

Les exemples à télécharger sont disponibles à l'adresse suivante :
<http://www.editions-eni.fr>
Saisissez la référence ENI de l'ouvrage **RI17JAV** dans la zone de recherche
et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

Chapitre 1 Présentation

1. Introduction	13
2. Historique	14
2.1 Pourquoi Java ?	14
2.2 Objectifs de la conception de Java	15
2.3 Essor de Java	16
3. Les caractéristiques du langage Java	18
3.1 Simple	18
3.2 Orienté objet	19
3.3 Distribué	20
3.4 Interprété	20
3.5 Robuste	21
3.6 Sécurisé	21
3.7 Indépendant des architectures	22
3.8 Portable	22
3.9 Performant	23
3.10 Multitâche	23
3.11 Dynamique	23
4. La plateforme Java SE	24
4.1 La machine virtuelle Java (JVM)	24
4.2 L'API Java	26
4.2.1 Les API de base	26
4.2.2 Les API d'accès aux données et d'intégration avec l'existant	28

4.2.3	Les API de gestion de l'interface des applications avec l'utilisateur	28
4.3	Les outils de déploiement des applications	29
4.4	Les outils d'aide au développement	29
5.	Les différentes implémentations de la plateforme	30
6.	L'environnement de développement	30
6.1	Installation du JDK sur Windows	31
6.1.1	Installation de la plateforme OpenJDK	31
6.1.2	Installation de la plateforme Oracle JDK	32
6.1.3	Configuration des variables d'environnement	34
6.1.4	Tester le bon fonctionnement	35
6.2	Installation du JDK sur Linux	36
6.2.1	Installation de la plateforme OpenJDK	36
6.2.2	Installation de la plateforme Oracle JDK	36
6.2.3	Configuration de la variable JAVA_HOME	38
6.2.4	Tester le bon fonctionnement	39
6.3	Installation d'Eclipse	40
6.3.1	Téléchargement de l'installeur	40
6.3.2	Installation d'Eclipse	41
6.3.3	Création d'un projet Java	46
6.4	Installation de MySQL	48
6.4.1	Windows	48
6.4.2	Linux	55
6.4.3	Utilisation de MySQL Workbench	57
7.	La javadoc	60
7.1	Consultation depuis un navigateur	60
7.2	Consultation depuis Eclipse	65
8.	Le premier programme	67
8.1	Écriture du code	67
8.2	Compilation et exécution du code	68
8.2.1	Théorie	68
8.2.2	Mise en œuvre	69

8.3 Utilisation d'Eclipse 71
 9. Conclusion 73

Chapitre 2
Bases du langage

1. Anatomie d'un programme 75
 2. Les variables 76
 2.1 Introduction 76
 2.2 Les emplacements 76
 2.3 Le nom 77
 2.4 Les types 78
 2.4.1 Présentation 78
 2.4.2 Les types valeurs 79
 2.4.3 Les types références 79
 2.5 La déclaration, l'initialisation, l'affectation 80
 2.5.1 La déclaration 80
 2.5.2 L'initialisation 80
 2.5.3 L'affectation 82
 2.6 Les types valeurs 82
 2.6.1 Les types numériques entiers 82
 2.6.2 Les types numériques décimaux 84
 2.6.3 L'utilisation de valeurs littérales 84
 2.6.4 Le type caractère 86
 2.6.5 Le type boolean 88
 2.7 Les types références 88
 2.7.1 Les tableaux 88
 2.7.2 Les chaînes de caractères 96
 2.7.3 La notion de Text Blocks 104
 2.7.4 Les dates et les heures 106
 2.8 Les valeurs par défaut 110
 2.9 La portée des variables 111
 2.10 La durée de vie des variables 112

