

Станислав Давыдов  
Алексей Ефимов



# IntelliJ IDEA

Профессиональное программирование на Java

- Описание интерфейса и настроек
- Эффективное создание и модификация кода
- Анализ кода на возможные ошибки
- Удобная работа с XML и компонентами J2EE
- Создание плагинов для IntelliJ IDEA

Наиболее  
полное  
руководство

+CD



В ПОДЛИННИКЕ®

**Станислав Давыдов  
Алексей Ефимов**

# **IntelliJ IDEA**

## **Профессиональное программирование на Java**

Санкт-Петербург

«БХВ-Петербург»

2005

УДК 681.3.068+800java  
ББК 32.973.26-018.2  
Д13

**Давыдов С. В., Ефимов А. А.**

Д13 IntelliJ IDEA. Профессиональное программирование на Java. —  
СПб.: БХВ-Петербург, 2005. — 800 с.: ил.

ISBN 5-94157-607-2

Книга представляет собой исчерпывающее руководство программиста по использованию среды разработки IntelliJ IDEA для создания Java-приложений. Рассмотрены основные особенности работы с IntelliJ IDEA: создание кода на основе информации о классах, редактор с мощными вспомогательными возможностями, встроенная поддержка рефакторинга кода, средства структурного анализа и проверки кода на наличие логических ошибок, потенциально опасных конструкций и возможных способов улучшения, поддержка J2EE-приложений, поддержка XML, визуальный редактор пользовательского интерфейса, встроенная поддержка unit-тестирования с использованием JUnit, поддержка основных систем контроля версий (SourceSafe, CVS, Starteam), полная поддержка Java 5, поддержка плагинов. Особое внимание уделено практическим приемам работы с IntelliJ IDEA. Компакт-диск содержит все необходимое для разработки Java-приложений.

*Для программистов*

УДК 681.3.068+800java  
ББК 32.973.26-018.2

### **Группа подготовки издания:**

Главный редактор	<i>Екатерина Кондукова</i>
Зам. гл. редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Алия Амирова</i>
Компьютерная верстка	<i>Натальи Караваевой</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Игоря Цырульникова</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 22.04.05.

Формат 70×100<sup>1/16</sup>. Печать офсетная. Усл. печ. л. 64,5.

Тираж 3000 экз. Заказ №

"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию  
№ 77.99.02.953.Д.006421.11.04 от 11.11.2004 г. выдано Федеральной службой  
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов  
в ГУП "Типография "Наука"  
199034, Санкт-Петербург, 9 линия, 12

ISBN 5-94157-607-2

© Давыдов С. В., Ефимов А. А., 2005  
© Оформление, издательство "БХВ-Петербург", 2005

# Оглавление

<b>Предисловие</b> .....	<b>15</b>
От авторов .....	15
Для кого эта книга .....	16
Структура книги .....	16
Благодарности .....	18
Замечания и предложения.....	18
<b>ЧАСТЬ I. ОБЩИЕ СВЕДЕНИЯ ОБ INTELLIJ IDEA</b> .....	<b>19</b>
<b>Глава 1. Прошлое и настоящее</b> .....	<b>21</b>
1.1. Идеология работы с файлами .....	22
1.2. Логическая модель проекта.....	23
1.3. Идеология работы с пользователями .....	24
<b>Глава 2. Установка и запуск</b> .....	<b>25</b>
2.1. Регистрация.....	25
2.2. Комплект поставки .....	26
2.2.1. Установка под Windows.....	26
2.2.2. Установка под Linux .....	26
2.2.3. Установка под Mac OS X .....	27
2.2.4. Установка под Generic Unix .....	27
2.2.5. Структура директорий .....	27
2.3. Запуск .....	28
2.4. Директории настроек и временной информации.....	28
2.4.1. Директория с настройками.....	28
2.4.2. Директория с временными файлами .....	29

<b>Глава 3. Интерфейс .....</b>	<b>30</b>
3.1. Организация рабочего пространства приложения .....	30
3.2. Меню.....	32
3.2.1. Меню <i>File (Файл)</i> .....	32
3.2.2. Меню <i>Edit (Редактирование)</i> .....	42
3.2.3. Меню <i>Search (Поиск)</i> .....	50
3.2.4. Меню <i>View (Просмотр)</i> .....	62
3.2.5. Меню <i>Go To (Перейти к)</i> .....	71
3.2.6. Меню <i>Code (Код)</i> .....	75
3.2.7. Меню <i>Analyze (Анализ)</i> .....	86
3.2.8. Меню <i>Refactor (Рефакторинг)</i> .....	87
3.2.9. Меню <i>Build (Сборка)</i> .....	89
3.2.10. Меню <i>Run (Запуск)</i> .....	92
3.2.11. Меню <i>Tools (Инструменты)</i> .....	98
3.2.12. Меню <i>Window (Окно)</i> .....	108
3.2.13. Меню <i>Help (Помощь)</i> .....	110
3.3. Панель инструментов ( <i>Tool Bar</i> ).....	115
3.4. Инструментальные окна ( <i>Tool Window</i> ) .....	116
3.4.1. Режимы отображения .....	117
3.4.2. Состояния окна.....	118
3.4.3. Статусы окна .....	118
3.4.4. Размещение кнопок вызова.....	118
3.4.5. Дополнительные команды управления .....	119
3.4.6. Настройки инструментальных окон .....	119
3.4.7. Поиск в инструментальных окнах .....	119
3.4.8. Базовые инструментальные окна .....	119
3.5. Панель статуса ( <i>Status Bar</i> ).....	141
3.6. Окна редактирования.....	142
3.6.1. Закладки редакторов.....	142
3.6.2. Поля редактора.....	144
3.6.3. Дополнительные визуальные элементы .....	147
3.6.4. Команды редактора.....	147
<b>Глава 4. Настройки.....</b>	<b>153</b>
4.1. <i>IDE Settings (Настройки IDE)</i> .....	155
4.1.1. Диалог настроек <i>General (Общие)</i> .....	156
4.1.2. Диалог настроек <i>Appearance (Внешний вид)</i> .....	158
4.1.3. Диалог настроек <i>Editor (Редактор)</i> .....	161
4.1.4. Диалог настроек <i>Code Completion (Завершение кода)</i> .....	166
4.1.5. Диалог настроек <i>Errors (Ошибки)</i> .....	170
4.1.6. Диалог настроек <i>Colors &amp; Fonts (Цвета и Шрифты)</i> .....	173
4.1.7. Диалог настроек <i>Global Code Style (Глобальные стили кода)</i> .....	179

