

Programmation GWT 2

**Développer des applications
HTML5/JavaScript en Java
avec Google Web Toolkit**

2^e édition

Sami Jaber

© Groupe Eyrolles, 2012, ISBN : 978-2-212-13478-0

EYROLLES



Table des matières

| | |
|--|-----------|
| Introduction à GWT | 1 |
| GWT et HTML 5 | 3 |
| Dix lignes de code GWT pour convaincre | 4 |
| Masquer la complexité du Web | 4 |
| Coder en Java | 5 |
| <i>Typage statique</i> | 5 |
| <i>Débogage</i> | 6 |
| <i>Refactoring</i> | 6 |
| <i>Tests unitaires</i> | 7 |
| <i>Compétences largement disponibles</i> | 7 |
| Support multinavigateur | 8 |
| Performances | 9 |
| Qu'est-ce qu'Ajax ? | 11 |
| La navigation en mode SPI | 13 |
| L'architecture RPC | 14 |
| Modèle 1.0 versus modèle 2.0 | 14 |
| GWT face aux autres frameworks Ajax | 16 |
| Quelle place pour GWT face à ses concurrents ? | 17 |
| Un projet collaboratif | 18 |
| Une communauté active | 19 |
| Des navigateurs de plus en plus performants | 20 |
| L'évolution et les nouveautés de GWT 2 | 20 |
| | |
| CHAPITRE 1 | |
| L'environnement de développement | 23 |
| Télécharger et installer GWT | 23 |
| Contenu du répertoire d'installation | 23 |
| L'ensemble logiciel GWT | 24 |
| Création du premier projet GWT | 24 |
| Exécuter l'application | 26 |
| Notion de module | 27 |

| | |
|--|----|
| Structure d'un projet GWT | 28 |
| Le package client | 29 |
| Le package serveur | 30 |
| Les fichiers de configuration | 30 |
| La structure du répertoire war | 31 |
| La page HTML hôte | 31 |
| Le mode développement | 33 |
| Le shell | 36 |
| Le conteneur de servlets Jetty | 37 |
| Le mode production | 38 |
| La structure d'un site compilé | 39 |
| Les types Java émülés par GWT | 40 |
| Le déploiement | 42 |
| Fichier Ant | 42 |
| Plug-in Maven | 43 |
| La fonctionnalité « Super DevMode » dans GWT 2.5 | 45 |

CHAPITRE 2

Les contrôles **47**

| | |
|---|----|
| Les classes UIObject et Widget | 47 |
| Les feuilles de styles CSS | 49 |
| La syntaxe | 50 |
| Les styles dépendants | 51 |
| Les styles prédéfinis | 52 |
| La gestion des événements | 54 |
| Tour d'horizon des widgets | 56 |
| Les composants de formulaires | 56 |
| SuggestBox | 59 |
| Les bundles d'images | 60 |
| Les hyperliens | 63 |
| Les conteneurs et gestionnaires de placement | 64 |
| Les conteneurs simples (Panels) | 64 |
| <i>FormPanel</i> | 65 |
| <i>LazyPanel</i> | 66 |
| Les conteneurs complexes | 67 |
| <i>Exemple d'utilisation</i> | 68 |
| <i>HTMLPanel</i> | 70 |
| Synthèse des conteneurs GWT | 72 |

CHAPITRE 3

Le modèle de placement CSS..... 75

| | |
|--|----|
| Pourquoi un modèle de placement ? | 75 |
| Les limites du modèle GWT 1.x | 76 |
| Une solution basée sur le standard CSS | 79 |
| Les API | 82 |
| Les nouveaux conteneurs de GWT 2 | 84 |
| Composant StackLayoutPanel | 85 |
| Le widget TabLayoutPanel | 86 |
| Sous le capot | 86 |

CHAPITRE 4

Les bibliothèques tierces..... 89

| | |
|--|-----|
| L'écosystème GWT | 89 |
| Les bibliothèques de composants graphiques | 90 |
| L'incubateur GWT | 90 |
| Sencha Ext-GWT (GXT v3) | 90 |
| SmartGWT | 96 |
| Glisser-déplacer avec GWT-DnD | 100 |
| Les courbes et graphiques | 102 |
| <i>GChart</i> | 103 |
| <i>GWT HighCharts</i> | 104 |
| Les frameworks complémentaires | 105 |
| Vaadin | 105 |
| La gestion des traces | 106 |
| Manipuler les services Google avec GWT | 107 |
| Gwt-google-apis | 107 |
| <i>GWT-GData</i> | 109 |
| Conclusion | 113 |