2.11	La conversion de type	112
2.11.1	Présentation	112
2.11.2	La conversion entre numériques	113
2.11.3	La conversion vers une chaîne de caractères	114
2.11.4	La conversion depuis une chaîne de caractères	116
2.12	L'inférence de type	118
3.	Les constantes	119
4.	Les énumérations	120
5.	Les arguments d'un programme	123
5.1	Fonctionnement	123
5.2	Utilisation dans Eclipse	124
6.	Les opérateurs	126
6.1	Les opérateurs unaires	126
6.2	L'opérateur d'affectation	127
6.3	Les opérateurs arithmétiques	128
6.4	Les opérateurs bit à bit	129
6.4.1	La représentation binaire des entiers	129
6.4.2	Les opérations logiques	131
6.5	Les opérateurs de comparaison	133
6.6	L'opérateur de concaténation	134
6.7	Les opérateurs logiques	136
6.8	Ordre d'évaluation des opérateurs	137
7.	Les structures de contrôle	137
7.1	Présentation	137
7.2	Structures de décision	138
7.2.1	Structure if	138
7.2.2	Structure ternaire	139
7.2.3	Structure switch historique	140
7.2.4	Structure switch nouvelle génération	142
7.3	Les structures de boucle	144
7.3.1	Structure while	145
7.3.2	Structure do ... while	145

- 7.3.3 Structure for 146
- 7.4 Interruption d'une structure de boucle 149
 - 7.4.1 break 149
 - 7.4.2 continue 150
 - 7.4.3 return. 151
- 8. Exercices 151
 - 8.1 Exercice 1. 151
 - 8.2 Exercice 2. 152
 - 8.3 Exercice 3. 152
 - 8.4 Exercice 4. 153
- 9. Corrections 153
 - 9.1 Exercice 1. 153
 - 9.2 Exercice 2. 155
 - 9.3 Exercice 3. 156
 - 9.4 Exercice 4. 157
- 10. Conclusion 158

Chapitre 3
Programmation objet

- 1. Introduction 159
- 2. Mise en œuvre avec Java 162
 - 2.1 Contexte 162
 - 2.2 Création d'une classe 163
 - 2.2.1 Déclaration de la classe 163
 - 2.2.2 Création des champs 164
 - 2.2.3 Création des méthodes. 166
 - 2.2.4 Création des surcharges de méthode. 168
 - 2.2.5 Passage de paramètres 171
 - 2.2.6 Création des accesseurs 172
 - 2.2.7 Création des constructeurs et des destructeurs 174
 - 2.2.8 Création de champs et méthodes statiques 175
 - 2.2.9 Utilisation des annotations 178

2.3	Utilisation d'une classe.	181
2.3.1	Création d'une instance	181
2.3.2	Initialisation d'une instance.	182
2.3.3	Destruction d'une instance	184
2.4	Héritage.	188
2.4.1	Création d'une classe fille.	188
2.4.2	this et super	191
2.4.3	Classes abstraites	196
2.4.4	Classes finales	197
2.4.5	Classes scellées	198
2.4.6	Conversion de type.	199
2.4.7	La classe Object	206
2.5	Interfaces.	213
2.5.1	Création d'une interface.	214
2.5.2	Utilisation d'une interface	215
2.5.3	Méthode par défaut	219
2.5.4	Méthode statique	222
2.5.5	Interfaces scellées	222
2.6	Classes imbriquées	223
2.6.1	Classes imbriquées statiques	223
2.6.2	Classes internes (d'instance)	224
2.6.3	Classes anonymes.	225
2.7	Les records.	231
3.	Les packages	233
3.1	Présentation	233
3.2	Création d'un package	234
3.3	Utilisation et importation d'un package	236
3.4	Importation des méthodes statiques.	238
4.	Les modules.	239
4.1	Mise en place.	239
4.2	Présentation	239
4.3	Le JDK est modulaire	240
4.4	Utilisation des modules	241

4.5	Création d'un nouveau module	242
5.	La gestion des erreurs	245
5.1	Les différents types d'erreurs	245
5.1.1	Les erreurs de syntaxe	245
5.1.2	Les erreurs d'exécution	246
5.1.3	Les erreurs de logique	246
5.2	La représentation objet des erreurs	246
5.3	Le traitement des exceptions	248
5.4	Les exceptions associées à des ressources	253
5.5	Utilisation des exceptions	254
5.6	Création et déclenchement d'exceptions	256
6.	Les génériques	258
6.1	Présentation	258
6.2	Classes génériques	260
6.2.1	Définition d'une classe générique	260
6.2.2	Utilisation d'une classe générique	265
6.2.3	Les méthodes génériques	268
6.3	Les génériques et l'héritage	269
6.4	Limitations des génériques	274
7.	Les collections	276
7.1	Présentation	276
7.2	La classe ArrayList	279
7.3	La classe HashSet	283
7.4	La classe LinkedList	289
7.5	La classe HashMap	290
7.6	Streams et pipelines	292
8.	Exercices	293
8.1	Exercice 1	293
8.2	Exercice 2	293
8.3	Exercice 3	293
8.4	Exercice 4	294

