

# 1 Введение

## 1.1 История создания языка

Язык Java создан в 1991 группой Джеймса Гослинга. Первоначальное название – Oak. Переименован в Java, ввиду того, что уже существовал одноименный язык. Причиной создания нового языка послужила необходимость платформонезависимого языка для встраивания в бытовую технику. Первым проектом на Java является система дистанционного управления Star 7. Впоследствии пришло осознание применимости языка для WWW. Одна из первых публикаций от языке Java (Java White Paper) : <http://java.sun.com/docs/white/langenv/>

## 1.2 Свойства языка

- Простой
- Объектно-ориентированный
- Распределенный
- Интерпретируемый
- Надежный
- Безопасный
- Архитектурно-нейтральный
- Переносимый
- Высокопроизводительный
- Многопоточный

Первоначальная версия языка базировалась на синтаксисе языка C++. Современная же версия имеет следующие отличия от C++:

- Отсутствие перегрузки операторов.
- Отсутствие множественного наследования.
- Автоматическое согласование типов.
- Отсутствие адресной арифметики.
- Отсутствие деструкторов.

Более подробно:

<http://java.sun.com/docs/white/langenv/>

## 1.3 Основные термины и инструментарий разработчика

- Средства разработки существуют для большинства аппаратных платформ.
- Виртуальная машина Java (Java Virtual Machine, JVM) гарантирует единообразие интерфейса с операционной системой.
- Переносимость: «Write once, run everywhere».
- Поставляется с исчерпывающей библиотекой классов JDK (Java Development Kit).
- JRE (Java Runtime Environment) – среда, позволяющая запустить программу, написанную на языке Java.



В состав JDK входит набор утилит для создания Java приложений. Среди них такие как:

- javac – компилятор языка Java.
- java – интерпретатор байт-кода.
- javah - создает заголовочные файлы.
- javadoc - формирует стандартную документацию.
- jar – создание дистрибутивов Java.
- javap – дизассемблер.
- apt – обработчик аннотаций.
- Другие базовые инструменты (appletviewer, jdb, extcheck).

## 1.4 Среда разработки

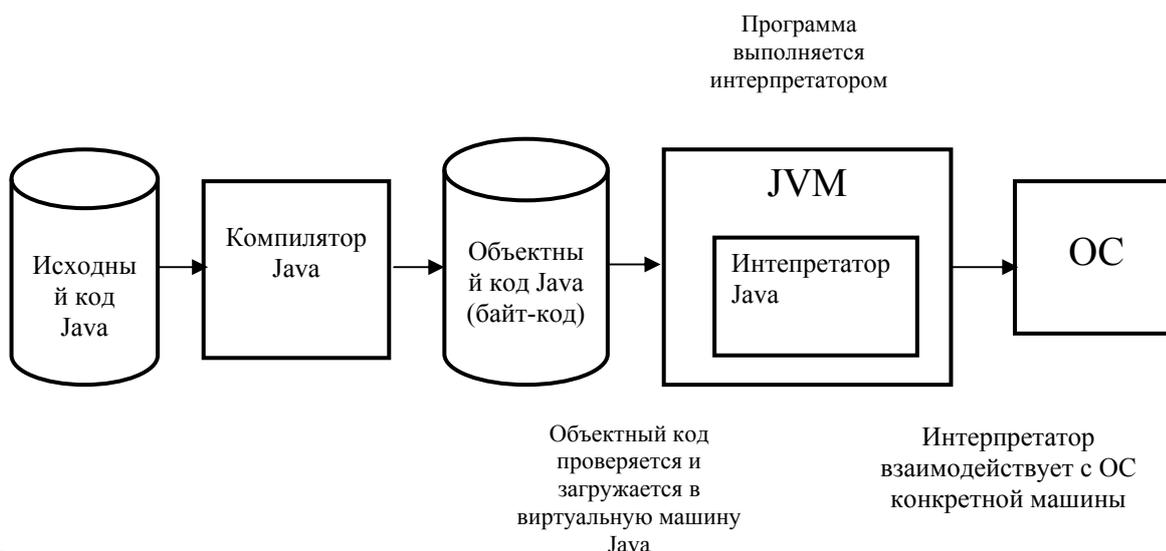
Среда разработки предлагает редактор, компилятор и набор вспомогательных программ, упрощающих разработку ПО.

