

Компьютеры и Технологии

Среда разработки Eclipse 4

Руководство разработчика

Основные возможности разработки
приложений с Eclipse 4

SWT и JFace

Создание Eclipse плагинов и RCP
приложений

Разработка приложений на основе
RAP, GWT, Riena, SCA, Scout, WTP,
DTP, BIRT и др.

Машнин Т.С.

Содержание

Введение. Краткий обзор проектов Eclipse. Отличие платформы Eclipse 4 от платформы Eclipse 3. Области применения Eclipse.....	7
Проекты Eclipse.....	9
Проект Business Intelligence and Reporting Tools (BIRT).....	9
Проект Eclipse Data Tools Platform (DTP).....	9
Проект Eclipse.....	9
Проект Eclipse Modeling.....	11
Проект Mylyn.....	12
Проект RT.....	13
Проект SOA Platform.....	14
Проект Technology.....	16
Проект Tools.....	19
Проект Eclipse Web Tools Platform.....	20
Некоторые новые возможности Eclipse 4 по сравнению с Eclipse 3.....	21
Области применения Eclipse.....	22
Для разработчиков программного обеспечения.....	22
Для пользователей программного обеспечения.....	23
Глава 1. Платформа Eclipse и средства разработки Java. Архитектура Eclipse. Начало работы с Eclipse. Разработка Java SE приложений.....	24
Архитектура платформы Eclipse и среда Eclipse SDK.....	24
Страница Welcome.....	26
Рабочая область Workbench.....	28
CSS и темы Eclipse.....	41
Разработка приложений платформы Java SE.....	42
Среда разработки Eclipse Standard.....	42
Среда разработки Eclipse IDE for Java Developers.....	60
Инструменты Mylyn.....	60
Интеграция с Maven.....	66
Средства работы с XML.....	72
Code Recommenders.....	80
Глава 2. Отладка, тестирование и рефакторинг кода. Автоматическое тестирование UI-интерфейса. Автоматическое тестирование Web-приложений.....	83
Отладка Java-кода.....	84
Тестирование Java-кода.....	97
Рефакторинг.....	105
Автоматическое тестирование UI-интерфейса.....	107
SWTBot.....	107
Jubula.....	112
Jubula-тестирование E4 RCP приложения.....	112
Автоматическое тестирование Web-приложений с Selenium 2.....	119
Selenium IDE.....	120
WebDriver.....	123
Глава 3. Командная разработка кода. Использование CVS, SVN, Git, Mercurial.....	128

CVS.....	129
Subversion.....	144
Плагин Subclipse.....	145
Локальный SVN-репозиторий.....	155
Плагин Subversive.....	157
Git.....	163
Mercurial.....	177
Глава 4. Интернационализация и локализация приложений.....	183
Глава 5. Графические системы SWT и JFace. Разработка SWT/JFace и XWT приложений.....	190
SWT-приложения.....	196
Связывание данных.....	218
JFace-приложения.....	223
XWT-приложения.....	226
Стилизация SWT-компонентов с помощью CSS.....	227
Глава 6. Разработка Eclipse-плагинов и OSGi-модулей. Пример: Создание Eclipse-плагина Rich Text Editor. Совместное использование SWT и JavaFX. Размещение плагина в Eclipse Marketplace. Полезные Eclipse-плагины. Пример: Создание плагина, добавляющего меню запуска внешней программы.....	230
Мастер Plug-in Project.....	232
Создание Eclipse-плагина.....	232
Создание OSGi-модуля.....	245
Мастер Fragment Project.....	249
Мастер Feature Project.....	250
Мастер Plug-in from Existing JAR Archives.....	253
Пример. Создание Eclipse-плагина Rich Text Editor. Совместное использование SWT и JavaFX.....	253
Размещение плагина в Eclipse Marketplace.....	262
Некоторые полезные Eclipse-плагины.....	263
Увеличение и уменьшение шрифта в редакторе.....	263
Перенос слов в редакторе.....	263
Декомпиляция Java кода.....	264
Создание Javadoc-документации.....	264
Визуализация Java-кода.....	264
Проверка орфографии.....	267
Просмотр PDF документов.....	267
Создание и редактирование PNG изображений.....	268
FTP клиент.....	269
Пример. Создание плагина, добавляющего меню запуска внешней программы.....	271
Глава 7. Создание RCP-приложений платформ Eclipse 3 и Eclipse 4. Пример: Создание E4 RCP приложения Rich Text Editor. Совместное использование SWT и JavaFX.....	275
Создание RCP-приложения, совместимого с Eclipse 3.....	276
Основные отличия платформы Eclipse 4 от платформы Eclipse 3.....	288
Создание Eclipse 4 RCP-приложения.....	288

Модель приложения.....	292
Добавление Addon-компонента.....	298
Создание Handler-обработчика и определение команд.....	302
Создание Window-окна.....	307
Создание Part-части.....	309
Переключение между перспективами.....	309
PartSashContainer и Part Stack.....	310
Part.....	310
Binding Contexts.....	314
Persisted State.....	315
Context Properties.....	316
Переменные контекста.....	317
Тэги элементов модели.....	318
Создание Part-части динамически.....	318
Тэги модели приложения.....	320
Стилизация E4-приложения с помощью CSS.....	321
Создание меню и панели инструментов.....	327
Mnemonics.....	328
DynamicMenuContribution.....	328
VisibleWhen Core Expression.....	330
View Menu.....	332
Popup Menu.....	332
Toolbar.....	333
Горячие клавиши UI-интерфейса.....	334
Модульность модели приложения.....	335
Фрагменты.....	335
Процессоры.....	339
Динамическое изменение модели приложения.....	340
Расширение модели приложения.....	341
Dependency Injection (DI) и Eclipse-контекст.....	347
Использование @Named и IServiceConstants.....	350
Использование аннотаций жизненного цикла.....	352
Взаимодействие между компонентами E4-приложения.....	353
Использование Snippets.....	355
Использование общих элементов.....	356
Использование элементов Contributions.....	357
Пример. Создание E4 RCP приложения Rich Text Editor. Совместное использование SWT и JavaFX.....	358
Глава 8. Создание Android-приложений. Отображение контента Android приложением. Отображение электронных книг в формате ePub и PDF.....	366
Установка ADT-плагина.....	367
Описание ADT-плагина.....	371
Перспектива DDMS.....	374
Перспективы Hierarchy View и Pixel Perfect.....	383
Wizard-мастера ADT-плагина.....	386

Мастер Android Project.....	386
Запуск Android-приложения из среды Eclipse.....	391
Подготовка к публикации Android-приложения.....	398
Activity-компонент.....	399
Layout-редактор ADT-плагина.....	403
Редактор файла AndroidManifest.xml ADT-плагина.....	410
Мастер Android XML File.....	424
Тип ресурса Layout.....	425
Тип ресурса Values.....	426
Тип ресурса Drawable.....	429
Тип ресурса Menu.....	433
Тип ресурса Color List.....	436
Тип ресурса Property Animation и Tween Animation.....	438
Тип ресурса AppWidgetProvider.....	443
Тип ресурса Preference.....	446
Тип ресурса Searchable.....	451
Мастер Android Icon Set.....	456
Мастер Android Test Project.....	457
Отображение контента Android приложением.....	459
Галерея изображений.....	459
ImageView.....	459
ViewPager.....	468
ImageView + Zoom + Scroll.....	479
Галерея HTML контента.....	479
WebView + Pagination.....	489
Отображение электронных книг в формате ePub.....	491
Отображение PDF.....	492
Глава 9. Создание RAP-приложений.....	495
RAP vs JavaFX.....	496
RAP vs GWT.....	497
Начало работы с RAP-платформой.....	497
Создание RAP-приложения.....	499
Преобразование SWT-приложения в RWT-приложение.....	505
Преобразование RCP-приложения в RAP-приложение.....	506
Глава 10. Создание GWT-приложений. Пример: Создание GWT приложения Content Delivery Node для платформы Google App Engine. Пример: Постраничное отображение HTML-контента. GWT и SEO. Пример: Визуализация карты сайта Sitemap с GWT.....	508
Пример. Создание GWT приложения Content Delivery Node для платформы Google App Engine.....	518
Пример. Постраничное отображение HTML-контента.....	548
GWT и SEO.....	556
Пример. Визуализация карты сайта Sitemap с GWT.....	561
Глава 11. Создание приложений на основе платформы Riena.....	581
Начало работы с платформой Riena.....	582

Создание Riena RCP приложения.....	583
Создание клиент-сервисного Riena-приложения.....	588
Глава 12. Разработка SCA-приложений.....	596
Глава 13. Разработка приложений на основе платформы Scout.....	605
Глава 14. Разработка Web-приложений на основе платформы WTP с использованием Servlet, JSP, JPA и EJB. Web-сервисы Axis2 и CXF. Создание статического Web-контента. Использование редактора Orion Editor. Пример: Организация комментариев на Web-странице с использованием TinyMCE, Servlet, JSP и JPA.....	615
Создание проекта динамического Web-приложения.....	616
Servlet + JSP.....	619
Servlet + JSP + JPA.....	623
Web + EJB.....	636
Application Client.....	641
Web-сервисы.....	643
Apache Axis2.....	643
Apache CXF.....	649
Создание статического Web-контента.....	652
Использование редактора Orion Editor для отображения HTML, CSS, JavaScript, Java кода на Web-странице.....	654
JQuery плагин редактора Orion.....	662
Создание статического Web-контента с Mylyn WikiText.....	663
Textile.....	663
MediaWiki.....	664
Confluence.....	665
TracWiki.....	665
TWiki.....	665
Пример. Организация комментариев на Web-странице с помощью Javascript HTML WYSIWYG редактора TinyMCE, Servlet, JSP, JPA.....	666
Глава 15. Управление данными с DTP.....	673
Глава 16. Создание отчетов с BIRT.....	683
Глава 17. Использование инструментов Eclipse Modeling Tools: EMF, GMF, Xtext и ATL.....	701
EMF.....	701
GMF.....	706
Xtext.....	711
ATL.....	714

Глава 1. Платформа Eclipse и средства разработки Java. Архитектура Eclipse. Начало работы с Eclipse. Разработка Java SE приложений

Архитектура платформы Eclipse и среда Eclipse SDK

Платформа Eclipse является фундаментом, на основе которого с помощью Eclipse-плагинов создаются все остальные Eclipse-продукты.

В свою очередь Eclipse-платформа состоит из набора подсистем, которые представлены также Eclipse-плагинами, работающими в среде выполнения Eclipse-платформы.

Из компонентов Eclipse-платформы можно выделить минимальный набор Eclipse-плагинов, известный как Rich Client Platform (RCP), на основе которого возможно создание любых клиентских приложений. Поэтому можно сказать, что та же среда Eclipse – это RCP-приложение. Платформа RCP включает в себя такие компоненты как среду выполнения на основе OSGi, библиотеки SWT и JFace, графическую многооконную Workbench-среду и связанные с ней компоненты.

Eclipse-платформа может быть структурирована на подсистемы согласно подпроектам проекта Eclipse Platform (см. Введение) или согласно набору основных предоставляемых функций. Такое деление по основной функциональности дает следующий набор компонентов Eclipse-платформы:

- Platform Runtime Core – основанная на OSGi среда выполнения, обеспечивающая запуск основы платформы, а также динамический поиск и запуск Eclipse-плагинов. Представлена плагинами `org.eclipse.osgi` и `org.eclipse.core.runtime`.
- Resource Management – плагин `org.eclipse.core.resources`, обеспечивающий доступ к проектам, папкам и файлам, связанным с Workspace-пространством. Workspace – рабочее пространство, физически представленное каталогом локальной файловой системы, в котором находятся Eclipse-проекты. Eclipse-платформа обеспечивает синхронизацию и управление Workspace-ресурсами, позволяя определить единые глобальные настройки для всех ресурсов в пределах одного рабочего пространства Workspace. Метаданные Workspace-пространства хранятся в папке `.metadata` его каталога. Создание своего Workspace-пространства для группы проектов определенного типа способствует грамотной организации процесса разработки. Eclipse-проект – это набор файлов, скомпонованных согласно типу проекта и сопровождаемых файлом `.PROJECT` метаданных проекта.