4.1.8. Диалог настроек <i>JDK &amp; Global Libraries</i> ( <i>JDK и Глобальные библиотеки</i> ) .....	195
4.1.9. Диалог настроек <i>Debugger</i> ( <i>Отладчик</i> ) .....	199
4.1.10. Диалог настройки <i>Resources</i> ( <i>Ресурсы</i> ) .....	202
4.1.11. Диалог настроек <i>File Types</i> ( <i>Типы файлов</i> ) .....	203
4.1.12. Диалог настроек <i>Local History</i> ( <i>Локальная история</i> ) .....	207
4.1.13. <i>Live Templates</i> ( <i>"Живые шаблоны"</i> ) .....	208
4.1.14. Диалог настройки <i>File Templates</i> ( <i>Шаблоны файлов</i> ) .....	221
4.1.15. Диалог настройки <i>Keypmap</i> ( <i>Сочетания клавиш</i> ) .....	228
4.1.16. Диалог настройки <i>External Tools</i> ( <i>Внешние программы</i> ) .....	234
4.1.17. Диалог настройки <i>Path Variables</i> ( <i>Местоположения</i> ) .....	241
4.1.18. Диалог <i>TODO</i> ( <i>Заметки</i> ) .....	242
4.1.19. Диалог настройки <i>Plugins</i> ( <i>Плагины</i> ) .....	247
4.1.20. Диалог настройки <i>Application Servers</i> ( <i>Серверы приложений</i> ) .....	250
4.1.21. Диалог настроек <i>Intention Power Pack</i> ( <i>Дополнительный пакет "подсказок"</i> ) .....	255
4.2. Настройки проекта .....	258
4.2.1. Диалог настройки <i>Paths</i> ( <i>Пути</i> ) .....	258
4.2.2. Диалог настройки <i>Compiler</i> ( <i>Компилятор</i> ) .....	259
4.2.3. Диалог настройки <i>Version Control</i> ( <i>Контроль версий</i> ) .....	263
4.2.4. Диалог настройки <i>Project Code Style</i> ( <i>Стиль кода проекта</i> ) .....	264
4.2.5. GUI Designer .....	266

## **ЧАСТЬ II. СОЗДАНИЕ КОДА.....267**

### **Глава 5. Управление проектами .....269**

5.1. Структура проекта .....	269
5.2. Создание проекта .....	269
5.2.1. Выбор JDK .....	270
5.2.2. Выбор количества модулей .....	271
5.2.3. Выбор типа модуля .....	271
5.3. Редактирование свойств модуля .....	289
5.3.1. Закладка <i>Paths</i> ( <i>Пути</i> ) .....	290
5.3.2. Закладка <i>Libraries</i> ( <i>Библиотеки</i> ) .....	293
5.3.3. Закладка <i>Dependencies</i> ( <i>Зависимости</i> ) .....	296
5.3.4. Закладка <i>Order/Export</i> ( <i>Порядок/Экспорт</i> ) .....	296
5.3.5. Закладка <i>Javadoc</i> .....	298
5.3.6. Закладка <i>Web Module Settings</i> ( <i>Свойства Web-модуля</i> ) .....	299
5.3.7. Закладка <i>J2EE Build Settings</i> ( <i>Настройка сборки J2EE</i> ) .....	301
5.3.8. Закладка <i>EJB Module Settings</i> ( <i>Свойства EJB-модуля</i> ) .....	304
5.3.9. <i>J2EE Application Module Settings</i> ( <i>Свойства модуля J2EE приложения</i> ) .....	305
5.4. Создание проекта из имеющихся файлов .....	307

<b>Глава 6. Работа с кодом.....</b>	<b>311</b>
6.1. Работа с редактором.....	312
6.1.1. Завершение кода (completion) .....	329
6.1.2. Подсветка.....	334
6.1.3. "Подсказки" (intention actions).....	336
6.1.4. Форматирование кода.....	341
6.1.5. Информация о коде.....	342
6.1.6. Буфер обмена.....	343
6.1.7. Сокращенное представление кода .....	343
6.2. Навигация по коду .....	344
6.2.1. Класс.....	344
6.2.2. Файл .....	344
6.2.3. Метод или поле.....	345
6.2.4. Переход к определению, имплементации, определению типа или базовому методу.....	345
6.2.5. Перемещение по коду "вперед-назад" .....	345
6.2.6. Список последних файлов.....	346
6.2.7. Закладки.....	346
6.2.8. Заметки (TODO) .....	346
6.2.9. Иерархия .....	346
6.3. Генерация кода .....	347
6.3.1. Шаблоны файлов.....	347
6.3.2. "Живые шаблоны" .....	347
6.3.3. Создание однотипных классов, интерфейсов и перечислений.....	347
6.3.4. Создание однотипных файлов.....	348
6.3.5. Создание методов <i>get</i> и <i>set</i> .....	348
6.3.6. Автоматическое создание конструкторов.....	348
6.3.7. Создание методов.....	348
6.3.8. Автоматическое создание блоков кода.....	348
6.3.9. Строки импорта .....	349
6.3.10. Комментарии .....	349
6.4. Анализ кода.....	350
6.4.1. Проверка кода (inspections).....	350
6.4.2. Анализ зависимостей.....	386
6.4.3. Анализ дубликатов.....	390
6.5. Структурный поиск.....	392
6.5.1. Редактирование переменных.....	393
6.5.2. Доступные шаблоны.....	395
6.5.3. Результаты поиска.....	396
6.5.4. Структурная замена .....	396
<b>Глава 7. Создание интерфейса пользователя .....</b>	<b>398</b>
7.1. Формы.....	398
7.1.1. Создание формы .....	399

7.1.2. Свойства компонент.....	415
7.1.3. Размещение компонент.....	420
7.2. Создание кода.....	425
7.3. Добавление собственных компонент.....	427
7.4. Дополнительные возможности.....	430
7.4.1. Настройка общих свойств визуального редактора.....	430
7.4.2. <i>Data Binding Wizard (Мастер связей)</i> .....	433
7.4.3. Компиляция форм.....	436
7.4.4. Интернализация (i18n).....	437
<b>Глава 8. Рефакторинг.....</b>	<b>442</b>
8.1. Рефакторинг <i>Rename (Переименовать)</i> .....	443
8.1.1. Переименование директорий.....	443
8.1.2. Переименование файлов.....	444
8.1.3. Переименование пакетов.....	444
8.1.4. Переименование классов.....	446
8.1.5. Переименование методов.....	446
8.1.6. Переименование полей класса.....	447
8.1.7. Переименование параметров вызова метода.....	448
8.1.8. Переименование локальных переменных.....	449
8.2. Рефакторинг <i>Change Signature (Изменить сигнатуру)</i> .....	449
8.3. Рефакторинг <i>Make Method Static (Сделать метод статичным)</i> .....	454
8.3.1. Простой случай.....	454
8.3.2. Сложный случай.....	456
8.4. Рефакторинг <i>Convert To Instance Method (Конвертировать в обычный метод)</i> .....	457
8.5. Рефакторинг <i>Move (Переместить)</i> .....	458
8.5.1. Перемещение директорий.....	459
8.5.2. Перемещение файлов.....	459
8.5.3. Перемещение пакетов.....	460
8.5.4. Перемещение классов.....	462
8.5.5. Перемещение членов класса.....	465
8.6. Рефакторинг <i>Copy (Копировать)</i> .....	467
8.6.1. Копирование директорий.....	467
8.6.2. Копирование файлов.....	468
8.6.3. Копирование классов.....	468
8.7. Рефакторинг <i>Safe Delete (Безопасное удаление)</i> .....	469
8.7.1. Удаление класса.....	469
8.7.2. Удаление метода.....	470
8.7.3. Удаление поля класса.....	470
8.8. Рефакторинг <i>Extract Method (Выделить метод)</i> .....	471
8.9. Рефакторинг <i>Replace Method Code Duplicates (Заменить методом повторяющийся код)</i> .....	474