- NetBeans <http://www.netbeans.org/>
- Eclipse <http://www.eclipse.org/>
- JBuilder, CodeGear <http://www.borland.com/>
- IntelliJ Idea <http://www.jetbrains.com/>

## 2 Создание приложения на Java

### 2.1 Схема разработки приложения

Общая схема разработки и запуска приложения на Java следующая.



### 2.2 Запуск приложения в командной строке

В текстовом редакторе создайте файл с исходным кодом программы на языке Java: Hello.java  
Скомпилируйте исходный код в промежуточный код командой

```
javac Hello.java
```

В результате получится файл Hello.class

Запустить приложение командой

```
java Hello
```

Если запускаемый класс принадлежит именованному пакету (пространство имен, которому принадлежит класс, package), например, **testpackage**, то скомпилированный класс необходимо поместить в папку, имя которой совпадает с именем указанного в классе пакета (если пакеты вложены друг в друга, то в несколько вложенных каталогов).

Запуск программы командой:

```
java testpackage.Hello
```



### 2.2.1 Простой пример – вывод строки на консоль

Единственный метод класса *Hello* является точкой входа программы (метод *main*). В результате работы выводит на консоль строку «Hello, World!».

```
public class Hello{
    public static void main(String []args){
        System.out.println("Hello, World");
    }
}
```

### 2.2.2 Определение и вызов метода

В данном классе 2 метода и один из них (*main*) вызывает другой (*sayHello*), передавая строку (*name*) в качестве параметра.

```
public class Hello{
    public static void main(String []args){
        sayHello("World");
    }
    public static void sayHello(String name){
        System.out.println("Hello, "+name+"!");
    }
}
```

### 2.2.3 Передача параметров через командную строку

Поскольку выполнение программы начинается с метода *main*, в него можно передать различные параметры из консоли, при запуске программы. Здесь первый введенный параметр интерпретируется как имя (*args[0]*).

```
public class Hello{
    public static void main(String []args){
        sayHello(args[0]);
    }
    public static void sayHello(String name){
        System.out.println("Hello, "+name+"!");
    }
}
```

### 2.2.4 Обработка неправильного ввода (отсутствие параметра при запуске)

В предыдущем примере мы предполагали, что программа запускается хотя бы с одним параметром. В ином случае сгенерировалось бы исключение. Здесь мы обрабатываем ситуацию запуска без параметра – при этом выводится «Hello, World!».

```
class Hello{
    public static void main(String []args){
        if (args.length==0){
            sayHello("World");
        }
        else{
            sayHello(args[0]);
        }
    }
    public static void sayHello(String rc){
        System.out.println("Hello, "+rc+"!");
    }
}
```



## 2.2.5 Перечисление всех параметров и использование вспомогательного класса

Создаем вспомогательный класс (*SayHello*). Он предоставляет функциональность для главного класса (*Hello*) – вывод строки на печать (метод *printString*). Необходимый параметр (имя) передается в конструктор при создании экземпляра класса *SayHello*.

```
public class Hello{
    public static void main(String []args){
        for (int i=0;i<args.length;++i) {
            FriendlyClass friend =new FriendlyClass (args[i]);
            friend.PrintString();
        }
    }
}

class FriendlyClass{
    String name;
    SayHello(String s){name=s;} //конструктор класса
    void PrintString(){
        System.out.println("Hello, "+name+"!");
    }
}
```

## 3 Знакомство со средой программирования

### 3.1 NetBeans IDE

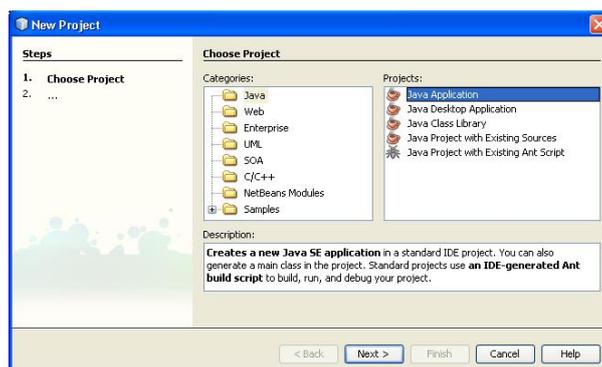
- Бесплатная интегрированная среда разработки.
- Не требует дополнительной настройки.
- Инструменты для разработки и отладки Java SE, ME и EE приложений.
- 100% Java.
- Открытый исходный код.
- Поддерживается сообществом.
- Поддержка контроля версий.
- Модульная архитектура.
- Поддержка рефакторинга.
- Проекты базируются на Ant (инструментарий для управления программными проектами на Java).