- **Workbench UI** – набор графических инструментов, созданных на основе библиотек SWT и JFace. Workbench обеспечивает реализацию GUI-интерфейса, основными блоками которого являются редакторы и View-представления, а также определяет точки расширения, позволяющие использовать существующие или создавать новые View-представления и редакторы.
- **Team support** – обеспечивает командную разработку кода под контролем версий.
- **Debug support** – плагины `org.eclipse.debug.core` и `org.eclipse.debug.ui` позволяют определить конфигурацию запуска приложения, а также реализовать отладку приложений.
- **Help System** – встроенная документация, содержащая набор электронных книг. При выборе меню **Help | Help Contents** открывается окно встроенного Web-браузера и запускается встроенный сервер Apache Tomcat, обеспечивающий отображение содержимого электронных книг, каждая из которых организована в виде Eclipse-плагина.

Набор Workbench-инструментов обеспечивает графический интерфейс пользователя Eclipse-платформы. Каждое Workbench-окно, открываемое при запуске среды Eclipse, содержит одну или несколько *перспектив*. Каждая перспектива Workbench-окна – это компоновка частей (*parts*) – редакторов и представлений (окон) в определенный набор, сопровождающийся определенными меню и панелями инструментов и соответствующий определенному типу выполняемой задачи. При этом одна перспектива Workbench-окна отличается от другой перспективы данного Workbench-окна отображаемым набором представлений, но использует общий набор редакторов.

Таким образом, визуально рабочая среда Workbench представлена окном, имеющим меню, панель инструментов и набор частей (редакторов и представлений).

Одновременно можно открыть несколько Workbench-окон с помощью выбора команды **New Window** в меню **Window**. При этом для каждого Workbench-окна может быть открыта только одна перспектива.

Сама по себе Eclipse-платформа содержит перспективы навигации ресурсов и поддержки командной разработки. Другие перспективы добавляются Eclipse-плагинами, расширяющими Eclipse-платформу до конкретной среды разработки Eclipse IDE. В частности JDT-плагин добавляет в Eclipse-платформу перспективы, помогающие в разработке Java-приложений.

Перспектива контролирует только первоначальное отображение компоновки представлений и окна редактора. Пользователь может перекомпоновать этот набор, который сохранится при закрытии среды Eclipse.

Новая перспектива открывается с помощью команды **Open Perspective** меню **Window**.

Eclipse-плагины добавляют к Eclipse-платформе новые типы редакторов, представлений и перспектив. К существующим редакторам, представлениям и перспективам могут добавляться новые действия в меню и панелях инструментов.

Eclipse-редакторы обеспечивают открытие, редактирование и сохранение объектов. Сама Eclipse-платформа содержит только редактор текстовых ресурсов, другие типы

редакторов добавляются Eclipse-плагины. Eclipse-редактор загружается в соответствующее окно рабочей области Workbench при двойном щелчке мышкой на ресурсе, отображаемом в представлении.

Eclipse-представления обеспечивают дополнительную информацию об объектах, с которыми идет работа в Workbench-окне. Eclipse-представления открываются с помощью команды **Show View** меню **Window**.

Проект Eclipse Platform является подпроектом проекта Eclipse, представляет который продукт Eclipse Standard, содержащий также Eclipse-плагины JDT (Java development tools) и PDE (Plug-in development environment).

Примечание

Далее описывается работа со средой Eclipse Standard в операционной системе Windows.

Перед инсталляцией среды Eclipse Standard требуется установка JDK (Java Development Kit) (<http://www.oracle.com/technetwork/java/javase/downloads/index.html>).

После скачивания ZIP-архива дистрибутива среды Eclipse Standard (<http://www.eclipse.org/downloads/>), требуется просто его распаковать. Для запуска среды Eclipse Standard дважды щелчком мышкой на исполняемом файле eclipse.exe каталога дистрибутива – после чего начнется загрузка Workbench-окна.

Перед тем как Workbench-окно будет открыто, появится диалоговое окно, запрашивающее расположение Workspace-пространства в локальной файловой системе компьютера.

Страница Welcome

Первое что появится на экране компьютера после определения Workspace-пространства – это страница приветствия Welcome (рис. 1.1).

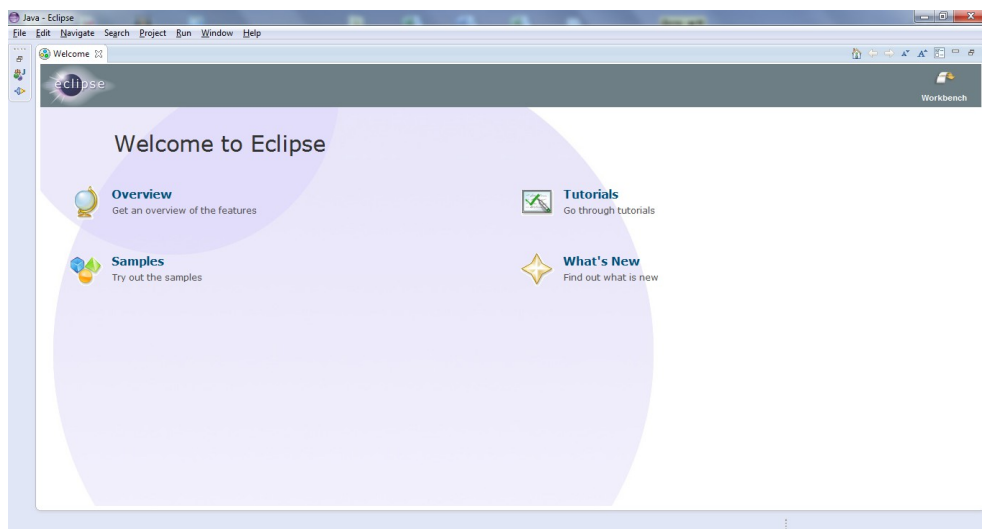


Рис. 1.1. Страница приветствия Welcome

Страницу Welcome можно также открыть с помощью команды **Welcome** меню **Help**.

Страница Welcome содержит кнопку **Workbench**, закрывающую страницу Welcome, а также гиперссылки:

- Overview – переход на страницу обзора среды Eclipse Standard, содержащую ссылки, которые открывают Help-документацию с электронными книгами Workbench User Guide, Java development user guide, Platform Plug-in Developer Guide, JDT Plug-in Developer Guide, Plug-in Development Environment Guide, Eclipse Marketplace User Guide и EGit Documentation. Эти же книги можно открыть с помощью команды **Help Contents** меню **Help**.
- Tutorials – переход на страницу учебных примеров, содержащую кнопки, которые открывают окна **Cheat Sheets** в Workbench-окне, последовательно проводящие пользователя через учебные примеры создания простого Java-приложения, SWT-приложения, командной разработки проекта, создания Eclipse-плагина и RCP-приложения.
- Samples – переход на страницу примеров, содержащую ссылки, которые открывают окна **Cheat Sheets** в Workbench-окне, последовательно проводящие пользователя через учебные примеры интеграции с Workbench-окном, создания пользовательского Java-редактора и SWT-примеры.
- What's new – переход на страницу обзора нововведений, содержащую ссылки, которые открывают соответствующие разделы Help-документации, запускают проверку обновлений и открывают страницы сайта <http://www.eclipse.org/> в Web-браузере.

Страница Welcome содержит в верхнем правом углу набор кнопок управления, среди которых есть кнопка **Customize page**, позволяющая изменить внешний вид и содержание страницы Welcome (рис. 1.2).

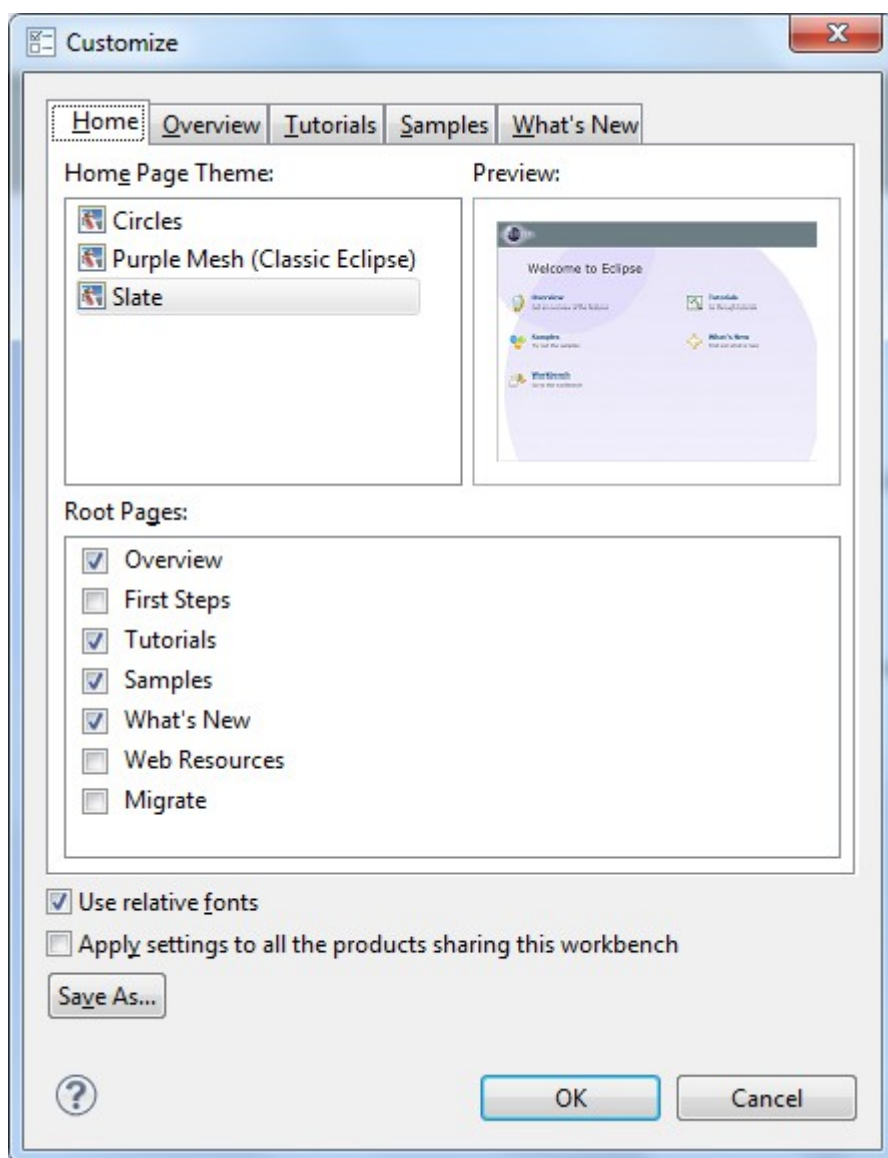


Рис. 1.2. Диалоговое окно изменения страницы Welcome

Рабочая область Workbench

После закрытия страницы Welcome на экране компьютера отобразится содержимое рабочей области Workbench.

Первоначально отобразится перспектива **Java**. Для того чтобы переключиться в перспективу **Resource** Eclipse-платформы выберем в меню **Window** команду **Open Perspective | Other | Resource**. В результате в Workbench-окне появятся три представления **Project Explorer**, **Outline** и **Tasks**, а также окно для редактора ресурсов (рис. 1.3).

