Prólogo

1. Contenido del libro **29**

2. Progresión del libro **30**

3. Destinado a profesores y alumnos **32**

4. Destinado a investigadores y doctores **33**

5. Destinado a aquellos que vienen de otro lenguaje **34**

Python en el paisaje informático

1. Breve historia de los lenguajes informáticos **35**

1.1 Informática teórica 35

1.2 Cronología de la informática 36

1.2.1 Evolución de las problemáticas vinculadas a la informática 36

1.2.2 Cronología de los lenguajes informáticos 37

1.3 Historia de Python 42

1.3.1 Génesis 42

1.3.2 Extensión del perímetro funcional 43

1.3.3 Evolución de la licencia

1.3.4 Porvenir 44

2. Tipología de los lenguajes de programación **45**

2.1 Paradigmas 45

2.1.1 Definición 45

2.1.2 Paradigma imperativo y derivados 46

2.1.3 Paradigma orientado a objetos y derivados 46

2.1.4 Programación orientada a aspectos 47

2.1.5 Paradigma funcional 47

2.1.6 Paradigma lógico 48

2.1.7 Programación concurrente 48

2.1.8 Síntesis 48

2.2 Interoperabilidad 49

2.3 Niveles de programación 51

2.3.1 Máquina 51

2.3.2 Bajo nivel 51

2.3.3 Alto nivel 52

2.4 Tipado 52

2.4.1 Débil vs. fuerte 52

2.4.2 Estático vs dinámico 53

2.5 Gramática

2.5.1 Lenguajes formales 53
2.5.2 Sintaxis 53
54

3. Ubicar Python en el paisaje informático 54

3.1 Tipología de Python 54
3.1.1 Gramática y sintaxis 54
3.1.2 Tipado 54
3.1.3 Nivel de programación 54
3.1.4 Paradigmas soportados 55
3.2 Posición estratégica del lenguaje Python 55
3.2.1 Segmentos de mercado 56
3.2.2 Nivel de complejidad 56
3.2.3 Fortalezas del lenguaje 56
3.2.4 Puntos débiles 56
3.3 Integración con otros lenguajes 57
3.3.1 Extensiones C 57
3.3.2 Integración de programas escritos en C 57
3.3.3 Integración de programas Python en C 58
3.3.4 Integración de programas escritos en Java 58
3.3.5 Integración de programas Python en Java 58
3.3.6 Otras integraciones 58

58

Presentación de Python

1. Filosofía **59**

1.1 Python en pocas líneas 59

1.1.1 ¿ De dónde proviene el nombre «Python» ? 59

1.1.2 Presentación técnica 60

1.1.3 Presentación conceptual 60

1.2 Comparación con otros lenguajes 60

1.2.1 Shell 60

1.2.2 Perl 61

1.2.3 C, C++ 61

1.2.4 Java 63

1.2.5 PHP 64

1.3 Grandes principios 66

1.3.1 El zen de Python 66

1.3.2 El desarrollador no es estúpido 66

1.3.3 Documentación 67

1.3.4 Python viene con todo incluido 67

1.3.5 Duck Typing 68

1.3.6 Noción de código pythónico

68

2. Gobierno **68**

2.1 Desarrollo 68

2.1.1 Ramas 68

2.1.2 Comunidad 69

2.2 Modo de gobierno 70

2.2.1 Creador del lenguaje 70

2.2.2 PEP 70

2.2.3 Toma de decisiones 70

3. ¿ Qué contiene Python ? **71**

3.1 Una gramática y una sintaxis 71

3.2 Varias implementaciones 71

3.3 Una librería estándar 72

3.4 Librerías de terceros 72

3.5 Frameworks 72

4. Fases de ejecución de un programa Python **73**

4.1 Carga de la máquina virtual 73

4.2 Compilación 73

4.3 Interpretación 73

Por qué escoger Python

1. Cualidades del lenguaje **75**

1.1 Cobertura funcional 75

1.2 Puerta de entrada 76

1.3 Garantías 77

2. Difusión **78**

2.1 Empresas 78

2.2 El mundo de la investigación 80

2.3 El mundo de la educación 80

2.4 Comunidad 81

3. Referencias **82**

3.1 Pesos pesados en la industria informática 82

3.1.1 Google 82

3.1.2 Mozilla 83

3.1.3 Microsoft 83

3.1.4 Canonical 84

3.1.5 Cisco 84

3.2 Empresas de innovación 85

3.2.1 Servicios de almacenamiento en línea 85
3.2.2 Cloud computing 85
3.2.3 Plataforma colaborativa (Forge) 85
3.2.4 Redes sociales 85
3.3 Editores de contenidos 86
3.3.1 Disney Animation Studio 86
3.3.2 YouTube 86
3.3.3 Box ADSL 86
3.3.4 Spotify 86
3.4 Fabricantes de software 86

