.5	Travailler avec des variables	342
1.5.1	Un nombre est non mutable	342
1.5.2	Modifier la valeur d'une variable	342
1.5.3	Opérateurs d'incrément	343
1.6	Statistiques	344
2.	Booléens	345
2.1	Le type booléen	345
2.1.1	Classe bool	345
2.1.2	Les deux objets True et False	346
2.1.3	Différence entre l'opérateur d'égalité et d'identité	346
2.2	Évaluation booléenne	346
2.2.1	Méthode générique	346
2.2.2	Objets classiques	346

Chapitre 3-5

Séquences et algorithmes appliqués

1. Présentation des différents types de séquences	349
1.1 Généralités	349
1.2 Les listes	350
1.3 Les n-uplets	351
1.4 Conversion entre listes et n-uplets	353
1.5 Socle commun entre liste et n-uplet	353
2. Notion d'itérateur	354
3. Utilisation des indices et des tranches	356
3.1 Définition de l'indice d'un objet et des occurrences	356
3.2 Utiliser l'indice pour adresser la séquence	358
3.3 Retrouver les occurrences d'un objet et leurs indices	359
3.4 Taille d'une liste, comptage d'occurrences	360
3.5 Utiliser l'indice pour modifier ou supprimer	361
3.6 Itération simple	363
3.7 Présentation de la notion de tranches (slices)	366
3.8 Cas particulier de la branche 2.x de Python	375
3.9 Utilisation basique des tranches	376
3.10 Utilisation avancée des tranches	377
4. Utilisation des opérateurs	379
4.1 Opérateur +	379
4.2 Opérateur *	380
4.3 Opérateur +=	383
4.4 Opérateur *=	384
4.5 Opérateur in	385
4.6 Opérateurs de comparaison	386
5. Méthodes de modifications	387
5.1 Ajouter des éléments dans une liste et un n-uplet	387
5.2 Supprimer un objet d'une liste et d'un n-uplet	389
5.3 Solutions de contournement pour la modification de n-uplets	393
5.4 Renverser une liste ou un tuple	394
5.5 Trier une liste	395
6. Utilisation avancée des listes	398
6.1 Opérations d'ensemble	398
6.2 Pivoter une séquence	399
6.3 Itérer correctement	400
6.4 Programmation fonctionnelle	401

6.5	Compréhensions de listes	403
6.6	Itérations avancées	405
6.7	Combinatoire	409
7.	Adapter les listes à des besoins spécifiques	412
7.1	Liste d'entiers	412
7.2	Présentation du type array	414
7.3	Utiliser une liste comme pile	415
7.4	Utiliser une liste comme file d'attente	416
7.5	Conteneur plus performant	416
7.6	Utiliser des listes pour représenter des matrices	417
7.7	Liste sans doublons	419
8.	Autres types de données	421

Chapitre 3-6 Ensembles et algorithmes appliqués

1.	Présentation	425
1.1	Définition d'un ensemble	425
1.2	Différences entre set et frozenset	426
1.3	Utilisation pour dédoubonner des listes	427
1.4	Rajouter une relation d'ordre	427
2.	Opérations ensemblistes	428
2.1	Opérateurs pour un ensemble à partir de deux autres	428
2.2	Opérateurs pour modifier un ensemble à partir d'un autre	429
2.3	Méthodes équivalentes à la création ou modification ensembliste	430
2.4	Méthodes de comparaison des ensembles	430
2.5	Exemples non classiques d'utilisation	431
3.	Méthodes de modification d'un ensemble	435
3.1	Ajouter un élément	435
3.2	Supprimer un élément	435
3.3	Vider un ensemble	436
3.4	Dupliquer un élément	436
3.5	Sortir une valeur d'un ensemble	437
3.6	Utiliser un ensemble comme un recycleur d'objets	438
3.7	Algorithmique avancée : résolution du problème des n-dames	441