9. Corrections	294
9.1 Correction de l'exercice 1	294
9.2 Correction de l'exercice 2	297
9.3 Correction de l'exercice 3	302
9.4 Correction de l'exercice 4	309

Chapitre 4

Les expressions lambda

1. Introduction	317
2. Fonctionnement	317
2.1 Les interfaces fonctionnelles	317
2.2 Les méthodes anonymes	320
2.2.1 Syntaxe générale	320
2.2.2 Déclaration des paramètres	320
2.2.3 Déclaration du corps	321
2.2.4 Utilisation des variables "externes"	322
2.3 Les références de méthodes	322
2.3.1 Méthode d'instance	322
2.3.2 Méthode de classe	323
2.3.3 Constructeur	323
2.4 L'API <code>java.util.function</code>	323
2.4.1 Présentation de l'API	323
2.4.2 Utilisation	324
3. Manipulation des collections	328
3.1 L'API <code>Stream</code>	328
3.2 Théorie	328
3.3 Obtenir un <code>Stream</code>	330
3.3.1 Obtenir un <code>Stream</code> générique	330
3.3.2 Obtenir un <code>Stream</code> de numérique	331
3.4 Utiliser un <code>Stream</code>	332
3.4.1 Utiliser un <code>Stream</code> générique	332
3.4.2 Utiliser un <code>Stream</code> numérique	336

3.5 La classe Optional<T> 338
 4. Conclusion 339

Chapitre 5
Application graphique

1. Introduction 341
 1.1 Les bibliothèques graphiques 342
 1.1.1 La bibliothèque AWT 342
 1.1.2 La bibliothèque Swing 343
 1.2 Constitution de l'interface graphique d'une application 343
 2. Conception d'une interface graphique 344
 2.1 Les fenêtres 344
 2.2 Le thread EDT 350
 2.3 La gestion des événements 353
 2.4 Aspect des composants 383
 2.5 Le positionnement des composants 386
 2.5.1 FlowLayout 387
 2.5.2 BorderLayout 389
 2.5.3 GridLayout 393
 2.5.4 BoxLayout 396
 2.5.5 GridBagLayout 400
 2.5.6 Sans gestionnaire de mise en page 405
 2.6 Les composants graphiques 407
 2.6.1 La classe JComponent 408
 2.6.2 Affichage d'informations 411
 2.6.3 Les composants d'édition de texte 418
 2.6.4 Les composants de déclenchement d'actions 425
 2.6.5 Les composants de sélection 430
 2.7 Les boîtes de dialogue 440
 2.7.1 La boîte de saisie 441
 2.7.2 La boîte de message 444
 2.7.3 La boîte de confirmation 444

2.8	Les traitements longs	446
2.8.1	Déléguer les traitements à un thread enfant	447
2.8.2	Mettre à jour l'IHM depuis un thread enfant	448
2.8.3	Utiliser la classe SwingWorker	450
3.	Conclusion	456

Chapitre 6

Accès aux bases de données

1.	Principe de fonctionnement d'une base de données	457
1.1	Terminologie	457
1.2	Le langage SQL	458
1.2.1	Recherche d'informations	459
1.2.2	Ajout d'informations	461
1.2.3	Mise à jour d'informations	461
1.2.4	Suppression d'informations	462
2.	Accès à une base de données à partir de Java	462
2.1	Présentation de JDBC	464
2.2	Chargement du pilote	465
2.3	Établir et manipuler la connexion	469
2.3.1	Établir la connexion	469
2.3.2	Manipuler la connexion	471
2.4	Exécution d'instructions SQL	477
2.4.1	Exécution d'instructions de base avec le type Statement	478
2.4.2	Exécution d'instructions paramétrées avec l'objet PreparedStatement	488
2.4.3	Exécution de procédures stockées avec l'objet CallableStatement	494
2.5	Utilisation des jeux d'enregistrements avec l'interface ResultSet	497
2.5.1	Positionnement dans un ResultSet	499
2.5.2	Lecture des données dans un ResultSet	503