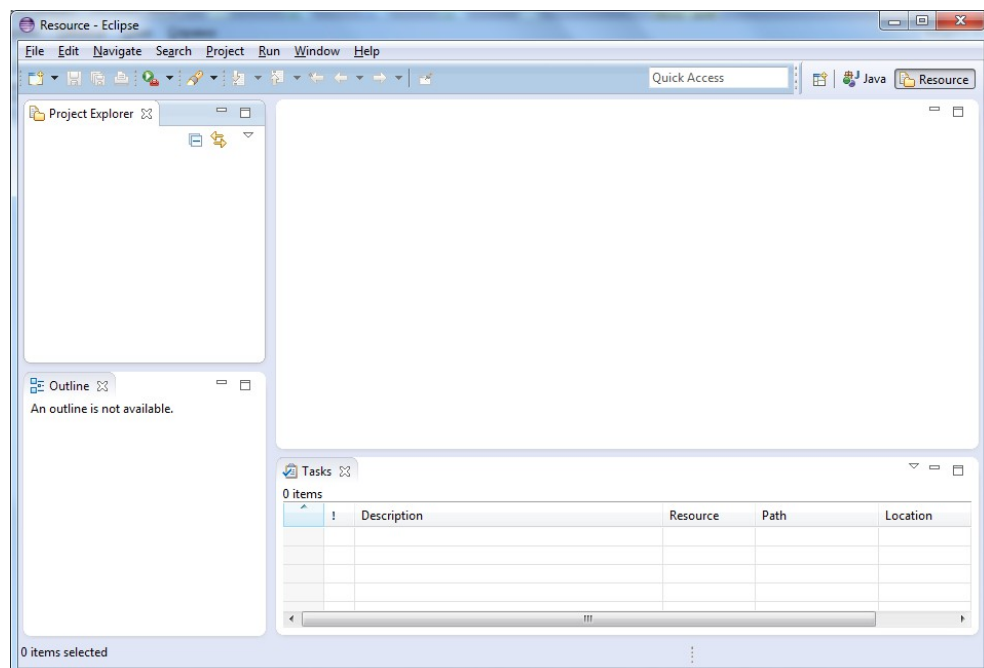


Рис. 1.3. Перспектива **Resource** среды Eclipse Standard

Для того чтобы изменить набор отображаемых представлений можно воспользоваться командой **Show View** меню **Window**. Также среда Eclipse дает возможность перетаскивать представления в Workbench-окне мышкой и изменять их размеры, минимизировать, максимизировать и закрывать представления, используя контекстное меню вкладки представления или его панель инструментов. Вернуться к первоначальному набору Eclipse-представлений позволяет команда **Reset Perspective** меню **Window**. Измененный набор представлений можно сохранить в виде новой перспективы с помощью команды **Save Perspective As** меню **Window**. Удалить перспективу можно в разделе **Perspectives** команды **Preferences** меню **Window**. Настроить перспективу позволяет команда **Customize Perspective** меню **Window**.

Eclipse-представление имеет три контекстных меню. Одно меню появляется при нажатии правой кнопкой мышки на вкладке представления, другое – при нажатии



кнопки **View Menu** панели инструментов представления, третье – при нажатии правой кнопкой мышки на области представления.

Eclipse-представление также можно определить в качестве Detached-представления – такое представление отображается в отдельном окошке вне Workbench-окна. Установить представление в качестве Detached-представления можно с помощью его перетаскивания левой кнопкой мышки за пределы Workbench-окна.

Платформа Eclipse имеет перспективы Resource, Team Synchronizing и CVS Repository Exploring.

Перспектива Resource имеет окно редактора и представления Project Explorer, Outline и Tasks (см. таблицу 1.1).

Таблица 1.1. Представления перспективы Resource

Представление	Описание
Project Explorer	Отображает дерево ресурсов.
Outline	Отображает структуру файла, открытого в данный момент в редакторе.
Tasks	Отображает список маркеров задач.

Перспектива Team Synchronizing имеет окно редактора и представления Synchronize, History, Tasks и Problems (см. таблицу 1.2).

Таблица 1.2. Представления перспективы Team Synchronizing

Представление	Описание
Synchronize	Обеспечивает сравнение локальных и удаленных ресурсов, обновление локальных ресурсов и их передачу в репозиторий.
History	Отображает список изменений ресурса в репозитории и локальную историю ресурса.
Tasks	Отображает список маркеров задач.
Problems	Отображает список ошибок и предупреждений.

Перспектива CVS Repository Exploring имеет окно редактора и представления History и CVS Repositories. Представление CVS Repositories отображает структуру CVS-хранилища, добавленного в Workbench-окно.

Создадим простой Eclipse-проект. Для этого в меню **File** выберем команду **New | Other | General | Project** и нажмем кнопку **Next** – появится Wizard-мастер создания проекта. Введем имя проекта SimpleProject и нажмем кнопку **Finish**. В результате средой Eclipse будет создана папка SimpleProject в каталоге workspace с файлом .PROJECT описания проекта. Файл .PROJECT идентифицирует набор файлов и папок как Eclipse-проект таким образом, что при переносе данного набора в другой каталог файловой системы, он мог бы импортирован в Workbench-окно с помощью команды **Import** меню **File**.

Среда Eclipse содержит большой набор Wizard-мастеров создания ресурсов, импорта и экспорта ресурсов и др. Wizard-мастера призваны помочь пользователю выполнить ту или иную задачу в Workbench-окне, и текущий набор Wizard-мастеров расширяется за счет Eclipse-плагинов.

Для создания папки и текстового файла проекта можно использовать команду **New** контекстного меню, появляющегося при нажатии правой кнопкой мышки на узле проекта в окне **Project Explorer**, можно воспользоваться кнопкой **New** панели инструментов Workbench-окна или можно использовать команду **New** меню **File**.

Выберем команду **New | Other | General | Folder**, введем имя папки projectfolder и нажмем кнопку **Finish**. В папке projectfolder создадим текстовый файл с помощью выбора команды **New | Other | General | File**, ввода имени файла text.txt и нажатии кнопки **Finish**.

В результате созданный файл text.txt будет открыт в текстовом редакторе Workbench-окна (рис. 1.4).

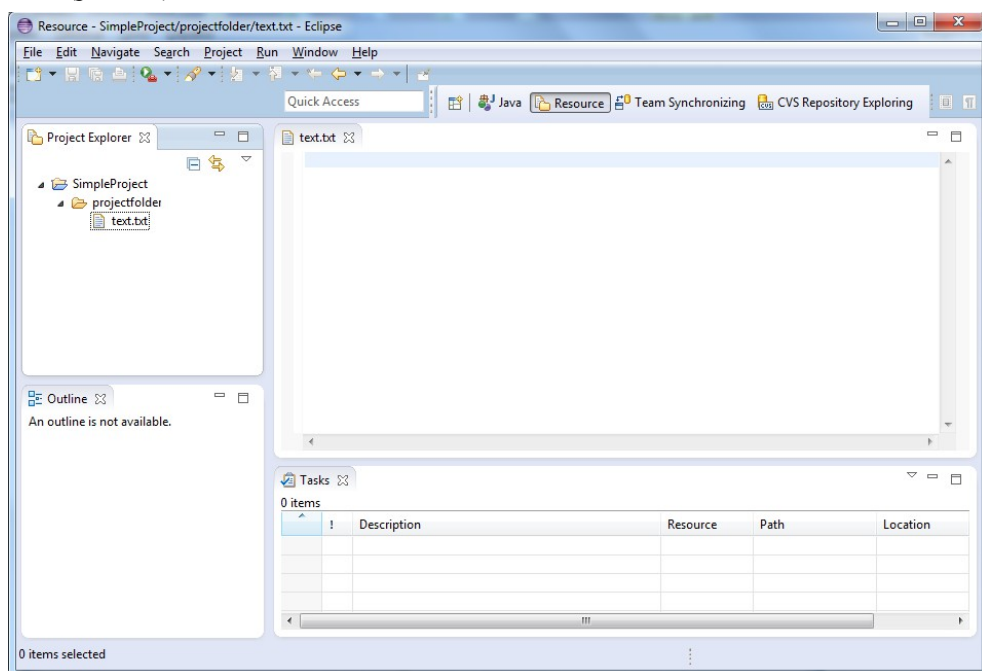


Рис. 1.4. Создание простого Eclipse-проекта с папкой и текстовым файлом

Если в редакторе набрать текст, то в закладке text.txt появится звездочка «*», указывающая что изменения файла text.txt не сохранены. Для сохранения изменений файла text.txt можно нажать кнопку **Save** панели инструментов Workbench-окна (рис. 1.5).

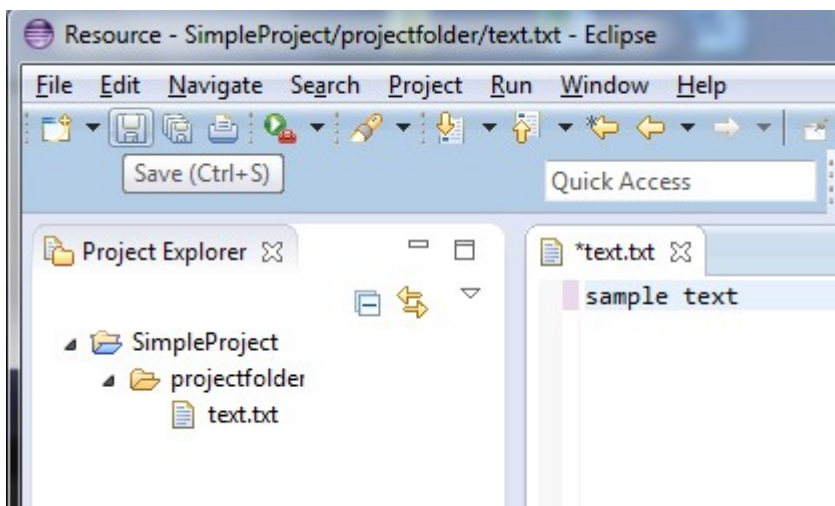


Рис. 1.5. Сохранение изменений текстового файла

Если создать текстовый файл с расширением, например, не .txt, а .doc, тогда среда Eclipse откроет созданный файл не в текстовом редакторе Eclipse-платформы, а в редакторе Microsoft Word операционной системы, который загрузится как OLE-объект в окно редактора (рис. 1.6).

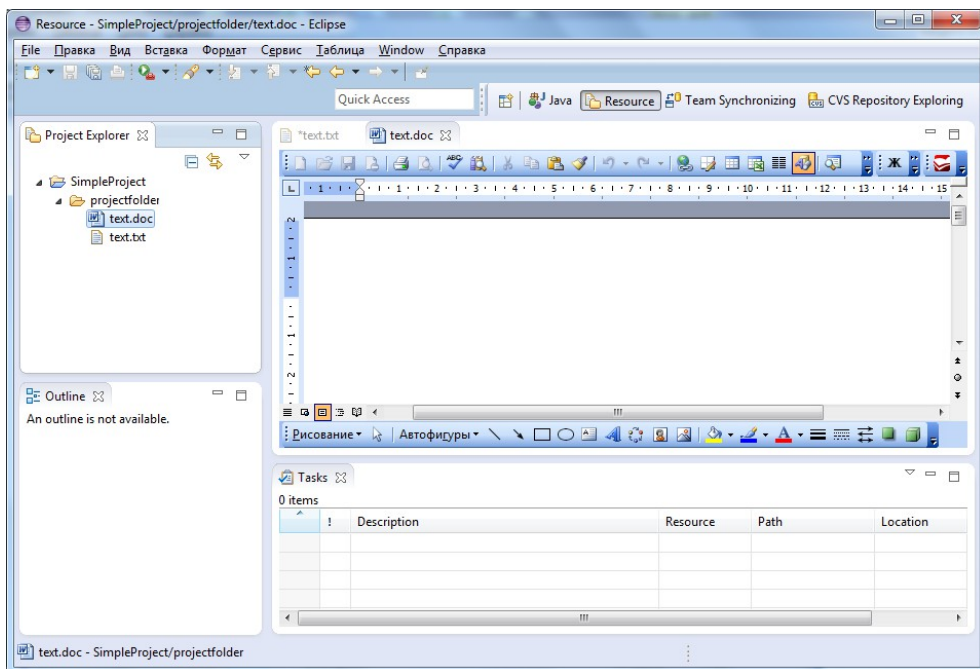


Рис. 1.6. Создание текстового файла с расширением, не совместимым с текстовым редактором Eclipse-платформы

Для того чтобы открыть файл в определенном редакторе можно воспользоваться командой **Open With** контекстного меню, открывающегося при нажатии правой кнопкой мышки на узле файла в окне **Project Explorer**. Например, при выборе команды **Open With | System Editor** текстовый файл text.txt может открыться в блокноте операционной системы Windows.