Chapitre 3-7

Chaînes de caractères et algorithmes appliqués

1.	Présentation	443
1.1	Définition	443
1.2	Vocabulaire	444
1.3	Spécificités de la branche 2.x	445
1.4	Changements apportés par la branche 3.x	446
1.5	Chaîne de caractères en tant que séquence de caractères.	448
1.6	Caractères	450
1.7	Opérateurs de comparaison	451
2.	Formatage de chaînes de caractères	454
2.1	Opérateur modulo.	454
2.2	Méthodes de formatage sur l'ensemble de la chaîne.	459
2.3	Nouvelle méthode de formatage des variables dans une chaîne	462
2.4	Littéraux formatés	465
3.	Opérations d'ensemble	466
3.1	Séquençage de chaînes	466
3.2	Opérations sur la casse	468
3.3	Recherche sur une chaîne de caractères	470
3.4	Informations sur les caractères.	470
4.	Problématiques relatives à l'encodage.	472
4.1	Encodage par défaut	472
4.2	Encodage du système	472
4.3	L'unicode, référence absolue.	472
4.4	Autres encodages.	473
4.5	Ponts entre l'unicode et le reste du monde	474
4.6	Revenir vers l'Unicode	475
5.	Manipulations de bas niveau avancées.	476
5.1	Opérations de comptage.	476
5.2	Une chaîne de caractères vue comme une liste.	477
5.3	Une chaîne de caractères vue comme un ensemble de caractères . . .	478
6.	Représentation mémoire	478
6.1	Présentation du type bytes.	478
6.2	Lien avec les chaînes de caractères.	479
6.3	Présentation du type bytearray	480
6.4	Gestion d'un jeu de caractères	482

Chapitre 3-8

Dictionnaires et algorithmes appliqués

1. Présentation	489
1.1 Définition	489
1.2 Évolutions et différences entre les branches 2.x et 3.x	490
1.3 Vues de dictionnaires	491
1.4 Instanciation	493
1.5 Compréhension de dictionnaire	494
2. Manipuler un dictionnaire	494
2.1 Récupérer une valeur d'un dictionnaire	494
2.2 Modifier les valeurs d'un dictionnaire	495
2.3 Supprimer une entrée d'un dictionnaire	497
2.4 Dupliquer un dictionnaire	497
2.5 Utiliser le dictionnaire comme agrégateur de données	498
2.6 Méthodes d'itération	499
3. Utilisation avancée des dictionnaires	499
3.1 Rajouter une relation d'ordre	499
3.2 Algorithmiques classiques	500
3.3 Adapter les dictionnaires à des besoins spécifiques	503
3.4 Représentation universelle de données	505

Chapitre 3-9

Données temporelles et algorithmes appliqués

1. Gérer une date calendaire	507
1.1 Notion de date calendaire	507
1.2 Travailler sur une date	508
1.3 Considérations astronomiques	509
1.4 Considérations historiques	509
1.5 Considérations techniques	509
1.6 Représentation textuelle	510
2. Gérer un horaire ou un moment d'une journée	512
2.1 Notion d'instant	512
2.2 Notion de fuseau horaire	513
2.3 Représentation textuelle	513
3. Gérer un instant absolu	514
3.1 Notion d'instant absolu	514
3.2 Rapport avec les notions précédentes	515
3.3 Représentation textuelle	517

3.4	Gestion des fuseaux horaires	517
3.5	Créer une date à partir d'une représentation textuelle.	517
4.	Gérer une différence entre deux dates ou instants	518
4.1	Notion de différence et de résolution	518
4.2	Considérations techniques	519
4.3	Utilisation avec des dates calendaires	520
4.4	Utilisation avec des horaires.	520
4.5	Utilisation avec des dates absolues	520
4.6	La seconde comme unité de base	521
4.7	Précision à la nanoseconde	521
5.	Spécificités des fuseaux horaires.	521
6.	Problématiques de bas niveau	522
6.1	Timestamp et struct_time	522
6.2	Mesures de performances	523
7.	Utilisation du calendrier	526
7.1	Présentation du module calendar	526
7.2	Fonctions essentielles du calendrier.	530