4. Experiencia **87**

4.1 Impresiones 87
4.2 Desarrollo en un contexto heterogéneo 88
4.3 Desarrollo rápido 89
4.4 Venta de desarrollos en Python 89
4.5 Navegar por lo desconocido 89

Instalar el entorno de desarrollo

1. Instalar Python **91**

1.1 Windows

1.2 Mac OS 91
1.3 UNIX/Linux 94
1.4 Compilación 94
1.5 Compilación de Python 3.4 95
2. Instalar librerías externas 96
2.1 Instalador o administrador de paquetes 96
2.2 PIP 96
2.3 Entorno virtual 96
3. Instalar un IDE 97
3.1 Consola y herramientas asociadas 98
3.2 Eclipse + PyDev 98
3.3 Aptana 98
3.4 Eric 99
3.5 PyCharm 99
3.6 Otras soluciones 99
3.7 StacklessPython 99
3.8 Entorno heterogéneo 99
4. Uso de la consola 100

4.1 Consola estándar 100
4.2 iPython 100

5. Distribuir sus propias aplicaciones 101

5.1 Herramienta de distribución 101
5.2 Creación de paquetes distribuibles 101
5.3 Distribución binaria para Windows 102
5.4 Distribución binaria para Mac 102
5.5 Distribución binaria multiplataforma 102
5.6 Distribución específica para un entorno 103

Algoritmos básicos

1. Delimitadores 105

1.1 Instrucción 105
1.2 Una línea de código = una instrucción 105
1.3 Comentario 106
1.4 Una instrucción en varias líneas 106
1.5 Palabras clave 107
1.6 Palabras reservadas 107
1.7 Indentación 108
1.8 Símbolos

1.9 Operadores 109
1.10 Uso del carácter de subrayado 113
1.11 PEP-8 115
1.12 PEP-7 117
1.13 PEP-257 117

2. Instrucciones 118

2.1 Definiciones 118
2.1.1 Variable 118
2.1.2 Función 119
2.1.3 Funciones lambda 120
2.1.4 Clase 121
2.1.5 Instrucción vacía 122
2.1.6 Borrado 122
2.1.7 Devolver el resultado de la función 123
2.2 Instrucciones condicionales 125
2.2.1 Definición 125
2.2.2 Condición 125
2.2.3 Instrucción if 125
2.2.4 Instrucción elif 126
2.2.5 Instrucción else

2.2.6 Instrucción switch 126
2.2.7 Interrupciones 128
2.2.8 Profundizando en las condiciones 128
2.2.9 Rendimiento 128
2.3 Iteraciones 129
2.3.1 Instrucción for 131
2.3.2 Instrucción while 131
2.3.3 Diferencias entre for y while 131
2.3.4 Instrucción break 132
2.3.5 Instrucción return 132
2.3.6 Instrucción continue 134
2.3.7 Instrucción else 134
2.3.8 Generadores 135
2.4 Construcciones funcionales 136
2.4.1 Generadores 139
2.4.2 Recorrido de listas 139
2.4.3 Recorrido de conjuntos 139
2.4.4 Recorrido de diccionarios 139
2.5 Gestión de excepciones 139
2.5.1 Breve presentación de las excepciones 140
2.5.2 Elevar una excepción 140

2.5.3 ¿ Por qué elevar una excepción ? 140
2.5.4 Aserciones 141
2.5.5 Capturar una excepción 142
2.5.6 Manejar una excepción 143
2.5.7 Gestionar la salida del bloque de captura 144
2.5.8 Gestionar que no se produzcan excepciones 146
2.5.9 Uso y liberación de recursos 147
2.6 Otros 148
2.6.1 Gestionar imports 150
2.6.2 Compartir espacios de nombres 150
2.6.3 Funciones print, help, eval y exec 150
152

Declaraciones

1. Variable **155**
1.1 ¿ Qué es una variable ? 155
1.1.1 Contenido 155
1.1.2 Continente 155
1.1.3 Formas de modificar una variable 158
1.2 Tipado dinámico 161
1.2.1 Asignación: recordatorio

