# Основы работы в Eclipse IDE

**Цель:**

1. Ознакомление со средой разработки Eclipse.Создание простого консольного приложенияHello, Java!!!.
2. Задачи на массивы, примитивные типы, классы, наследование
3. **Среда разработки Eclipse**

В настоящем разделе описываются основы использования платформы Eclipse в качестве среды разработки Java-приложений. Рассматриваются основные структурные элементы и функциональные возможности EclipseWorkbench на примере создания простого Java-приложения. В разделе рассмотрены не все возможности, используемые при выполнении лабораторных работ, однако изучение базовых принципов Eclipse позволит легко освоить новые элементы среды, необходимость в которых будет появляться в ходе выполнения заданий.

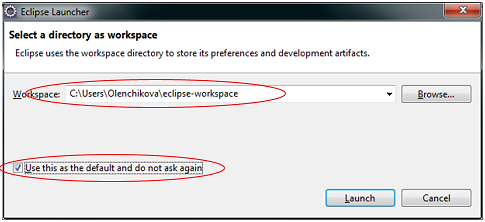
**1.1Запуск Eclipse**

Для запуска Eclipseвыполните:

Пуск→Все программы→Программирование→EclipseJEE.

**1.2 Создание рабочей области (workspace)**

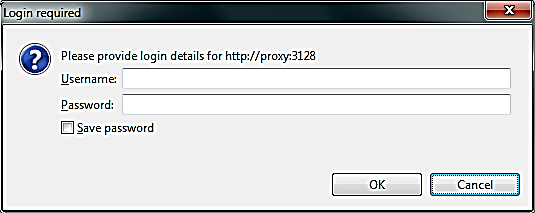
При первом запуске Eclipse предлагает выбрать каталог рабочей области (Workspace). Это каталог, в котором будут храниться все файлы, относящиеся к проектам – настройки проектов, исходные файлы программ, результаты компиляции и сборки и т.д.



Укажите каталог рабочей области (свой рабочий диск), установите флажок Usethisasthedefault… и нажмите Launch.

При дальнейшей работе, переключение между рабочими областями, а также создание новой рабочей области выполняется с помощью команды меню File/SwitchWorkspace.

В появившемся окне



нажмите Cancel

**1.3Основные представления, перспективы и редакторы**

Прежде чем приступить к созданию проекта Eclipse, познакомимся поближе с основными элементами EclipseWorkbench.

EclipseWorkbench состоит из (рис. 1):

* перспектив (perspectives);
* представлений (views);
* редакторов (editors).

**Перспектива** – группа представлений и редакторов в окне Workbench. Количество перспектив в одном окне Workbench не ограничено. Так же и перспектива может содержать неограниченное количество представлений и редакторов. В одном окне каждая из перспектив может иметь различные наборы представлений, но все перспективы делят между собой один и тот же набор редакторов.

Для подключения перспективы используется команда **основного меню**Window/Open Perspective.

Перечень открытых перспектив по умолчанию отображаются в правой верхней части экрана. С помощью этого же списка выполняется переключение между перспективами:

1

**Представление** – это визуальный компонент Workbench. Обычно он используется для перемещения по иерархии некоторой информации (например, структуры проекта), открытия редакторов, просмотра результатов выполнения и т.д. Все изменения, сделанные в представлениях, немедленно сохраняются. В большинстве случаев, в окне Workbench может существовать только один экземпляр представления некоторого типа.

Многие представления входят в состав тех или иных перспектив и отображаются при их открытии. Тем не менее, с помощью команды меню Window/Show View можно открыть любое представление, независимо от текущей перспективы.

Восстановить набор представлений текущей перспективы можно с помощью команды Window/Reset Perspective.

**Редактор** также является визуальным компонентом Workbench. Используется для просмотра и редактирования некоторого ресурса, например,Java-класса или файла, содержащего SQL-запросы. Изменения, делающиеся в редакторе, подчиняются модели жизненного цикла «открыть-сохранить-закрыть» (anopen-save-closelifecyclemodel). В Workbench может существовать несколько экземпляров редакторов.

