Практические занятия в компьютерном класса

Занятие №1 «Программы простой структуры»

Вводное занятие. Техника безопасности. Разработка программ простой структуры в интегрированной среде программирования **Eclipse**. Знакомство с основами ввода / вывода, применение математических функций.

Задание №1

Написать программу, которая выводит на экран строку текста — приветствие:

Привет, группа ТЯ-31!

Замечание: каждый студент пишет название своей группы.

Задание №2

Написать программу, которая запрашивает у пользователя значение переменной строкового типа (фамилию, имя и отчество), значение переменной целого типа (курс) и вещественного типа (средний балл). Вывести на экран полученную от пользователя информацию.

Приведем возможный вариант диалога:

Ваше ФИО: *Иванов Иван Иванович*
Курс: *3*
Средний балл: *98.96*

Иванов Иван Иванович
Вы учитесь на 3 курсе
Ваш средний балл: 98.96

Замечание: курсивом отмечены ответы пользователя.

Задание №3

Написать программу, которая получает от пользователя значение вещественных переменных x, y, z и вычисляет значение выражения:

$$a=\frac{\sqrt{|x-1|}-\sqrt[3]{|y|}}{1+\frac{x^2}{2}+\frac{y^2}{4}},\ b=x\left(\operatorname{arctg}(z)+\mathrm{e}^{-(x+3)}\right).$$

Замечание: для вычислений организовать статический импорт пакета с математическими функциями; результат расчетов вывести на экран с шестью цифрами после запятой.

Тестовые значения:

$x=1.1$, $y=2.2$, $z=3.3$ $\qquad$ $a=-0.349685$, $b=1.42245$.

Рекомендации по выполнению заданий

Перед выполнением заданий нужно создать на логическом разделе жесткого диска, предназначенном для сохранения файлов пользователя (диск **D** или **E** в зависимости от *ПК*) папку для своих файлов. Затем можно запускать **Eclipse** для работы. Ярлык для запуска **Eclipse** находится в правом верхнем углу рабочего стола.

Для того, чтобы лучше организовать свои файлы, будем называть проекты Theme1, Theme2, Theme3 … в который будем создавать пакеты classWork1, classWork2, classWork3 …, соответственно.

Основы работы в Eclipse

Eclipse — один из лучших и популярных инструментов для *Java*, созданных за последние годы. SDK **Eclipse** представляет собой интегрированную среду разработки (*IDE, Integrated Development Environment*) с открытым исходным кодом. В начале своего существования **Eclipse** появилась как коммерческий продукт, но в ноябре 2001 г. его исходные коды были опубликованы.

Поддержкой и разработкой **Eclipse** в настоящее время занимается организация *Eclipse Foundation* и сообщество **Eclipse**, информацию о которых можно найти на официальном сайте в сети Интернет http://www.eclipse.org.

При первоначальном знакомстве со средой IDE **Eclipse** она может показаться несколько сложной для неподготовленного пользователя. Для того чтобы понять основы работы с системой, нужно хорошо разобраться с основными концепции среды: *рабочее пространство*, *инструменты*, *компоновки*, *редакторы* и *представления*.

Рабочее пространство

В простейшем случае *рабочее пространство (workspace)* — это папка для проектов пользователя. В этой папке располагаются все файлы, относящиеся к проектам. Все, что находится внутри этого каталога, считается частью рабочего пространства.

Инструментальные средства Eclipse

Инструментальные средства **Eclipse** становятся доступны сразу после запуска приложения. Это по существу сама платформа с набором различных функциональных возможностей главного меню, где прежде всего выделяется набор операций по управлению проектом. Фактическая обработка, как правило, осуществляется дополнениями (плагинами), например редактирование и просмотр файлов проектов осуществляется JDT, и т. д. К инструментам (*workbench*) относится набор соответствующих *редакторов* и *представлений*, размещенных в рабочей области **Eclipse**. Для конкретной задачи определенный набор редакторов и представлений называют *перспективой* или *компоновкой*.

Компоновки

Компоновка (*perspective*) — это набор представлений и редакторов, расположенных в том порядке, который вам требуется. В каждой компоновке присутствует свой набор инструментов, некоторые компоновки могут иметь общие наборы инструментов. В определенный момент времени активной может быть только одна компоновка. Переключение между различными компоновками осуществляется нажатием клавиш <Ctrl+F8>. Используя компоновки, вы можете настроить свое рабочее пространство под определенный тип выполняемой задачи. Мы будем использовать компоновки, связанные с программированием на *Java*, такие, как: *Debug*, *Java Browsing*, *Java*.

В **Eclipse** имеется также возможность создавать свои компоновки. Открыть компоновку можно командой *Window* | *Open Perspective*.

Редакторы

Редакторы представляют собой программные средства, позволяющие осуществлять операции с файлами (создавать, открывать, редактировать, сохранять и др.).

Представления

Представления по существу являются дополнениями к редакторам, где выводится информация сопроводительного или дополнительного характера, как правило, о файле,

находящемся в редакторе. Открыть представления можно командой *Window* | *Show View.*

Проект

Проект (*project*) представляет собой набор файлов приложения и сопутствующих дополнений.

Дополнение

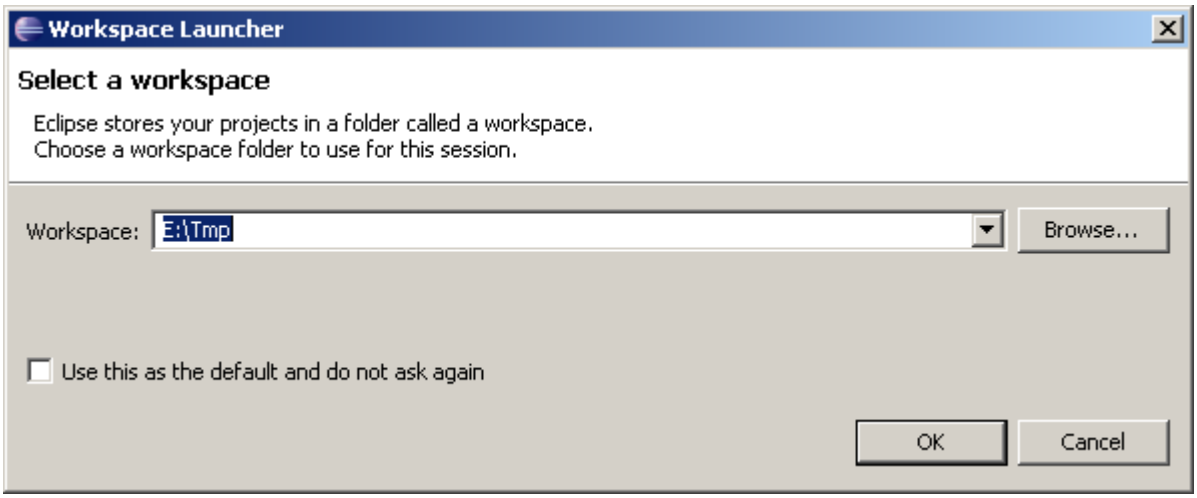
Дополнением (*plug-in*) называют приложение, которое дополнительно может быть установлено в **Eclipse**. Примером дополнения может выступать *JDT* (*Java Development Tools* — набор плагинов платформы **Eclipse** для организации *Java IDE*.)

Мастера

Мастер — это программное средство, которое помогает пользователю в настройках и проведении сложной операции. В **Eclipse** имеется , большое количество различных мастеров, которые делают работу пользователя в системе удобной и эффективной, беря часть рутинных операций на себя. Примером мастера может выступить мастер создания нового класса, который помогает пользователю в таких операциях, как создание нового файла в нужной директории, создание начального кода класса, автоматическая расстановка модификаторов и т.д. .

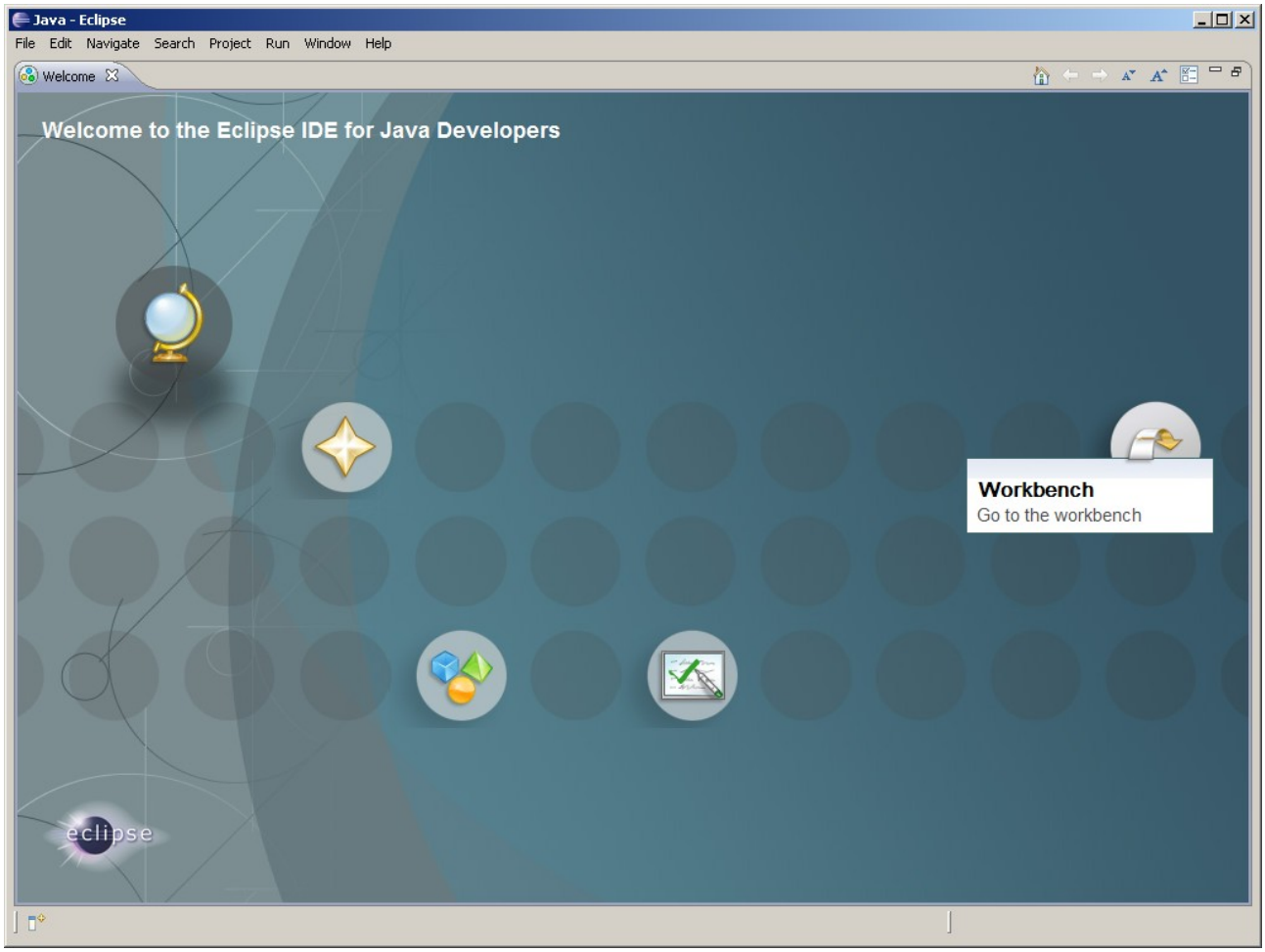
Первый запуск Eclipse

Первое окно, которое отобразится на экране — это диалоговое окно выбора рабочего пространства (*workspace*):



В данном окне нужно выбрать папку, в которой будут храниться файлы ваших проектов *Java*. Если вы уже работали за этим *ПК,* то выбрать ранее установленную папку можно с помощью раскрывающегося списка. Если вы первый раз работаете за этим компьютером, или хотите создать новое рабочее пространство в новой папке, то нажимаете кнопку *Browse...*, вызываете стандартное диалоговое окно выбора папок и указываете нужную. Путь к выбранной папке отображается в поле списка.