1.2.2 Primitiva type y naturaleza del tipo 161
1.2.3 Características del tipado Python 161
1.3 Visibilidad 162
1.3.1 Espacio global 164
1.3.2 Noción de bloque 164
2. Función 165
2.1 Declaración **169**
2.2 Parámetros 169
2.2.1 Firma de una función 171
2.2.2 Noción de argumento o de parámetro 171
2.2.3 Valor por defecto 171
2.2.4 Parámetros nombrados 172
2.2.5 Declaración de parámetros extensibles 173
2.2.6 Paso de parámetros con asterisco 174
2.2.7 Firma universal 176
2.2.8 Obligar a un parámetro a ser nombrado (keyword-only) 176
2.2.9 Anotaciones 177
3. Clase 179
3.1 Declaración **183**
183

3.1.1 Firma 183
3.1.2 Atributo 183
3.1.3 Método 184
3.1.4 Bloque local 184
3.2 Instanciación 185
3.2.1 Sintaxis 185
3.2.2 Relación entre la instancia y la clase 185

4. Módulo 186

4.1 Declaración 186
4.2 Instrucciones específicas 187
4.3 ¿ Cómo conocer el contenido de un módulo ? 187

Modelo de objetos

1. Todo es un objeto 189

1.1 Principios 189
1.1.1 Qué sentido dar a «objeto» 189
1.1.2 Adaptación de la teoría de objetos en Python 190
1.1.3 Generalidades 191
1.2 Clases 192
1.2.1 Introducción

1.2.2 Declaración imperativa de una clase 192
1.2.3 Instancia 192
1.2.4 Objeto en curso 193
1.2.5 Declaración por prototipo de una clase 195
1.3 Métodos 195
1.3.1 Declaración 197
1.3.2 Invocar al método 197
1.3.3 Métodos y atributos especiales 199
1.3.4 Constructor e inicializador 202
1.3.5 Gestión automática de atributos 206
1.3.6 Interés del paradigma orientado a objetos 207
1.3.7 Relación entre objetos 207
1.4 Herencia 208
1.4.1 Polimorfismo por subtipado 209
1.4.2 Sobrecarga de métodos 209
1.4.3 Sobrecarga de operadores 210
1.4.4 Polimorfismo paramétrico 212
1.4.5 Herencia múltiple 212
2. Otras herramientas de la programación orientada a objetos 214
2.1 Principios **217**

2.2 Interfaces 217
2.3 Atributos 217
2.4 Propiedades 220
2.5 Ubicaciones 223
2.6 Metaclases 225
2.7 Clases abstractas 226
2.8 Zope Component Architecture 229
2.8.1 Presentación 232
2.8.2 Instalación 232
2.8.3 Definir una interfaz y un componente 233
2.8.4 Otras funcionalidades 233
2.8.5 Ventajas de la ZCA 234

3. Funciones principales y primitivas asociadas 235

3.1 Personalización 235
3.1.1 Clases 235
3.1.2 Instancias 235
3.1.3 Comparación 237
3.1.4 Evaluación booleana 238
3.1.5 Relaciones de herencia o de clase a instancia 239
3.2 Clases particulares 239

3.2.1 Iterador 240
3.2.2 Contenedores 240
3.2.3 Instancias similares a funciones 242
3.2.4 Recursos que hay que proteger 242
3.2.5 Tipos 243
244

Tipos de datos y algoritmos aplicados

1. Números **245**
1.1 Tipos 245
1.1.1 Enteros 245
1.1.2 Reales 246
1.1.3 Cosas en común entre números enteros y reales 247
1.1.4 Métodos dedicados a los números enteros 248
1.1.5 Métodos dedicados a los números reales 249
1.1.6 Complejos 249
1.2 La consola Python, la calculadora por excelencia 251
1.2.1 Operadores matemáticos binarios 251
1.2.2 Operadores binarios particulares 252
1.2.3 Operadores matemáticos unarios 253
1.2.4 Redondeo

1.2.5 Operadores de comparación 254
1.2.6 Operaciones matemáticas n-arias 257
1.2.7 Funciones matemáticas usuales 258
1.3 Representaciones de un número 259
1.3.1 Representación decimal 265
1.3.2 Representación por un exponente 265
1.3.3 Representación por una fracción 265
1.3.4 Representación hexadecimal 265
1.3.5 Representación octal 266
1.3.6 Representación binaria 268
1.3.7 Operaciones binarias 268
1.3.8 Longitud de la representación en memoria de un entero 268
1.4 Conversiones 271
1.4.1 Conversión entre enteros y reales 272
1.4.2 Conversión entre reales y complejos 272
1.4.3 Conversión en un booleano 273
1.5 Trabajar con variables 273
1.5.1 Un número es inmutable 274
1.5.2 Modificar el valor de una variable 274
1.5.3 Operadores incrementales 275
1.6 Estadísticas 276