8.10. Рефакторинг <i>Introduce Variable (Ввести переменную)</i> .....	476
8.11. Рефакторинг <i>Introduce Field (Ввести поле класса)</i> .....	477
8.11.1. Локальная переменная статического метода .....	477
8.11.2. Локальная переменная обычного метода.....	479
8.12. Рефакторинг <i>Introduce Constant (Ввести константу)</i> .....	480
8.13. Рефакторинг <i>Introduce Parameter (Ввести параметр)</i> .....	481
8.14. Рефакторинг <i>Extract Interface (Выделить интерфейс)</i> .....	482
8.15. Рефакторинг <i>Extract Superclass (Выделить базовый класс)</i> .....	485
8.16. Рефакторинг <i>Use Interface Where Possible</i> <i>(Использовать интерфейс, где это возможно)</i> .....	488
8.17. Рефакторинг <i>Pull Members Up (Перемещение членов класса вверх)</i> .....	490
8.18. Рефакторинг <i>Push Members Down (Перемещение членов класса вниз)</i> .....	492
8.19. Рефакторинг <i>Replace Inheritance With Delegation</i> <i>(Заменить наследование делегированием)</i> .....	495
8.20. Рефакторинг <i>Inline (Подстановка в код)</i> .....	497
8.20.1. Подстановка для метода .....	497
8.20.2. Подстановка для статического поля класса.....	499
8.21. Рефакторинг <i>Convert Anonymous to Inner</i> <i>(Конвертировать анонимный класс во внутренний)</i> .....	500
8.22. Рефакторинг <i>Encapsulate Fields (Инкапсулировать поля)</i> .....	502
8.23. Рефакторинг <i>Replace Temp with Query</i> <i>(Заменить временное использование вызовом)</i> .....	505
8.24. Рефакторинг <i>Replace Constructor With Factory Method</i> <i>(Заменить конструктор методом фабрики)</i> .....	507
8.25. Рефакторинг <i>Generify (Генерификация)</i> .....	508
<b>Глава 9. Компиляция, запуск, отладка</b> .....	<b>512</b>
9.1. Компиляция .....	512
9.1.1. Компиляция Java-классов .....	512
9.1.2. Компиляция RMI.....	514
9.1.3. Валидация JSP .....	514
9.2. Типы приложений .....	514
9.2.1. Java-приложения (закладка <i>Application</i> ) .....	516
9.2.2. Апплеты (закладка <i>Applet</i> ) .....	517
9.2.3. Тесты (закладка <i>JUnit</i> ) .....	520
9.2.4. Удаленные приложения (закладка <i>Remote</i> ) .....	524
9.2.5. Web-приложения для Tomcat (закладка <i>Tomcat Server</i> ) .....	526
9.2.6. Приложения для JSR45-совместимых серверов (JSR-45 Compatible Server).....	530
9.2.7. Приложения для сервера WebLogic (закладка <i>WebLogic Instance</i> )....	533
9.3. Запуск .....	538
9.3.1. Java-приложения (закладка <i>Application</i> ) .....	538
9.3.2. Апплеты.....	540

9.3.3. Тесты JUnit .....	544
9.3.4. J2EE-приложения.....	546
9.4. Отладка .....	549
9.4.1. Точки останова.....	550
9.4.2. Пошаговая отладка .....	561
9.4.3. Информация о потоках (закладка <i>Threads</i> ).....	569
9.4.4. Информация о переменных (закладка <i>Frame</i> ).....	571
9.4.5. <i>Watches (Наблюдения)</i> .....	576
9.4.6. Вычисление выражений.....	577
9.4.7. Функция <i>HotSwap</i> .....	579
<b>Глава 10. J2EE .....</b>	<b>580</b>
10.1. Работа с XML.....	580
10.1.1. Редактирование XML .....	580
10.1.2. Поддержка типизации .....	582
10.2. Серверы приложений.....	583
10.3. Работа с EJB .....	583
10.3.1. Создание EJB.....	585
10.3.2. Установка связей между EJB .....	598
10.4. Web-приложения .....	600
10.4.1. Сервлеты .....	603
10.4.2. Фильтры .....	605
10.4.3. JSP.....	606
10.5. Приложения J2EE.....	608
<b>Глава 11. Поддержка новых возможностей Java 5.....</b>	<b>610</b>
11.1. Generics .....	611
11.2. Цикл <i>for</i> для коллекций .....	612
11.3. Auto(un)boxing .....	613
11.4. Перечислимый тип <i>enum</i> .....	614
11.5. Переменное число параметров метода.....	616
11.6. Статические импорты .....	617
11.7. Аннотации .....	618
<b>ЧАСТЬ III. ИНТЕГРАЦИЯ.....</b>	<b>621</b>
<b>Глава 12. Системы контроля версий .....</b>	<b>623</b>
12.1. Утилита Diff.....	623
12.1.1. Сравнение двух файлов .....	623
12.1.2. Сравнение частей файлов .....	624
12.1.3. Представление результатов сравнения .....	624

12.1.4. Дополнительные команды .....	625
12.2. Локальная система контроля версий.....	625
12.2.1. Версии файла.....	626
12.2.2. Версии директорий .....	629
12.2.3. Метки.....	630
12.3. CVS.....	631
12.3.1. Настройки .....	631
12.3.2. Работа с проектом .....	641
12.3.3. Визуальное выделение файлов .....	648
12.3.4. Команды.....	648
12.4. SourceSafe.....	665
12.4.1. Настройки .....	665
12.4.2. Визуальное выделение файлов .....	667
12.4.3. Команды.....	668
12.5. StarTeam.....	672
12.5.1. Настройки .....	672
12.5.2. Визуальное выделение файлов .....	672
12.5.3. Команды.....	672
12.6. Дополнительные команды.....	676
<b>Глава 13. Утилита Ant.....</b>	<b>677</b>
13.1. Создание и подключение файлов сборки.....	677
13.2. Редактирование файлов сборки .....	680
13.2.1. Список доступных команд и параметров.....	680
13.2.2. Вывод дополнительной информации .....	681
13.2.3. Проверка ошибок.....	682
13.2.4. Редактирование зависимостей .....	682
13.2.5. Использование свойств .....	683
13.3. Настройка параметров .....	683
13.3.1. Общие свойства.....	684
13.3.2. Параметры для передачи файлу сборки .....	684
13.3.3. Выполнение .....	684
13.3.4. Дополнительные библиотеки.....	685
13.3.5. Фильтры .....	686
13.4. Привязка целей Ant.....	687
13.5. Запуск целей Ant.....	689
13.6. Генерация файла сборки для проекта.....	690
<b>Глава 14. Тестирование.....</b>	<b>692</b>
14.1. Представление тестов в проекте .....	692
14.2. Запуск тестов.....	693
14.3. Просмотр результатов .....	696