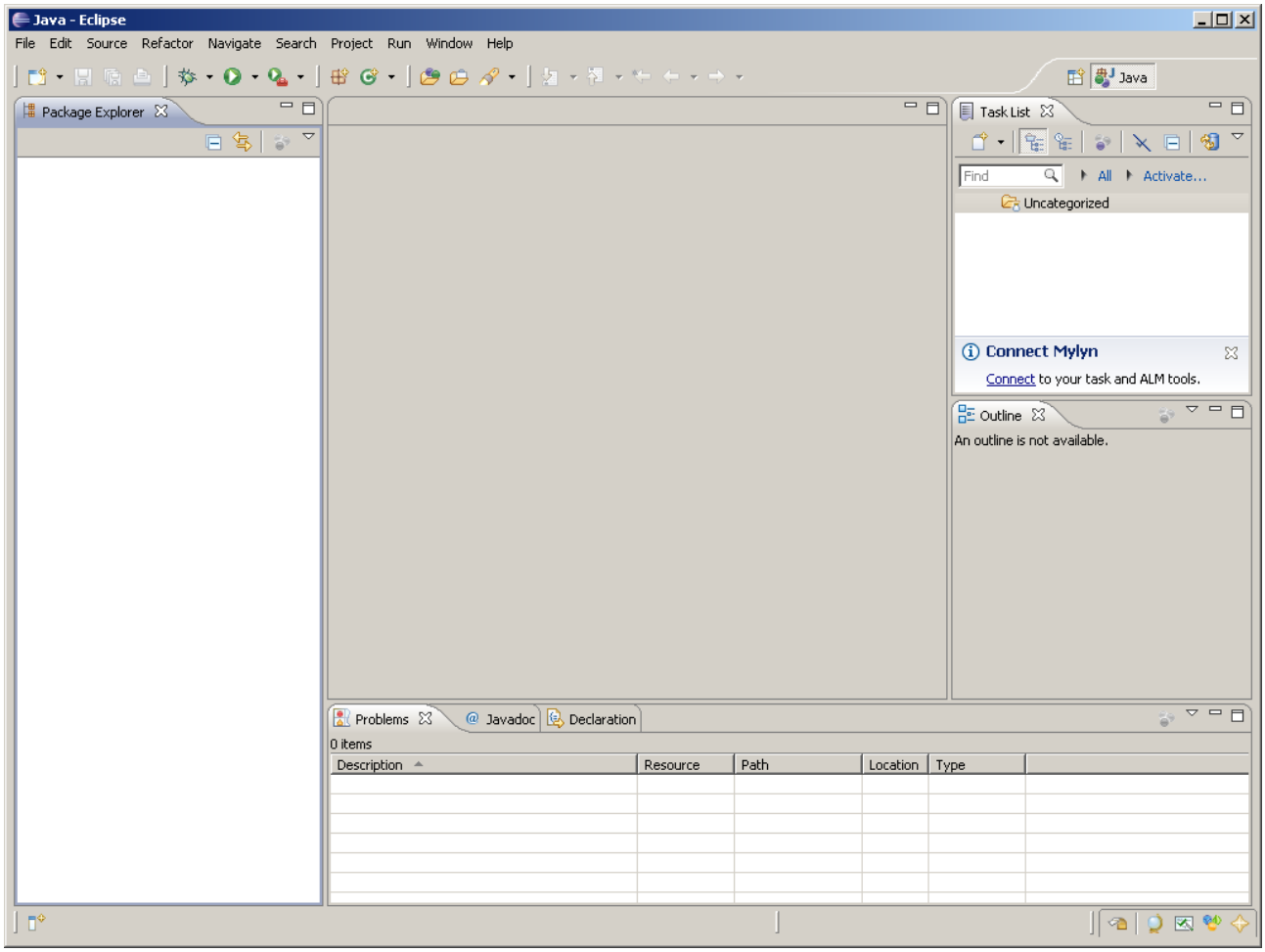
Установка флажка (*Use this as a default and do not ask again*), который расположен сразу под списком *workspace* в этом диалоговом окне, даст указание оболочке использовать данное рабочее пространство по умолчанию, и больше при начале работы с **Eclipse** данное окно появляться не будет. В этом случае, для изменить рабочее пространство можно будет только после старта системы и загрузки рабочего пространства по умолчанию. Для этого нужно выбрать команду меню *File* | *Switch Workspace*. Поэтому, данный флажок устанавливать НЕ НУЖНО!!! Файлы каждого проекта — исходные тексты программ (.java, .jsp), файлы настроек (.xml) и прочие данные будут храниться в указанном вами рабочем пространстве. В качестве рабочего пространства укажите вашу рабочую папку.



После того, как будет указан путь к рабочему пространству и нажата кнопка *OK*, на экран будет выведена страница приветствия с пятью графическими кнопками:

- *Overview* — обзор, содержащий ссылки на обучающие интернет-ресурсы **Eclipse**;
- *Tutorials* — уроки, содержит несколько примеров создания простейших приложений *Java*;
- *What's new* — «что нового», содержит обзор основных нововведений;
- *Samples* — примеры, содержит несколько примеров разработки, которые должны быть предварительно установлены для того, чтобы их можно было просмотреть;
- *Workbench* — «рабочий стол» — это рабочая область программиста.

Для того, чтобы приступить к работе, нажмите кнопку *Workbench*. На рисунке со страницей приветствия выделена именно эта графическая кнопка. По умолчанию откроется универсальный рабочий стол.

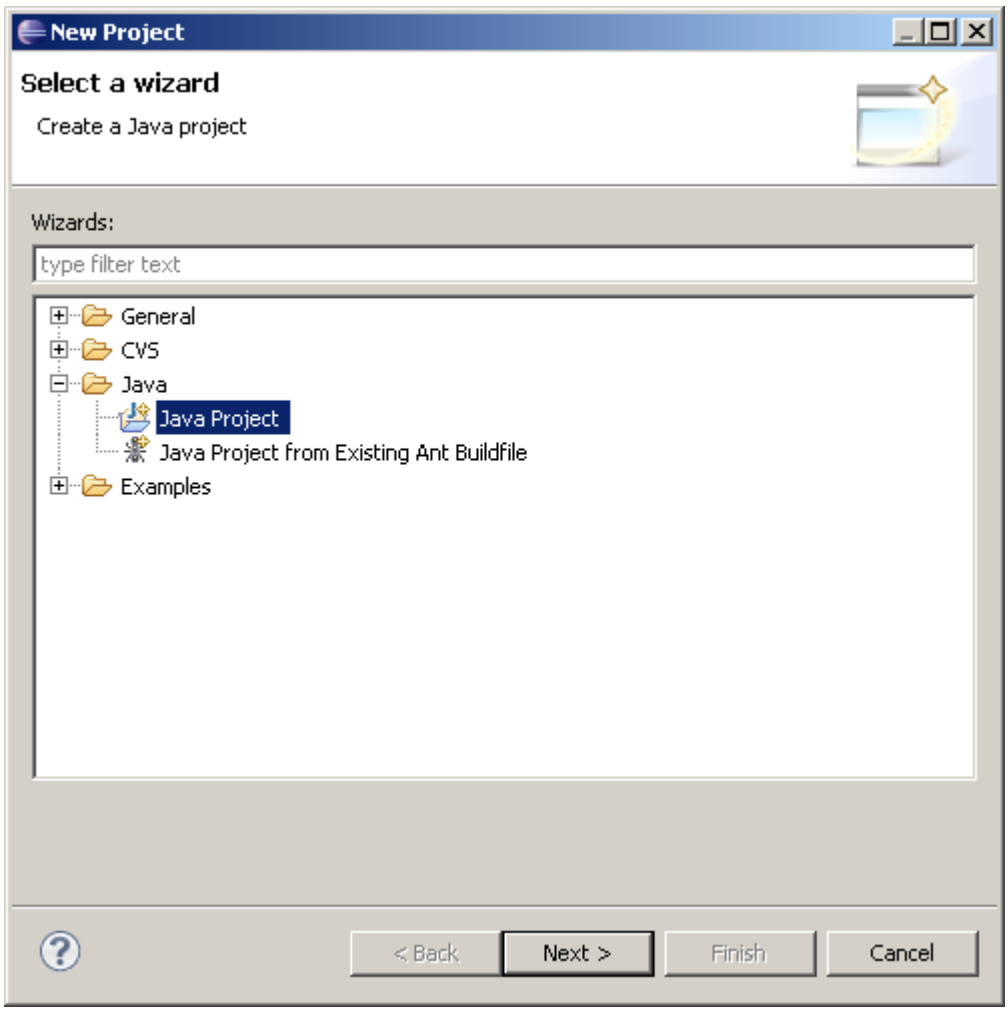


Этот универсальный рабочий стол находится в режиме работы *Java*. В правом верхнем углу рисунка расположена метка, которая отображает текущий режим рабочего стола (*Java*). Для того, чтобы переключиться в другой режим, нужно нажать кнопку, находящуюся слева от выделенной метки, и в раскрывающемся списке выбрать нужный режим. В нашем случае нужно выбрать *Java*. Обычно этот режим выбран по умолчанию.

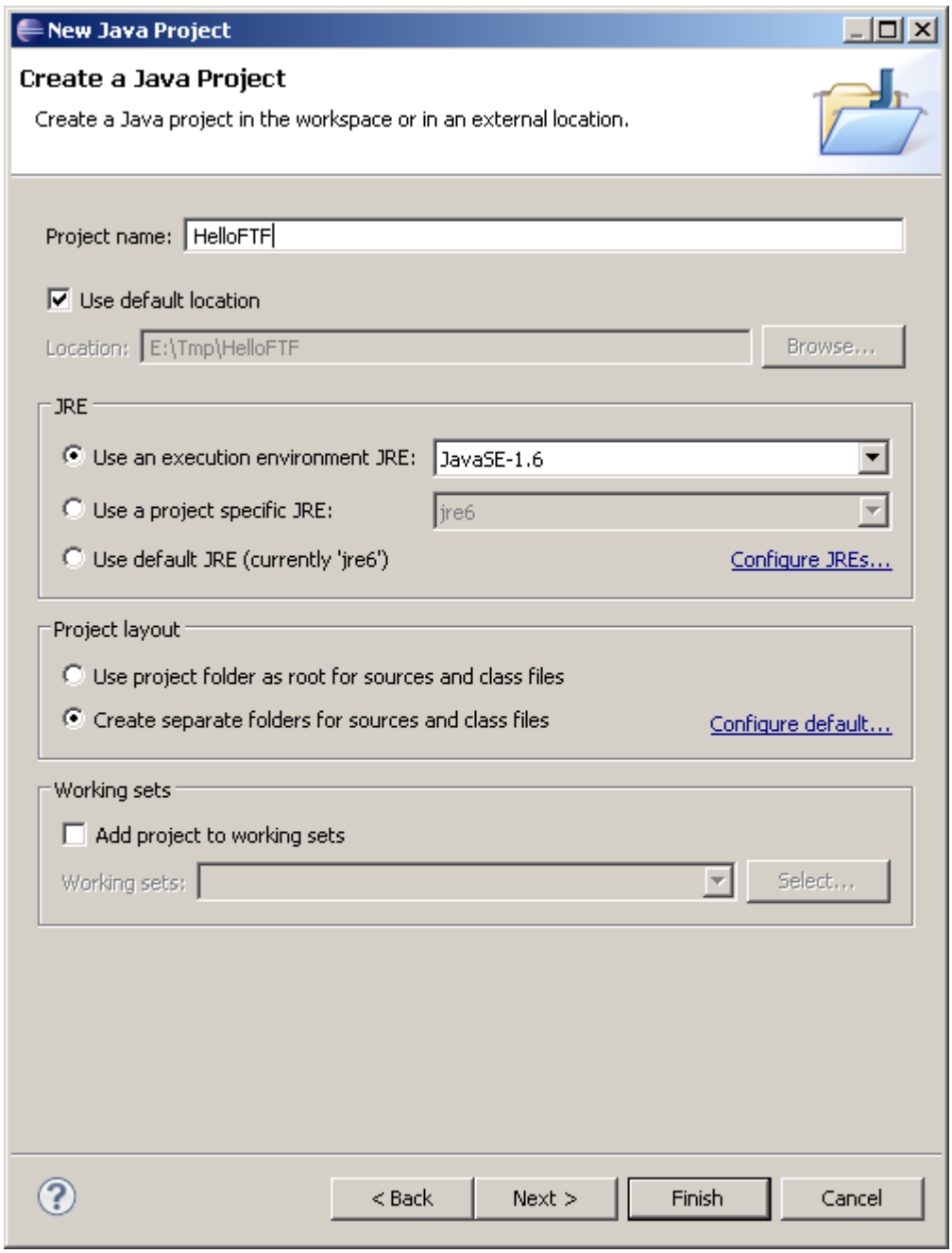
Создание проекта

Проект *Java* представляет собой каталог на жестком диске, содержащий библиотеки *JRE*, исходные коды, картинки и прочее. Управлять содержимым пакета лучше с помощью среды. Хотя никто не запрещает изменять содержимое файлов и структуру каталогов в проекте с помощью сторонних редакторов и браузеров, все же рекомендуется это делать средствами *IDE*, поскольку возможны серьезные ошибки и сбои при компиляции и работе приложения в случае использования других программ.