1.7 Cálculo científico 277
1.7.1 El cálculo científico ¿ para qué hacerlo ? 278
1.7.2 Python, una alternativa libre y con credibilidad 278
1.7.3 Algunas librerías 279

2. Secuencias **280**

2.1 Presentación de los distintos tipos de secuencias 280
2.1.1 Generalidades 280
2.1.2 Las listas 281
2.1.3 Las n-tuplas 282
2.1.4 Conversión entre listas y n-tuplas 284
2.1.5 Cosas en común entre una lista y una n-tupla 284
2.1.6 Noción de iterador 285
2.2 Uso de índices y tramos 288
2.2.1 Definición de índice de un objeto y sus ocurrencias 288
2.2.2 Utilizar el índice para recorrer la secuencia 289
2.2.3 Encontrar las ocurrencias de un objeto y sus índices 290
2.2.4 Tamaño de una lista, contar ocurrencias 292
2.2.5 Utilizar el índice para modificar o eliminar 293
2.2.6 Iteración simple 295
2.2.7 Presentación de la noción de tramos (slices)

2.2.8 Caso particular de la rama 2.x de Python 299
2.2.9 Uso básico de tramos 308
2.2.10 Uso avanzado de tramos 309
2.3 Uso de operadores 311
2.3.1 Operador + 313
2.3.2 Operador * 313
2.3.3 Operador += 314
2.3.4 Operador *= 317
2.3.5 Operador in 318
2.3.6 Operadores de comparación 319
2.4 Métodos de modificación 320
2.4.1 Agregar elementos a una lista y a una n-tupla 322
2.4.2 Eliminar un objeto de una lista y de una n-tupla 322
2.4.3 Soluciones alternativas para la modificación de n-tuplas 324
2.4.4 Invertir una lista o una tupla 328
2.4.5 Ordenar una lista 329
2.5 Uso avanzado de listas 331
2.5.1 Operaciones de conjunto 333
2.5.2 Pivotar una secuencia 333
2.5.3 Iterar correctamente 334
2.5.4 Programación funcional 336

2.5.5 Recorrido de listas 337
2.5.6 Iteraciones avanzadas 339
2.5.7 Combinatoria 341
2.6 Adaptar las listas a necesidades específicas 346
2.6.1 Lista de enteros 348
2.6.2 Presentación del tipo array 348
2.6.3 Utilizar una lista como una pila 350
2.6.4 Utilizar una lista como una lista de espera 352
2.6.5 Utilizar las listas para representar matrices 352
2.6.6 Lista sin duplicados 354
2.7 Otros tipos de datos 355
3. Conjuntos 358
3.1 Presentación **360**
3.1.1 Definición de un conjunto 360
3.1.2 Diferencias entre set y frozenset 360
3.1.3 Uso para eliminar valores duplicados de las listas 361
3.1.4 Agregar una relación de orden 362
3.2 Operaciones sobre conjuntos 362
3.2.1 Operadores para un conjunto a partir de otros dos 363
3.2.2 Operadores para modificar un conjunto a partir de otro 363

3.2.3 Métodos equivalentes a la creación o modificación de conjuntos 365
3.2.4 Métodos de comparación de conjuntos 365
3.2.5 Ejemplos de uso poco clásicos 366
3.3 Métodos de modificación de un conjunto 367
3.3.1 Agregar un elemento 371
3.3.2 Eliminar un elemento 371
3.3.3 Vaciar un conjunto 371
3.3.4 Duplicar un elemento 372
3.3.5 Sacar un valor de un conjunto 372
3.3.6 Utilizar un conjunto como un almacén de objetos 373
3.3.7 Algorítmica avanzada: resolución del problema de las n reinas 374
4. Cadenas de caracteres 377
4.1 Presentación **379**
4.1.1 Definición 379
4.1.2 Vocabulario 379
4.1.3 Especificidades de la rama 2.x 380
4.1.4 Cambios aportados por la rama 3.x 382
4.1.5 Cadena de caracteres como secuencia de caracteres 383
4.1.6 Caracteres 385
4.1.7 Operadores de comparación 388