Для настройки текстового редактора, а также определения соответствий файловых расширений определенным редакторам, можно использовать раздел **Editors** команды **Preferences** меню **Window** (рис. 1.7).

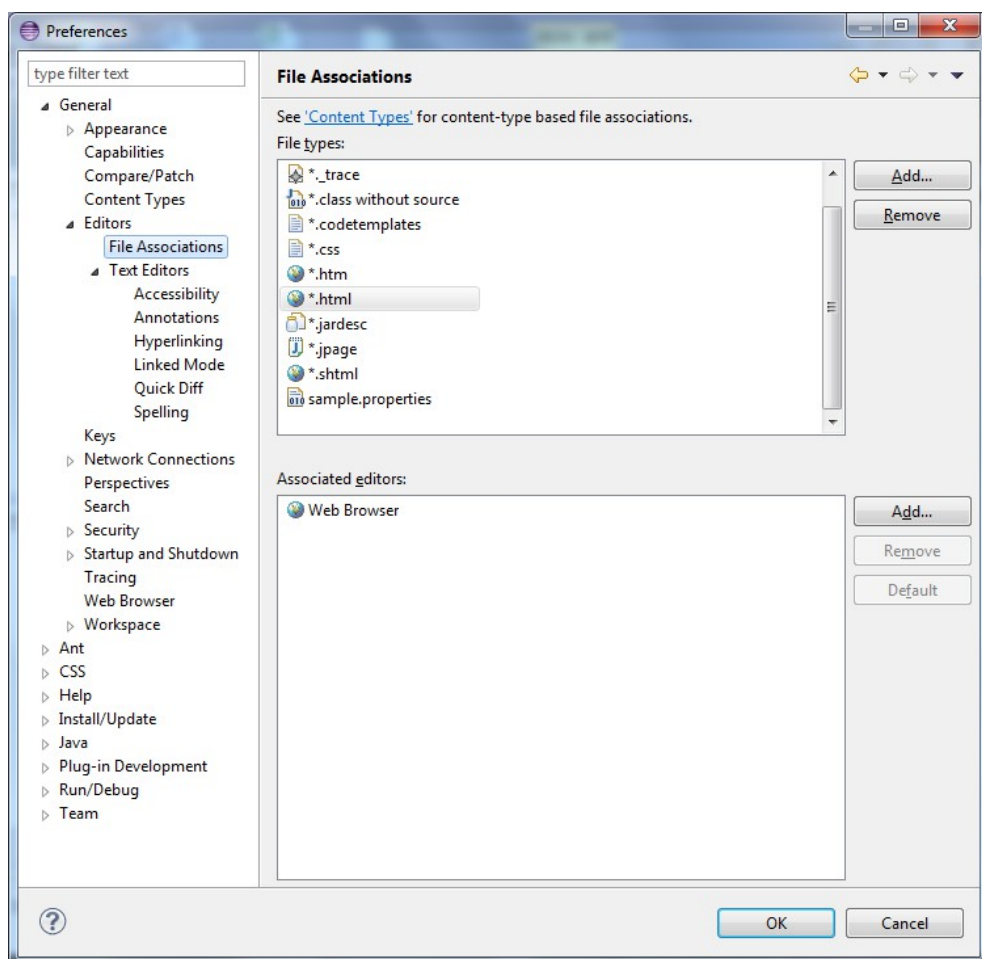


Рис. 1.7. Настройка соответствия файлового расширения определенному редактору

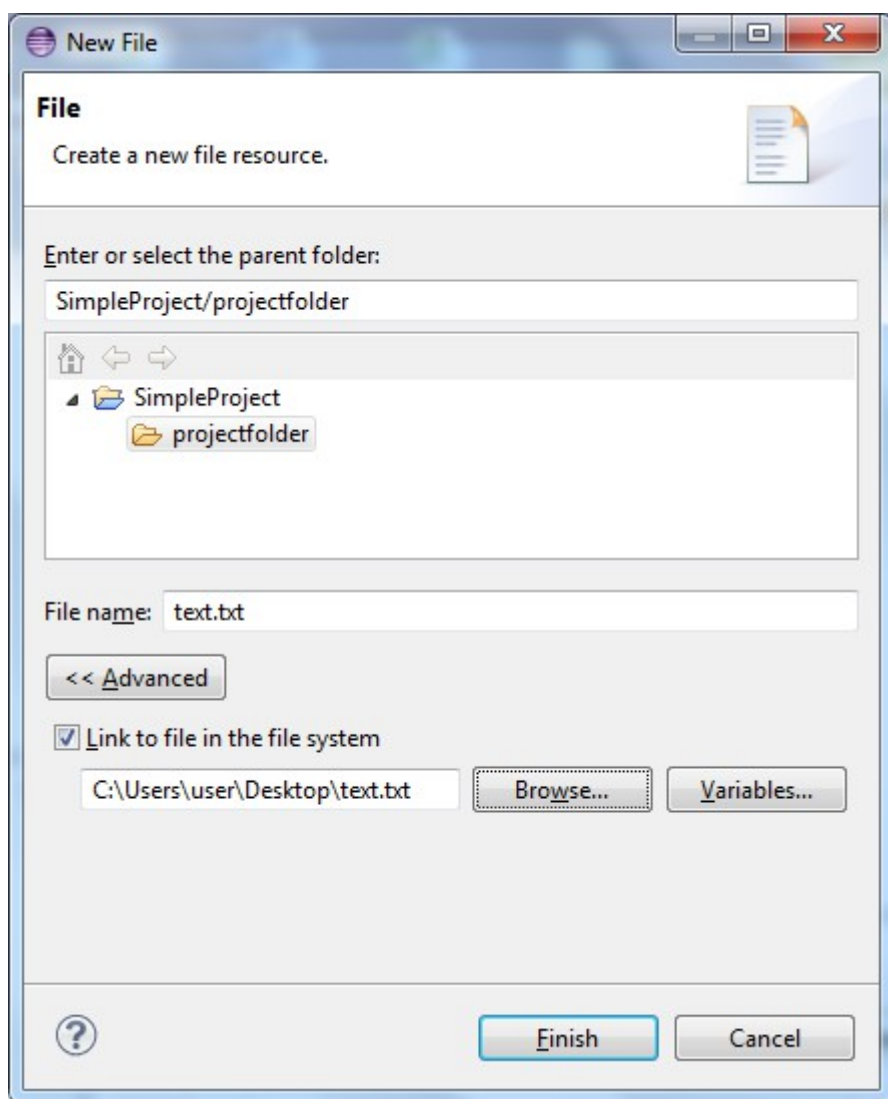


Рис. 1.8. Создание связанного файла

Список связанных ресурсов может быть отредактирован с помощью команды **Properties | Resource | Linked Resources** контекстного меню узла проекта. Отключить саму опцию связанных ресурсов можно используя раздел **General | Workspace | Linked Resources** команды **Preferences** меню **Window**.

Связанные ресурсы могут быть организованы в Eclipse-проекте в иерархическую структуру с помощью виртуальных папок. Виртуальная папка физически не существует в файловой системе, а присутствует в качестве узла Workbench-окна.

Создать виртуальную папку можно с помощью команды **New | Folder**, нажатия кнопки **Advanced** и выбора переключателя **Folder is not located in the file system (Virtual Folder)**.

Создать ресурс проекта можно не только с помощью команды **New**. Готовую папку или файл также можно импортировать в Eclipse-проект с помощью перетаскивания мышкой из файловой системы компьютера в Workbench-окно, используя операцию Copy/Paste или команду **Import** контекстного меню окна **Project Explorer**.

Среда Eclipse разрешает и обратную операцию экспорта папок или файлов из Workbench-окна в файловую систему компьютера с помощью перетаскивания мышкой, используя операцию Copy/Paste или команду **Export** Workbench-окна.

Удалить ресурс проекта в Workbench-окне можно с помощью команды **Delete** контекстного меню окна **Project Explorer**, выбора ресурса и нажатия кнопки Del клавиатуры или используя команду **Delete** меню **Edit**.

Контекстное меню окна **Project Explorer** позволяет также переименовывать и перемещать ресурсы проекта с помощью команд **Rename** и **Move** соответственно.

Поиск ресурсов или текста осуществляется с помощью меню **Search** Workbench-окна. При этом поиск файлов может быть реализован с учетом файлового расширения и с учетом содержащегося в них текста (рис. 1.9). Результаты поиска отображаются в открывающемся представлении **Search** Workbench-окна.

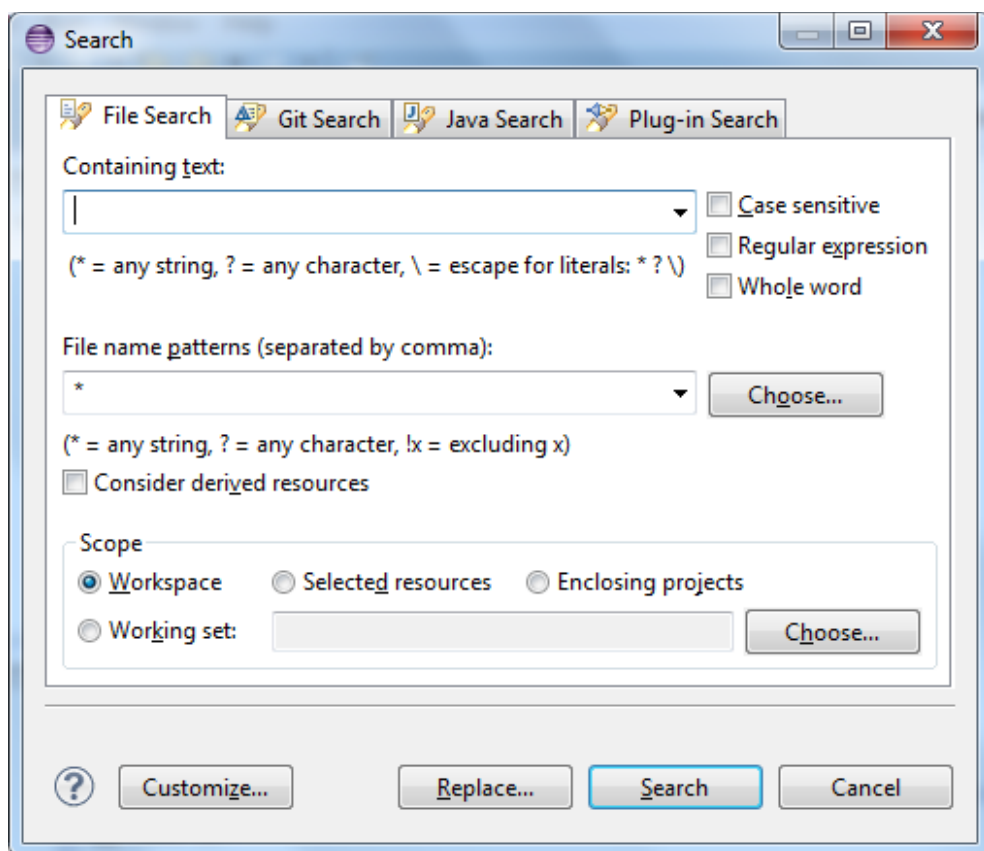


Рис. 1.9. Диалоговое окно настройки поиска ресурсов

Кроме того, быстрый поиск позволяет осуществлять поле **Quick Access** панели инструментов (рис. 1.9.1).