Для создания нового проекта выберем команду меню *File* | *New* | *Project*. Запустится мастер создания нового проекта.

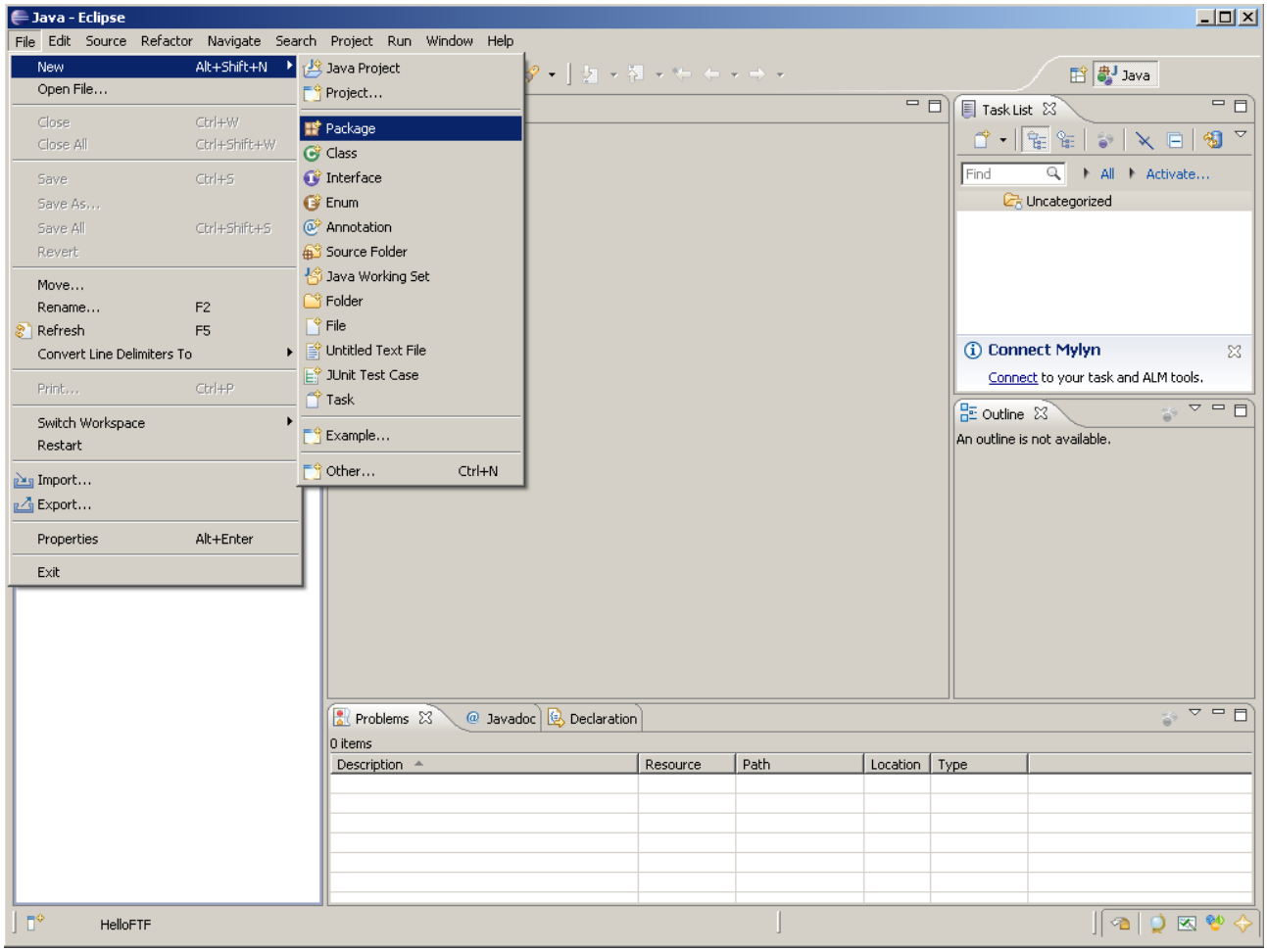


В первом окне нужно выбрать нужный тип проекта *Java Project* (*Проект Java*). Нажав кнопку *Next >* переходим к следующему окну мастера проектов. Это будет окно с начальными настройками проекта:

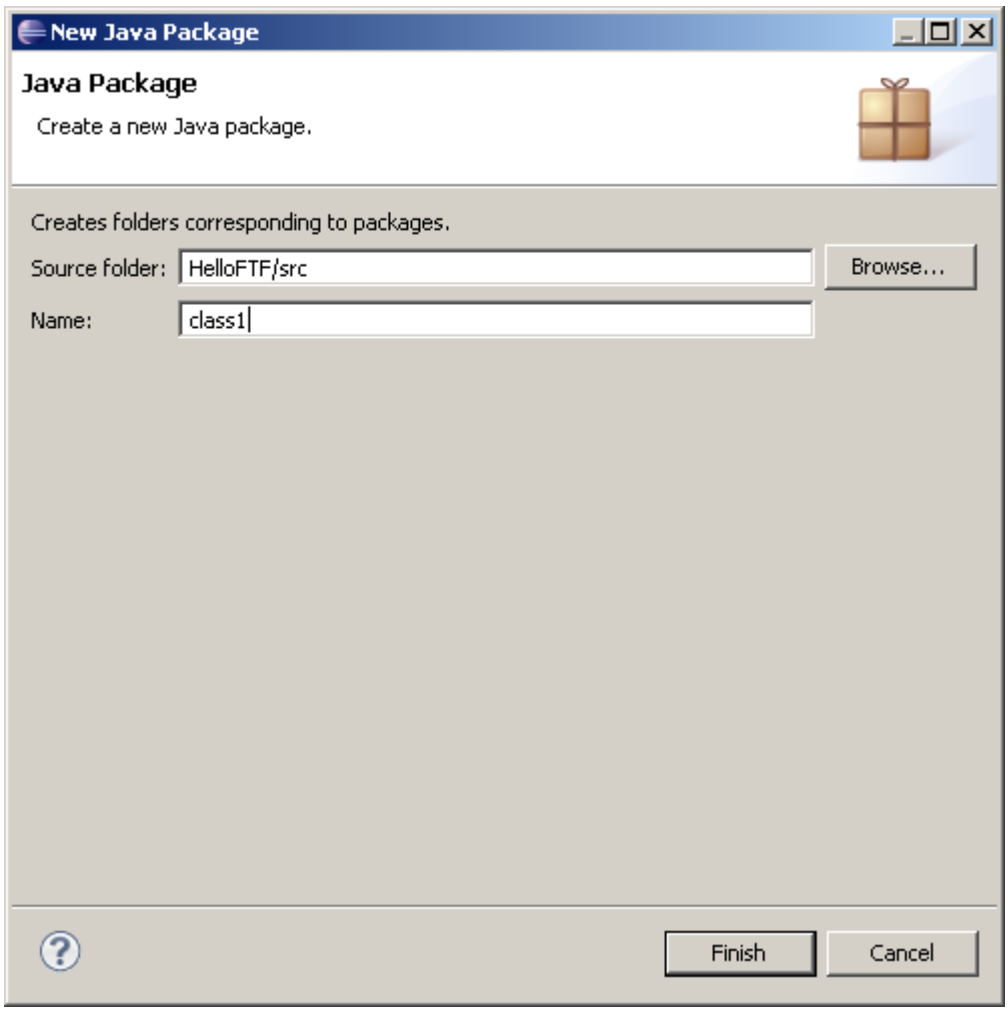


В этом окне в поле *Project name* (*Имя проекта*) нужно введисти название нового проекта Theme1. Остальные значения можно оставить в положении по умолчанию и для завершения работы мастера нажимаем кнопку *Finish*.

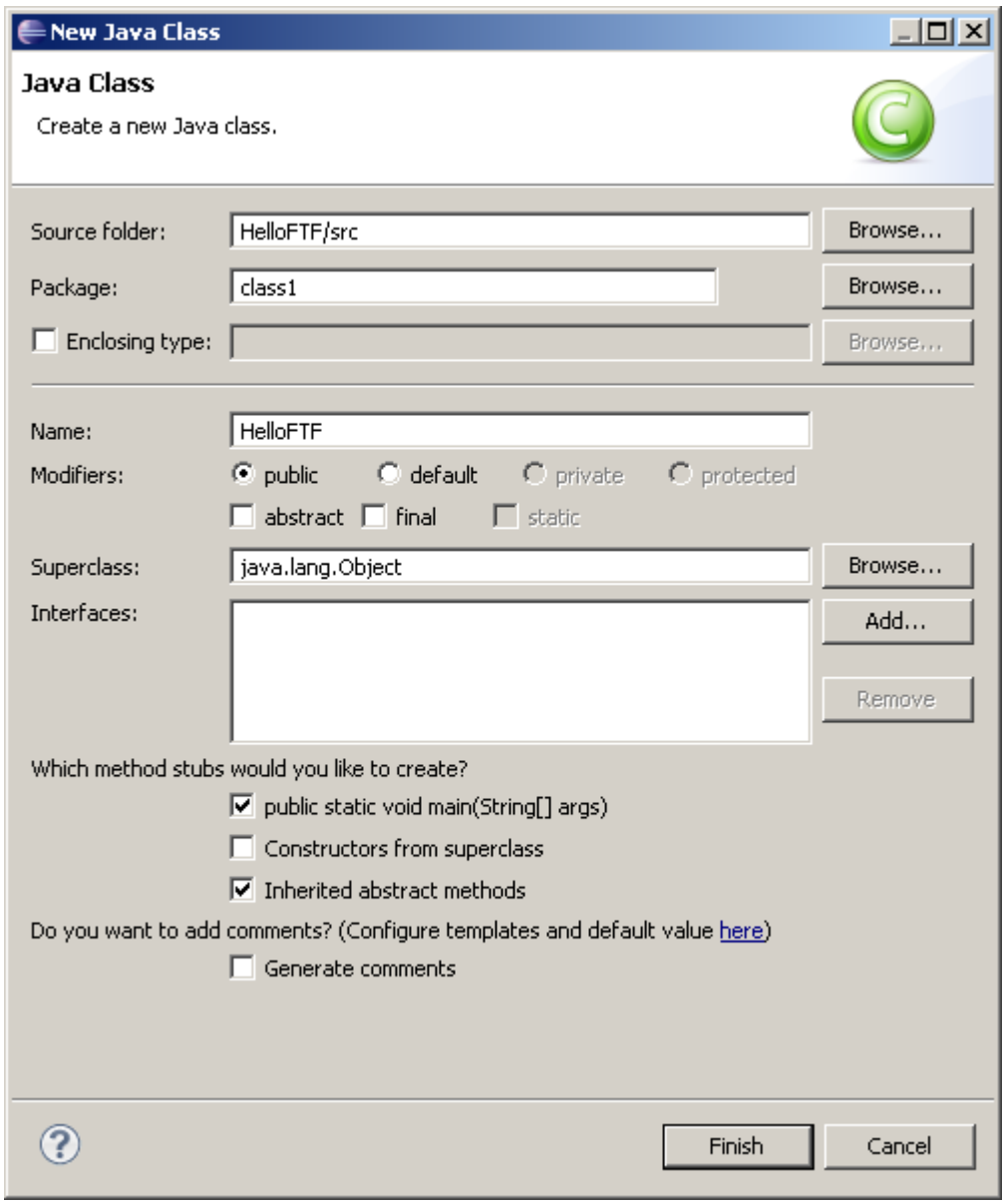
Обычно, все классы в *Java* хранятся в каком-либо *пакете*. С помощью мастера создадим новый пакет. Для этого выберем команду меню *File* | *New* | *Package*.