CHAPITRE 5

L'intégration de code JavaScript..... 115

| | |
|---|-----|
| Comprendre JSNI | 115 |
| Mise en pratique | 116 |
| Intégration d'un fichier JavaScript externe | 118 |
| Invoquer une méthode Java en JavaScript | 119 |
| Accéder à des attributs Java en JavaScript | 121 |
| Correspondance des types entre Java et JavaScript | 122 |
| Instancier un type Java en JavaScript | 123 |
| Le type JavaScriptObject (JSO) | 124 |

| | |
|--|-----|
| Undefined vs null | 126 |
| Gestion des exceptions JSNI | 127 |
| Appeler une méthode Java à partir d'un code JavaScript externe | 128 |
| Les types Overlay | 128 |
| Un peu d'histoire | 129 |
| Mise en pratique des Overlay | 131 |
| Intégration Overlay et JSON | 134 |
| Sous le capot | 135 |
| L'implémentation unique du type JSO | 138 |
| Les contraintes associées à un JSO | 140 |
| Effet de JSNI sur le framework GWT | 140 |
| La magie interne de JSNI | 141 |

CHAPITRE 6

La création de composants personnalisés..... 143

| | |
|---|-----|
| Quelques mots sur le DOM | 143 |
| La mécanique des événements | 146 |
| Pourquoi JavaScript fuit-il ? | 146 |
| Propagation par bouillonnement et capture | 147 |
| Expando et fuite mémoire | 149 |
| Créer un composant dérivé de Widget | 152 |
| Aller plus loin avec l'API événementielle | 157 |
| Dériver de la classe Composite | 158 |
| Dériver de la classe UIObject | 161 |
| Attachement dans un conteneur | 161 |

CHAPITRE 7

Les services RPC..... 163

| | |
|--|-----|
| L'architecture RPC | 164 |
| Les étapes de la construction d'un service | 165 |
| Créer les deux interfaces de service | 166 |
| Créer les objets d'échange | 168 |
| Coder l'implémentation | 169 |
| Coder et configurer le client | 170 |
| La sérialisation | 172 |
| La gestion des exceptions | 173 |
| Exceptions non vérifiées | 175 |
| Accès à la requête HTTP | 176 |
| Bonnes pratiques et mode asynchrone | 177 |
| Déploiement | 178 |

| | |
|---|------------|
| Déploiement des classes | 179 |
| Configuration du fichier web.xml | 179 |
| Configuration dans un réseau sécurisé avec un frontal web | 179 |
| Classe RequestBuilder et services REST | 181 |
| Appel d'URL basique | 181 |
| Architecture REST | 186 |
| | |
| CHAPITRE 8 | |
| L'intégration J2EE..... | 187 |
| Le modèle par délégation | 187 |
| Le modèle d'extensibilité | 189 |
| L'option <code>-noserver</code> | 193 |
| Intégration avec les EJB 3 et JPA | 193 |
| Le problème des classes instrumentées | 202 |
| Intégration des protocoles RMI, Corba et Soap | 207 |
| | |
| CHAPITRE 9 | |
| Le chargement à la demande | 209 |
| Principe général | 209 |
| Les types de fragments | 213 |
| Positionner efficacement les points de rupture | 218 |
| Le design pattern Async package | 223 |
| Forcer le chargement des fragments | 225 |
| CodeSplitting V2 | 227 |
| Sous le capot | 229 |
| Conclusion | 231 |
| | |
| CHAPITRE 10 | |
| La liaison différée | 233 |
| Principe général | 233 |
| Mise en pratique | 236 |
| Le script de sélection | 239 |
| Propriétés, règles et conditions | 239 |
| Propriétés | 240 |
| Propriétés de configuration | 242 |
| Les propriétés conditionnelles | 243 |
| Règles de liaison | 244 |
| Générateurs de code | 246 |
| Dans la pratique | 247 |
| Déboguer | 251 |

| | |
|-----------------------------|-----|
| Conditions de liaison | 252 |
| Conclusion | 252 |