- 2.5.3 Modification des données dans un ResultSet 508
- 2.5.4 Suppression de données dans un ResultSet 510
- 2.5.5 Ajout de données dans un ResultSet 512
- 2.6 Gestion des transactions 514
 - 2.6.1 Mise en œuvre des transactions 516
 - 2.6.2 Points de sauvegarde 518
 - 2.6.3 Niveaux d'isolement 518

Chapitre 7

Déploiement d'applications

- 1. Archive Java 521
 - 1.1 Présentation 521
 - 1.2 Manipulation d'une archive 522
 - 1.2.1 Création d'une application 522
 - 1.2.2 Création d'une archive 523
 - 1.2.3 Visualisation du contenu 525
 - 1.2.4 Extraction 525
 - 1.2.5 Mise à jour 526
 - 1.2.6 Exécution 526
 - 1.3 Le manifest 527
 - 1.3.1 Présentation 527
 - 1.3.2 Création 528
 - 1.4 La gestion des dépendances 530
 - 1.4.1 Dans un sous-répertoire 530
 - 1.4.2 À l'intérieur même de l'archive 532
- 2. Création d'une application autonome avec jlink 534
 - 2.1 La création de l'archive Java 535
 - 2.2 La création du module de l'application 536
 - 2.3 La recherche des dépendances avec jdeps 536
 - 2.4 La création de l'arborescence avec jlink 538
 - 2.5 L'ajout des dépendances non modulaires 540
 - 2.6 Le test de l'application 541

3. Externalisation des paramètres	541
3.1 Création du fichier de configuration	541
3.2 Utilisation du fichier de configuration	543
3.3 Déploiement de l'application	544
4. Création d'un installeur avec jpackage	545
5. Conclusion	547
Index	549

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

Avant-propos

Chapitre 1

Installation et configuration initiale

1. Introduction	13
2. Installation du kit de développement Java	14
2.1 Avant de commencer : note sur la plateforme Java	14
2.1.1 Les éléments de la plateforme : JVM, JRE, JDK	14
2.1.2 Focus sur la JVM.	14
2.1.3 Un environnement pour chaque utilisation	15
2.2 Quel JDK sélectionner ?	15
2.3 Quelle version sélectionner ?	17
3. Installation de NetBeans	19
4. Configuration initiale	23

Chapitre 2

Créer son premier projet avec NetBeans

1. Introduction	35
2. Les types de projets Java	36
3. Création d'un projet Java simple avec Maven	42
4. Notions de programmation orientée objet	44
4.1 Définition d'une classe	44

2 _____ Apache NetBeans

Développez vos applications en Java

4.2	Modélisations, classes, types et exécution en Java	45
4.2.1	Les classes métier	46
4.2.2	Les points d'entrée d'exécution du programme	47
4.3	Notion de package	47
4.3.1	Définition	47
4.3.2	Règle de nommage	48
5.	Présentation de l'interface de NetBeans	49
6.	Création d'une classe Java avec NetBeans	51
6.1	Création d'une classe	51
6.2	Création d'un attribut	53
6.3	Création d'un accesseur et d'un mutateur	54
6.4	Création d'une méthode	56
7.	Définition d'un point d'entrée pour l'exécution de son programme	57
8.	Compilation d'un projet	58
8.1	Définition du compilateur	58
8.2	Compilation d'un projet avec NetBeans	58
9.	Exécution d'un projet	59
9.1	Sélectionner le point d'entrée du programme	59
9.2	Construire son projet avec NetBeans	60
10.	Documentation d'une application	63
10.1	Tags Javadoc	63
10.2	Analyse de la Javadoc	67
10.3	Exécution de la Javadoc	68
11.	Exercice corrigé : Calcul géométrique	72