Partie 4 : Les fonctionnalités

Chapitre 4-1

Manipulation de données

1.	Manipuler des fichiers	533
1.1	Ouvrir un fichier	533
1.2	Lire un fichier	534
1.3	Écrire un fichier	535
1.4	Comparer deux fichiers.	536
2.	Utilitaire de sauvegarde	538
3.	Lire un fichier de configuration.	538
4.	Format d'export/import	539
4.1	CSV	539
4.1.1	Exploiter un fichier CSV	540
4.1.2	Génération d'un fichier CSV	543
4.2	JSON	546
4.3	Base64	548
4.4	Pickle	549

5.	Compresser et décompresser un fichier	551
5.1	Tarfile	551
5.2	Gzip	553
5.3	BZ2	554
5.4	Zipfile	554
5.5	Interface de haut niveau	556
6.	Outils de manipulation de données	557
6.1	Générer des nombres aléatoires	557
6.2	Expressions régulières	558
7.	Cryptographie légère	562
7.1	Nombre aléatoire sécurisé	562
7.2	Fonctions de chiffrement	563
7.3	Code d'authentification de message	565
7.4	Empreinte de fichier	566
7.5	Stéganographie	566
7.6	Communication inter-applicative sécurisée	570

Chapitre 4-2 Bases de données

1.	Introduction	573
2.	Accès à une base de données relationnelle	573
2.1	Point d'entrée	573
2.2	MySQL	574
2.3	PostgreSQL	579
2.4	SQLite	581
2.5	Oracle	582
3.	Utilisation d'un ORM	582
3.1	Qu'est-ce qu'un ORM ?	582
3.2	ORM proposés par Python	582
3.3	SQLAlchemy	583
3.3.1	Introspection sur une table existante	583
3.3.2	Manipuler des données sur une table existante	586
3.3.3	Décrire une base de données par le code	590
4.	Autres bases de données	591
4.1	CSV	591
4.2	NoSQL	597
4.3	Base de données orientée objet : ZODB	598
4.4	Base de données orientée graphe : Neo4j	602

4.5	Base de données de type clé-valeur : Redis	604
4.6	Bases de données orientées documents : CouchDB et MongoDB . . .	605
4.7	Bases de données natives XML : BaseX, eXist	606
4.8	Cassandra	607
4.9	Bases de données orientées colonnes : HBase	607
4.10	Big Data : l'écosystème Hadoop	609
5.	LDAP	612
5.1	Protocole	612
5.2	Serveurs	612
5.3	Terminologie	612
5.4	Installation	613
5.5	Ouvrir une connexion à un serveur	613
5.6	Effectuer une recherche	614
5.7	Synchrone vs asynchrone	615
5.8	Connexions sécurisées	616

Partie 5 : Mise en pratique

Chapitre 5-1

Créer un environnement de travail en 10 minutes

1.	Description de l'application à construire	617
2.	Containers	618
2.1	Portainer	618
2.2	Base de données	619
3.	Créer son container Docker	621
4.	Installer ses bibliothèques Python	623

Chapitre 5-2

Créer une application web en 30 minutes

1.	Description de l'application à construire	625
2.	Mise en place	626
2.1	Isolation de l'environnement	626
2.2	Création du projet	627
2.3	Paramétrage	627
2.4	Premiers essais	629