CHAPITRE 11

La gestion des ressources 253

| | |
|--|-----|
| La problématique des ressources | 253 |
| Installation et configuration | 255 |
| Les différents types de ressources | 256 |
| Les ressources textuelles (TextResource) | 257 |
| Les ressources textuelles asynchrones | 259 |
| Les ressources binaires externes | 263 |
| Les ressources images | 264 |
| Les options de la liaison différée | 267 |
| L'injection dynamique CSS | 267 |
| L'injection différée | 269 |
| Les constantes | 270 |
| La substitution à l'exécution | 271 |
| Les fonctions de valeur | 272 |
| Les directives conditionnelles | 274 |
| Les préfixes de style | 276 |
| Les sprites d'images | 277 |

CHAPITRE 12

Sous le capot de GWT 279

| | |
|---|-----|
| Introduction au compilateur | 280 |
| Vive les fonctions JavaScript ! | 280 |
| Les étapes du compilateur | 283 |
| Lecture des informations de configuration | 284 |
| Création de l'arbre syntaxique GWT | 284 |
| La création de code JavaScript et les optimisations | 285 |
| <i>La réduction de code (pruning)</i> | 288 |
| <i>La finalisation de méthodes et de classes</i> | 290 |
| <i>La substitution par appels statiques.</i> | 290 |
| <i>La réduction de type</i> | 291 |
| <i>L'élimination de code mort</i> | 292 |
| <i>L'inlining</i> | 292 |
| Tracer les optimisations | 293 |
| Les options du compilateur | 295 |
| Accélérer les temps de compilation | 299 |
| Les linkers | 299 |

| | |
|---------------------------------------|-----|
| La pile d'erreurs en production | 305 |
| Table des symboles | 309 |

CHAPITRE 13

L'internationalisation 311

| | |
|--|-----|
| La problématique | 311 |
| Paramétrer et définir la locale courante | 312 |
| L'API i18n | 313 |
| Les dictionnaires à constantes statiques | 314 |
| Dictionnaire par recherche dynamique de constantes | 316 |
| Les messages | 317 |
| Notion de langue par défaut | 318 |
| Signification, exemple et description | 319 |
| Les formes plurielles | 320 |
| Conversion des types | 322 |
| Formats monétaires | 322 |
| Date et formats horaires | 323 |
| Création automatique de dictionnaires | 325 |
| Bénéfices de l'internationalisation statique | 326 |
| Externalisation dynamique | 327 |
| L'outillage | 328 |
| i18nCreator | 328 |
| I18nSync | 329 |

CHAPITRE 14

L'environnement de tests 331

| | |
|---|-----|
| GWT et la problématique des tests | 331 |
| La mixité des tests | 332 |
| Créer un test unitaire | 332 |
| Les suites de tests | 335 |
| Une architecture modulaire et extensible | 336 |
| Le style HtmlUnit – moteur de test par défaut | 338 |
| Le style manuel ou interactif | 340 |
| Le style Selenium | 340 |
| Le style distant | 342 |
| Le style externe | 342 |
| Synthèse des différentes options et annotations | 343 |
| Tests de charge avec la classe Benchmark | 343 |
| Les compteurs intégrés de performance | 346 |
| Tests fonctionnels robotisés : scénarios joués | 348 |

| | |
|--|-----|
| Selenium IDE | 349 |
| <i>Le module WebDriver</i> | 354 |
| Les stratégies de test par bouchon (mocking) | 356 |
| Quel est l'atelier de tests idéal ? | 359 |