4.2 Dar formato a cadenas de caracteres 389
4.2.1 Operador módulo 391
4.2.2 Métodos para dar formato al conjunto de la cadena 391
4.2.3 Nuevo método para dar formato a variables en una cadena 397
4.3 Operaciones de conjunto 400
4.3.1 Secuenciación de cadenas 404
4.3.2 Operaciones sobre mayúsculas y minúsculas 404
4.3.3 Búsqueda en una cadena de caracteres 406
4.3.4 Información sobre los caracteres 407
4.4 Problemáticas relativas a la codificación 408
4.4.1 Codificación por defecto 410
4.4.2 Codificación del sistema 410
4.4.3 Unicode, referencia absoluta 410
4.4.4 Otras codificaciones 410
4.4.5 Puntos entre el Unicode y el resto del mundo 412
4.4.6 Volver a Unicode 412
4.5 Manipulaciones de bajo nivel avanzadas 414
4.5.1 Operaciones para contar 415
4.5.2 Una cadena de caracteres vista como una lista 415
4.5.3 Una cadena de caracteres vista como un conjunto de caracteres 416
4.6 Representación en memoria 417

4.6.1 Presentación del tipo bytes 417
4.6.2 Vínculo con las cadenas de caracteres 417
4.6.3 Presentación del tipo bytearray 418
4.6.4 Gestión de un juego de caracteres 420
421

5. Diccionarios 427

5.1 Presentación 427
5.1.1 Definición 427
5.1.2 Evolución y diferencias entre las ramas 2.x y 3.x 428
5.1.3 Vistas de diccionarios 429
5.1.4 Instanciación 432
5.1.5 Recorrer un diccionario 432
5.2 Manipular un diccionario 433
5.2.1 Recuperar un valor de un diccionario 433
5.2.2 Modificar los valores de un diccionario 434
5.2.3 Eliminar una entrada de un diccionario 435
5.2.4 Duplicar un diccionario 435
5.2.5 Utilizar un diccionario como un agregador de datos 436
5.2.6 Métodos de iteración 437
5.3 Uso avanzado de diccionarios 437
5.3.1 Agregar una relación de orden

5.3.2 Algorítmica clásica 437
5.3.3 Adaptar los diccionarios a necesidades específicas 441
5.3.4 Representación universal de datos 442
443

6. Booleanos **445**

6.1 El tipo booleano 445
6.1.1 Clase bool 445
6.1.2 Los dos objetos True y False 445
6.1.3 Diferencia entre el operador de igualdad y de identidad 446
6.2 Evaluación booleana 446
6.2.1 Método genérico 446
6.2.2 Objetos clásicos 446

7. Datos temporales **447**

7.1 Gestionar una fecha del calendario 447
7.1.1 Noción de fecha del calendario 447
7.1.2 Trabajar con una fecha 448
7.1.3 Consideraciones astronómicas 449
7.1.4 Consideraciones históricas 449
7.1.5 Consideraciones técnicas 449
7.1.6 Representación textual 451

7.2 Gestionar un horario o un momento de la jornada 452
7.2.1 Noción de instante 452
7.2.2 Noción de huso horario 453
7.2.3 Representación textual 454
7.3 Gestionar un instante absoluto 455
7.3.1 Noción de instante absoluto 455
7.3.2 Relación con las nociones anteriores 455
7.3.3 Representación textual 457
7.3.4 Gestión de los husos horarios 458
7.3.5 Crear una fecha a partir de una representación textual 458
7.4 Gestionar una diferencia entre dos fechas o instantes 459
7.4.1 Noción de diferencia y de resolución 459
7.4.2 Consideraciones técnicas 460
7.4.3 Uso con fechas del calendario 461
7.4.4 Uso con horarios 461
7.4.5 Uso con fechas absolutas 461
7.4.6 El segundo como unidad básica 462
7.5 Especificidades de los husos horarios 462
7.6 Problemáticas de bajo nivel 464
7.6.1 Timestamp y struct_time 464
7.6.2 Medidas de rendimiento 465

7.7 Uso del calendario 467
7.7.1 Presentación del módulo calendar 467
7.7.2 Funciones esenciales del calendario 472

Patrones de diseño

1. Definición **475**
1.1 Situación respecto a la noción de objeto 475
1.2 Organización del capítulo 476
1.3 Situación respecto a otros conceptos 477

2. Creación **477**
2.1 Singleton 477
2.2 Fábrica 478
2.2.1 Presentación de la problemática 478
2.2.2 Soluciones 478
2.2.3 Conclusiones 481
2.3 Fábrica abstracta 481
2.3.1 Presentación de la problemática 481
2.3.2 Solución 481
2.4 Constructor 481
2.4.1 Presentación de la problemática