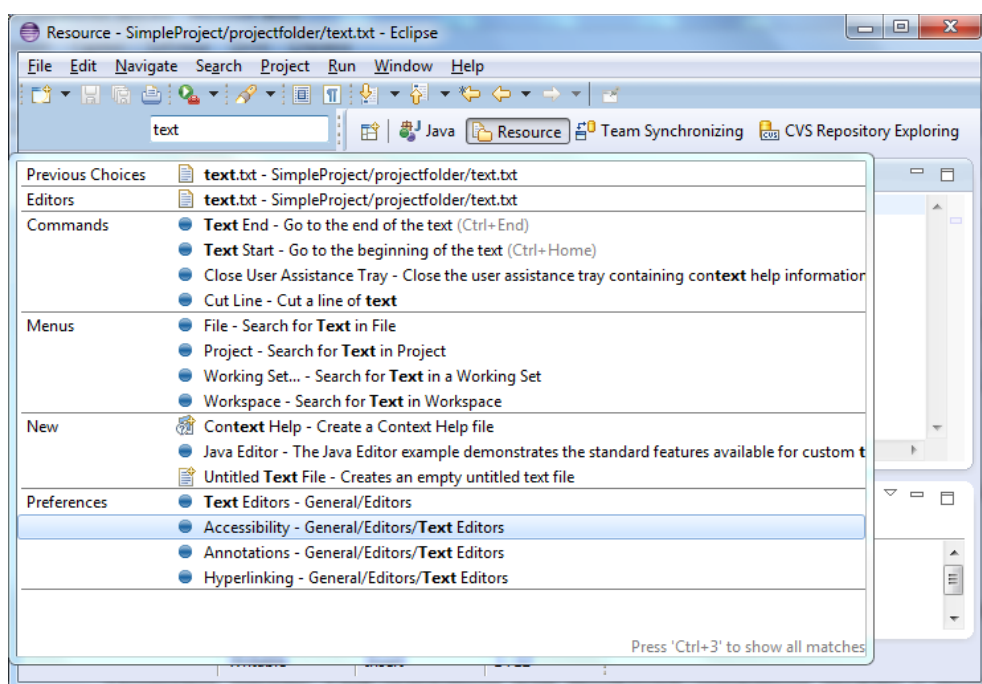


Рис. 1.9.1. Быстрый поиск Quick Access

Среда Eclipse дает возможность пометить ресурсы такими маркерами как задачи Tasks и закладки Bookmarks.

Пометить ресурс Task-маркером или Bookmark-маркером можно с помощью команд **Add Task** или **Add Bookmark** меню **Edit Workbench**-окна, или используя команды **Add Task** или **Add Bookmark** контекстного меню, которое появляется при нажатии правой кнопкой мышки на самом левом крае текстового редактора (рис. 1.10).

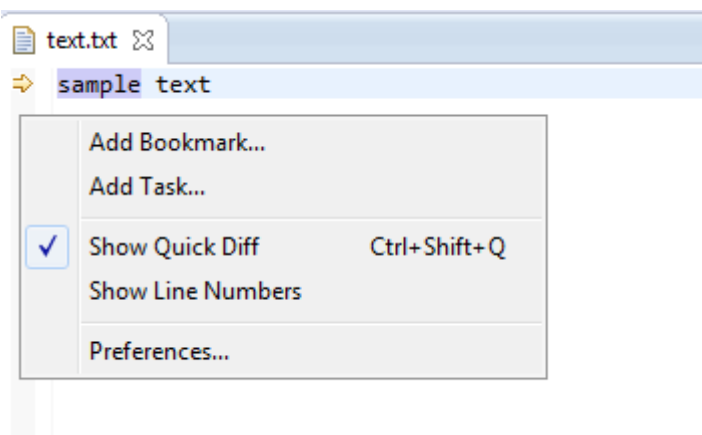


Рис. 1.10. Добавление маркеров с помощью контекстного меню текстового редактора

После создания Task-маркера он появится в окне **Tasks**, а после создания Bookmark-маркера – маркер появится в окне **Bookmarks**, открыть которое можно с помощью команды **Show View** меню **Window**. Управление созданными маркерами обеспечивают контекстные меню соответствующих представлений.

Eclipse-платформа обеспечивает сравнение ресурсов (проектов, папок и файлов) между собой и сравнение версий редактируемого файла согласно его локальной истории с отображением результатов сравнения в представлении **Compare**.

Для сравнения двух ресурсов между собой необходимо в окне **Project Explorer** щелкнуть левой кнопкой мышки на одном ресурсе, нажать кнопку **Ctrl** и щелкнуть левой кнопкой мышки на другом ресурсе для одновременного выбора сразу двух ресурсов. Затем щелкнуть правой кнопкой мышки на выделенных ресурсах и в контекстном меню выбрать команду **Compare With | Each Other**. В результате будет открыто окно **Compare** с отображением различий между двумя ресурсами (рис. 1.11).

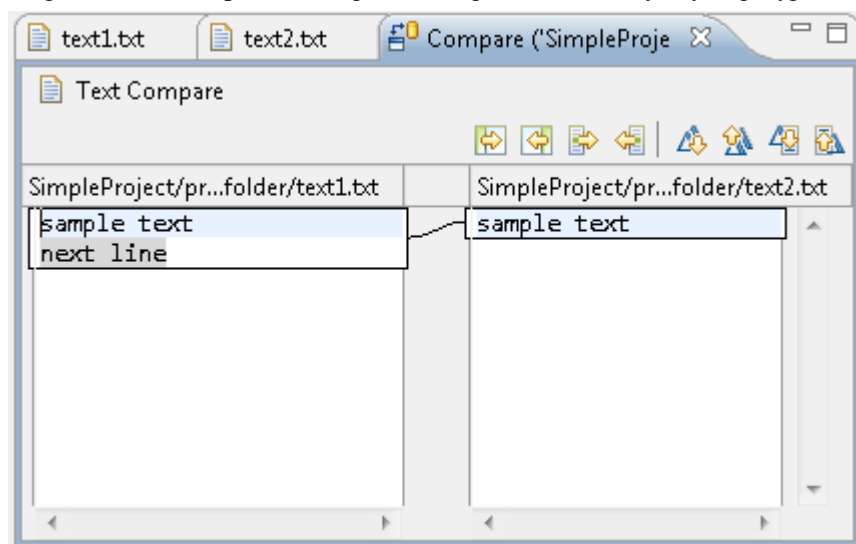


Рис. 1.11. Сравнение содержимого двух текстовых файлов

Для сравнения различных версий редактируемого файла согласно его локальной истории в окне **Project Explorer** щелкнем левой кнопкой мышки на узле файла, нажмем правой кнопкой мышки и в контекстном меню выберем команду **Compare With | Local History**. В появившемся окне **History** два раза щелкнем левой кнопкой мышки на интересующей локальной версии файла – в результате будет открыто окно **Compare** с отображением различий между текущей и предыдущей версиями файла. При этом панель инструментов представления **Compare** обеспечивает функции копирования и навигации.

Локальная история файла организуется средой Eclipse при создании файла и при его модификации. При сохранении отредактированного файла его копия, имеющая идентификатор в виде даты и времени сохранения, также сохраняется, образуя локальную историю файла с возможностью ее просмотра в представлении **History Workbench**-окна. Настраивается локальная история с помощью раздела **General | Workspace | Local History** команды **Preferences** меню **Window**.

Для отображения ресурсов в окне **Project Explorer** можно применять различные фильтры, для создания которых можно воспользоваться кнопкой **Add** раздела **Resource | Resource Filters** команды **Properties** контекстного меню узла проекта.

Ограничить набор отображаемых в Eclipse-представлении ресурсов можно также с помощью рабочего набора Working Set, для применения которого к представлению нужно открыть меню кнопкой **View Menu** панели инструментов представления и выбрать команду **Select Working Set** (представление **Project Explorer**) или **Configure Contents** (представление **Tasks**).

Для того чтобы определить используемый рабочий набор Working Set можно выбрать команду **Customize Perspective** меню **Window** и во вкладке **Command Groups Availability** отметить переключатели **Window Working Set** и **Working Set Manipulation** (рис. 1.12). В результате в панели инструментов Workbench-окна появятся кнопки **Modify window working set**, **Add the selected elements to a working set**, **Remove the selected elements from a working set** (рис. 1.13).

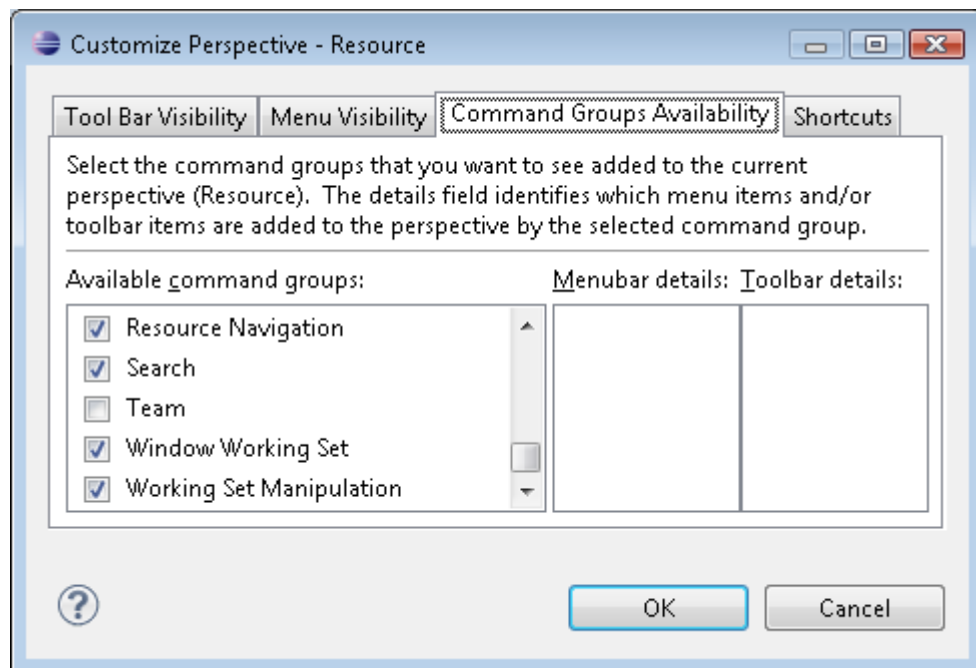


Рис. 1.12. Подключение инструментов работы с набором Working Set

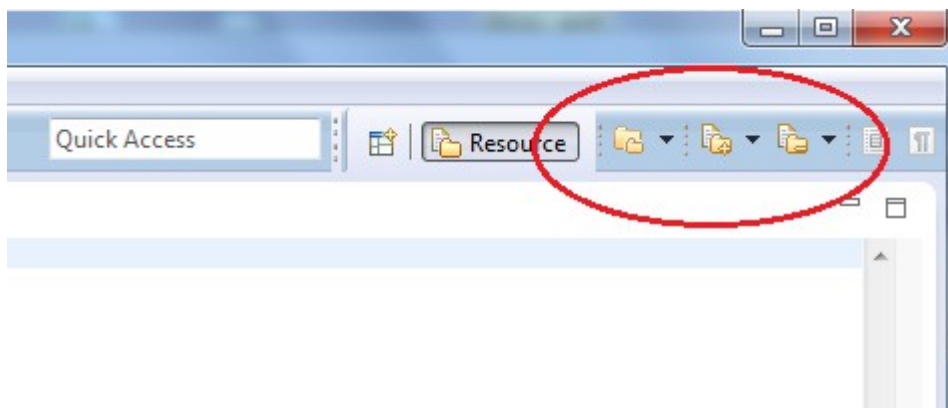


Рис. 1.13. Кнопки управления набором Working Set

CSS и темы Eclipse

Платформа Eclipse обеспечивает поддержку CSS-стилей, что позволяет декларативным способом определять внешний вид GUI-интерфейса основанных на платформе Eclipse приложений, в том числе и среды разработки Eclipse.