Редакторы того или иного типа обычно отображаются автоматически при двойном щелчке на соответствующий ресурс (например, Java-класс). Сохранение данных выполняется с помощью команды меню File/Save, либо нажатием на клавиши Ctrl-S.

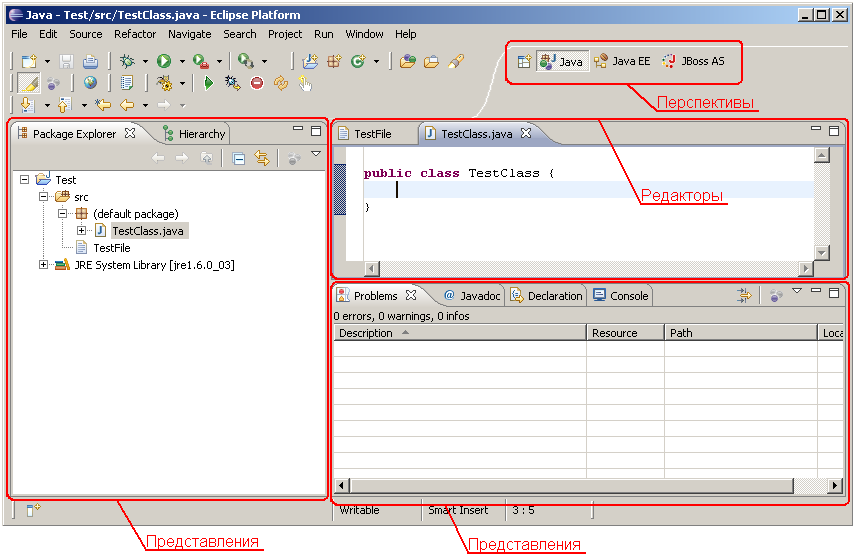


Рисунок 1 – Основное окно Eclipse

В данный момент времени может быть активным только одно преставление или редактор. Активным является представление или редактор у которого подсвечен заголовок. На активный элемент будут воздействовать общие операции вырезки, копирования и вставки (cut, copy, paste). Также активный элемент обуславливает содержимое строки состояния (statusline). Если закладка редактора белая, то это означает, что данный редактор неактивен, однако представления могут отображать информацию, полученную из редактора, бывшего активным последним.

**1.3 Помощь и обучение работе со средой**

Eclipse IDE имеет встроенный Help и Titorial для быстрого ознакомления со средой.

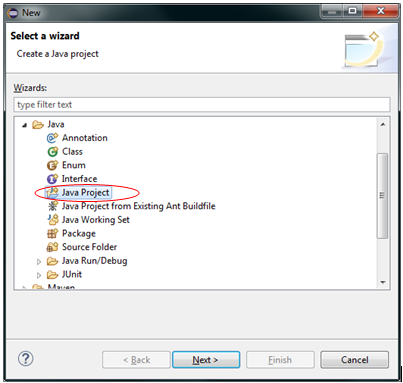
**1.4 Создание проекта и ресурсов проекта**

Проект Eclipse – это совокупность служебных и конфигурационных файлов Eclipse и ресурсов проекта, относящихся к конкретному разрабатываемому приложению или подсистеме приложения. Физически на диске проект представляет собой каталог по имени проекта внутри каталога рабочей области, внутри которого содержатся файлы – составляющие проекта.