3.	Réalisation de l'application	630
3.1	Modèles	630
3.2	Templates	632
3.3	Vues	634
4.	Pour aller plus loin	638

Chapitre 5-3

Créer une application console en 10 minutes

1.	Objectif	639
2.	Enregistrer le script	640
3.	Création des données	640
4.	Parseur d'arguments	641

Chapitre 5-4

Créer une application graphique en 20 minutes

1.	Objectif	643
1.1	Fonctionnel	643
1.2	Technique	643
2.	Présentation rapide de Gtk et d'astuces	644
2.1	Présentation	644
2.2	Astuces	644
3.	Démarrer le programme	645
4.	Interface graphique avec Glade	648
5.	Créer le composant graphique	650
6.	Contrôleur	652
7.	Autres bibliothèques graphiques	653
7.1	TkInter	653
7.2	wxPython	653
7.3	PyQt	653
7.4	PySide	654
7.5	Autres	654

Chapitre 5-5

Créer un jeu en 30 minutes avec PyGame

- 1. Présentation de PyGame 655
- 2. Réalisation d'un jeu Tetris 656
 - 2.1 Présentation du jeu 656
 - 2.2 Présentation des problématiques 657
 - 2.3 Création des constantes 657

Annexes

- 1. Objets mutables et non mutables. 669
- 2. Table Unicode 672
- 3. Bytes 672
- 4. Guide de portage vers Python 3 675
- 5. Comment déboguer 677
- 6. Comment tester la performance. 678

Index 681

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

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

Saisissez la référence de l'ouvrage **EIPYQT6** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

Chapitre 1 De Qt à PyQt

1. À l'origine, le framework Qt	13
1.1 Contexte historique	13
1.2 Qt comme réponse	13
1.3 Philosophie de Qt	14
2. Des éléments d'architecture de Qt.	14
2.1 Modules principaux	15
2.2 Quelques concepts spécifiques à Qt	15
2.3 Licences, forks et dérivés de Qt	16
2.4 Quelques solutions modernes avec Qt	17
3. Les interfaces de liaison : PyQt et le concept de binding de bibliothèques	18
3.1 Binding de bibliothèques	18
3.2 Le cas de PyQt.	18
3.3 Les autres bindings relatifs à Qt	19
3.4 Le paradigme induit pour le développeur PyQt	19
3.4.1 Une autre manière de travailler	19
3.4.2 Pourquoi la documentation Qt est indispensable en PyQt	19

2 Framework PyQt

Développez vos interfaces graphiques en Python avec PyQt6

4. De PyQt5 à PyQt6	21
4.1 Le contexte de cet ouvrage PyQt6	21
4.2 PyQt6	21
4.2.1 La sortie de PyQt6	21
4.2.2 Pourquoi Qt6/PyQt6	21
4.3 Les évolutions de Qt5 à Qt6 (PyQt5 à PyQt6)	22
4.3.1 Refonte des modules	22
4.3.2 Changement des noms de modules en PyQt6	22
4.3.3 Évolutions dans la gestion des types et des énumérations	23
4.3.4 Améliorations des performances et du rendu graphique	24
4.3.5 Refonte du module QtQuick et QML	24
4.3.6 Gestion de l'accessibilité et de l'internationalisation	25
5. De PyQt5 à PyQt6, une esquisse de méthodologie	25
5.1 Changements dans les imports	25
5.2 Changements dans les énumérations	26
5.3 Changements dans la gestion des types	26
5.4 Changements dans la gestion des signaux/slots	27
6. Conclusion	27

Chapitre 2

Notions générales et inventaire des widgets

1. Notions générales en PyQt	29
1.1 Installer PyQt	29
1.1.1 Installation sur Windows	29
1.1.2 Installation sur macOS	29
1.1.3 Installation sur Linux	30
1.1.4 Vérification de l'installation	31
1.2 Le rôle central de QApplication	31
1.3 QWidget et QMainWindow	32
1.4 L'organisation des widgets et la notion de layout	34