Chapitre 3
Les outils du développeur

- 1. Introduction 83
- 2. Débogage d'une application grâce à NetBeans 83
 - 2.1 Découvrir le mode debug 84
 - 2.2 Apprendre les notions importantes 87
 - 2.2.1 Les déplacements pas à pas 87
 - 2.2.2 Les points d'arrêts 88
 - 2.2.3 Les observateurs 89
 - 2.3 Déboguer un programme 90
- 3. Capacités de génération de code 93
 - 3.1 Les attributs 93
 - 3.2 Les constructeurs 94
 - 3.3 Les méthodes equals() et hashCode() 95
 - 3.4 La méthode toString() 97
 - 3.5 Les méthodes comportementales 98
 - 3.6 La méthode clone() 100
 - 3.6.1 Le clonage en surface 101
 - 3.6.2 Le clonage en profondeur 103
- 4. Refactoring avec NetBeans 104
 - 4.1 L'introduction de variables 104
 - 4.2 L'introduction de constantes 105
 - 4.3 L'introduction d'attributs 106
 - 4.4 L'introduction de méthodes 107
 - 4.5 L'introduction de paramètres 109
- 5. Internationalisation d'un projet 110
- 6. Diagrammes de classes UML avec NetBeans 117
 - 6.1 Installation du plugin easyUML 120
 - 6.2 Génération d'un diagramme de classes 122
- 7. Groupement de projets 128

4 Apache NetBeans

Développez vos applications en Java

Chapitre 4

Le développement d'applications riches

1. Introduction	133
2. Objectif des applications riches	134
3. Programmation graphique : composants et évènements	135
4. Bonnes pratiques pour la programmation au sein des frameworks graphiques	136
4.1 Séparation de la couche métier et de la couche de présentation	137
4.2 Intrication du code métier et du code de présentation	138
4.3 Débordement de la couche métier dans la couche graphique .	139
5. Cas concret d'utilisation d'une librairie graphique et implémentation avec NetBeans	139
6. Implémentation d'un exemple avec Swing	141
6.1 Maquettage avec Swing	141
6.1.1 Création d'un projet Java pour démarrer avec Swing . .	141
6.1.2 Fonctionnement global de Swing et intégration dans NetBeans	145
6.1.3 Édition des propriétés, génération d'évènements	146
6.1.4 Utilisation de l'historique	148
6.2 Composants et programmation des interactions	149
6.2.1 Manipulation du JTable	152
6.2.2 Manipulation du JFileChooser	154
6.2.3 Mise en place d'un thème	155
7. Implémentation d'un exemple avec JavaFX	157
7.1 Différence d'approche entre Swing et JavaFX	157
7.2 Maquettage avec JavaFX	158
7.2.1 Création d'un projet JavaFX	159
7.2.2 Configuration de Scene Builder dans NetBeans	161
7.2.3 Première exécution	162
7.2.4 Fonctionnement global de JavaFX	164
7.2.5 Prise en main de SceneBuilder	165

- 7.2.6 Création d'un panneau (Scene) 167
- 7.2.7 Programmation des interactions avec l'utilisateur 171
- 7.3 Composants et programmation des interactions 174
 - 7.3.1 Manipulation du TableView 174
 - 7.3.2 Manipulation du FileChooser 177
- 8. Swing ou JavaFX? 178

Chapitre 5

Vers l'industrialisation du logiciel

- 1. Introduction 181
- 2. Travail en équipe : intégration avec Git 181
 - 2.1 Fonctionnement de Git 182
 - 2.1.1 Git, un système distribué 182
 - 2.1.2 L'aspect concurrent de Git 183
 - 2.2 Git : branches et tags 184
 - 2.3 Partage et modification d'un projet avec Git et NetBeans. . . . 186
 - 2.3.1 Intégration initiale 186
 - 2.3.2 Réalisation et intégration (locale et distante)
de modifications 188
 - 2.3.3 Intégration de la modification sur le repository local .. 189
 - 2.3.4 Intégration de la modification
dans un repository distant 190
 - 2.4 Suivi des modifications 191
 - 2.4.1 Intégration dans l'éditeur de code 191
 - 2.4.2 Affichage de l'historique d'un fichier 191
 - 2.4.3 Gestion des conflits 193
- 3. Construction du projet : utilisation de Maven dans NetBeans. . . . 194
 - 3.1 Fonctionnement de Maven 194
 - 3.1.1 Maven : la convention comme philosophie 194
 - 3.1.2 Structuration du projet 195
 - 3.1.3 Gestion des dépendances 196
 - 3.1.4 Système de coordonnées 196