2.4.2 Solución 481
2.4.3 Conclusiones 482
2.5 Prototipo 484
2.5.1 Presentación de la problemática 484
2.5.2 Solución 484
2.5.3 Conclusiones 485
3. Estructuración 487
3.1 Adaptador 487
3.1.1 Presentación de la problemática 487
3.1.2 Solución 487
3.1.3 Conclusiones 488
3.2 Puente 490
3.2.1 Presentación de la problemática 491
3.2.2 Solución 491
3.2.3 Conclusiones 492
3.3 Composite 494
3.3.1 Presentación de la problemática 494
3.3.2 Solución 494
3.3.3 Conclusiones 494
3.4 Decorador 496

3.4.1 Presentación de la problemática 497
3.4.2 Solución 497
3.4.3 Conclusiones 499
3.5 Fachada 500
3.5.1 Presentación de la problemática 500
3.5.2 Solución 500
3.5.3 Conclusiones 502
3.6 Peso mosca 502
3.6.1 Presentación de la problemática 502
3.6.2 Solución 502
3.6.3 Conclusiones 503
3.7 Proxy 503
3.7.1 Presentación de la problemática 503
3.7.2 Solución 504
3.7.3 Conclusiones 506

4. Comportamiento **506**

4.1 Cadena de responsabilidad 506
4.1.1 Presentación de la problemática 506
4.1.2 Solución 506
4.1.3 Conclusiones

4.2 Solicitud 507
4.2.1 Presentación de la problemática 507
4.2.2 Solución 507
4.2.3 Conclusiones 508
4.3 Iterador 509
4.3.1 Presentación de la problemática 509
4.3.2 Solución 509
4.3.3 Conclusiones 510
4.4 Memento 512
4.4.1 Presentación de la problemática 512
4.4.2 Solución 512
4.4.3 Conclusiones 512
4.5 Visitante 513
4.5.1 Presentación de la problemática 513
4.5.2 Solución 513
4.5.3 Conclusiones 514
4.6 Observador 514
4.6.1 Presentación de la problemática 515
4.6.2 Solución 515
4.6.3 Conclusiones 515
4.7 Estrategia 516

4.7.1 Presentación de la problemática 516
4.7.2 Solución 516
4.7.3 Conclusiones 516
4.8 Retrollamada 517
4.8.1 Presentación de la problemática 517
4.8.2 Solución 517
4.8.3 Conclusiones 517
5. ZCA **518**
5.1 Consideraciones 518
5.2 Adaptador 518
5.2.1 Declaración 519
5.2.2 Uso 519
5.3 Utilidad 520
5.3.1 Declaración 521
5.3.2 Uso 521
5.4 Fábrica 522
5.4.1 Declaración 522
5.4.2 Uso 522
5.5 Para ir más allá 523