На экран будет выведено окно мастера создания нового пакета.



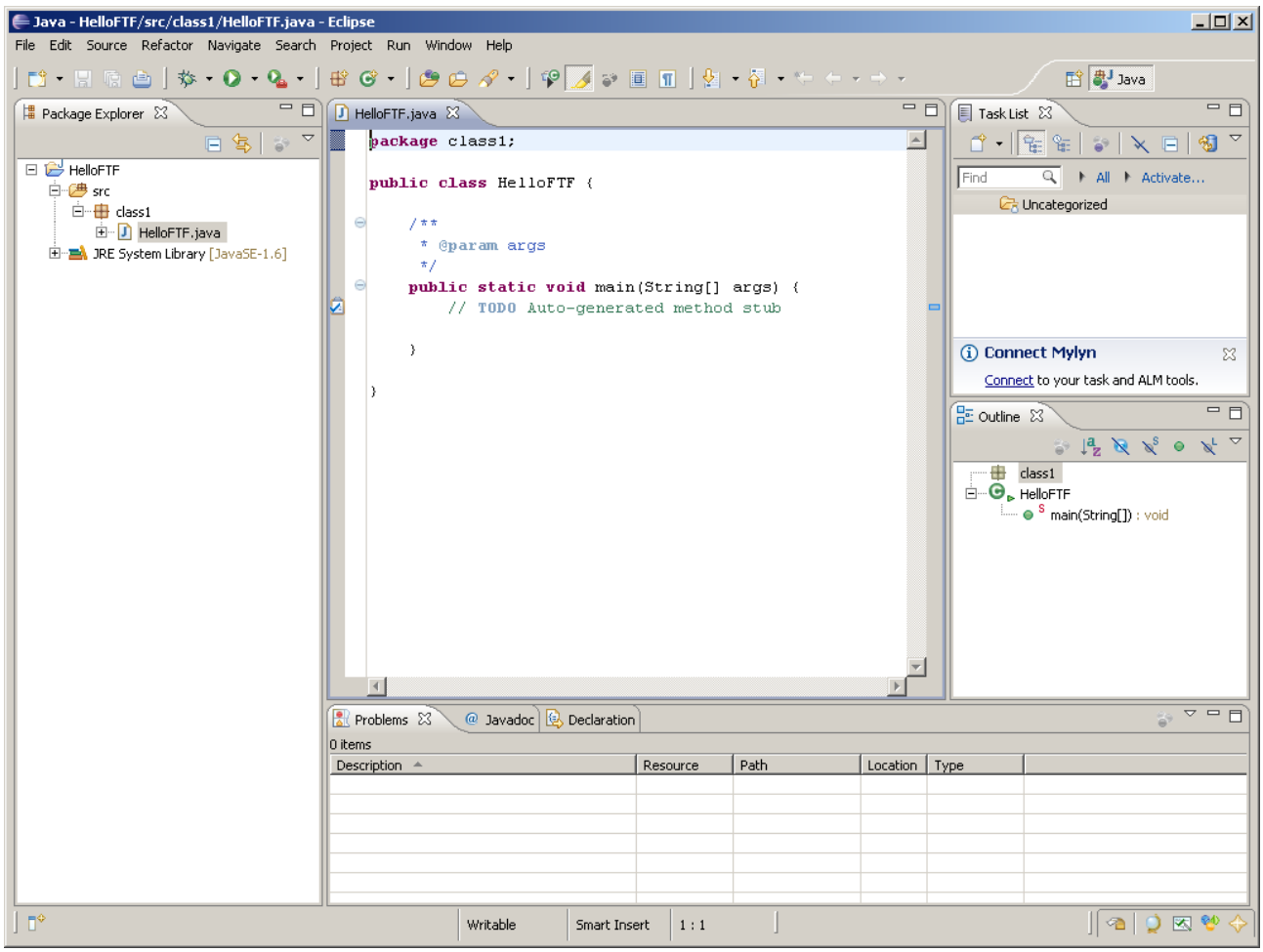
В поле *Name* нужно указать имя создаваемого пакета classWork1 и нажать кнопку *Finish*. После этого в созданном пакете создадим наш класс: в главном меню программы выберем команду *File* | *New* | *Class*. В выведенном на экран диалоговом окне мастера создания классов укажем нужную информацию:



В поле *Name* указываем имя создаваемого класса `Task1`. Здесь можно также выбрать набор модификаторов для класса, класс-родитель и интерфейсы класса. В нижней части окна имеется группа из трех флажковых кнопок, которая позволяет автоматически добавить некоторые основные методы:

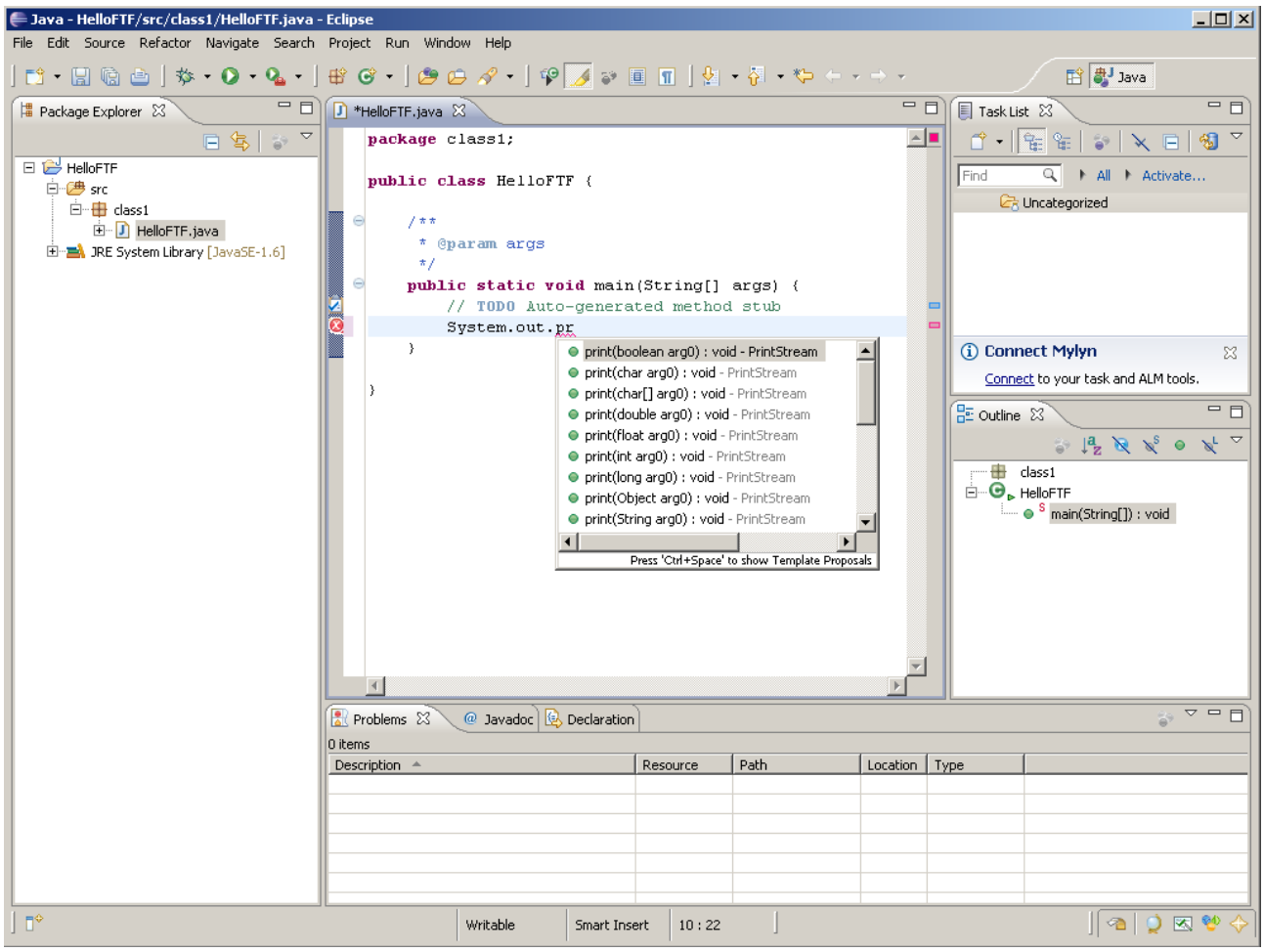
- *public static void main(String[] args)* — уже известный нам метод main();
- *Constructor from superclass* — конструктор класса-родителя;
- *Inherited abstract methods* — наследованные абстрактные методы.

Выбираем автоматическое создание функции main(). Остальное оставляем в значении по умолчанию и нажимаем кнопку *Finish*. Теперь в нашем проекте есть один класс `Task1`, после чего можно приступить к созданию программы.



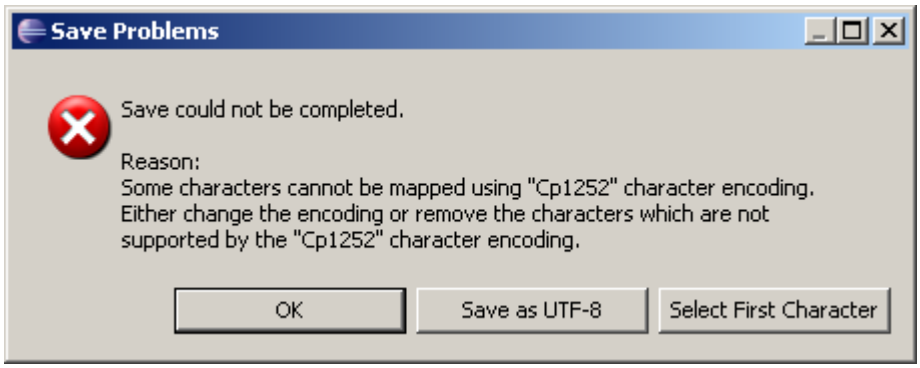
При наличии нескольких проектов лучше закрыть предыдущий проект перед открытием или созданием нового проекта. Для закрытия проекта нужно выделить в окне *Package Explorer* нужный проект, затем в строке меню выбрать команду *Project* | *Close*, либо, щелкнув правой кнопкой мыши на проект в *Package Explorer*, в раскрывающемся меню выбрать пункт *Close*.

Во время написания исходного кода программы среда предлагает варианты окончания кода.