<b>Глава 15. Плагины</b> .....	<b>698</b>
15.1. Общие сведения о плагинах.....	698
15.1.1. Сообщество разработчиков плагинов IDEA.....	698
15.2. Разработка плагинов.....	698
15.2.1. Подготовка IntelliJ IDEA к разработке плагинов.....	699
15.2.2. Общая компонентная модель.....	709
15.2.3. Диагностика ошибок.....	715
15.2.4. Сохранение и восстановление состояния компонент.....	718
15.2.5. Система действий в IntelliJ IDEA.....	723
15.2.6. Стандартные визуальные компоненты IntelliJ IDEA.....	731
15.2.7. Редактор IntelliJ IDEA.....	746
15.2.8. Виртуальная файловая система IntelliJ IDEA (VFS).....	763
15.2.9. Система контроля версий IntelliJ IDEA (VCS).....	766
15.2.10. Структурный программный интерфейс IntelliJ IDEA (PSI).....	768
15.3. Структура директорий плагина.....	771
15.4. Дескриптор плагина.....	772
15.4.1. Требования к дескрипторам.....	772
15.4.2. Общая информация о плагине.....	772
15.4.3. Описания компонент.....	774
15.4.4. Описание действий.....	774
15.5. Заключение.....	777
<b>ЧАСТЬ IV. БЛИЖАЙШЕЕ БУДУЩЕЕ</b> .....	<b>779</b>
<b>Глава 16. IDEA 5</b> .....	<b>781</b>
16.1. Поддержка XML, HTML, CSS.....	781
16.1.1. XML.....	781
16.1.2. HTML.....	782
16.1.3. CSS.....	782
16.2. Поддержка J2EE.....	782
16.3. Работа с проектом.....	783
16.4. Настройки.....	783
16.5. Рефакторинги.....	783
16.6. Системы контроля версий.....	784
16.7. Плагины и OpenAPI.....	784
16.8. Поддержка J2ME.....	785
16.9. Отладка.....	785
16.10. Дополнительные функции редактора.....	785
16.11. Прочие улучшения.....	785
<b>Заключение</b> .....	<b>787</b>
<b>Список ссылок</b> .....	<b>789</b>
<b>Содержимое компакт-диска</b> .....	<b>791</b>
<b>Предметный указатель</b> .....	<b>792</b>

В 1900 году лорд Кельвин, выступая в Лондонском Королевском Обществе в канун Нового года, сказал: "Я удовлетворен современным состоянием науки. Все основные законы открыты, и научный небосвод чист. За исключением двух небольших облачков".

В 2000 году на рынок постепенно выходили Java IDE (Integrated Development Environment — интегрированные среды разработки) больших уважаемых гигантов: Borland, Symantec, Sun. Разработкой IDE заниматься было не модно, все равно, что начинать писать свой Windows — какой смысл, если все уже, в общем-то, сделано и открыто до нас.

Как раз тогда возникла IntelliJ IDEA и за полтора года, по общему признанию, опередила по качеству неповоротливых монстров. Это теперь мы со смехом вспоминаем убогость тогдашних "систем четвертого поколения" и уже точно знаем, что и теперешним их потомкам есть куда развиваться. А словосочетание "интегрированная среда разработки" сейчас чем-то напоминает "ЭВМ": перед глазами встает машинный зал с рядами крутящихся бобин с интеллектом нынешнего калькулятора.

Чем IntelliJ IDEA выделяется среди аналогичных IDE? Прежде всего, как неоднократно замечают пользователи, тем, что она "does not get in the way", то есть не диктует свой способ, свой стиль использования, а, наоборот, помогает программисту решать свои задачи с наименьшими усилиями. Ну и, наконец, спросите у фанатов Apple Computer, почему им приятно брать в руки свои Маки?

— Ну, фиг его знает. Это так стильно!

Я думаю, Стас Давыдов сумел передать в своей книге это ощущение стиля, другой философии работы с IntelliJ IDEA.

Алексей Кудрявцев, Senior Software Engineer, JetBrains, Inc.

# Предисловие

## От авторов

С момента появления IntelliJ IDEA на свет в 2000 году прошло уже достаточно много времени. За это время IntelliJ IDEA превратилась в серьезный продукт, обеспечивший Java-разработчиков практически всем необходимым для комфортной работы. Однако до сих пор об IntelliJ IDEA не было написано ни одной книги. Мы хотим исправить эту, с нашей точки зрения, ошибку.

Одна из причин, побудивших нас написать эту книгу — это то, что IntelliJ IDEA обладает колоссальными возможностями, о которых мало кто знает, даже многие разработчики из самой компании JetBrains. Но мы даже не предполагали, сколько на самом деле этих возможностей, скрытых в глубинах функциональности IntelliJ IDEA!

Книга, которую вы держите в руках, — первая книга в мире, целиком и полностью посвященная IntelliJ IDEA.

IntelliJ IDEA имеет англоязычный интерфейс, поэтому при описании элементов интерфейса, названий и понятий нам пришлось переводить их на русский язык самостоятельно. Мы старались подбирать как можно более точный и понятный перевод терминов, однако в нескольких случаях нам это сделать не удалось. Так, к примеру, остались без перевода английские термины `generics` и `auto(un)boxing`. Также мы полностью оставили без перевода термины `J2EE`, поскольку, как мудро заметил один из наших технических помощников в этой области, в таком переводе можно дойти до абсурда (в качестве примера он привел совершенно дурной перевод термина "Message Driven Bean" — "Боб, погоняемый сообщением", который он видел в одной из книг по `J2EE`).

## Для кого эта книга

Прежде чем начать эту книгу, мы долго думали, для кого мы будем ее писать.

С одной стороны, хотелось рассказать обо всех тонкостях работы с IntelliJ IDEA как можно подробнее, чтобы даже не очень опытные программисты (или незнакомые с IntelliJ IDEA) смогли легко использовать ее в своей повседневной работе и получать от этого максимум удовольствия и результата. Однако в этом случае книга получилась бы очень большой и, возможно, не очень интересной для других программистов — тех, кто либо уже работают с IntelliJ IDEA, либо имеют богатый опыт работы с другими средствами разработки. К тому же у нас ушло бы на ее написание слишком много времени — информация могла бы уже устареть.

С другой стороны, ограничив материал книги лишь разными "продвинутыми" техниками работы с IntelliJ IDEA, мы бы потеряли наших возможных читателей из числа начинающих Java-разработчиков.

Поэтому мы решили пойти на компромисс с собой и сделать следующее: разбить книгу на три части, в первой из которых информация дается максимально подробно, с примерами и пояснениями. Она будет понятна даже начинающим разработчикам. Вторая часть несколько более сложная и требует от читателя знаний в области Java и смежных технологиях, таких, например, как рефакторинг или J2EE. Третья часть носит справочный характер — каждый может взять из нее то, что ему нужно — информацию об интеграции с системами контроля версий, утилитами сборки и тестирования. В последней главе третьей части содержится информация о системе плагинов IntelliJ IDEA. Эта глава будет интересна Java-разработчикам, создающим модули расширения для IntelliJ IDEA.

## Структура книги

Информация об IntelliJ IDEA структурирована в виде шестнадцати глав, разделенных на четыре части.

Первая часть включает главы с первой по четвертую.

*Глава 1* содержит общую информацию об IntelliJ IDEA, так сказать для общего развития ☺.

*Глава 2* повествует об установке IntelliJ IDEA, а также о различных особенностях, сопровождающих этот процесс на различных операционных системах.

*Глава 3* содержит подробное описание интерфейса IntelliJ IDEA, включая описание системы меню, панели инструментов, команд, инструментальных окон и окон редактора. Информация в данной главе организована таким образом, чтобы ее можно было легко использовать в качестве справочного материала при работе с IntelliJ IDEA.

*Глава 4* содержит подробные описания всевозможных настроек IntelliJ IDEA (коих не так уж и мало). Данную главу также можно использовать в качестве справочника.

Вторая часть содержит главы с пятой по одиннадцатую.

*Глава 5* рассказывает об управлении программными проектами в рамках IntelliJ IDEA. Из этой главы вы узнаете о структуре проекта IntelliJ IDEA и его составных частях — модулях.

*Глава 6* содержит информацию, полезную с практической стороны: здесь описана работа с редактором (включая основные моменты работы с редактором при написании кода), навигация, генерация и анализ кода. В этой главе также рассказано о функции структурного поиска и замены.

*Глава 7* повествует о встроенном визуальном редакторе IntelliJ IDEA — средстве для создания интерфейса пользователя с помощью технологии Swing.