1.5	Signaux et slots	38
1.5.1	Signaux	38
1.5.2	Slots	38
1.5.3	Exemple signal/slot	39
1.5.4	Signaux/slots personnalisés	40
2.	Inventaire des widgets PyQt	41
2.1	Widget QWidget	41
2.2	Widget QLabel	42
2.3	Widget QLineEdit	44
2.4	Widget QPushButton	45
2.5	Widget QComboBox	46
2.6	Widget QSpinBox	47
2.7	Widgets QCheckBox et QRadioButton	48
2.8	Widget QDateTimeEdit	52
2.9	Widget QProgressBar	54
2.10	Widgets QTextEdit, QPlainTextEdit	56

Chapitre 3

Gestion des styles et apparences en PyQt6

1.	Introduction	59
2.	Manipulation des polices et l'objet QFont	63
2.1	Utilisation de QFont	63
2.2	La question des polices de caractères non installées	65
3.	Manipuler les images en PyQt	69
3.1	Utilisation de QPixmap pour les images	69
3.1.1	Exemple d'usage de QPixmap	69
3.1.2	Plus loin avec QPixmap pour les images	72
3.2	Utilisation de QPixmap pour les icônes	73
3.2.1	Introduction à la gestion des icônes	73
3.2.2	Fonctionnement général de la gestion des icônes	74
3.2.3	Exemple d'utilisation	74

4 Framework PyQt

Développez vos interfaces graphiques en Python avec PyQt6

4. Usage des ressources en PyQt	76
4.1 Introduction	76
4.2 Ressources et PyQt6	77
4.2.1 Format d'un fichier QRC	78
4.2.2 Génération du fichier Python de ressources	78
4.2.3 Utilisation des ressources dans le code	78
5. Les palettes en PyQt	79
5.1 Introduction	79
5.2 Utilisation de QPalette	80
5.3 Tableaux QPalette.ColorGroup et QPalette.ColorRole	82
5.3.1 QPalette.ColorGroup	82
5.3.2 QPalette.ColorRole	83
6. Les pinceaux en PyQt	84
6.1 Introduction	84
6.2 Utilisation de QBrush	84
6.3 Tableau des styles de pinceaux disponibles	86
7. Gestion des styles en PyQt	87
7.1 Introduction	87
7.2 Première utilisation de QSS	88
7.3 Tableaux des propriétés QSS	91
8. Conclusion	94

Chapitre 4

Les événements en PyQt6

1. Introduction	95
2. Signal et slot	96
2.1 Introduction	96
2.2 Ajout d'un signal et d'un slot	97
2.3 Autre manière de connecter un signal à un slot	99

3. Plus loin dans la gestion d'événements	100
3.1 La transmission de données	100
3.2 Les limitations issues de C++	104
4. Développer sa propre gestion d'événements	105
4.1 Introduction	105
4.2 Utilisation de PyQtSignal	105
4.3 Exemple de gestion personnalisée (PyQtSignal)	105
4.4 Passage de paramètres supplémentaires à des slots	108
4.4.1 Utilisation de lambda	108
4.4.2 Utilisation de partial	109
4.4.3 Exemple d'utilisation	109
5. Conclusion	111

Chapitre 5

Disposition en PyQt/Le modèle MVC

1. Introduction	113
2. Disposition en PyQt	114
2.1 Introduction	114
2.2 QHBoxLayout et QVBoxLayout	115
2.2.1 Présentation	115
2.2.2 Cas pratique avec QHBoxLayout	115
2.2.3 Transposition de l'exemple avec QHBoxLayout	117
2.3 Disposition en grille, QGridLayout	118
2.3.1 Introduction	118
2.3.2 Exemple d'utilisation : création d'une calculatrice	118
2.3.3 Plus loin avec la fonction addWidget	120
2.4 Disposition en formulaire, QFormLayout	123
2.4.1 Introduction	123
2.4.2 Exemple d'utilisation	123