### 3.2 Установка среды

- Дистрибутив NetBeans можно скачать с сайта <http://www.netbeans.org/>
- Необходима JDK (может поставляться вместе с дистрибутивом NetBeans). Можно скачать с сайта <http://java.sun.com/>
- Для установки с параметрами по умолчанию достаточно последовательно отвечать на вопросы инсталлятора.

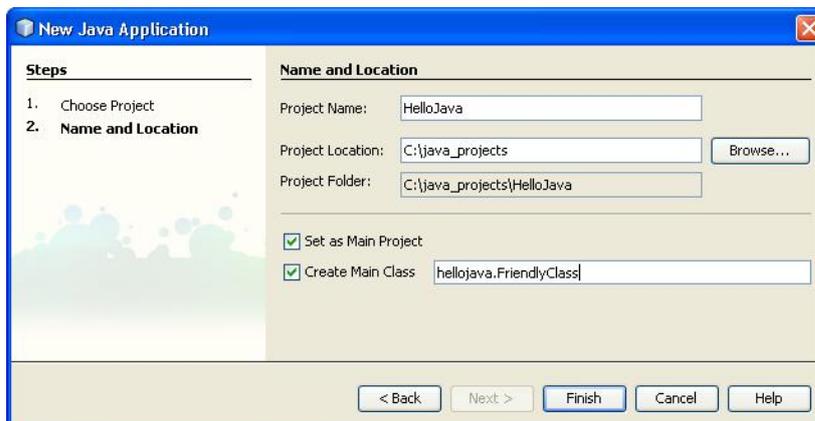
### 3.3 Лабораторная работа №1: «Разработка приложения HelloJava»

- 1) Запустите среду разработки.
- 2) Выберите в меню **File** пункт **New Project**.
- 3) В появившемся диалоговом окне выберите первый пункт «Java» в левом списке. В списке, появившемся справа, выберите пункт «Java Application». Нажмите **Next**.
- 4) В поле **Project Name** введите имя проекта (HelloJava) и укажите путь к нему в поле **Project location**. Здесь это «C:\java\_projects», но можно выбрать любой другой. Оставьте отмеченными флажки **Set as Main Project** и **Create Main Class**.



В текстовом поле рядом с последним флажком введите имя главного класса приложения. Оно должно начинаться с имени пакета (hellojava.FriendlyClass). Можете оставить значение по умолчанию (hellojava.Main).

Нажмите **Finish**.



5) Проект готов! Тело программы пока что пусто.

В левом окне, во вкладке **Projects** вы увидите структуру файлов проекта. Пока в него входит лишь один класс – FriendlyClass.java. Он расположен в пакете hellojava, созданном по умолчанию.



6) В правой части окна замените строку  
`// TODO code application logic here`  
следующей строкой

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

Метод main является точкой входа в программу – именно с него начнется ее выполнение. Пока внутри метода лишь одна строчка. При запуске программы на консоль будет выведен текст «Hello, Java!».

Полный текст программы имеет следующий вид.

```
public class FriendlyClass {
    public static void main(String []args) {
        System.out.println("Hello, Java!");
    }
}
```

7) В нижнем левом углу приложения нажмите Output (если данной кнопки нет, выберите в меню Window пункт Output>Output). Появившаяся область экрана – консоль среды разработки. В ней можно видеть результаты различных действий с приложением.

### 8) Компиляция проекта.

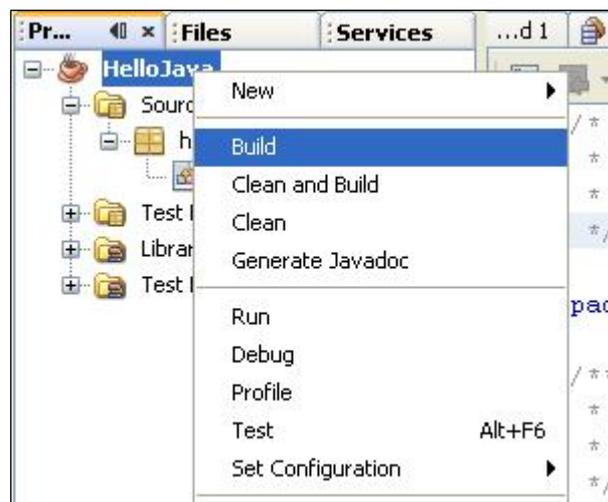
Нажмите правой кнопкой мыши на название проекта HelloJava. В появившемся контекстном меню выберите пункт **Build**.