*Глава 8* содержит справочную информацию по функциям IntelliJ IDEA, связанным с рефакторингом кода.

*Глава 9* рассказывает о компиляции и сборке проектов, а также о запуске и отладке различных типов Java-приложений (в том числе J2EE-приложений).

*Глава 10* содержит справочную информацию о поддержке J2EE в IntelliJ IDEA. В главе рассмотрены вопросы создания и настройки компонентов EJB, Web- и J2EE-приложений.

*Глава 11* повествует о поддержке новых возможностей языка Java версии 5 в IntelliJ IDEA, таких как параметризованные классы (generics), классы-перечисления (enum) и т. д.

Третья часть книги содержит главы с двенадцатой по пятнадцатую.

*Глава 12* рассказывает о взаимодействии IntelliJ IDEA с системами контроля версий: встроенной системе, CVS, SourceSafe и StarTeam.

*Глава 13* содержит информацию о поддержке утилиты сборки Java проектов Ant.

*Глава 14* повествует о поддержке в IntelliJ IDEA unit-тестирования Java-кода с помощью утилиты JUnit.

*Глава 15* содержит информацию о системе плагинов IntelliJ IDEA — специальных подключаемых модулях расширения функциональности IntelliJ IDEA.

Последняя, четвертая часть содержит *главу 16*, рассказывающую о новых возможностях, которые будут доступны в следующей (пятой) версии IntelliJ IDEA, которая должна выйти в начале лета 2005 года.

В конце книги приведен алфавитный указатель.



## Благодарности

Мы хотим выразить нашу признательность людям и компаниям, благодаря которым эта книга смогла увидеть свет.

Во-первых, мы бы хотели поблагодарить компанию JetBrains, Inc. и лично Евгения Беляева, Сергея Дмитриева и Валентина Кипяткова — создателей этого замечательного программного продукта IntelliJ IDEA.

Во-вторых, мы хотим поблагодарить издательство "БХВ-Петербург" за то, что оно поверило в наш проект и претворило его в жизнь, издав эту книгу, а также лично наших редакторов: Игоря Шишигина и Алию Амирову.

В-третьих, мы хотим поблагодарить всех людей, которые помогли нам при написании этой книги: Сергея Баранова и Алексея Кудрявцева из компании JetBrains за ценные советы и ответы на наши вопросы, возникавшие по ходу работы над книгой; Антона Катилина из компании YourKit LLC за ценные замечания по поводу описания визуального редактора IntelliJ IDEA; активных читателей "Живого Журнала", посвященного созданию этой книги, за ценные советы: insolite, nettoyeur и др.

Кроме этих людей, помогавших нам в создании этой книги, у каждого из нас найдется, кого поблагодарить лично.

Станислав Давыдов: мне хотелось бы отдельно поблагодарить Гоблина и Гнума, которые стойко терпели мой стук по клавишам в течение долгой работы над книгой.

Алексей Ефимов: я бы хотел поблагодарить Юльку за терпение и понимание, Дмитрия Кашина, создателя плагина Properties Editor, который помогал в разрешении некоторых вопросов и давал ценные замечания, и Гоблинов, без которых вообще ничего не получилось бы.

Извините, если мы кого-то забыли упомянуть здесь. Сообщите нам об этом, чтобы в следующем издании мы исправили эту ошибку.

Станислав Давыдов ([davidovsv@yandex.ru](mailto:davidovsv@yandex.ru))

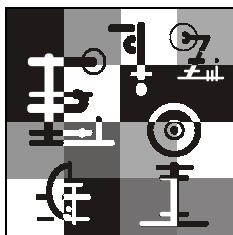
Алексей Ефимов ([aefimov-box@yandex.ru](mailto:aefimov-box@yandex.ru))

Санкт-Петербург — Москва, 2005 год.

## Замечания и предложения

Если у вас возникнут любые замечания или предложения по поводу материалов, представленных в этой книге, мы будем рады получить от вас эту информацию. Также мы будем признательны за любые дополнения или информацию об ошибках и неточностях.

Вы можете обращаться к нам лично по электронной почте, либо использовать сообщество в "Живом Журнале", посвященное этой книге (<http://www.livejournal.com/community/ideabook>).



## **Часть I**

# **Общие сведения об IntelliJ IDEA**

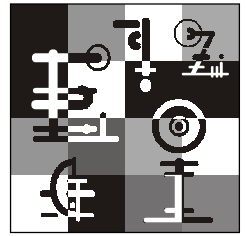
**Глава 1. Прошлое и настоящее**

**Глава 2. Установка и запуск**

**Глава 3. Интерфейс**

**Глава 4. Настройки**

# Глава 1



## Прошлое и настоящее

IntelliJ IDEA — очень молодое и динамично развивающееся Java IDE (Integrated Development Environment — интегрированная среда разработки). Молодость в данном случае выступает в роли достоинства — программа еще не успела обрасти пережитками, тянущимися со старых версий. Динамичность же обеспечивается новейшими методиками разработки, которые использует команда компании JetBrains, Inc., разрабатывающая IntelliJ IDEA.

По сравнению с известными мировыми брендами в этой области с точки зрения рядовых Java-разработчиков IntelliJ IDEA выглядит чрезвычайно привлекательно, поскольку старается воплотить все новые веяния в процессе создания программного обеспечения на Java. Кроме того, она, ориентируясь больше на ручное написание (в отличие от визуальных систем), позволяет писать код максимально быстро, удобно и без ошибок. Даже если от IntelliJ IDEA оставить один лишь редактор, она все равно останется чрезвычайно полезной.

По заявлению компании, *IntelliJ IDEA создается Java-разработчиками для Java-разработчиков*. Стоит лишь несколько часов пописать код в IntelliJ IDEA, как становится понятно, что это действительно так. Если продолжить писать код в IntelliJ IDEA около недели, окажется, что все остальные среды разработки выглядят громоздкими и неудобными. Авторы сами убедились в этом в свое время.

Кроме богатых возможностей редактирования кода, в IDEA есть много других возможностей. Из числа наиболее полезных вещей можно назвать встроенную поддержку рефакторинга, гибкие возможности по запуску и отладке приложений, включая возможности по загрузке измененного кода "на лету" — HotSwap, отладку J2EE-приложений и многое другое.

Долгое время в IDEA не было вообще никаких визуальных средств, однако, начиная с четвертой версии появился встроенный визуальный редактор форм (использующий Swing).

## 1.1. Идеология работы с файлами

В IntelliJ IDEA используется очень дружелюбная к пользователю система работы с файлами. Если вы обратите внимание на меню IntelliJ IDEA, там не обнаружится многочисленных команд сохранения файлов. Одна-единственная команда **Save All** (Сохранить все) (*см. раздел 3.2.1.13*) скорее дань психологическому спокойствию разработчика, нежели реальная необходимость.

Отсутствие команд сохранения неслучайно.

За историю существования программ, предназначенных для изменения каких-либо данных, разработчики этих программ приучили нас сохранять каждые несколько минут. Иначе, малейший сбой в программе или компьютере — и результат целого дня работы может быть потерян. Данный подход к организации сохранения изменений, произошедших в результате работы программы, чрезвычайно неудобен.

Другой подход — сохранять все и всегда — впервые в массовом масштабе был реализован в портативных устройствах PDA (Personal Data Assistant — персональный помощник) или КПК (карманный персональный компьютер). Суть его весьма проста и исходит из архитектуры КПК, в которых память, используемая для размещения программ и данных, также является и оперативной памятью. То есть если на КПК некоторый файл загружен в редактор и там изменяется, то он также изменяется и в месте своего постоянного хранения. Это достаточно грубое описание, однако для обычного пользователя все выглядит именно так.