6 Apache NetBeans

Développez vos applications en Java

3.1.5	Référentiels (repository) d'artefacts	197
3.1.6	Nature d'une dépendance : packaging et classifier	199
3.2	Configuration de la construction du projet	200
3.2.1	Flux de construction d'un projet Maven	200
3.2.2	Instructions de construction grâce aux plugins	201
3.2.3	Exemple : mise en œuvre avec la construction de la Javadoc	202
3.2.4	Configuration des plugins Maven	204
3.3	Déclenchement de constructions depuis NetBeans	205
3.3.1	Première construction	205
3.3.2	Historiques et enregistrement des paramètres de construction dans NetBeans	206
3.4	Ajout de dépendances	208
3.4.1	Ajout assisté dans le pom.xml	208
3.4.2	Visualisation de l'arbre des dépendances	208
3.4.3	Conflits et exclusion de dépendance	211
3.5	Réutilisation de projets : héritage et composition	212
3.5.1	Définition d'un projet parent	213
3.5.2	Déclaration du parent dans les projets fils	214
3.5.3	Effet de la déclaration d'un parent sur le projet	215
3.5.4	Limites de l'héritage	216
3.5.5	Composition grâce au BOM	216
3.6	Construction groupée de projet : projets multimodules	217
3.6.1	Conception d'un projet multimodule	217
3.6.2	Distinction entre le réacteur et le parent	218
4.	Exécution des tests unitaires dans NetBeans	219
4.1	Méthodologie de définition du test unitaire	219
4.1.1	Approche systématique : hypothèse, action, vérification	219
4.1.2	Principe d'indépendance et de transparence des tests	220
4.1.3	Méthodologie de développement intégrant les tests : BDD et TDD	221
4.1.4	Exemple de formalisation du test BDD	222

- 4.1.5 Différence entre une erreur (error) et un échec (failure) . 223
- 4.2 Définition d'un test unitaire avec JUnit 224
 - 4.2.1 Écriture d'un premier test 225
 - 4.2.2 Exécution des actions systématiques
grâce aux annotations 229
 - 4.2.3 Assertions 230
 - 4.2.4 Interprétation du résultat des tests 231
 - 4.2.5 Génération des tests unitaires par NetBeans 232
- 5. Analyse de la qualité de code intégrée à l'outil 235
 - 5.1 Utilisation de l'analyseur de code intégré de NetBeans 235
 - 5.1.1 Analyse globale 236
 - 5.1.2 Analyse sur critère unique 237
 - 5.1.3 Analyse en vue d'une migration 237
 - 5.2 Couverture des tests 237
 - 5.2.1 Couverture en lignes et couverture en branches 237
 - 5.2.2 Configuration et analyse de la couverture de test
dans NetBeans 239
 - 5.2.3 Comparaison avec la couverture de test
de JaCoCo - Java Code Coverage 244
 - 5.3 Intégration de SonarQube 246
 - 5.3.1 Introduction à SonarQube 246
 - 5.3.2 Installation d'un serveur SonarQube local 247
 - 5.3.3 Les sept axes de la qualité selon SonarQube 248
 - 5.3.4 Installation de SonarLint4Netbeans 249
 - 5.3.5 Analyse d'un projet grâce au plugin Maven sonar 251
- 6. Exercice corrigé : Conception d'un service de lecture
de fichiers CSV en approche TDD 254
 - 6.1 Spécifications de l'exercice 254
 - 6.1.1 Préparation du projet 254
 - 6.1.2 Modélisation des données 254
 - 6.1.3 Désérialisation grâce à la classe CSVReader 255

6.2	Solution de l'exercice	255
6.2.1	Préparation du projet	255
6.2.2	Modélisation des données	257
6.2.3	Désérialisation grâce à la classe de service CSVReader	258
6.2.4	Recherche d'amélioration	263