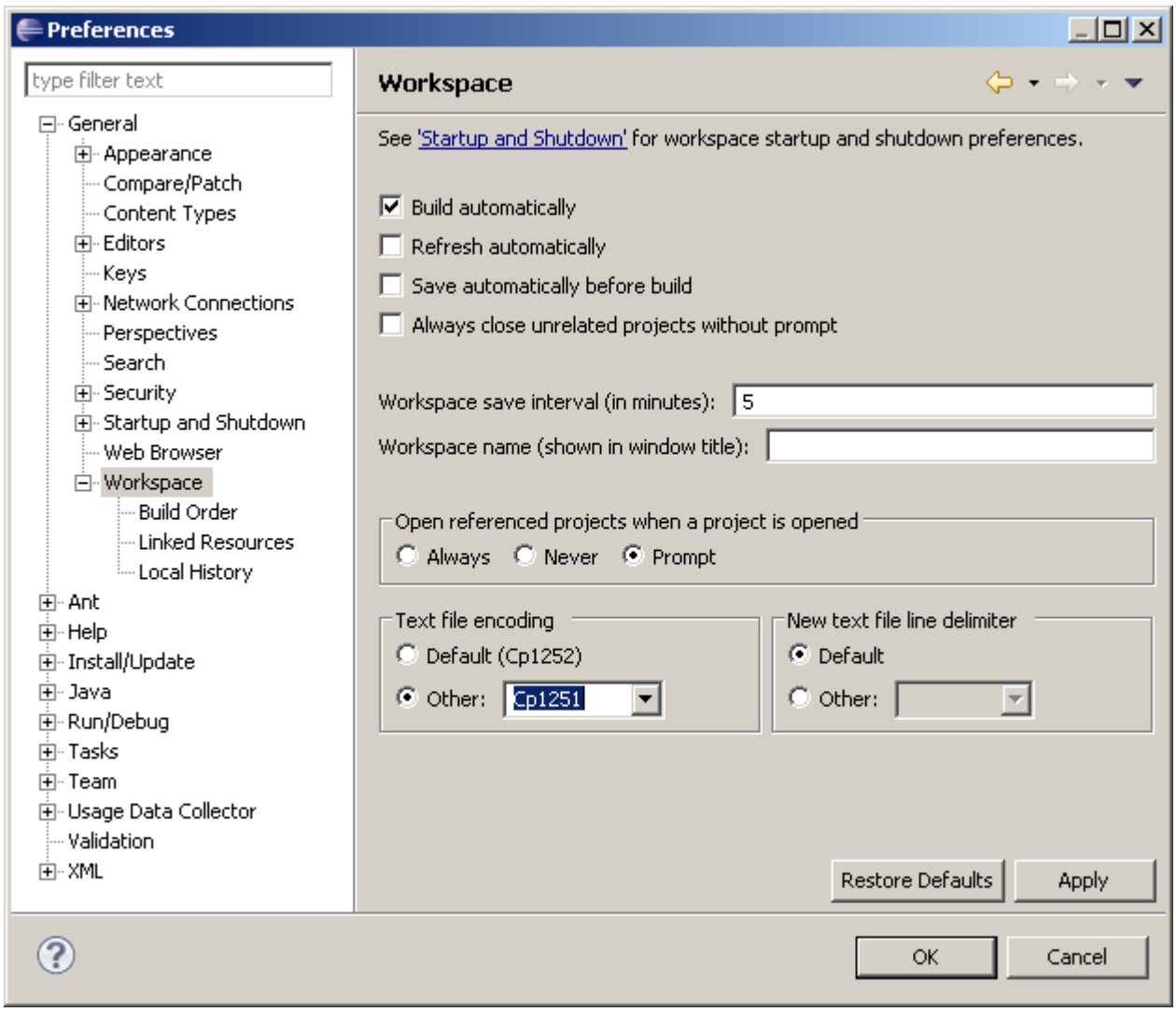


Во всплывающем окне можно выбрать нужный вариант и нажатием клавиши *Enter* подтвердить свой выбор.

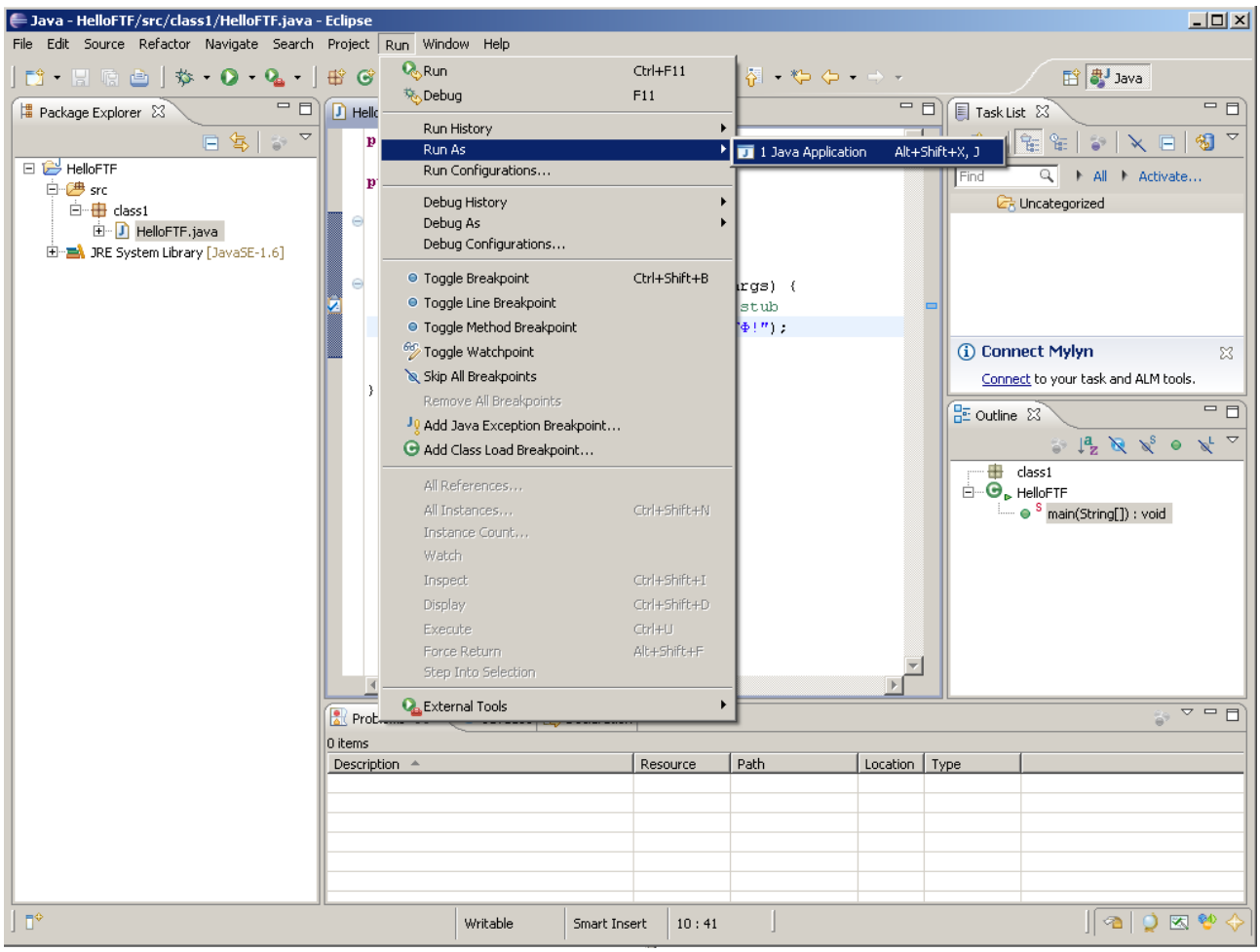
Иногда, когда операционная система настроена не оптимальным образом, система не может сохранить исходный текст программы с русскими буквами. При этом на экран выводится диалоговое окно с описанием проблемы:

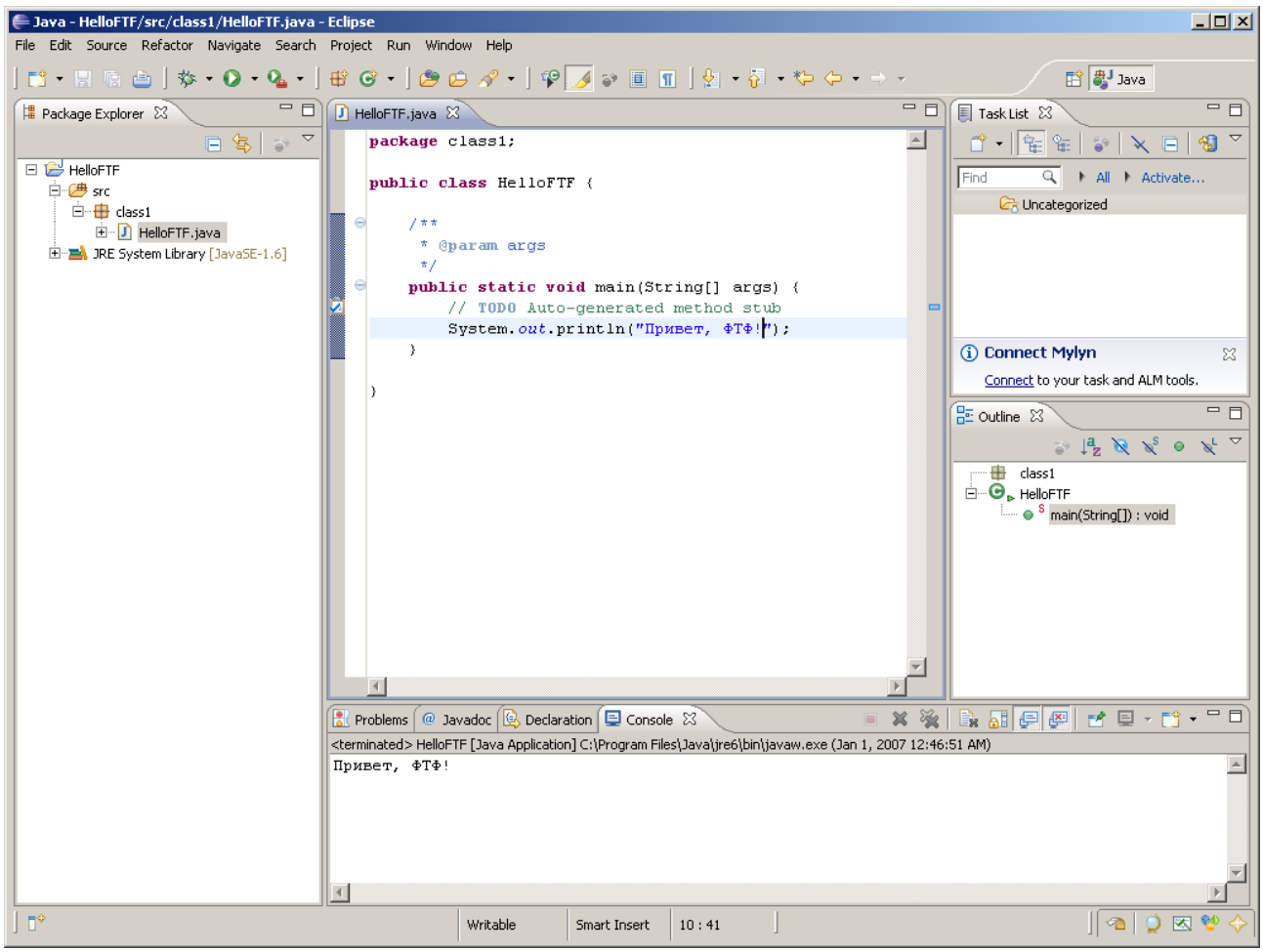


В этом случае нужно изменить настройки рабочего пространства. Для этого выбираем команду главного меню *Window* | *Preferences* | *General* | *Workspace* и указываем нужную кодовую таблицу:

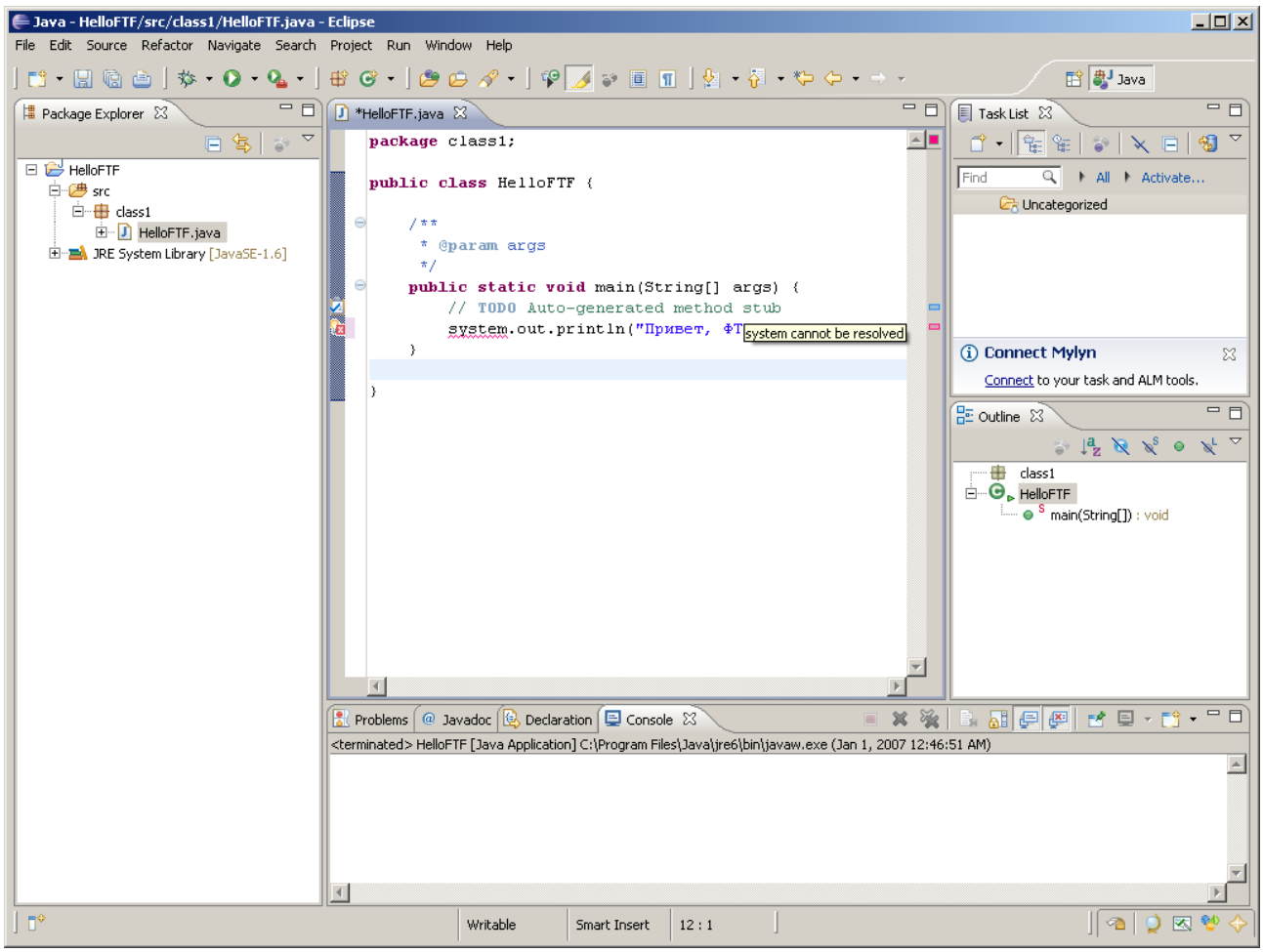


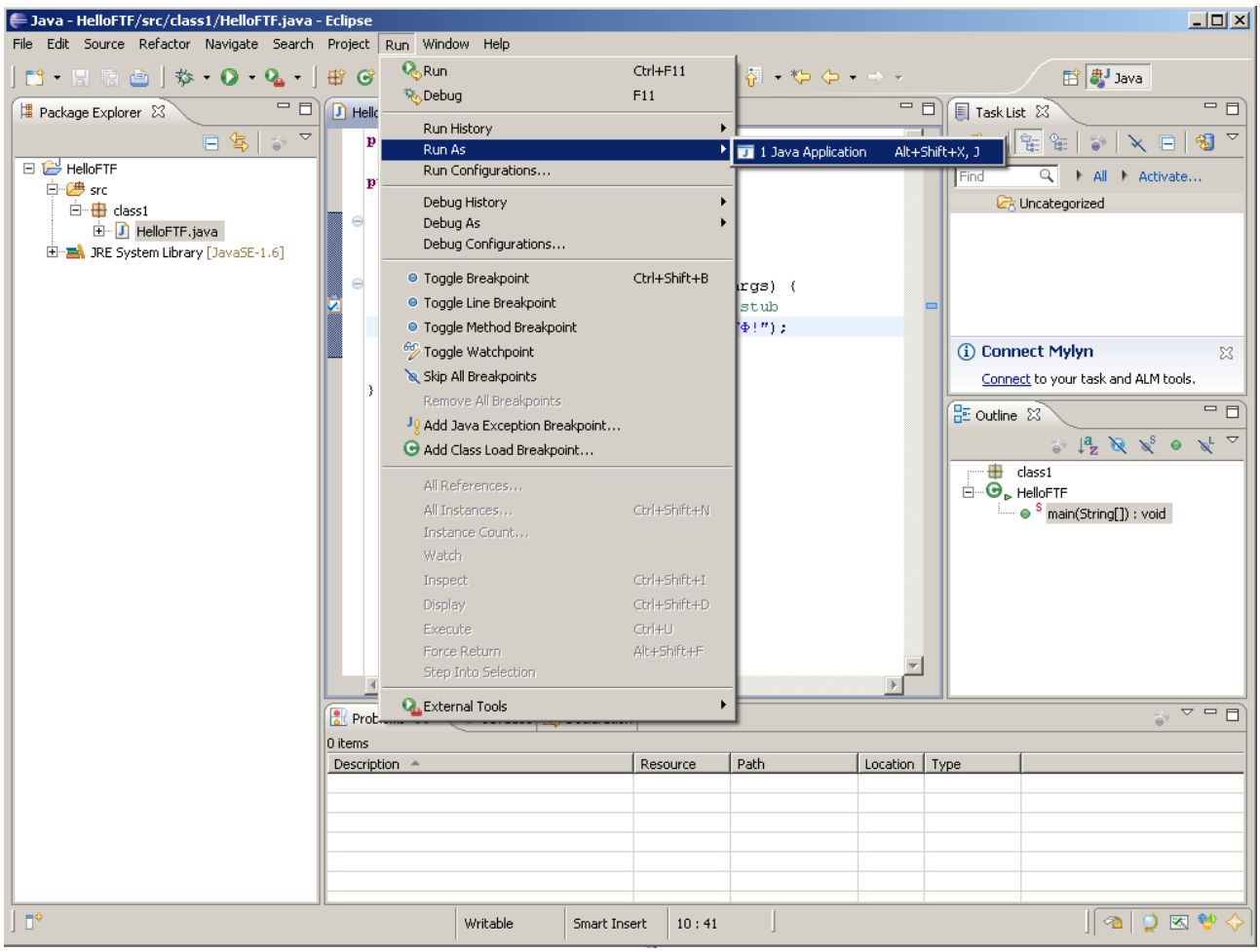
При сохранении документа среда разработки сразу компилирует исходный текст вашего класса. Для запуска приложения нужно выбрать пункт главного меню программы *Run | Run As | Java Application*. Результат работы программы будет отображаться во вкладке *Console*, расположенной в нижней части главного окна программы.


Java - HelloFTF/src/class1/HelloFTF.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Run Ctrl+F11
Debug F11
Run History
Run As
Run Configurations...
Debug History
Debug As
Debug Configurations...
Toggle Breakpoint Ctrl+Shift+B
Toggle Line Breakpoint
Toggle Method Breakpoint
Toggle Watchpoint
Skip All Breakpoints
Remove All Breakpoints
Add Java Exception Breakpoint...
Add Class Load Breakpoint...
All References...
All Instances... Ctrl+Shift+N
Instance Count...
Watch
Inspect Ctrl+Shift+I
Display Ctrl+Shift+D
Execute Ctrl+U
Force Return Alt+Shift+F
Step Into Selection
External Tools
1 Java Application Alt+Shift+X, J
Package Explorer
HelloFTF
src
class1
HelloFTF.java
JRE System Library [JavaSE-1.6]
Task List
Find
All
Activate...
Uncategorized
Connect Mylyn
Connect to your task and ALM tools.
Outline
class1
HelloFTF
main(String[]) : void
0 items
Description
Resource
Path
Location
Type
Writable
Smart Insert
10 : 41



Если при создании программы были допущены синтаксические ошибки, информация о них отображается во вкладке *Problems*, а также слева и справа от того места в тексте программы, где компилятор увидел ошибку. Можно подвести указатель мыши к значку, соответствующему ошибке и в виде подсказки появится краткая информация об ошибке. См. рисунок ниже.


Java - HelloFTF/src/class1/HelloFTF.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package Explorer
HelloFTF
src
class1
HelloFTF.java
JRE System Library [JavaSE-1.6]
*HelloFTF.java
package class1;
public class HelloFTF {
/**
* @param args
*/
public static void main(String[] args) {
// TODO Auto-generated method stub
system.out.println("Привет, ФТ
system cannot be resolved
}
}
Task List
Find
All
Activate...
Uncategorized
Connect Mylyn
Connect to your task and ALM tools.
Outline
class1
HelloFTF
main(String[]) : void
Problems
Javadoc
Declaration
Console
<terminated> HelloFTF [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Jan 1, 2007 12:46:51 AM)
Writable
Smart Insert
12 : 1


Java - HelloFTF/src/class1/HelloFTF.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Run Ctrl+F11
Debug F11
Run History
Run As
Run Configurations...
Debug History
Debug As
Debug Configurations...
Toggle Breakpoint Ctrl+Shift+B
Toggle Line Breakpoint
Toggle Method Breakpoint
Toggle Watchpoint
Skip All Breakpoints
Remove All Breakpoints
Add Java Exception Breakpoint...
Add Class Load Breakpoint...
All References...
All Instances... Ctrl+Shift+N
Instance Count...
Watch
Inspect Ctrl+Shift+I
Display Ctrl+Shift+D
Execute Ctrl+U
Force Return Alt+Shift+F
Step Into Selection
External Tools
1 Java Application Alt+Shift+X, J
Package Explorer
HelloFTF
src
class1
HelloFTF.java
JRE System Library [JavaSE-1.6]
Task List
Find
All
Activate...
Uncategorized
Connect Mylyn
Connect to your task and ALM tools.
Outline
class1
HelloFTF
main(String[]) : void
0 items
Description
Resource
Path
Location
Type
Writable
Smart Insert
10 : 41

Занятие №2 «Управление процессом вычислений»

Знакомство с основными вычислительными конструкциями: выбор, циклы разного типа и соответствующими операторами языка *Java* их реализующими. Работа с примитивными типами данных. Разбор основных вычислительных алгоритмов: вычисления сумм, произведений, пределов с заданной точностью.

Задание №1

Вычислить значение выражения:

$$\begin{cases} -a x^3 - b & \text{при } x+c<0 \text{ и } a\neq 0 \\ \dfrac{x-a}{x-c} & \text{при } x+c>0 \text{ и } a=0 \\ \dfrac{x}{c}+\dfrac{c}{x} & \text{в остальных случаях} \end{cases}$$

где a, b, c — действительные числа.

Результат должен быть действительным выражением, если:

$(\lfloor a \rfloor \bmod 2) > (\lfloor b \rfloor \bmod 2)$ или $(\lfloor b \rfloor \bmod 2) > (\lfloor c \rfloor \bmod 2)$, $\lfloor\ \rfloor$ — целая часть числа,

и целым в противном случае.

Значения a, b, c, x ввести с клавиатуры.

Задание №2

Проверить, попадает ли точка с заданными с клавиатуры координатами (x, y) в заштрихованную область на графике (см. рис. 1). Заштрихованная область задается уравнениями:

$$f_1(x)=x,\ f_2(x)=2-\sqrt{4-(x-4)^2},\quad f_3(x)=-\sqrt{4-(x-2)^2},$$

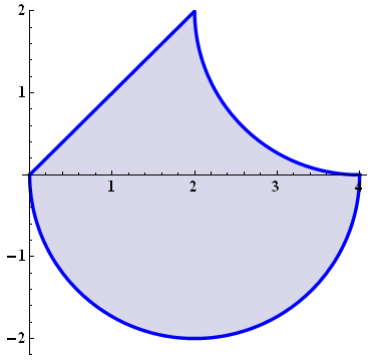


Рисунок 1. Область для Задания № 2

Задание №3

Программа генерирует целое псевдослучайное число в диапазоне $[1;100]$ — оценку студента. Вывести на экран эту оценку и ее буквенный эквивалент по Болонской системе, если:

1 – 49	Fx
50 – 59	E
60 – 69	D
70 – 79	C

80 — 89	B
90 — 100	A

Задание №4

Вычислить значение выражения:

$$\prod_{i=2}^{100} \frac{i+1}{i+2}$$

Значение для проверки: 0.0294118

Задание №5

Вычислить значение выражения:

$$\sum_{i=1}^{n} (-1)^i \frac{x^i}{i!}$$

Значения n, x ввести с клавиатуры. Осуществить проверку введенного значения для переменной n ($n \geq 1$)

Значение для проверки: $x = 0.1$, $n = 25$ сумма равна -0.0951626

Задание №6

Вычислить значение выражения:

$$\sum_{i=1}^{20} \prod_{j=1}^{50} \frac{i^2}{j^2}$$

Значение для проверки: 13.7856

Задание №7

Вычислить для заданного аргумента x значение функции ошибок $\mathrm{erf}(x)$ с заданной точностью ε.