При таком подходе пользователю просто не нужно беспокоиться, сохранил он файл или нет.

В IntelliJ IDEA использован именно такой подход — избавление программиста от лишних действий по сохранению файлов, т. к. IntelliJ IDEA все файлы сохраняет сама. Делать она это может по трем событиям:

- ❑ потеря фокуса окна IntelliJ IDEA;
- ❑ периодически, в случае, если приложение неактивно (отсутствуют события с клавиатуры или мыши);
- ❑ по команде пользователя **File | Save All** (Файл | Сохранить все) или при нажатии клавиш `<Ctrl>+<S>`.

А чтобы не беспокоиться, не затерли ли мы какой-нибудь важный файл, IDEA ведет журнал всех изменений. Эта система называется LVCS (Local version control system или просто Local History) (*см. раздел 12.2*). Она позволяет просматривать изменения в файлах, возвращать к жизни давно забытые идеи и легко отвечать на вопросы типа: "Покажи все изменения в моем классе, произошедшие со времени последнего успешного запуска тестов".

## 1.2. Логическая модель проекта

Основой модели разработки программ в IntelliJ IDEA является *проект* (Project). Проект — это некоторый набор файлов (классов, ресурсов, JSP-страниц и пр.) и настроек, актуальных для всех этих файлов. Все файлы в проекте объединены в *модули*. Модуль — это структурная единица проекта, которая позволяет разделять файлы проекта некоторым логическим образом, а также определить для них некоторые специфичные настройки, отличные от общих настроек проекта.

Проще всего назначение модулей можно понять из следующего примера. Допустим, мы пишем некоторое Web-приложение, которое состоит из JSP-файлов, а также нескольких сервлетов, предназначенных для авторизации. Кроме самих классов с сервлетами, имеется много вспомогательных классов, необходимых для функционирования самих сервлетов. Будет логично, если в нашем проекте мы выделим два модуля:

1. Модуль с сервлетами и сопутствующими классами (в IntelliJ IDEA такой модуль называется Java Module).
2. Модуль с JSP-файлами (в IntelliJ IDEA такой модуль называется Web-модуль).

Почему было выбрано именно такое разделение? Во-первых, наши классы сервлетов работают с собственными библиотеками (т. е. должны компилироваться вместе с ними), во-вторых, мы хотим, чтобы после сборки приложения все классы были объединены в некоторую отдельную библиотеку (JAR-файл), которая просто размещается в специфическом для Web-приложения месте (папка /WEB-INF/lib). В-третьих, все остальные составляющие нашего Web-приложения — JSP-файлы, не имеют прямой связи с классами из этой библиотеки, однако работают совместно с ними.

Такое разделение — не единственно возможное. В принципе, можно было просто "свалить" все файлы в один модуль, однако тогда пострадала бы логическая модель представления нашего приложения. Чем проще и нагляднее представлен наш проект, тем меньше вероятность допустить в нем какие-то ошибки, связанные с его структурной организацией. При грамотном разделении проекта на модули IntelliJ IDEA сделает некоторую работу за вас, например, проверит целостность ссылок между различными классами проекта, а также позволит еще до этапа компиляции обнаружить некоторые ошибки (если они, конечно, есть 😊).

Количество модулей в проекте не ограничено. Чем сложнее приложение, тем больше в нем модулей. Например, в проекте самой IntelliJ IDEA на момент написания этой главы содержится порядка 50 модулей.

## 1.3. Идеология работы с пользователями

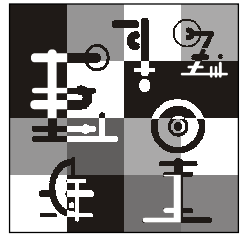
Каким бы ни был замечательным программный продукт, в нем всегда найдутся ошибки. Каждый, кто работает с любым программным продуктом, рано или поздно замечает, что одни вещи можно было бы сделать более удобными, другие — более быстрыми, третьи вообще неплохо было бы просто иметь в продукте.

IntelliJ IDEA — коммерческий программный продукт, созданный специально для Java-разработчиков. Основная цель его создания — сделать работу Java-программистов максимально удобной, качественной и быстрой. Компания JetBrains, Inc. — создатель IntelliJ IDEA — прикладывает к этому все силы.

Разработка самой IntelliJ IDEA построена по очень интересному принципу. Во-первых, каждый пользователь IntelliJ IDEA может попросить разработчиков включить в IntelliJ IDEA ту функциональность, которая ему нужна. Для таких запросов используется система онлайн учета запросов необходимых изменений (Change Request). Все пожелания пользователей учитываются, систематизируются и самые важные (те, о которых попросило максимальное количество пользователей) включаются в следующую версию IntelliJ IDEA. То есть фактическим заказчиком разработки программного продукта в данном случае может выступать каждый из нас — самый обычный Java-разработчик.

Все ошибки, которые возникают у разработчиков при работе IntelliJ IDEA также собираются и учитываются в специальной онлайн-системе (Bug Tracker). Ошибки оперативно исправляются и периодически выпускаются обновления той или иной версии продукта.

Благодаря тесной работе с пользователями IntelliJ IDEA, продукт получается удобным и качественным. Да, сама IntelliJ IDEA разрабатывается с помощью IntelliJ IDEA — это тоже немаловажный момент.



## Глава 2

# Установка и запуск

В данной главе будет рассказано о том, откуда можно скачать IntelliJ IDEA, как ее установить и запустить.

## 2.1. Регистрация

IntelliJ IDEA является коммерческим программным продуктом (стоимость однопользовательской коммерческой лицензии составляет порядка \$500, а некоммерческой — для образовательных учреждений — \$100). Тем не менее компания JetBrains, производящая IntelliJ IDEA, предоставляет возможность любому желающему бесплатно попробовать поработать в IntelliJ IDEA в течение 30 дней. Для этого нужно зарегистрироваться на сайте компании (<http://www.jetbrains.com>) и получить на указанный электронный адрес 30-дневный лицензионный ключ, который активирует пробную лицензию (evaluation license). Эта лицензия ограничивает лишь срок использования продукта, но никак не ограничивает функциональность.

Подробнее о регистрации лицензии IntelliJ IDEA можно прочитать в *разделе 3.2.13.5*.

Кроме платной версии IntelliJ IDEA, имеется еще и так называемая предварительная версия, которую можно получить, участвуя в "Программе предварительного доступа" (Early Access Program — EAP). Данная программа дает возможность любому желающему использовать версию IntelliJ IDEA, которая только разрабатывается.

Например, на момент написания данной книги в качестве такой предварительной версии для EAP предоставляется IntelliJ IDEA 5.0 ("Irida" code name).

EAP-версия может использоваться на всем протяжении разработки очередной версии IntelliJ IDEA (т. е. не менее полугода) без ограничений. Конечно, данная версия содержит значительно больше ошибок по сравнению со стабильной коммерческой версией. Тем не менее EAP-версия также

содержит и функции, запланированные на новую версию IntelliJ IDEA, и поэтому пользуется немалой популярностью.

EAP-версию IntelliJ IDEA можно скачать с сайта IntelliJ Technology Network (ITN — <http://www.intellij.net>), для чего необходимо предварительно зарегистрироваться на этом сайте. По сути, EAP делает каждого пользователя новой версии добровольным бета-тестером продукта в обмен на возможность его бесплатного использования.