Chapitre 6

La conception et l'exploitation de services

1.	Introduction	265
2.	Créer une base de données	266
2.1	Créer une base de données H2	266
2.2	Initialiser la base de données H2	269
3.	Développer des services REST pour exposer des APIs	272
3.1	Notion d'API	274
3.2	Principe KISS	275
3.3	Veiller à l'aspect Stateless de l'API	276
3.4	Verbes de la grammaire REST	277
3.5	Installation d'un serveur d'applications WildFly	279
3.6	Création d'un projet d'application Web	281
3.7	Génération d'entités et services REST	283
3.7.1	À propos des entités et de JPA	283
3.7.2	Manipulation	283
3.7.3	Bonnes pratiques de conception	287
3.8	Configuration et test de services REST	288
4.	Développer des services SOAP	292
4.1	Créer un projet Spring Boot	294
4.2	Générer les entités Java	295
4.3	Créer des objets de transfert de données	298
4.4	Développer des services Soap	301
4.5	Définir l'adresse d'un service web	304
4.6	Exécuter un projet	305

4.7	Tester des services Soap	306
5.	Exercice corrigé : développer une API REST avec Spring	309
5.1	Créer un projet Spring Boot	310
5.2	Créer l'entité Etudiant	311
5.3	Créer la classe EtudiantDAO	313
5.4	Créer la classe EtudiantController	316
5.5	Tester l'API REST	319

Chapitre 7

NetBeans, Java et le Web

1.	Introduction	323
2.	Présentation de l'architecture Java EE	324
2.1	Le pattern Modèle - Vue - Contrôleur	325
2.1.1	Composants de base du Web en Java	327
2.1.2	MVC, MVC 2 et les frameworks Java	329
2.2	Le pattern MVP	330
3.	Création d'une interface web avec JSF	332
3.1	Créer une application web	333
3.2	Générer les entités Java	334
3.3	Générer les pages JSF	336
3.4	Tester l'interface web	339
3.5	Manipuler la palette JSF de NetBeans	344
4.	Création d'une interface JSP	350
4.1	Créer une première page JSP	350
4.2	Développer une application en JSP	353
4.2.1	Les balises	353
4.2.2	Les directives	354
4.2.3	Le développement d'une page JSP	355
4.2.4	Le développement d'une servlet	367
4.2.5	Le développement du JavaScript	371

5. Exercice corrigé : développement d'une application web JSF	373
5.1 Créer une application Web	374
5.2 Créer l'entité Etudiant	375
5.3 Créer la classe EtudiantDAO	377
5.4 Créer la classe EtudiantPresenter	379
5.5 Créer les pages JSF	382
5.6 Tester l'application JSF	386
5.7 Aller plus loin	389

Chapitre 8

Le profilage d'applications Java

1. Introduction	391
2. Avant de commencer : les grands principes de fonctionnement de la JVM	392
2.1 Gestion automatique de la mémoire	392
2.1.1 Les grandes zones mémoire de la JVM	392
2.1.2 Les mécanismes de recyclage : garbage collector	394
2.2 Aspect multithread	395
3. Premier profilage d'une application avec NetBeans	396
3.1 Déclenchement et configuration du premier profilage	396
3.1.1 Lancement et calibration	398
3.1.2 Choix des métriques	400
3.1.3 Télémétrie globale de l'application	402
3.2 Techniques de relevés pour le profilage	403
3.2.1 Le sampling	404
3.2.2 L'observation continue	406
4. Analyses possibles relatives à la mémoire	406
4.1 Analyse globale de la mémoire	406
4.1.1 Observation et interprétation du comportement de la mémoire	407
4.1.2 Observation de la mémoire dans un cas de mauvaise conception	410

- 4.2 Analyse des instances en cours d'exécution 412
 - 4.2.1 Observation des instances en mode sampling 412
 - 4.2.2 Observation continue des instances en cours d'exécution 415
 - 4.2.3 Avantages et limites de l'observation de la mémoire en cours d'exécution 418
- 4.3 Analyses de la mémoire à froid 418
 - 4.3.1 Réalisation d'un dump et découverte du HeapWalker . . 419
 - 4.3.2 Comparaison de dumps 423
 - 4.3.3 Console OQL 425
- 5. Analyses possibles relatives à l'utilisation du processeur 427
- 6. Autres possibilités du profileur de NetBeans 429

- Index 431