Альтернативный вариант меню **Build**, пункт **Build Main Project** (горячая клавиша F11). Работает лишь если ваш проект выбран главным (стартовым).

На консоли должна появиться следующая надпись:

**BUILD SUCCESSFUL (total time: ? seconds)**

– знак того, что компиляция прошла успешно.



```

Output - HelloJava (jar)
init:
deps-jar:
Created dir: C:\java_projects\HelloJava\build\classes
Compiling 1 source file to C:\java_projects\HelloJava\build\classes
compile:
Created dir: C:\java_projects\HelloJava\dist
Building jar: C:\java_projects\HelloJava\dist\HelloJava.jar
Not copying the libraries.
To run this application from the command line without Ant, try:
java -jar "C:\java_projects\HelloJava\dist\HelloJava.jar"
jar:
BUILD SUCCESSFUL (total time: 0 seconds)
    
```

**9) Запуск проекта.**

Вновь войдите в контекстное меню для проекта (пункт 8) и выберите пункт Run для того чтобы запустить программу.

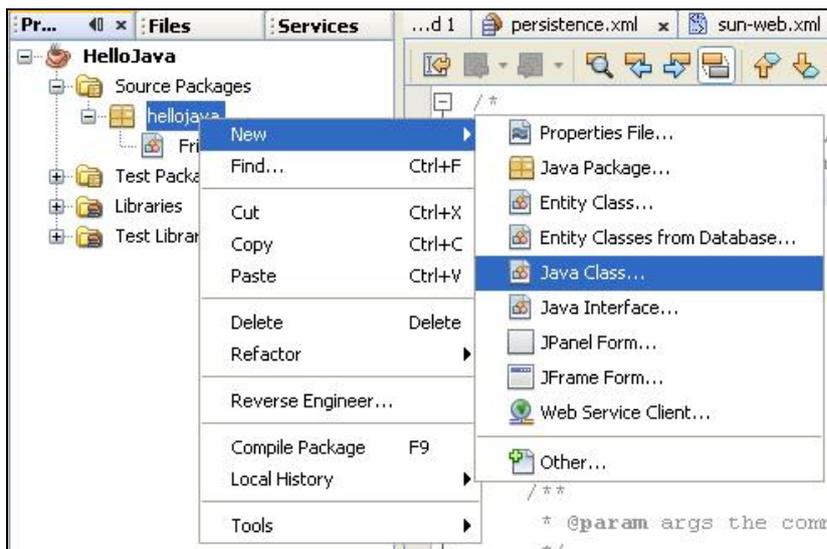
Альтернативный вариант меню **Run**, пункт **Run Main Project** (горячая клавиша F6). Работает лишь если ваш проект выбран главным (стартовым).

В консоли можно увидеть результат работы программы – строку «Hello, Java!».

```

Output - HelloJava (run)
init:
deps-jar:
compile:
run:
Hello, Java!
BUILD SUCCESSFUL (total time: 0 seconds)
    
```

**10) Добавление нового класса.**



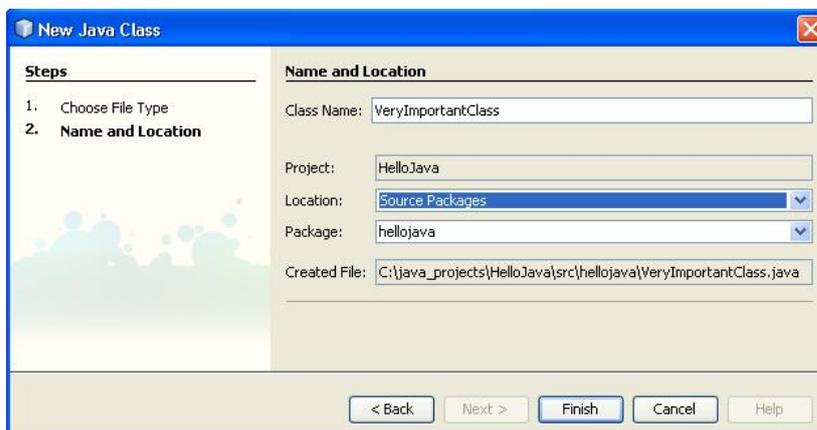
Щелкните правой кнопкой мыши по проекту (или пакету hellojava). В появившемся контекстном меню выберите **New > Java Class..**

11) Введите имя нового класса в поле **Class Name**.