Каждый участник EAP получает в свое распоряжение лицензию на использование IntelliJ IDEA сроком на один месяц, которая продлевается каждый месяц (т. е. пользователю необходимо заходить на сайт ITN минимум раз в месяц).

## 2.2. Комплект поставки

IntelliJ IDEA — приложение, целиком и полностью написанное на Java, соответственно, оно может работать на любой платформе, на которой имеется JRE (Java Runtime Environment). Минимальная версия JRE, которая требуется для работы IntelliJ IDEA — JRE 1.4.2\_04. Для компиляции Java-классов также требуется наличие JDK (Java Development Kit).

Для операционных систем семейства Windows и Linux IntelliJ IDEA поставляется в виде самораспаковывающегося дистрибутива, содержащего программу-инсталлятор.

В комплекте к некоторым дистрибутивам, кроме основного приложения — IntelliJ IDEA, прилагается и необходимое Java окружение (JRE) для запуска IntelliJ IDEA.

### 2.2.1. Установка под Windows

Для операционной системы Windows дистрибутив IntelliJ IDEA занимает порядка 56 Мбайт. Дистрибутив представляет собой одиночный EXE-файл с инсталлятором и IntelliJ IDEA (например, `idea-4.5.3.exe`).

Процесс инсталляции вполне стандартен. Если на вашем компьютере установлены предыдущие версии IntelliJ IDEA, указав инсталлятору соответствующую опцию, можно импортировать настройки от старой версии.

В комплекте с дистрибутивом поставляется JRE версии 1.4.2\_04 (для IntelliJ IDEA версии 4.5.3).

### 2.2.2. Установка под Linux

Для операционной системы Linux существует два вида дистрибутива IntelliJ IDEA:

- архив с директорией установки IntelliJ IDEA;
- дистрибутив, поставляемый в виде одиночного самораспаковывающегося файла (например, `idea-4.5.3.bin`).



### 2.2.2.1. Дистрибутив tar.gz

Первый вариант прост в установке — достаточно лишь выполнить команду разархивации `tar -xvzf idea-4.5.3.tar.gz` в требуемом каталоге.

В комплекте с данным дистрибутивом JRE не поставляется, поскольку этот дистрибутив также используется для установки IntelliJ IDEA на прочих Unix-совместимых платформах (см. раздел 2.2.4). Поэтому JRE необходимо установить отдельно.

JRE следует установить в директорию `je`, расположенную внутри корневой директории с установленной IntelliJ IDEA.

Данный дистрибутив занимает около 40 Мбайт.

### 2.2.2.2. Самораспаковывающийся дистрибутив

Данный дистрибутив содержит кроме самой IntelliJ IDEA программу-инсталлятор, производящую все необходимые действия по установке, включая импорт настроек из старых версий IntelliJ IDEA и создание ссылок на программу запуска IntelliJ IDEA.

В составе данного дистрибутива имеется JRE версии 1.4.2\_04.

Размер данного дистрибутива составляет около 59 Мбайт.

## 2.2.3. Установка под Mac OS X

Для операционной системы Mac OS X дистрибутив IntelliJ IDEA представляет собой файл типа Disk Image (dmg), который может быть установлен в систему стандартным образом.

Для установки под Mac OS X необходимо дважды щелкнуть левой кнопкой мыши на дистрибутиве, после чего выполнится стандартная операция Disk Copy, которая смонтирует диск с IntelliJ Labs.

Размер дистрибутива составляет порядка 34 Мбайт.

## 2.2.4. Установка под Generic Unix

Если некоторая операционная система Unix имеет поддержку Java и JDK версии не ниже, чем 1.4.2\_04, на данную систему можно установить IntelliJ IDEA из дистрибутива tar.gz, как было описано в разделе 2.2.1.

Нам доподлинно известно, что IntelliJ IDEA успешно запускали на операционных системах Sun Solaris и FreeBSD.

## 2.2.5. Структура директорий

После установки IntelliJ IDEA создается следующая структура директорий:

□ `bin/` — файлы, необходимые для запуска IntelliJ IDEA;

- help/ — файлы справочной системы;
- jre/ — JRE;
- lib/ — файлы библиотек, необходимых для работы IntelliJ IDEA;
- license/ — файлы с лицензиями IntelliJ IDEA и библиотек;
- plugins/ — модули-расширения IntelliJ IDEA;
- redist/ — библиотеки, необходимые для сборки визуальных форм IntelliJ IDEA сторонними средствами;
- build.txt — информация о порядковом номере сборки IntelliJ IDEA;
- changes.txt — информация о внесенных изменениях;
- features.txt — информация о возможностях IntelliJ IDEA;
- knownissues.txt — информация об известных ошибках и проблемах;
- readme.txt — краткая информация.

## 2.3. Запуск

В зависимости от операционной системы запуск IntelliJ IDEA производится:

- на Windows с помощью файла `idea.exe` или `idea.bat`, расположенного в каталоге `bin`;
- на Linux и прочих Unix-системах с помощью файла `idea.sh` в каталоге `bin`;
- на Mac OS X — после того как образ диска смонтирован в системе (см. раздел 2.2.3), IntelliJ IDEA можно запускать прямо с этого смонтированного диска.

## 2.4. Директории настроек и временной информации

IntelliJ IDEA имеет две дополнительные директории, которые обычно создаются в пользовательской директории:

- директория с индивидуальными пользовательскими настройками IntelliJ IDEA — `USER_HOME/.IntelliJIDEA/config`;
- директория с временными файлами, используемыми для работы IntelliJ IDEA — `USER_HOME/.IntelliJIDEA/system`.

### 2.4.1. Директория с настройками

Пользовательские настройки хранятся в формате XML и, в принципе, могут быть модифицированы в любом текстовом редакторе, однако мы бы не советовали этого делать — лучше доверить это IntelliJ IDEA. Также в директории

настроек находится файл с лицензионным ключом к IntelliJ IDEA. Для IntelliJ IDEA версий 4.0 и выше этот файл называется `idea40.key`.

Кроме настроек, в поддиректории `config/plugins` хранятся плагины IntelliJ IDEA, установленные пользователем.

## 2.4.2. Директория с временными файлами

В директории с временными файлами содержатся различные кэшированные данные, которыми приложение IntelliJ IDEA управляет самостоятельно. Объем, занимаемый этими файлами, может достигать весьма внушительных размеров. Так, например, при работе над проектом с примерами к этой книге (достаточно небольшой проект) временные файлы заняли порядка 256 Мбайт. При работе над большими проектами временные файлы могут занимать значительно больше памяти.

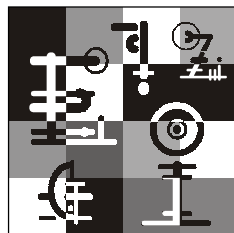
Кроме кэшированных данных среди временных файлов содержится также репозиторий локальной системы контроля версий, встроенной в IntelliJ IDEA (см. раздел 12.2).

Мы бы не советовали удалять временные файлы без лишней необходимости. Во-первых, они существенно ускоряют работу IntelliJ IDEA, а во-вторых, даже если эти файлы удалить, IntelliJ IDEA создаст их вновь после следующего старта и открытия проекта.

Среди временных файлов в директории `system/log` хранятся также журнальные файлы работы IntelliJ IDEA, которые могут быть использованы для диагностирования неисправностей и ошибок в работе программы.