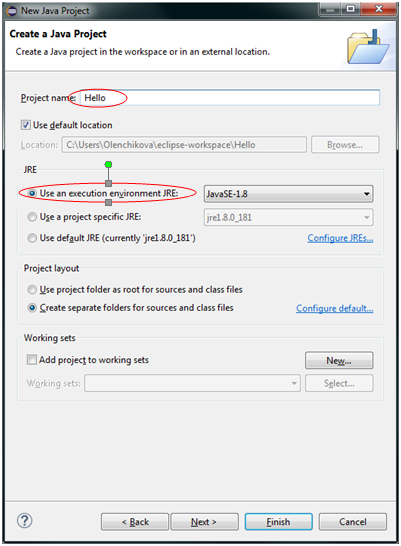
Под **ресурсом проекта** понимаются любые элементы, из которых строится приложение – Java-классы, Java-библиотеки, HTML-страницы, изображения, текстовые файлы, наборы SQL-скриптов и т.д. Строго говоря, сам проект также является ресурсом. Физически каждый ресурс проекта представлен каталогами и файлами соответствующего типа.

Существует несколько способов создания проектов и ресурсов проекта. Основным из них является команда основного меню File/New. С помощью подпункта Other можно отобразить все типы ресурсов, доступных для создания. Для удобства выбора типы создаваемых проектов и ресурсов делятся на условные категории, обычно в зависимости от типа разрабатываемого приложения. Например, в категории Java размещаются обычные Java-проекты, Java-классы, интерфейсы, пакеты и т.д., а в категории Web – Web-проекты, HTML-страницы, JSP-страницы и т.д.

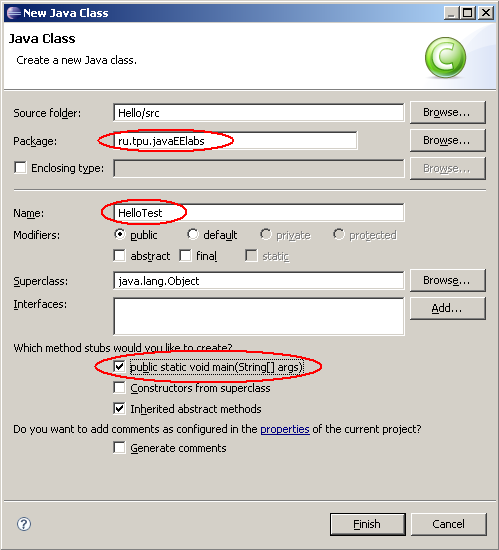
1. Закройте перспективу Welcome, появившуюся на экране после первого запуска Eclipse.
2. Выберите в меню File/New/Other.
3. Выберите категорию Java, а в ней JavaProject и нажмите Next.



1. Введите имя проекта – Hello. Затем, если в разделе JRE не указана или указана неверная настройка JRE, выберите Configuredefault…Нажмите Finish.



1. В результате создания проекта автоматически открывается перспектива Java, в левой части окна открывается представление PackageExplorer, в котором отражается новый проект Hello, содержащий подкаталогsrc, предназначенный для хранения исходных файлов программы
2. Далее, нам необходимо добавить новый ресурс проекта – обычный Java-класс:
3. В представлении PackageExplorer нажмите правой кнопкой мыши на значок проекта Hello и выберите New/Class. То же самое действие можно выполнить с помощью команды File/New/Other, затем выбрав категорию Java и типресурса– Class.
4. В появившемся окне укажите имя пакета и имя нового класса. Имя пакета может состоять из нескольких уровней, разделенных точкой. Также выберите флажок publicstaticvoidmain(…) для автогенерации main-метода.
5. Нажмите Finish



1. В методе main запишите команду вывода сообщения на экран:

public static void main(String[] args) {

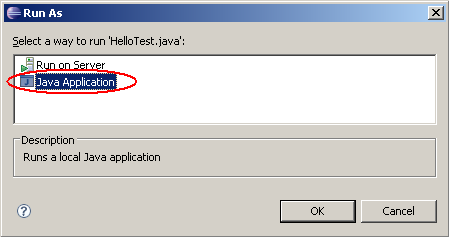
System.out.println("Hello Java!!!");