Для среды Eclipse Standard файлы CSS-стилей находятся в папке `plugins\org.eclipse.platform_4.3.0.v20130605-2000\css` дистрибутива.

CSS-стили среды Eclipse можно редактировать с помощью внешнего редактора или использовать инструмент Lightweight CSS Editor.

Редактор Lightweight CSS Editor обеспечивает редактирование CSS-стилей темы GUI-интерфейса непосредственно в разделе **General | Appearance** команды **Preferences** меню **Window** среды Eclipse.

Для установки редактора Lightweight CSS Editor выберем команду **Install New Software** меню **Help**, в поле **Work With** введем адрес <http://download.eclipse.org/e4/updates/0.14> и отметим флажок **CSS file editor**, нажмем кнопку **Next** и установим редактор.

Для редактирования темы откроем раздел **General | Appearance** команды **Preferences** меню **Window** среды Eclipse (рис. 1.13.1).

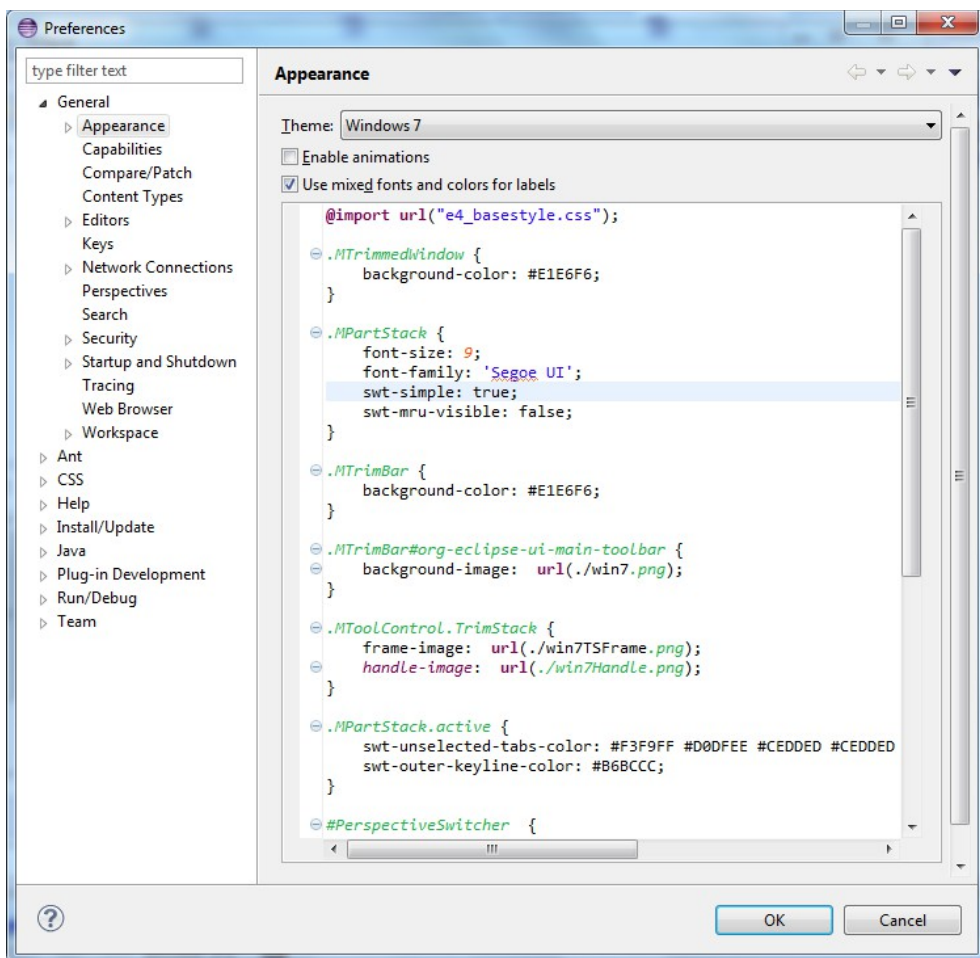


Рис. 1.13.1. Редактор Lightweight CSS Editor

Разработка приложений платформы Java SE

Среда разработки Eclipse Standard

Среда Eclipse Standard содержит плагин Java development tools (JDT), расширяющий Eclipse-платформу до интегрированной среды разработки Java IDE, добавляя перспективы **Java**, **Java Browsing**, **Java Type Hierarchy**, **Debug** и набор представлений, редакторов, мастеров и других инструментов для работы с Java-кодом. JDT-плагин служит фундаментом для разработки любых Java-приложений, включая создание Eclipse-плагинов. JDT-плагин содержится во всех остальных

Eclipse-продуктах, предназначенных для создания Java-приложений на основе различных платформ, и сам по себе помогает в разработке Java-кода платформы Java SE.

Перспектива Java содержит окно редактора и представления Package Explorer, Outline, Problems, Javadoc, Declaration (см. таблицу 1.3).

Таблица 1.3. Представления перспективы Java

Представление	Описание
Package Explorer	Отображает Java-проект с его структурой, определяемой сборкой проекта, в виде узлов папок и библиотек, Java-пакетов, Java-файлов с их внутренней структурой.
Outline	Отображает компилируемую структуру редактируемого в данный момент Java-файла.
Problems	Отображает ошибки и предупреждения сборщика проекта.
Javadoc	Отображает документацию выбранного в данный момент Java-элемента.
Declaration	Отображает исходный код выбранного в данный момент Java-элемента.

Перспектива Java Browsing содержит окно редактора и представления Projects, Packages, Types, Members (см. таблицу 1.4).

Таблица 1.4. Представления перспективы Java Browsing

Представление	Описание
Projects	Отображает Java-проект, его папки и библиотеки без возможности их раскрытия в данном представлении.
Packages	Отображает при выборе в окне Projects узла список его Java-пакетов.
Types	Отображает при выборе в окне Packages узла список его Java-типов.
Members	Отображает при выборе в окне Types узла его содержимое.

Перспектива Java Type Hierarchy содержит окно редактора и представление Type Hierarchy, отображающее иерархию Java-типа с помощью команды **Open Type Hierarchy** контекстного меню.

Перспектива Debug содержит окно редактора и представления Debug, Breakpoints, Variables, Outline, Console, Tasks (см. таблицу 1.5).

Таблица 1.5. Представления перспективы Debug

Представление	Описание
Debug	Обеспечивает управление процессом отладки и запуска Java-кода.
Breakpoints	Отображает список контрольных точек отладки Java-кода.
Variables	Отображает информацию о переменных выбранного узла окна Debug.
Outline	Отображает компилируемую структуру редактируемого в данный момент Java-файла.
Console	Отображает системный вывод выполнения Java-кода.
Tasks	Отображает список маркеров задач проекта.

Для создания простого Java-приложения откроем перспективу **Java** среды Eclipse Standard и в меню **File** выберем команду **New | Other | Java | Java Project** и нажмем кнопку **Next**, введем имя проекта Hello и нажмем кнопку **Finish**.

В окне **Package Explorer** нажмем правой кнопкой мышки на узле проекта и в контекстном меню выберем команду **New | Other | Java | Class**, нажмем кнопку **Next**, в поле **Package:** введем имя пакета hello, в поле **Name:** введем имя класса Hello, отметим переключатель **public static void main(String[] args)** создания точки входа в приложение и нажмем кнопку **Finish** (рис. 1.14).

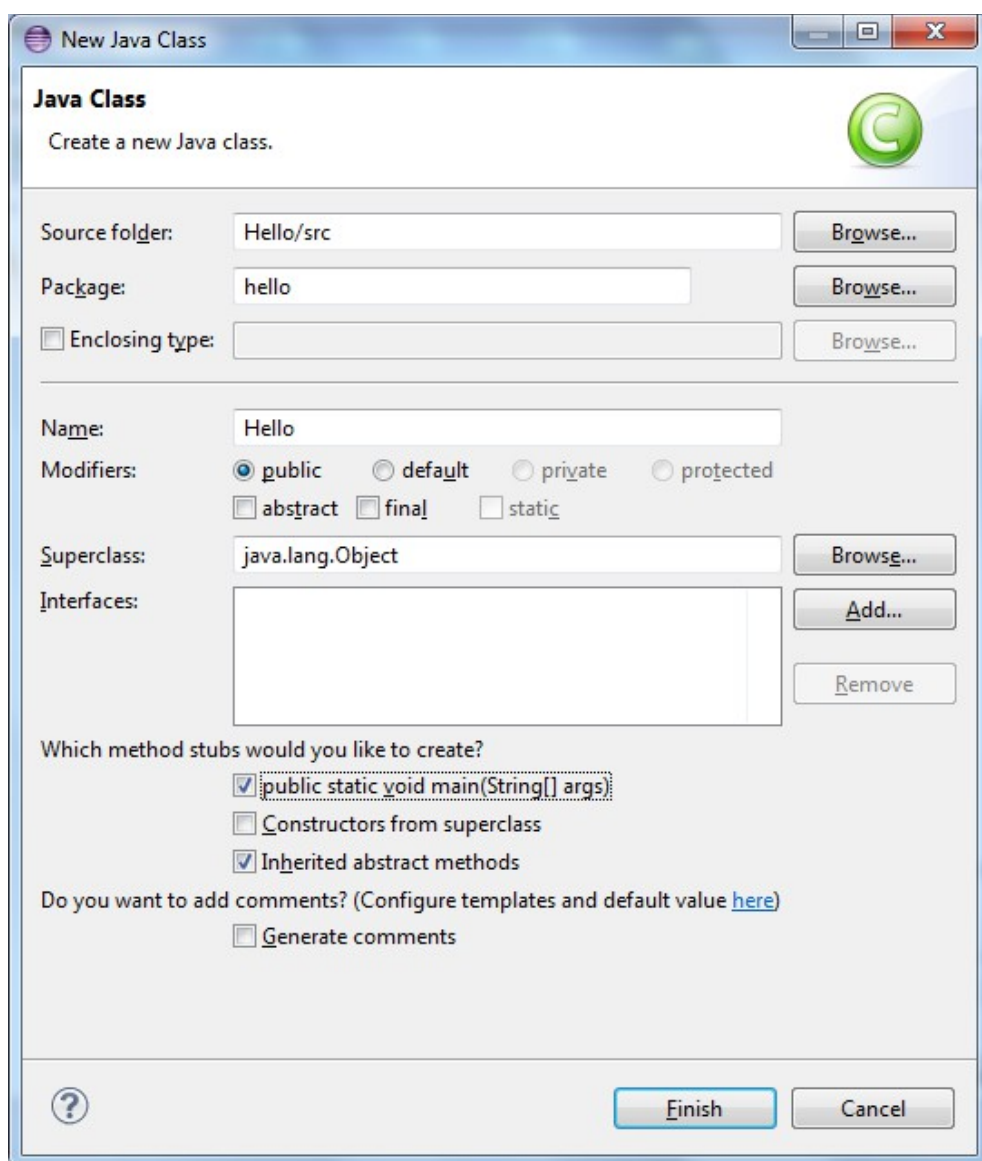


Рис. 1.14. Мастер создания Java-класса

В результате в окне **Package Explorer** среды Eclipse отобразится иерархия проекта Hello, в окне редактора будет открыт файл Hello.java, а в Workspace-каталоге будет создана папка Hello с файлами .PROJECT и .CLASSPATH и папками .settings, bin и src.