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{n!(2n+1)}$$

С помощью программы проверить, что $\mathrm{erf}(-x) = -\mathrm{erf}(x)$.

Значение для проверки: $\mathrm{erf}(0.95) = 0.820891$

Задание №8

Дано натуральное число k, действительное число a. С заданной точностью ε вычислить значение $\sqrt[k]{a}$ по итерационной формуле:

$$x_0 = a,\ x_i = \frac{1}{k}\left((k-1)x_{i-1} + \frac{a}{x_{i-1}^{k-1}}\right).$$

После вычислений сравнить найденное значение с результатом, возвращаемым функцией возведения в степень.

Задание №9

Вычислить сумму цифр заданного с клавиатуры натурального числа n.

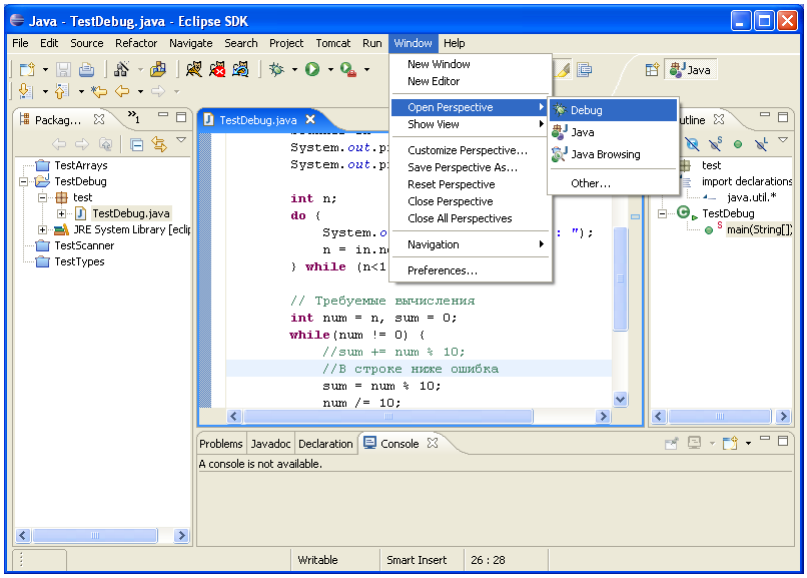
Задание №10

Пользователь вводит с клавиатуры два натуральных числа n_1 и n_2 $(n_1 < n_2)$. Вычислить все простые числа из заданного интервала.

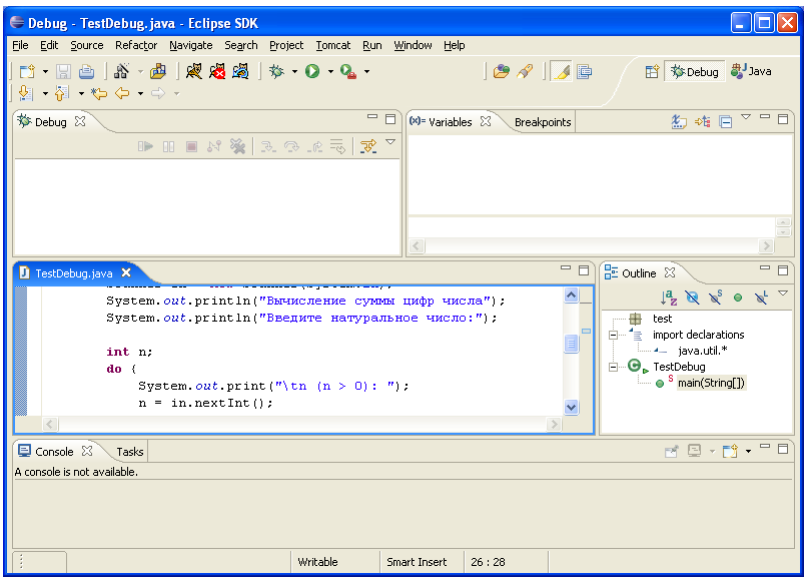
Отладка программ в среде Eclipse.

Во время разработки программ часто бывает так, что программа не имеет синтаксических ошибок и нормально понимается компилятором, но во время выполнения программы либо возникает ошибка, либо программа дает неверный результат. Для того, чтобы понять почему такое происходит и исправить ошибки необходимо воспользоваться отладчиком.

Для работы с отладчиком существует специальная компоновка (перспектива): *Debug*, которую можно активировать с помощью команды меню *Window* | *Open Perspective* | *Debug*.



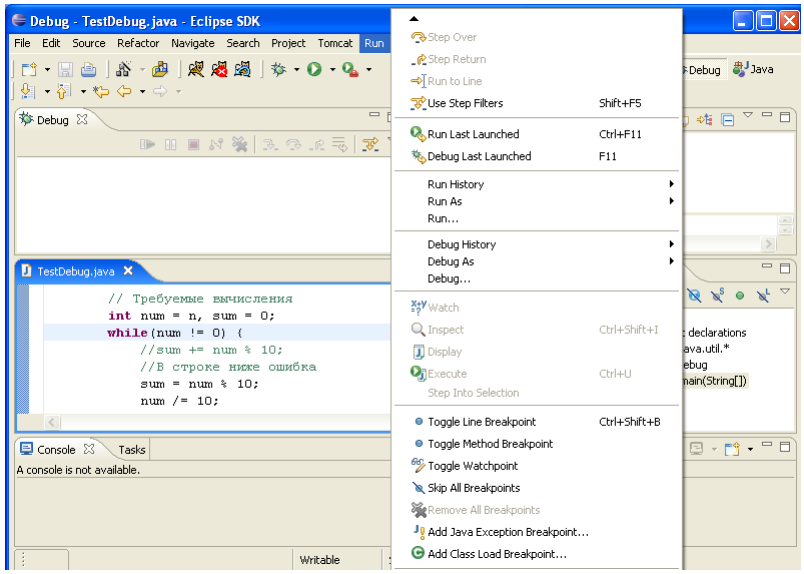
Доступ к этой перспективе можно получить с помощью кнопки переключения компоновок, расположенной рядом с кнопкой активной компоновки (на рисунке Java), выше представления *Outline*. Кроме того, компоновку *Debug* можно активизировать сразу вызвав отладчик. Для этого нужно нажать кнопку с изображением жука, расположенную на панели инструментов. После того, как компоновка *Debug* будет активизирована изменится набор представлений в главном окне программы.



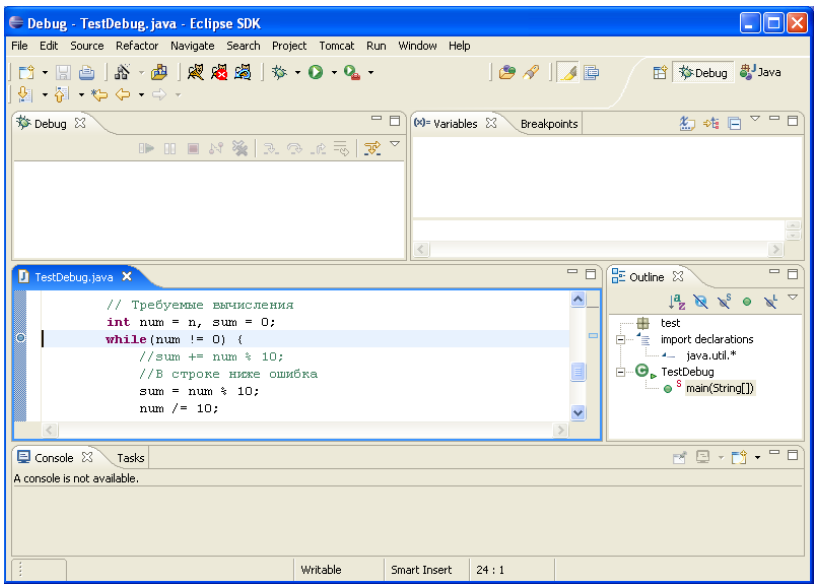
В компоновке *Debug* можно выделить такие основные представления:

- Окно *Debug*. В данном окне отображаются используемые в отладке элементы, а так же панель управления процессом отладки.
- Окно состояния переменных и точек прерывания. В этом окне на вкладке *Variables* представлен список переменных, задействованных в текущей точке прерывания. Данное окно очень важно в процессе отладки, так как с его помощью можно узнать значения всех переменных в каждой точке прерывания, отследить поведение изучаемого объекта и изменение его свойств в процессе выполнения программы. На вкладке *Breakpoints* содержится список установленных программистом точек прерывания. Точки прерывания можно как отключать, так и включать в процессе отладки, устанавливая или удаляя маркер напротив нужной точки.
- Панель *Outline*, редактор кода и консоль, все как и в компоновке *Java*.

Затем, в исходном тексте программы нужно расставить точки прерывания. Для этого нужно установить текстовый курсор в нужной строке исходного текста программы и вызвать команду меню *Run | Toggle Run Breakpoint*, либо нажать комбинацию клавиш *Ctrl + Shift + B*.

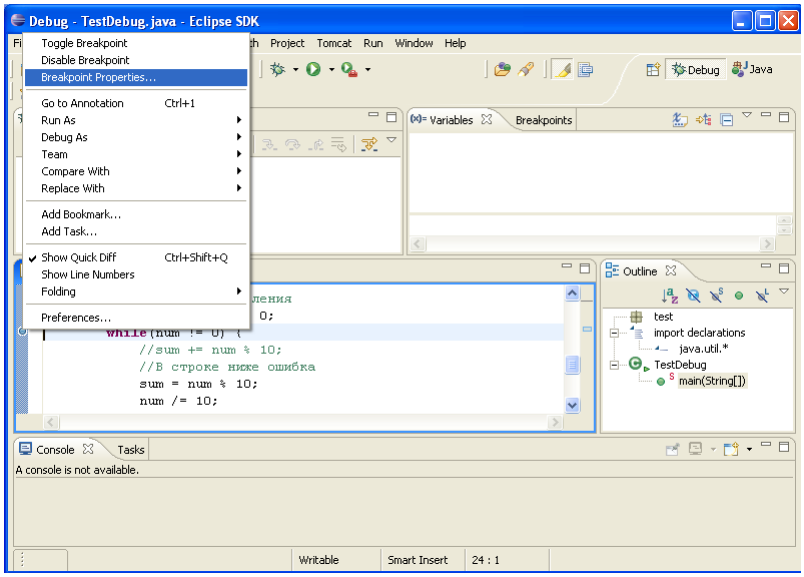


В результате, слева от выбранной стоки кода будет отображаться значок точки прерывания — графическая точка.

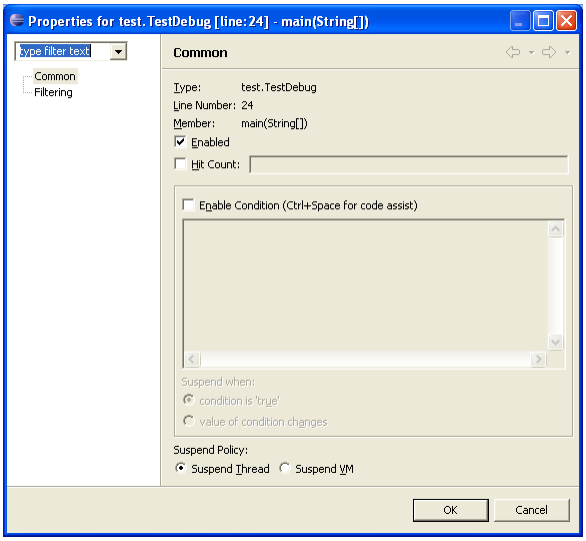


Во время отладки программы отладчик, последовательно выполняя все операторы программы, доходит до первой точки прерывания и приостанавливает выполнение программы. Программист может проконтролировать значения переменных, какие они имеют в момент прерывания работы программы. После того, как проанализировано состояние программы можно продолжить ее выполнение либо в пошаговом режиме, либо до следующей точки прерывания.