CHAPITRE 15

Les design patterns GWT..... 361

| | |
|--|-----|
| Pourquoi des bonnes pratiques ? | 361 |
| Gestion de la session (cliente et serveur) | 362 |
| Limiter les besoins mémoire de la session cliente | 363 |
| La gestion côté serveur | 364 |
| Session et onglets des navigateurs | 365 |
| Gestion de l'historique | 367 |
| Que signifie le contexte précédent avec Ajax ? | 370 |
| Les traitements longs | 372 |
| La classe Timer | 373 |
| La classe Scheduler | 374 |
| <i>Les traitements différés</i> | 374 |
| <i>Les traitements incrémentaux</i> | 375 |
| Séparer présentation et traitement | 378 |
| Le pattern Commande | 378 |
| L'approche Modèle Vue Contrôleur | 381 |
| <i>MVC par l'exemple avec le framework PureMVC</i> | 382 |
| <i>Le pattern MVP</i> | 384 |
| Les failles de sécurité | 386 |
| Injection SQL | 386 |
| Cross-site Scripting (XSS) | 387 |
| CSRF (Cross-Site Request Forgery) | 390 |
| Les autres attaques | 392 |
| L'authentification | 392 |
| Authentification Basic et Digest | 392 |
| Authentification par formulaire | 393 |
| Les limites de la session HTTP par cookies | 395 |

CHAPITRE 16

La création d'interfaces avec UIBinder 397

| | |
|--|-----|
| Présentation | 398 |
| Styles et ressources | 402 |
| Incorporation d'images | 408 |
| Intégration des ressources de type données | 409 |

| | |
|---|------------|
| Gestionnaires d'événements | 410 |
| Intégration d'un flux HTML standard | 413 |
| Internationalisation | 414 |
| Les emplacements | 416 |
| <i>Cas des balises imbriquées</i> | 418 |
| Traduire les attributs | 419 |
| Liaison avec des beans externes | 420 |
| Modèles composites et constructeurs | 424 |
| | |
| CHAPITRE 17 | |
| Le plug-in Eclipse pour GWT | 429 |
| Le cas AppEngine | 429 |
| Le plug-in GWT | 430 |
| Création d'un projet GWT | 430 |
| Les assistants de création | 433 |
| Création d'un point d'entrée | 433 |
| Création d'un nouveau module | 433 |
| Création d'une page HTML hôte | 434 |
| Création d'un squelette ClientBundle | 435 |
| Création d'un squelette UIBinder | 435 |
| Aide à la saisie de code JSNI | 437 |
| Assistants RPC | 438 |
| | |
| CHAPITRE 18 | |
| Les composants CellWidget | 439 |
| Philosophie de ce nouveau modèle de composants | 439 |
| Utilisation du modèle de données | 443 |
| Le pattern Appearance pour le rendu graphique | 445 |
| Modèle de présentation : SafeHtmlTemplate | 448 |
| Les autres composants CellWidget | 450 |
| DataGrid et CellTable | 451 |
| Chargement des données de manière asynchrone | 453 |
| Mise à jour des données et gestion des événements cellule | 454 |
| Gestion de la sélection | 458 |
| | |
| CHAPITRE 19 | |
| Activités et places | 459 |
| Objectif et philosophie | 459 |
| Les notions | 460 |
| Les emplacements | 460 |

| | |
|---|-----|
| Les activités | 460 |
| Les vues | 461 |
| Le contrôleur d'emplacement (PlaceController) | 461 |
| Le dictionnaire d'activités (ActivityMapper) | 462 |
| Le bus d'événements (EventBus) | 462 |
| Activity and Places par l'exemple | 462 |
| L'application Gestion des courriers | 464 |
| Communication entre vues et bus d'événements | 475 |
| À ne pas mettre entre toutes les mains | 476 |

CHAPITRE 20

L'API Request Factory..... 479

| | |
|--|-----|
| Objectifs | 479 |
| La requête et son contexte côté client | 480 |
| Premiers pas avec l'API | 480 |
| Les entités | 481 |
| Les classes proxies entités | 483 |
| <i>Les proxies de valeur</i> | 484 |
| RequestFactory et l'interface des requêtes | 484 |
| Intégration des services et localisateurs | 485 |
| Utilisation côté client et receveurs | 489 |
| Création et modification d'un objet | 492 |
| Validation et JSR 303 | 492 |
| Les pièges à éviter | 494 |
| L'API AutoBean | 495 |
| Autres fonctionnalités avancées | 497 |

CHAPITRE 21

L'API Editors..... 499

| | |
|---|-----|
| Objectifs et concepts | 499 |
| Modèle de fonctionnement | 500 |
| Les délégués | 503 |
| La gestion des collections d'objets | 505 |
| Gestion de la validation | 509 |

Index..... 511