Файл `.PROJECT` определяет папку `Hello` как Eclipse-проект, имеющий тип Java-проект (тэг `<natures>`), и содержит команду вызова Eclipse-компилятора исходного Java-кода при инкрементальной сборке проекта (тэг `<buildCommand>`).

Файл `.CLASSPATH` содержит определения папок и файлов, участвующих в построении и запуске проекта, и автоматически дополняется новыми определениями при использовании команды **Build Path** контекстного меню окна **Package Explorer** среды Eclipse.

Папка `.settings` хранит установки плагина, папка `src` – исходный Java-код, а открыв папку `bin` можно обнаружить уже откомпилированный и готовый к запуску Java-код. Произошло это из-за того, что в разделе **General | Workspace** команды **Preferences** меню **Window** отмечен переключатель **Build automatically**. Если убрать отметку данного переключателя, тогда среда Eclipse не будет автоматически компилировать Java-код, а в контекстном меню окна **Package Explorer** появится команда **Build Project**.

Раздел **Java** команды **Preferences** меню **Window** позволяет определить такие установки как используемая среда выполнения JRE, путь сборки проекта, установки Java-редактора и Java-компилятора и др.

Редактор Java-кода среды Eclipse обеспечивает подсветку синтаксиса, включая выделение цветом комментариев к коду, ключевых слов, текстовых строк, проверку синтаксиса, автозавершение кода, форматирование кода, подсказки Quick Fix, интегрированные опции отладки кода.

Открыть Java-файл в Java-редакторе можно, щелкнув два раза левой кнопкой мышки на узле файла, или любого из элементов структуры Java-кода, отображаемых в представлении. При этом с Java-редактором связано представление **Outline**, отображающее компилируемую структуру Java-кода с возможностью ее фильтрации с помощью панели инструментов представления (рис. 1. 15).

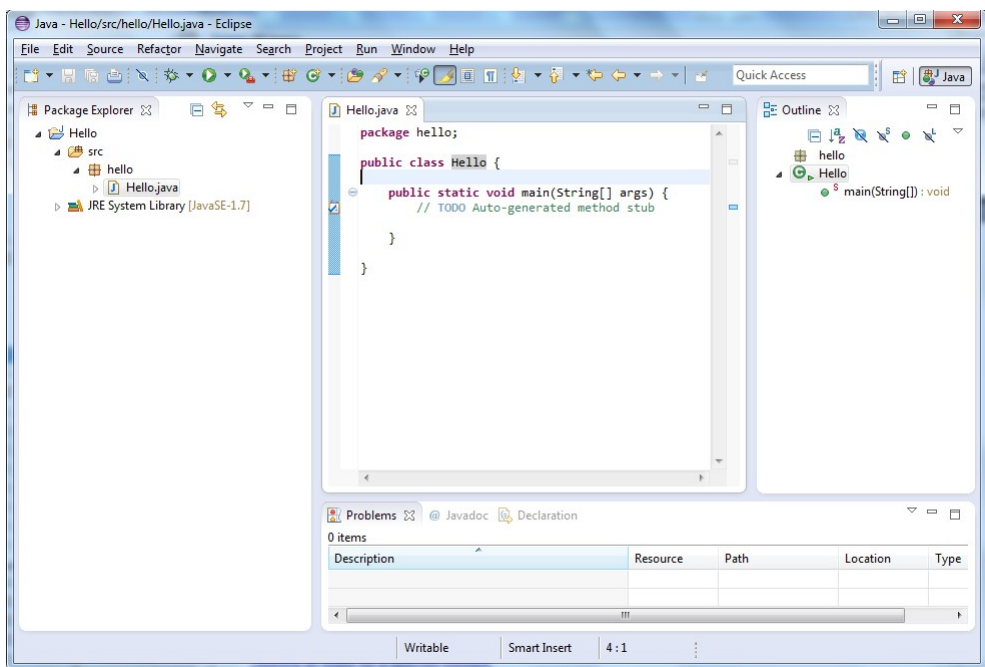


Рис. 1.15. Редактор Java-кода и связанное с ним представление **Outline**

В верхней части Java-редактора с помощью кнопки **Toggle Breadcrumb** можно открыть панель навигации, отображающую структуру проекта (рис. 1.16).

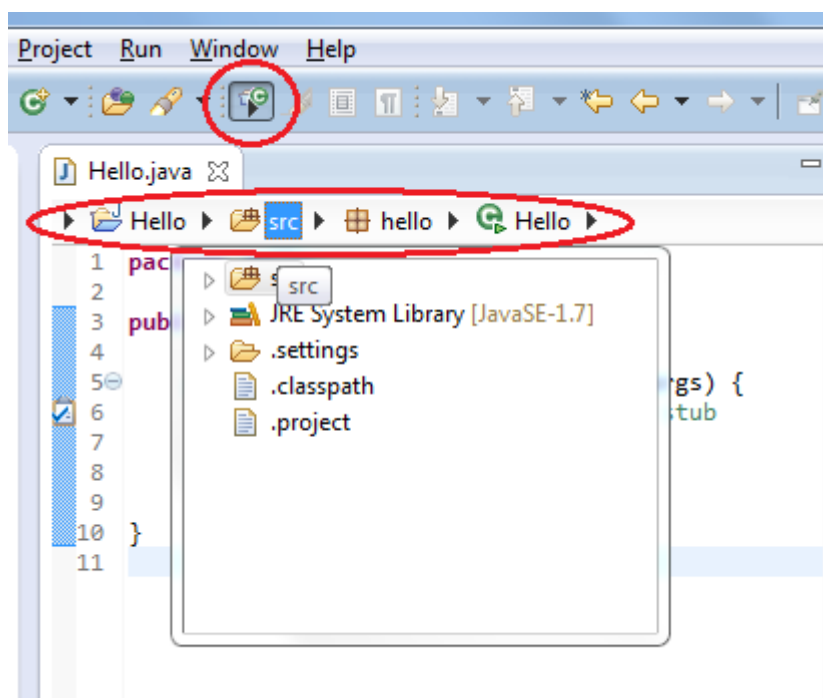



Рис. 1.16. Breadcrumb-панель навигации редактора Java-кода

Выбрав команду **Show Line Numbers** контекстного меню крайней левой полосы Java-редактора, которое открывается нажатием правой кнопкой мышки, в окне редактора будут отображаться номера строк Java-кода.

Если нажать кнопку  **Link With Editor** окна **Package Explorer**, тогда в представлении **Package Explorer** будет подсвечен именно тот файл, который открыт в данный момент в Java-редакторе.

Для навигации Java-кода в Java-редакторе можно открыть окно **Quick Outline**, используя команду **Quick Outline** контекстного меню редактора, открываемого нажатием правой кнопкой мышки в окне редактора. Окно **Quick Outline** является аналогом представления **Outline** и отображает структуру редактируемого Java-кода, при выборе одного из элементов которой – он подсвечивается в Java-редакторе. В верхнем поле окна **Quick Outline** есть возможность ввода интересующего элемента, при этом окно **Quick Outline** автоматически фильтрует отображаемое дерево Java-структуры. Меню окна **Quick Outline** позволяет произвести его настройки (рис. 1.17).

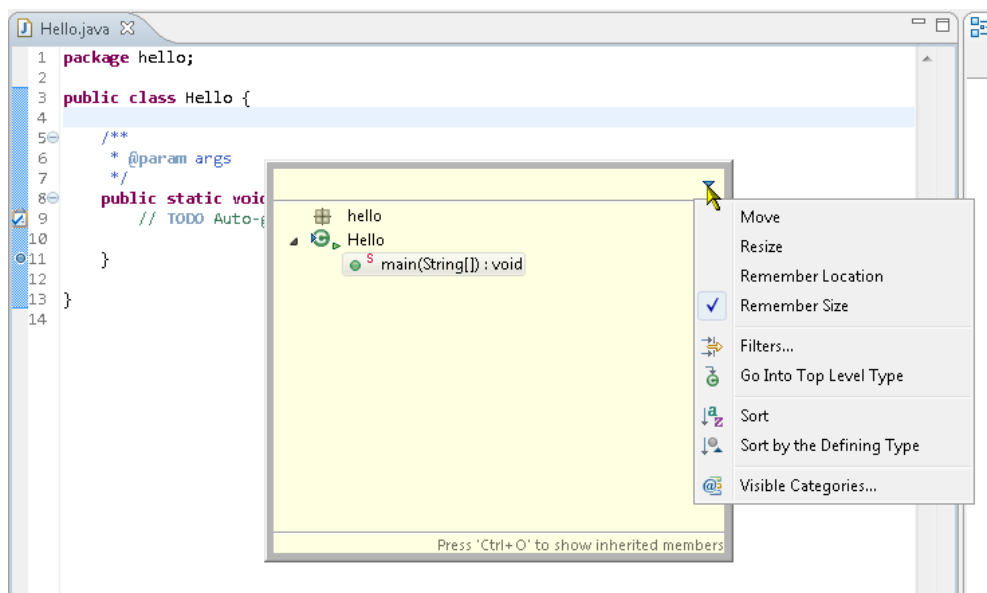


Рис. 1.17. Окно **Quick Outline** редактора Java-кода

Широкие возможности навигации также предоставляет меню **Navigate Workbench** окна.

В редакторе Java-кода начнем добавлять новый метод класса Hello. При этом Java-редактор будет автоматически осуществлять проверку синтаксиса и в окне редактора появятся маркеры ошибок с подсказками (рис. 1.18).

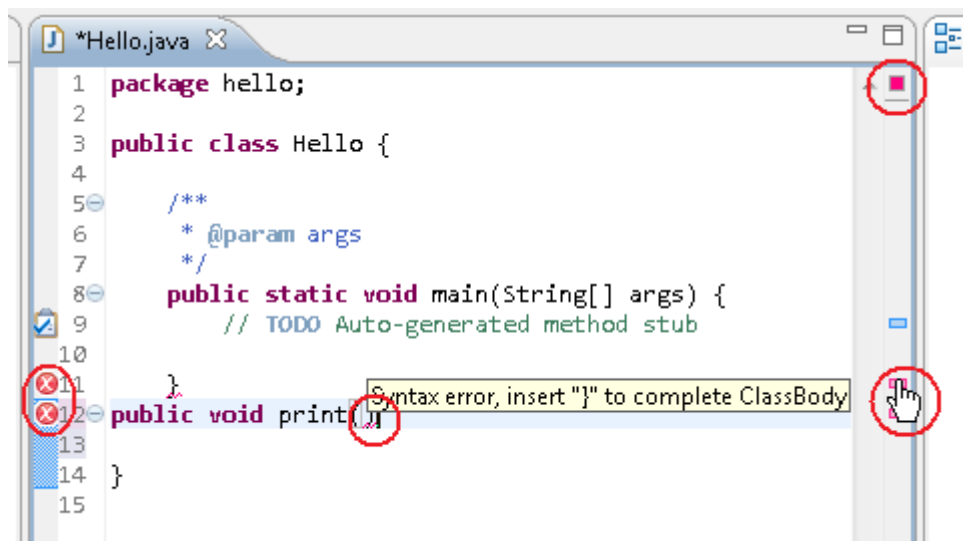


Рис. 1.18. Проверка синтаксиса Java-редактора

При вводе нового метода класса, он автоматически добавится в окно **Package Explorer** и окно **Outline**.

При синтаксически правильном завершении ввода нового метода маркеры ошибок исчезнут.

В новом методе осуществим вывод строки текста в консоль. При наборе Java-кода можно использовать два типа подсказок. Подсказку Content Assist можно вызвать нажатием комбинации клавиш **Ctrl + Space**, или данная подсказка появляется сама при вводе разделителя «.»». Подсказка Content Assist обеспечивает автозавершение кода путем выбора одного из вариантов предлагаемого списка (рис. 1.19 и рис. 1.20).