}

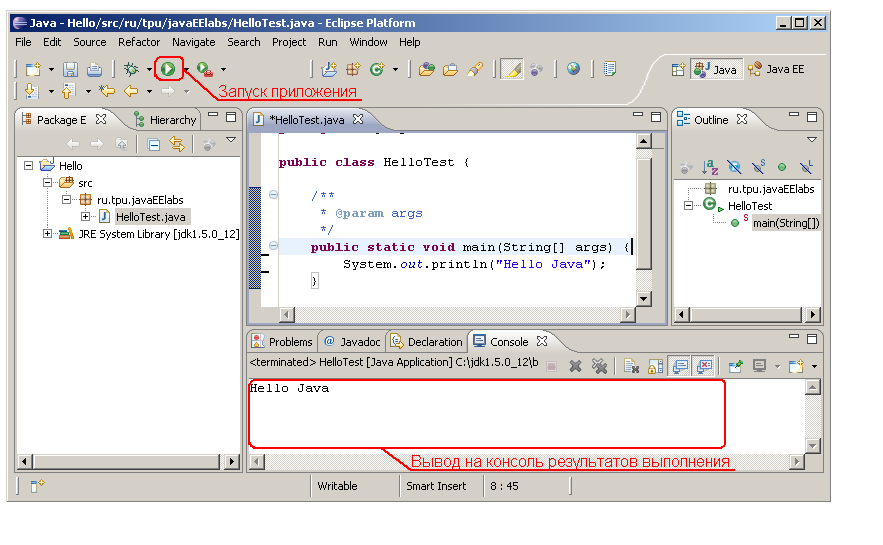
1. Сохраните файл, нажав на клавиши Ctrl-S.

**5. Запуск приложения**

1. Для запуска приложения выделите файл Hello.java в представлении PackageExplorer, выберите команду меню Run/Run или нажмите на соответствующую кнопку панели инструментов. Запустить приложение можно также нажав правой кнопкой мыши на Hello.java в представлении PackageExplorer и выбрав команду RunAs/JavaApplication.
2. В появившемся окне RunAs… выберите JavaApplication и нажмите OK.

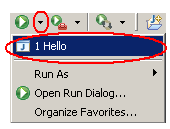


Если появилось окно сохранения файлов перед запуском SaveandLaunch, выберите флажок Alwayssaveresources… и нажмите ОК.



1. После запуска приложения в нижней части экрана в окне Console отображается результат выполнения программы.

При первом запуске приложения создается конфигурация запуска, которая используется при последующих запусках программы. В дальнейшем, при выполнении лабораторных работ, рабочая область будет содержать несколько проектов, и соответственно несколько Java-классов, содержащих метод main(). Для того, чтобы выбрать приложение для запуска, следует выделить соответствующий java-файл в представлении PackageExplorer и выполнить Run/Run, либо нажав правой кнопкой мыши на java-файл в представлении PackageExplorer и выполнить команду RunAs/JavaApplication.Для повторных запусков приложения можно нажать на кнопку Run панели инструментов для запуска последнего запускавшегося приложения, либо открыть список конфигураций запуска и выбрать другое приложение:



**6. Автогенерация иавтоформатирование кода**

Ниже приведены некоторые полезные функции по генерации стандартных конструкций в Java-коде, а также автоматическое форматирование кода. Функции автогенерации кода сосредоточены в меню SourceиначинаютсясословаGenerate(кромеOverride/ImplementMethods). Эти функции отрабатывают только на коде активного редактора java-кода.

Override/ImplementMethods переопределить/реализовать метод базового класса/интерфейса

GenerateGettersAndSetters создатьметодыgetи/илиset

для полей класса

GenerateDelegateMethods создать методы-делегаты для

методов объекта,

определенного в классе

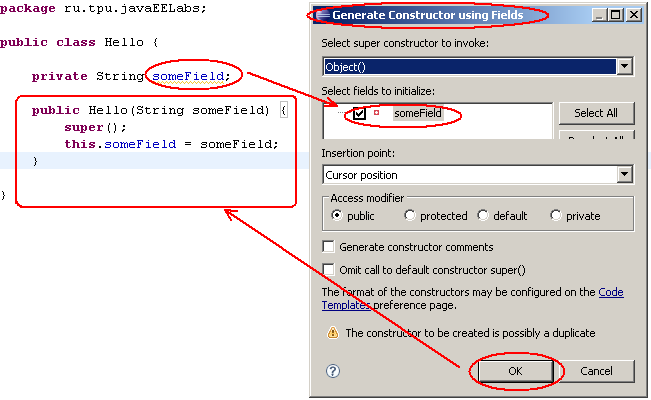
GeneratehashCode() andequals() переопределить и сгенерировать содержимое методов hashCode() и equals()