В поле **Location** оставьте пункт **Source Packages**.

В поле **Package** выберите имя пакета (hellojava).

Нажмите **Finish**.



## 12) Немного об интерактивности

Созданное нами ранее приложение отлично работает, но оно всегда ведет себя одинаково, что бы ни делал пользователь. Добавим в него немного интерактивности.

1) Обратите внимание на параметры главного метода:

```
main(String []args)
```

Параметры можно задать при запуске приложения, разделяя их пробелами (их может быть произвольное число). Впрочем, в этом нет никакого смысла, пока мы не добавим в метод код, обрабатывающий вводимые параметры.

2) Перепишем код следующим образом:

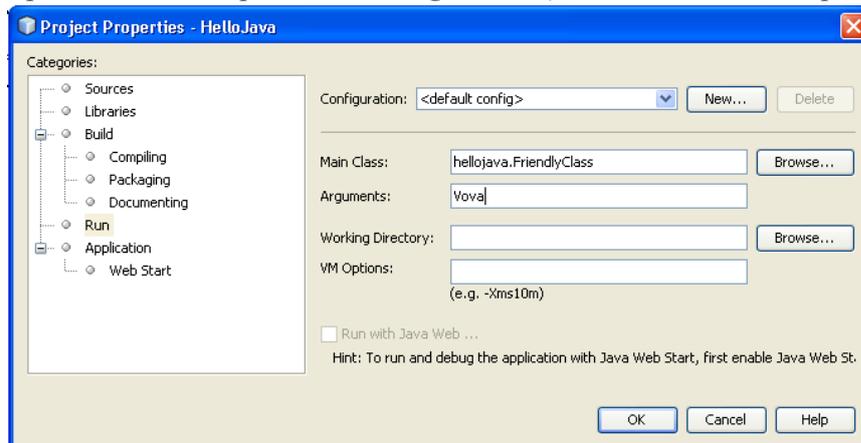
```
public class FriendlyClass{
    public static void main(String []args){
        System.out.println("Hello, "+args[0]+"!");
    }
}
```

3) Теперь, передав в качестве параметра запуска свое имя, мы получим индивидуальное приветствие. Сделаем это, не выходя из среды разработки.

Нажмите правой клавишей мыши на названии проекта HelloJava. В появившемся контекстном меню выберите последний пункт **Properties**. Откроется диалоговое окно свойств проекта (в нем можно задать много полезных настроек!).

Выберите пункт **Run** из списка (слева).

Введите параметры командной строки в поле **Arguments** (можно ваше имя, например, Vova).



4) Запустите программу

5) На консоль будет выведено личное приветствие «Hello, Vova!».

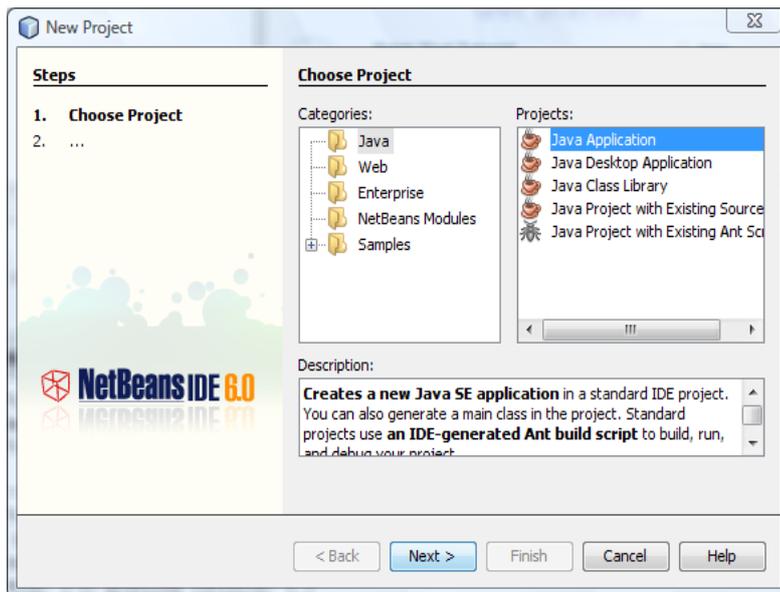
## 3.4 Возможности NetBeans

- Текстовый редактор
- Дизайнер графического интерфейса
- UML
- Системы контроля версий
- Отладчик
- И др.