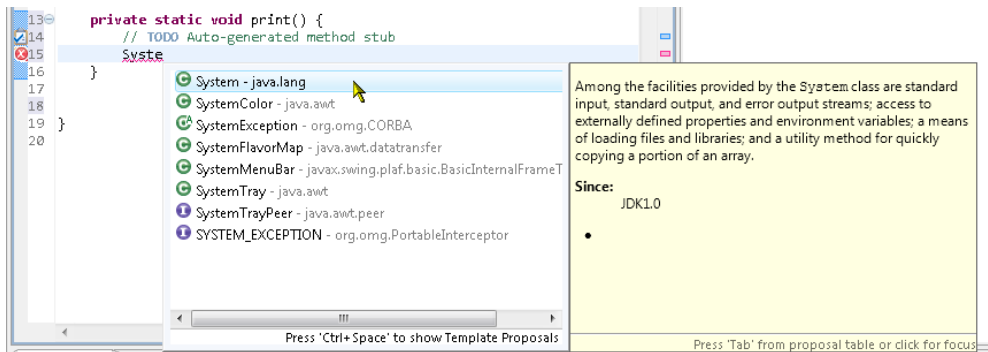


Рис. 1.19. Вызов подсказки Content Assist нажатием комбинации клавиш **Ctrl + Space**

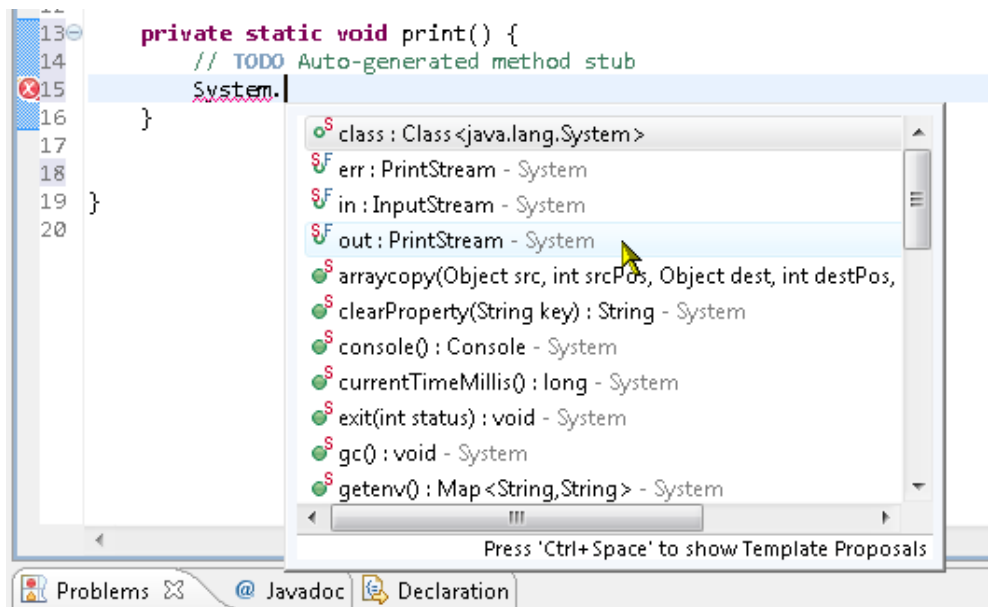


Рис. 1.20. Подсказка Content Assist появляется сама при вводе разделителя «.»

Подсказка Quick Fix вызывается командой **Quick Fix** контекстного меню Java-редактора и предлагает различные варианты исправления ошибки, связанной с объявлением пакета, импортом, созданием Java-типов, конструкторов, методов, полей и переменных, обработкой исключений, путем приложения и др. (рис. 1.21). Подсказка Quick Fix также обеспечивает автозавершение кода путем выбора одного из предлагаемых вариантов решения проблемы.

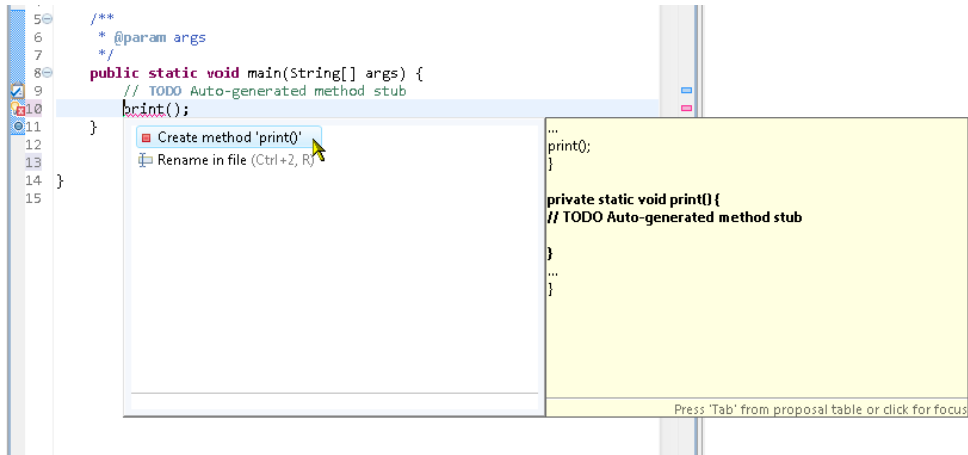


Рис. 1.21. Вызов подсказки Quick Fix

Отменить произведенный ввод кода позволяет команда **Undo Typing** меню **Edit** или контекстного меню Java-редактора.

Java-редактор поддерживает форматирование кода. Например, если перенести в строке:

```
public static void main(String[] args) {
```

`main` на другую строку:

```
public static void
    main(String[] args) {
```

и в меню **Source** Workbench-окна выбрать команду **Format** – код примет свой первоначальный вид.

Помимо форматирования исходного кода меню **Source** Workbench-окна или команда **Source** контекстного меню Java-редактора обеспечивает такие действия как генерация блока комментариев, организация импорта, генерация методов Get/Set и др.

Так как созданный класс `Hello` имеет статический метод `main` – точку входа в приложение, его можно развернуть как настольное приложение. Для запуска кода

класса `Hello` из среды Eclipse можно нажать кнопку



Run панели

инструментов Workbench-окна или нажать правой кнопкой мышки на узле файла Hello.java в окне **Package Explorer** и в контекстном меню выбрать команду **Run As | Java Application**. В результате в окне **Console** отобразится строка «Hello World» (рис. 1.22).

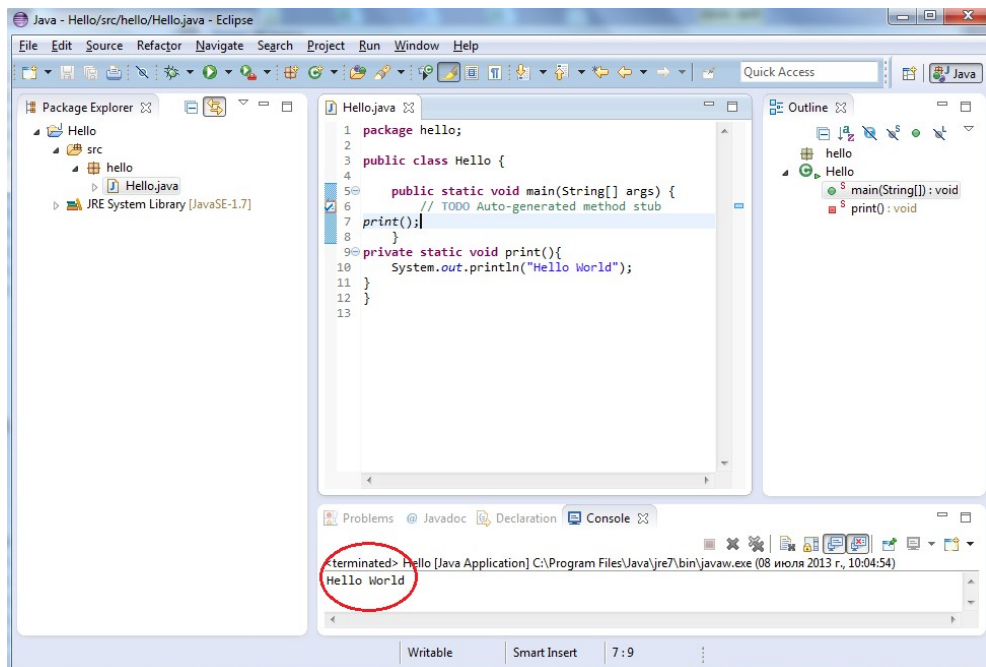


Рис. 1.22. Результат запуска кода класса Hello

Команда **Run As | Run Configurations** позволяет настроить запуск Java-кода (рис. 1.23).

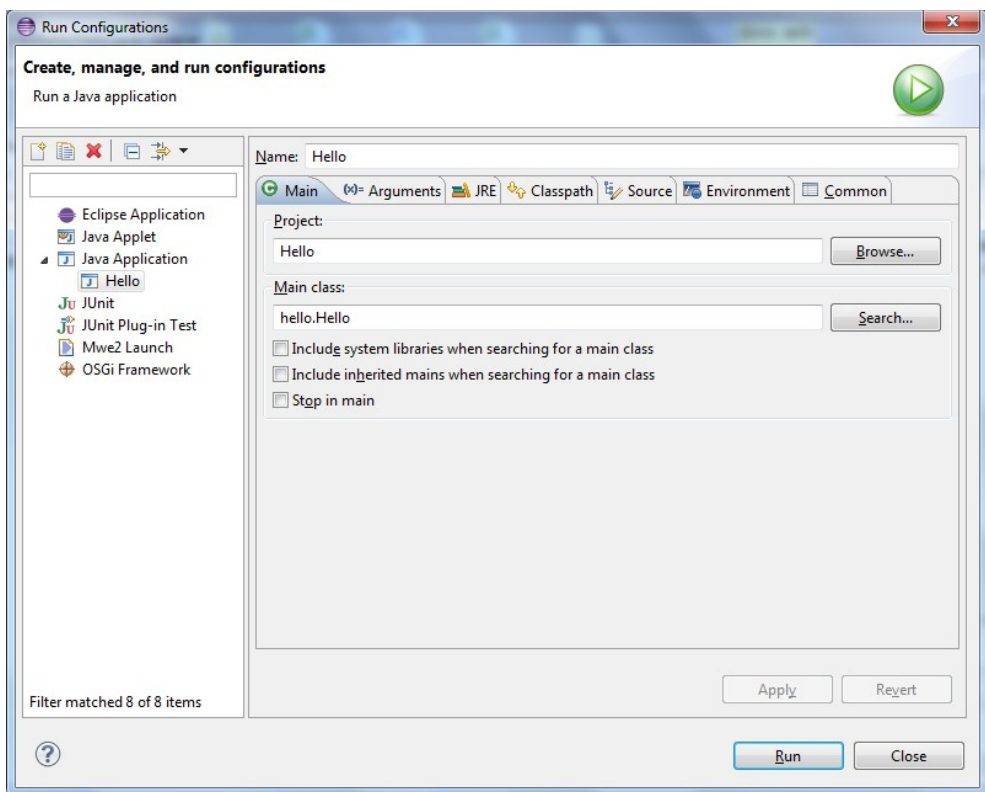


Рис. 1.23. Мастер настройки конфигурации запуска Java-кода

Например, изменим код класса Hello следующим образом:

```
package hello;

public class Hello {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        print(args[0]+" "+args[1]);
    }
    private static void print(String str) {
        // TODO Auto-generated method stub
        System.out.println(str);
    }
}
```

Выберем команду **Run As | Run Configurations** и в поле **Program arguments** вкладки **Arguments** окна **Run Configurations** введем Hello World, нажмем кнопку **Run** – в результате в окне **Console** отобразится строка «Hello World».