6 **Framework PyQt**

Développez vos interfaces graphiques en Python avec PyQt6

2.5	Disposition en onglets, QTabWidget	126
2.5.1	Introduction	126
2.5.2	Exemple d'utilisation	126
2.6	QGroupBox.	130
2.6.1	Introduction	130
2.6.2	Exemple d'utilisation	131
2.7	QToolBar.	133
2.7.1	Introduction	133
2.7.2	Exemple d'utilisation	134
3.	Modèle-Vue-Contrôleur (MVC) et PyQt	136
3.1	Introduction	136
3.2	Les fondements de Modèle-Vue-Contrôleur (MVC)	137
3.2.1	Le modèle (M).	137
3.2.2	La vue (V)	137
3.2.3	Le contrôleur (C)	137
3.2.4	Exemple dans le cadre d'une application web.	138
3.3	Développement d'une application de calculatrice	139
3.3.1	Cahier des charges	139
3.3.2	Développement de la vue.	140
3.3.3	Gestion des données avec le modèle	142
3.3.4	Code principal.	145
3.4	Apparence écran LCD.	149
3.5	Développement d'un éditeur de texte simple	151
3.5.1	Introduction	151
3.5.2	Programmation de l'application.	153
3.5.3	Le code complet de l'application Notepad	157
3.6	Développement d'une application d'éditeur de texte enrichi	163
3.6.1	Introduction	163
3.6.2	Programmation de l'application.	164
3.6.3	Code global de l'application d'édition	169
4.	Conclusion	174

Chapitre 6

Développement rapide avec Qt Designer

1. Le développement rapide d'applications	175
1.1 Introduction	175
1.2 Exemples dans d'autres technologies.	176
1.2.1 Microsoft Access.	176
1.2.2 Ruby on Rails	176
1.2.3 MATLAB.	176
1.2.4 Flutter	176
1.3 PyQt6 et le RAD.	177
2. L'outil Qt Designer	177
2.1 Installation avec l'outil pip.	177
2.2 Avec PyQt6-tools	178
2.3 Avec PySide6.	178
2.4 Premier lancement de l'outil	179
3. Présentation de Qt Designer	180
3.1 Introduction	180
3.2 Les principales sections du logiciel	180
3.2.1 Section Widgets	180
3.2.2 Section Propriétés.	181
3.2.3 Section Objet (Object Inspector)	181
3.2.4 Zone principale (Form Editor)	181
3.3 Déroulé de la création d'une fenêtre	182
3.3.1 Création d'un projet	182
3.3.2 Ajout de widgets.	182
3.3.3 Organisation des widgets.	183
3.4 Conversion du fichier .ui en Python	183
3.4.1 Utilisation de l'outil pyuic6	183
3.4.2 Chargement direct avec QUiLoader	183
3.4.3 Gestion des événements.	184
3.4.4 Utilisation des QSS avec Qt Designer.	184
4. Conclusion	185

Chapitre 7

Les animations en PyQt

1. Introduction	187
2. Premières animations en PyQt	188
2.1 Introduction	188
2.2 Code avant animation	189
2.3 Une première animation géométrique avec PyQt6	190
2.4 Les limites de l'usage de QPropertyAnimation	193
2.4.1 Héritage obligatoire de QObject	193
2.4.2 Les propriétés éligibles	194
2.4.3 Les types de données supportés	194
2.4.4 Cas particulier : la gestion des couleurs	195
3. Thématique couleurs et animation en PyQt	195
3.1 Introduction	195
3.2 Le code explicite	195
4. Animations plus élaborées avec QPropertyAnimation	199
4.1 Introduction	199
4.2 Animations et courbes de Bézier	200
4.3 Création et animation d'un objet Italique avec une courbe de Bézier	201
4.3.1 Introduction	201
4.3.2 Définition de la classe Italique	201
4.3.3 Définition de la courbe de Bézier	202
4.3.4 Animation le long de la courbe de Bézier	202
4.3.5 Mise en œuvre finale	203
5. Notion de scène graphique en PyQt	206
5.1 Introduction	206
5.2 Exemple d'utilisation	206
5.2.1 Définition de la classe Italique	207
5.2.2 Création de la classe Fenetre	207
5.2.3 Code principal	208
6. Conclusion	212