Manipulación de datos

1. Bases de datos **525**

1.1 Presentación 525

1.2 Acceso a una base de datos relacional 526

1.2.1 Punto de entrada 526

1.2.2 MySQL 526

1.2.3 PostgreSQL 531

1.2.4 SQLite 533

1.2.5 Oracle 534

1.3 Uso de un ORM 534

1.3.1 ¿ Qué es un ORM ? 534

1.3.2 ORM propuestos por Python 535

1.3.3 SQLAlchemy 535

1.4 Otras bases de datos 542

1.4.1 CSV 542

1.4.2 NoSQL 550

1.4.3 Base de datos orientada a objetos: ZODB 550

1.4.4 Base de datos de tipo clave-valor: REDIS 555

1.4.5 Bases de datos orientadas a documentos: CouchDB y MongoDB 557

2. LDAP

558

2.1 Presentación 558

2.1.1 Protocolo 558

2.1.2 Servidores 559

2.1.3 Terminología 559

2.2 Instalación 559

2.3 Abrir una conexión a un servidor 560

2.4 Realizar una búsqueda 561

2.5 Síncrono vs asíncrono 562

2.6 Conexiones seguras 563

3. XML **564**

3.1 XML y las tecnologías relacionadas 564

3.1.1 Definición de XML, terminología asociada 564

3.1.2 Noción de esquema 565

3.1.3 Ventajas e inconvenientes de XML 566

3.1.4 Distintas maneras de recorrer un archivo XML 567

3.1.5 Módulos Python dedicados a XML 568

3.2 Validar un documento XML 569

3.2.1 Documento XML 569

3.2.2 Esquema DTD 570

3.2.3 Esquema XSD

3.2.4 Esquema RNG (RelaxNG) 570
3.2.5 Schematron 571
3.3 DOM 572
3.3.1 Lectura 572
3.3.2 Escritura 572
3.4 SAX 573
3.4.1 Soporte de SAX en lxml 575
3.4.2 API SAX ligera 575
3.5 XPath 576
3.6 XSLT 577
3.7 El caso concreto de los archivos HTML 580
3.7.1 Problemática 581
3.7.2 Parsear un archivo HTML según DOM 581
3.7.3 Parsear un archivo HTML según SAX 581

4. Herramientas de manipulación de datos 583

4.1 Encriptar un dato **585**
4.1.1 Funciones de hash 585
4.1.2 Código de autenticación del mensaje 585
4.1.3 Esteganografía 587
4.2 Generar números aleatorios 588

4.3 Expresiones regulares 592

5. Trabajar con medios gráficos **593**

5.1 Imágenes **597**

5.1.1 Representación informática de una imagen 597

5.1.2 Presentación de Pillow 597

5.1.3 Formatos de imágenes matriciales 598

5.1.4 Recuperar la información de una imagen 600

5.1.5 Operaciones de conjunto sobre una imagen 602

5.1.6 Trabajar con capas o con píxeles 603

605

Generación de contenido

1. PDF **609**

1.1 Presentación 609

1.1.1 Formato PDF 609

1.1.2 Ventajas 609

1.1.3 Inconvenientes 610

1.1.4 Presentación de la librería libre 610

1.2 Bajo nivel 610

1.2.1 Librería de datos 610

1.2.2 Canvas 613
1.3 Alto nivel 614
1.3.1 Estilos 614
1.3.2 Flujo de datos 616
1.3.3 Creación de un elemento visual 618
1.3.4 Plantilla de página 619
1.3.5 Página que contiene varias zonas 620

2. OpenDocument 623
2.1 Instalación 623
2.2 OpenDocument Texto 623
2.2.1 Hello World 623
2.3 OpenDocument Hoja de cálculo 623
2.3.1 Principios generales respecto al texto 623
2.3.2 Ir más allá 624

Programación paralela

1. Terminología 625
1.1 Proceso 625
1.2 Tarea 626

2. Uso de una tarea **626**

2.1 Gestión de una tarea 626

2.1.1 Presentación 626

2.1.2 Creación 627

2.2 Gestión de varias tareas 630

2.2.1 Arranque y control 630

2.2.2 Oportunidad de utilizar una tarea 632

2.3 Resolución de problemáticas asociadas 634

2.3.1 Sincronización 634

2.3.2 Sincronización condicional 637

2.3.3 Semáforo 640

3. Uso de procesos **641**

3.1 Gestión de un proceso 641

3.1.1 Presentación 641

3.1.2 Creación 642

3.2 Gestión de varios procesos 644

3.2.1 Sincronización 644

3.2.2 Paralelizar un trabajo 645

3.3 Resolución de problemáticas asociadas 648

3.3.1 Comunicación interproceso 648

3.3.2 Compartir datos entre procesos 649
3.4 Oportunidad de utilizar los procesos 650
3.5 Demonio 651

4. Ejecución asíncrona 653

4.1 Introducción 653
4.2 Presentación 654
4.3 Programación asíncrona 660

Programación de sistema y de red

1. Presentación 663

1.1 Definición 663
1.2 Objetivos del capítulo 664

2. Escribir scripts de sistema 664

2.1 Conozca su sistema operativo 664
2.1.1 Advertencia 664
2.1.2 Sistema operativo 664
2.1.3 Procesos en curso 665
2.1.4 Usuarios y grupos 666
2.1.5 Constantes para el sistema de archivos

2.1.6 Gestionar las rutas 668
2.2 Gestión de archivos 669
2.2.1 Abrir un archivo 670
2.2.2 Leer un archivo 670
2.2.3 Escribir un archivo 671
2.2.4 Cambiar los permisos de un archivo 672
2.2.5 Cambiar de propietario o de grupo 673
2.2.6 Recuperar información relativa al archivo 675
2.2.7 Eliminar un archivo 676
2.3 Alternativas sencillas a los comandos bash habituales 676
2.3.1 Carpetas 677
2.3.2 Archivos 677
2.3.3 Módulo de alto nivel 679
2.3.4 Buscar un archivo 680
2.4 Ejecutar comandos externos 682
2.4.1 Ejecutar y mostrar el resultado 683
2.4.2 Ejecutar y recuperar el resultado 683
2.5 Herramientas 684
2.5.1 Diferencias entre archivos 685
2.5.2 Herramienta de salvaguarda 685
2.5.3 Leer un archivo de configuración 687