GenerateConstructorsusingFields создать конструктор, принимающий поля класса в качестве параметров

GenerateConstructorsusingSuperclass создатьконструктор

используя суперкласс

Ниже приведен пример работы команды GenerateConstructorsusingFields:



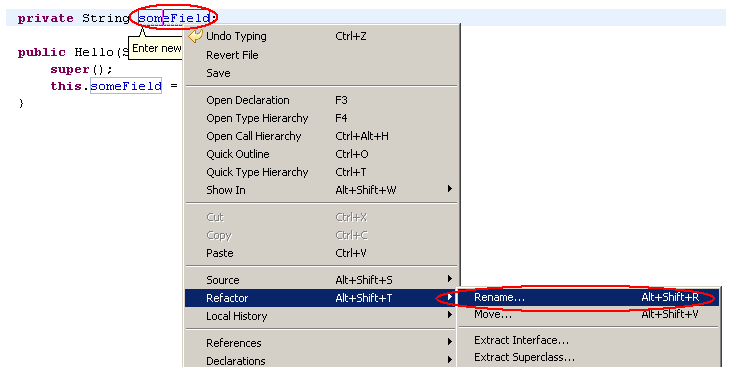
Автоформатирование кода выполняется с помощью команды меню Source/Format, либо с помощью комбинации клавиш Ctrl-Shift-F. Автоформатирование выполняется над текущим файлом открытым в окне редактора.

**7. Рефакторинг**

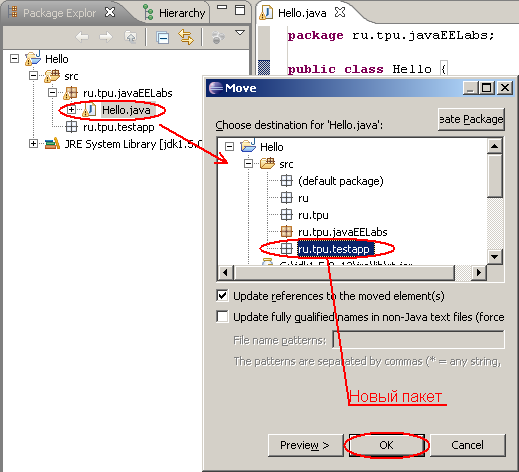
Рефакторинг – это безопасное изменение структуры ресурсов проекта. Действия по рефакторингу включают в себя изменение названий классов, полей и методов, изменение принадлежности классов пакетам, извлечение интерфейсов из описания класса и пр. При выполнении действия рафактор (refactor) обеспечивает сохранение целостности кода.Например, изменение имени класса приводит к изменению этого имени во всех элементах кода, где используется этот класс.

Команды рефакторинга вызываются над определенным элементом – классом, полем, методом и т.д. Функции рафактора (refactor) удобнее всего вызывать путем выбора элемента, вызова щелчком правой кнопкой мыши контекстного меню и выбора в нем одной из команд меню Refactor. Также команды рефакторинга можно вызвать, используя основное меню Refactor.

Ниже приведен пример вызова команды переименования над полем класса. Перед вызовом меню курсор редактора предварительно устанавливается на изменяемое поле someField.