Chapitre 8

Gestion et usage des bases de données en PyQt

1. Introduction	213
1.1 Les bases de données relationnelles	213
1.2 Le système de base de données SQLite	214
1.2.1 Contexte d'utilisation.	214
1.2.2 Pourquoi SQLite ? Un choix pragmatique	214
1.2.3 Utilisation de SQLite (avec PyQt).	215
1.2.4 Limites de SQLite	216
2. Exemple détaillé PyQt et SQLite	217
2.1 Création de la base de données	217
2.2 Connexion à la base de données	224
2.3 Première insertion de données dans la base de données.	225
2.4 Insertion des autres données dans la base de données	229
2.5 Modification de données existantes dans la base de données .	231
2.6 Exemple de requête utilisant une jointure	233
2.7 Requête préparée en PyQt	235
3. Plus loin avec l'utilisation de QSqlTableModel	237
3.1 Introduction	237
3.2 Exemple d'utilisation	238
4. L'utilisation de QSqlRelationalTableModel	241
4.1 Introduction	241
4.2 Exemple d'utilisation	242
5. Le mapping de données avec QDataWidgetMapper	245
5.1 Introduction	245
5.2 Exemple d'utilisation	246
6. Conclusion	250

Chapitre 9

Son et vidéo avec PyQt

1. Introduction	253
2. Le son dans PyQt	254
2.1 Jouer un simple son	254
2.2 Lire et enregistrer un son	257
2.2.1 Introduction	257
2.2.2 Vérification de la présence d'un micro	258
2.3 Développement d'une application de lecture et d'enregistrement audio.	261
2.3.1 Présentation de l'application	261
2.3.2 Développement de l'application	261
3. La vidéo avec PyQt	274
3.1 Introduction	274
3.2 Développement de l'application	275
4. Conclusion	282

Chapitre 10

Réseaux et gestion des couches basses en PyQt

1. Introduction	283
2. PyQt et les sockets	284
2.1 Introduction	284
2.2 L'adressage IP	285
2.3 Protocoles de transmission TCP et UDP.	287
2.4 Réalisation d'une messagerie instantanée locale basée sur UDP.	288
2.5 Réalisation d'une messagerie instantanée locale basée sur TCP	293

3. PyQt et les requêtes HTTP	298
3.1 Introduction	298
3.2 Exemples sur la thématique PyQt et HTTP	299
3.2.1 Récupérer le contenu d'une page web	299
3.2.2 Une requête POST en PyQt	302
4. Conclusion	303

Chapitre 11

Le multithreading en PyQt et sujets connexes

1. Introduction	305
1.1 Contexte du chapitre	305
1.2 Outils PyQt6 pour le multithreading	306
2. Utilisation de la classe QTimer	307
2.1 Introduction	307
2.2 Application utilisant QTimer	308
3. Le multithreading en PyQt	312
3.1 Introduction	312
3.2 Illustration de l'utilisation de QThread avec PyQt6	312
3.3 Exemple avancé avec QThread	318
3.4 Pool de threads en PyQt	323
3.4.1 Introduction	323
3.4.2 Exemple pédagogique autour de QThreadPool	324
4. Conclusion	328

Chapitre 12

Usages étendus en PyQt

1. Introduction	329
2. Diagrammes de données en PyQt	331
2.1 Introduction	331
2.2 Exemple de diagramme en PyQt : le pourcentage CPU	332

12 **Framework PyQt**

Développez vos interfaces graphiques en Python avec PyQt6

3. Web browsing en PyQt	337
3.1 Introduction	337
3.2 Développement d'un navigateur web	338
4. Modélisation 3D en PyQt	342
4.1 Introduction	342
4.2 Modélisation 3D en PyQt	342
Index	349