Обычно, в случае серьезной ошибки служба технической поддержки IntelliJ IDEA просит прислать эти файлы.

## Глава 3



# Интерфейс

Пользовательский интерфейс IntelliJ IDEA представляет собой стандартную форму MDI<sup>1</sup>-интерфейса, в котором общее окно соответствует одному проекту.

## 3.1. Организация рабочего пространства приложения

Общий вид окна проекта в IntelliJ IDEA представлен на рис. 3.1.

Кроме стандартного для большинства приложений меню, панели инструментов (ToolBar), панели статуса (StatusBar) и рабочей области, в IntelliJ IDEA есть интересное интерфейсное дополнение — *инструментальные окна* (ToolWindow — по аналогии с панелью инструментов — ToolBar). Инструментальные окна имеют кнопки вызова, расположенные по периметру рабочей области, с пиктограммой, подписью и числовым обозначением (последнее не обязательно). Если на эти кнопки нажать, рядом с ними откроются окошки с некоторой вспомогательной функциональностью (рис. 3.2). Подробнее об инструментальных окнах написано в *разделе 3.4*.

Обзор интерфейса IntelliJ IDEA мы начнем с меню, как с наиболее часто употребляемого списка команд, доступных в IntelliJ IDEA.

---

<sup>1</sup> MDI — Multi Document Interface — многодокументный интерфейс.

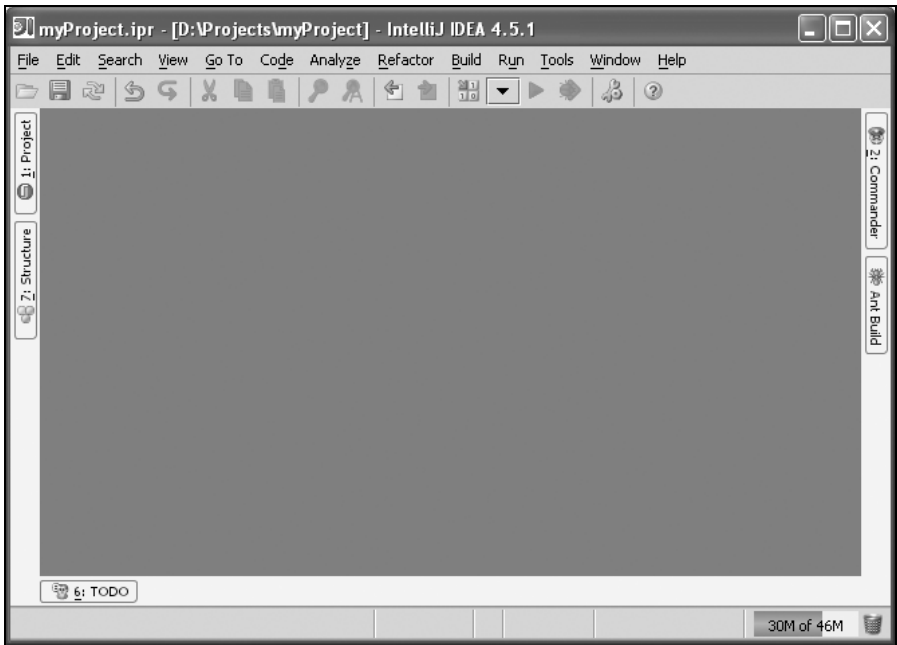


Рис. 3.1. Общий вид окна проекта в IntelliJ IDEA

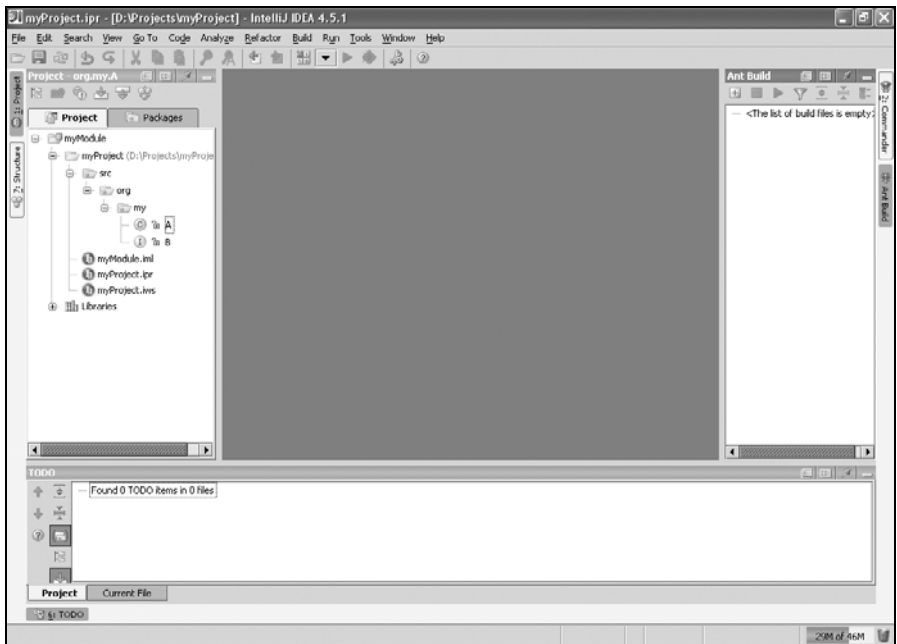


Рис. 3.2. Инструментальные окна

## 3.2. Меню

Главное меню IntelliJ IDEA состоит из следующих пунктов:

- ❑ **File** (Файл) — команды для работы с проектами, модулями, файлами;
- ❑ **Edit** (Редактирование) — стандартные команды для редактирования;
- ❑ **Search** (Поиск) — различные типы поиска и замены;
- ❑ **View** (Просмотр) — команды для отображения различных элементов интерфейса;
- ❑ **Go To** (Перейти) — команды навигации по проекту;
- ❑ **Code** (Код) — вспомогательные команды для написания кода;
- ❑ **Analyze** (Анализ) — команды для анализа кода;
- ❑ **Refactor** (Рефакторинг) — набор команд для проведения рефакторинга;
- ❑ **Build** (Сборка) — команды для компиляции и сборки проекта;
- ❑ **Run** (Запуск) — запуск и отладка приложений;
- ❑ **Tools** (Утилиты) — набор различных вспомогательных утилит;
- ❑ **Window** (Окно) — работа с окнами в IntelliJ IDEA;
- ❑ **Help** (Помощь) — контекстная помощь, справка о программе.

Пункты меню могут быть связаны с кнопками на панели инструментов, а также иметь *"горячие" клавиши* (hot keys) для быстрого вызова команд. В IntelliJ IDEA любой команде в меню можно назначить *"горячую"* клавишу. Для некоторых наиболее часто используемых команд *"горячие"* клавиши заданы по умолчанию. В описании таких команд мы будем указывать соответствующие им *"горячие"* клавиши в тексте сразу после названия.

Далее мы рассмотрим все приведенные пункты меню. Некоторые команды, присутствующие в меню, имеют весьма сложную функциональность, например, команды структурного поиска или рефакторинга. Таким командам посвящены отдельные главы или разделы, а в этой главе мы лишь приведем их краткое описание.

### 3.2.1. Меню *File* (Файл)

В меню **File** (Файл) представлены команды, с помощью которых можно выполнить некоторые действия с файлами — открыть, закрыть, вывести на печать и пр.

#### 3.2.1.1. *New Project* (Новый проект)

Команда создания нового проекта (запускается мастер создания нового проекта, см. раздел 5.2).