Точку прерывания можно отключить — для этого следует выбрать команду меню *Run | Disable Breakpoint*. При этом точка прерывания остается в указанном месте и просто не участвует в вычислениях. Для того, чтобы убрать точку прерывания следует, либо находясь на строке кода с точкой, либо вызвав контекстное меню для нужной точки прерывания выбрать еще раз команду меню *Toggle Breakpoint*. Точка прерывания будет убрана из текста программы. Для того, чтобы сразу убрать все точки прерывания из текста программы можно выполнить команду меню *Run | Remove All Breakpoints*.



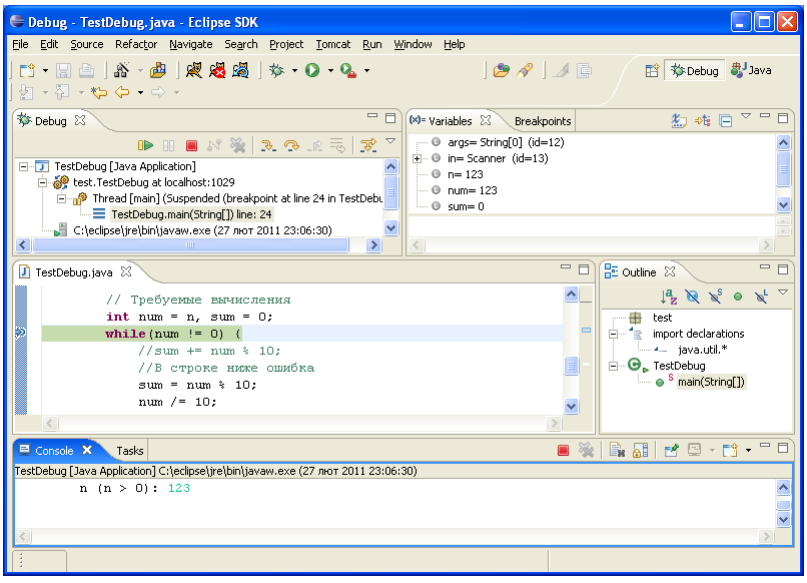
Кроме обычных, безусловных точек прерывания во время отладки можно создавать точки прерывания с дополнительными условиями. Для того, чтобы указать такое условие следует в контекстном меню для точки прерывания выбрать команду *Breakpoint Properties...*. На экран будет выведено диалоговое окно свойств выбранной точки прерывания.



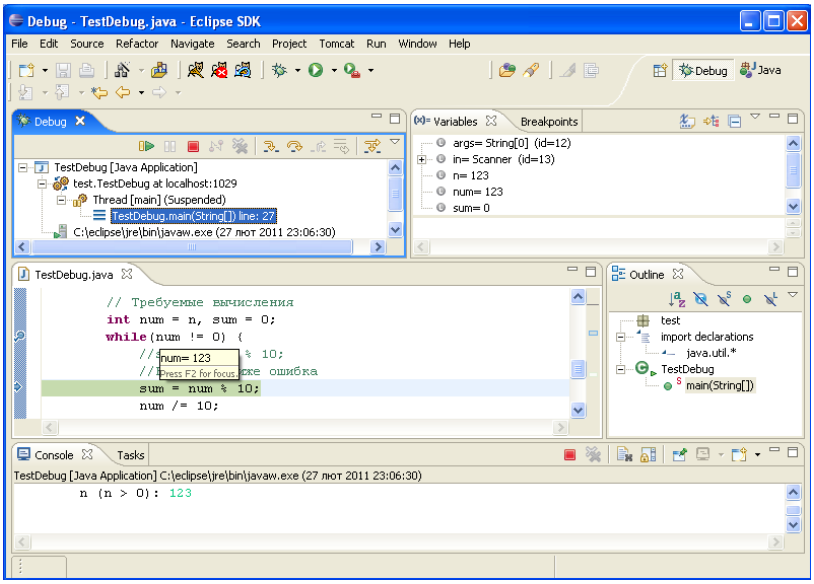
Для указания условия необходимо установить флажок *Enable Condition* и в поле ввода, расположенном ниже, указать необходимое условие (в условие могут входить переменные из исходного текста программы), при котором будет происходить останов на данной точке. Здесь же можно выбрать тип условия прерывания — под полем ввода находится две кнопки выбора — *condition is 'true'* и *value of condition changes*. При выборе первого, останов будет происходить каждый раз, когда введенное логическое выражение примет значение «*истина*», при выборе второго типа условия прерывания, останов будет происходить каждый раз, когда введенное значение или константа изменит свое значение.

С помощью параметра *Suspend Policy* можно установить политику прерывания. При выборе значения *Suspend Thread* останов будет происходить на уровне текущего потока *thread*, при выборе *Suspend VM* — на уровне самой виртуальной машины *Java*.

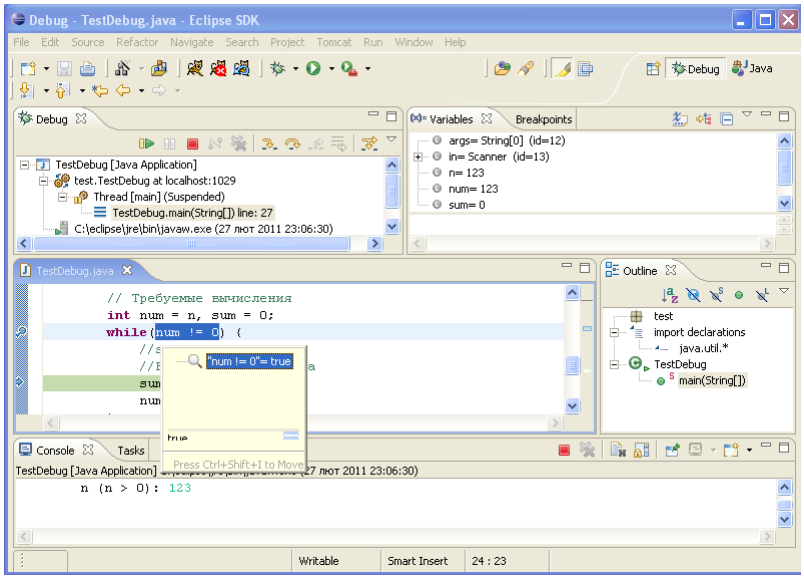
Кроме того, можно выбрать флажок *Hit count* и в расположенном рядом текстовом поле указать количество проходов по точке прерывания после которых она сработает.



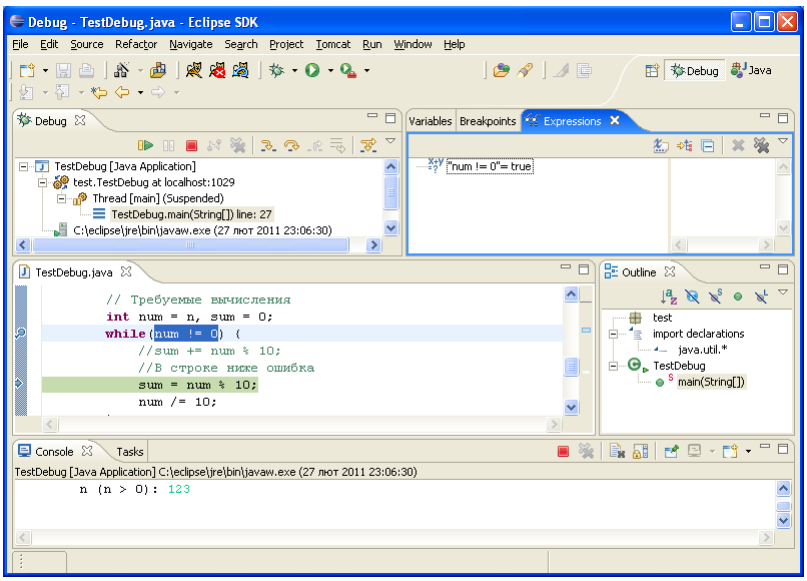
В отладчике **Eclipse** есть еще одна полезная функция, с помощью которой можно увидеть значения сложных выражений. Например, нужно проверить значение логического выражения (условия), при котором выполняется цикл. Для этого выделите нужный текст программы, в нашем случае `num != 0` и выберите команду меню *Run | Watch*. В окне просмотра значений переменной появится вкладка *Expressions* в которой отображается выбранное выражение и его значение в текущий момент времени.



Если же нужно просто просмотреть значение переменной примитивного типа, можно навести указатель мыши на имя переменной в тексте программы. Появится всплывающее окно с текущим значением переменной.



После того, как расставлены точки прерывания программу можно запускать в режиме отладки. Для этого можно выбрать команду меню *Run | Debug Last Launched* (или клавишу *F11*). Программа выполняется до ближайшей точки прерывания, а затем выполнение программы приостанавливается. Для возобновления вычислений программу можно запустить в пошаговом режиме *Run | Step Into* (клавиша *F5*) для выполнения с отладкой методов или *Run | Step Over* (клавиша *F6*) без захода в методы, либо продолжить выполнение программы в автоматическом режиме до конца или следующей точки прерывания *Run | Resume* (клавиша *F8*).



Для быстрого доступа к этим командам на вкладке окна *Debug* расположены соответствующие кнопки.

Для немедленного прекращения отладки можно выбрать команду *Run* | *Terminate*, или нажать соответствующую кнопку на вкладке окна *Debug*. Для того, чтобы перейти из компоновки *Debug* в обычную компоновку *Java* можно воспользоваться либо соответствующей командой меню, либо кнопкой переключения компоновок, расположенной выше окна просмотра значений переменных и точек прерывания.

Занятие №3 «Одномерные и двумерные массивы»

Знакомство с основными приемами работы с одномерными и многомерными массивами.

Задание №1

Написать программу, которая проверяет, есть ли в одномерном массиве элементы с одинаковыми значениями.

Задание №2

Написать программу, которая проверяет, представляют ли элементы одномерного массива возрастающую или убывающую последовательность.

Задание №3

В одномерном массиве $a(n)$ найти и напечатать номера (индексы) локальных максимумов, то есть таких элементов массива a_i, для которых выполняется условие $a_{i-1} < a_i > a_{i+1}$.

Задание №4

Каждый из элементов x_i одномерного массива $x(n)$ заменить средним значением первых i элементов этого массива.

Задание №5

Пусть дана последовательность из n различных целых чисел. Найдите среднее арифметическое чисел этой последовательности, расположенных между минимальным и максимальными числами (в сумму включить и оба эти числа).

Задание №6

Пусть даны координаты n точек на плоскости: (x_1, y_1), (x_2, y_2), …, (x_n, y_n). Найдите номера двух точек, расстояние между которыми наибольшее (считайте, что такая пара точек единственная).

Задание №7

Написать программу, которая, с помощью встроенного метода бинарного поиска упорядоченно заполняет массив данными, введенными с клавиатуры.

Задание №8

Дан массив целых чисел. Написать программу, которая, преобразовывает массив таким образом, чтобы в первой его части располагались все положительные элементы, а после них — все отрицательные. Отсортировать положительные элементы массива по возрастанию, а отрицательные — по убыванию.

Задание №9

Написать программу, которая в квадратной матрице M в строках с отрицательным элементом на главной диагонали находит наибольший элемент из всех элементов строки и сумму всех таких максимальных элементов.

Задание №10

Дана целочисленная матрица размером $m \times n$ Найти матрицу, получающуюся из данной:

- перестановкой столбцов — первого с последним, второго с предпоследним и т.д.
- перестановкой строк — первой с последней, второй с предпоследней и т.д.

Задание №11

В двумерном массиве X все числа различны. В каждой строке находится минимальный элемент, затем среди этих чисел выбирается максимальное. Напечатать номер строки и столбца массива X на пересечении которых расположено выбранное число.

Задание №12

Дан двумерный массив X размером $m \times n$. Получите из него новый двумерный массив, удалением κ -й строки и l -го столбца ($k \leqslant n$, $l \leqslant m$).