2.5.4 Pickle 688
2.6 Comprimir y descomprimir un archivo 689
2.6.1 Tarfile 692
2.6.2 Gzip 692
2.6.3 Bz2 694
2.6.4 Zipfile 695
2.6.5 Interfaz de alto nivel 695
3. Trabajar con argumentos 697
3.1 Presentación **699**
3.2 Implementación 699
4. Programación de red 699
4.1 Escribir un servidor y un cliente **703**
4.1.1 Uso de un socket TCP 703
4.1.2 Uso de un socket UDP 703
4.1.3 Creación de un servidor TCP 707
4.1.4 Creación de un servidor UDP 710
4.1.5 Un poco más allá 712
4.2 Utilizar un protocolo estándar 713
4.2.1 HTTP 714
714

4.2.2 Proxy 719
4.2.3 Cookies 720
4.2.4 FTP y SFTP 720
4.2.5 SSH 723
4.2.6 POP y POPS 725
4.2.7 IMAP e IMAPS 726
4.2.8 SMTP y SMTPS 727
4.2.9 NNTP 732
4.2.10 IRC 733
4.3 Servicios web 737
4.3.1 REST 737
4.3.2 SOAP 738
4.3.3 Pyro 740

5. Uso de hardware 741

5.1 Wake-on-LAN 741
5.1.1 Requisitos previos 741
5.1.2 Implementación 741
5.2 Uso del puerto serie 742

Buenas prácticas

1. Desarrollo guiado por pruebas **745**

1.1 Pruebas unitarias 745

1.1.1 Principios 745

1.1.2 Interpretación 746

1.1.3 Cobertura 747

1.1.4 Herramientas 747

1.2 Pruebas de no regresión 750

1.2.1 Acciones de desarrollo 750

1.2.2 Gestión de las anomalías detectadas 750

1.3 Pruebas funcionales 751

1.4 Pruebas de rendimiento 752

1.5 Integración continua 755

2. Programación dirigida por la documentación **756**

2.1 Documentación interna 756

2.1.1 Destinada a los desarrolladores 756

2.1.2 Destinada a los usuarios 757

2.2 Documentación externa 758

2.2.1 Presentación 758

2.2.2 Inicio rápido 758

2.2.3 Resultado

760

3. Optimización **761**

3.1 Medir la calidad 761

3.2 Herramientas de depuración 763

3.3 Herramientas de perfilado 764

3.4 Reglas de optimización 765

3.4.1 ¿ Por qué optimizar ? 765

3.4.2 Reglas generales 766

3.4.3 Optimizar un algoritmo 767

3.4.4 Optimizar el uso de la memoria 778

Crear una aplicación web en 30 minutos

1. Descripción de la aplicación que se va a construir **781**

2. Implementación **782**

2.1 Aislar el entorno 782

2.2 Creación del proyecto 783

2.3 Configuración 784

2.4 Primeros ensayos 785

3. Realizar la aplicación **785**

3.1 Modelos 786
3.2 Vistas 788
3.3 Controladores 790

4. Para ir más allá 795

Crear una aplicación de consola en 10 minutos

1. Objetivo 797

2. Registrar el script 798

3. Creación de los datos 798

4. Parser de argumentos 799

Crear una aplicación gráfica en 20 minutos

1. Objetivo 801

1.1 Funcional 801
1.2 Técnica 801

2. Breve presentación de Gtk y algunos trucos 802

2.1 Presentación 802
2.2 Trucos 803

3. Iniciar el programa 804

4. Interfaz gráfica con Glade 806

5. Crear el componente gráfico 809

6. Controlador 811

7. Otras librerías gráficas 812

7.1 TkInter 812
7.2 wxPython 812
7.3 PyQt 813
7.4 PySide 813
7.5 Otras 813

Crear un juego en 30 minutos con PyGame

1. Presentación de PyGame 815

2. Construcción de un juego Tetris 817

2.1 Presentación del juego

2.2 Presentación de la problemática 817
2.3 Creación de constantes 817
818

Anexos

1. Tabla UNICODE **833**
1.1 Script 833
2. Bytes **834**
2.1 Script 834
2.2 Resultado 834
índice 839