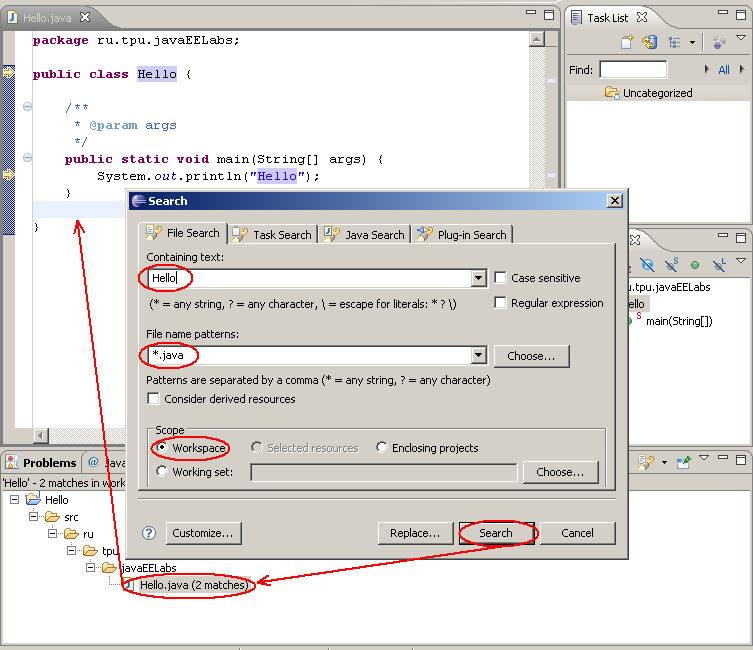
Следующий пример демонстрирует возможность смены пакета, в котором размещается класс. Действие рефактораMove вызывается над классом Hello в представлении PackageExplorer. В результате выполнения действия класс перемещается из пакета ru.tpu.javaEELabs в пакет ru.tpu.testapp.



**8. Поиск**

EclipseWorkbench позволяет производить поиск файлов и строк, а также замену (без соблюдения целостности проекта) одной строки на другую. Рассмотрим пример поиска файлов в рабочей области, содержащих слово “Hello”.

1. Нажмите на кнопку поиска 12 или сочетание клавиш Ctrl-H
2. В появившемся окне вберите закладку FileSearch и в поле ContainingText введите строку Hello.
3. Опция FileNamePatterns определяет шаблон имен файлов для поиска. Изменить шаблон можно путем прямого ввода нового значения в поле, либо с помощью кнопки Choose. По умолчанию выбран шаблон \*.java, т.е. все файлы с расширением java.
4. Опция Scope определяет область поиска. Убедитесь, что в этом разделе выбран пункт Workspace (по всей рабочей области).
5. Нажмите Search.
6. В результате выполнения поиска отображается представление Search, в котором показаны результаты поиска. Щелчок на найденный файл открывает его в окне редактора.



**10. Основные горячие клавиши**

Ниже приведен список некоторых горячих клавиш EclipseWorkbench:

Ctrl-N создать новый ресурс

Ctrl-S сохранить файл, открытый в активном редакторе

Ctrl-Shift-S сохранить все файлы проекта

Ctrl-F11 запуск последней конфигурации

F11 запуск последней конфигурации

в режиме отладки

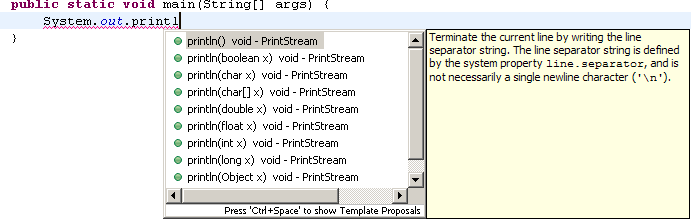
Ctrl-Shift-F автоформатирование текущего файла

Ctrl-H вызов окна сложного поиска/замены

Ctrl-F вызов окна простого текстового   
поиска/замены по текущему файлу

Ctrl-Shift-O автоматический импорт класса

Ctrl-Space вызов контекстной подсказки редактора



Ctrl-щелчок мыши на классе, ссылке или вызове метода осуществляет переход к объявлению этого элемента.

**Контрольные вопросы**