### 3.5 Выбор типа проекта

General (Java)	Обычные приложения и библиотеки на Java.
Web	Вэб-приложения. Среди шаблонов: JSF и Struts.
Enterprise	ЕJB и вэб-сервисы.
Mobile	Приложения для портативных устройств.
NetBeans Plug-in Modules	Разработка подключаемых модулей для среды разработки.
Samples	Примеры приложений.

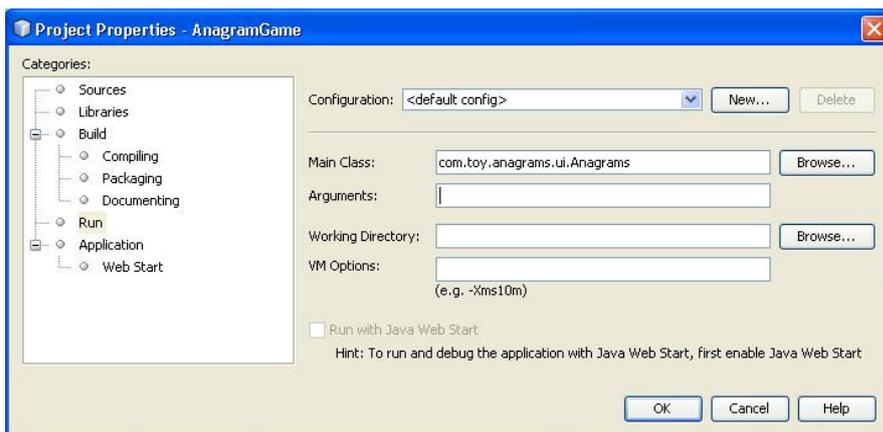


Каждый проект может быть создан из существующих исходных кодов или на основе готового сценария Ant.

### 3.6 Настройка окружения

Можно конфигурировать:

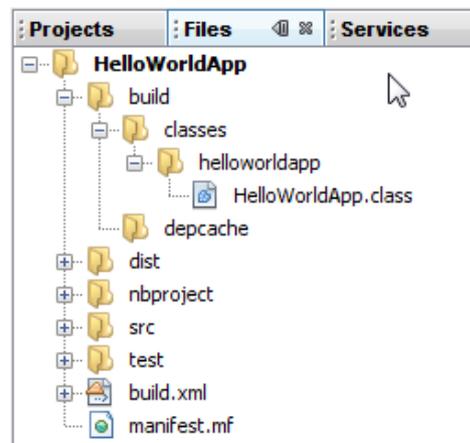
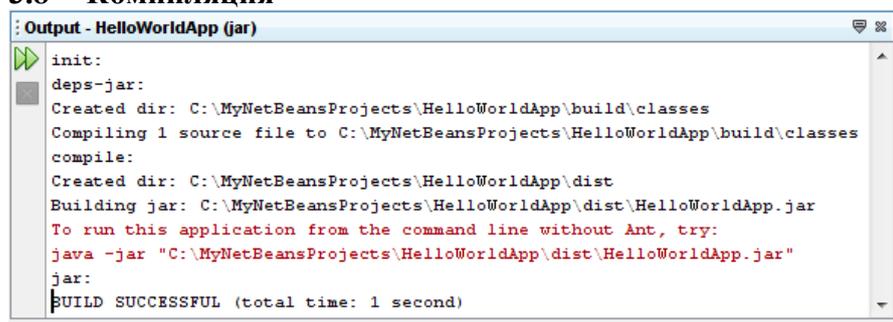
- Используемый проектом JDK.
- Переменные окружения.
- Зависимости между проектами.
- Используемые библиотеки.
- Параметры виртуальной машины и приложения.



### 3.7 Файловая структура

- При создании файла можно использовать шаблоны документов.
- Каждому типу файла соответствует редактор с соответствующими возможностями.
- Файлы .java можно организовывать в пакеты.
- Реализована удобная навигация по файлам проектов.

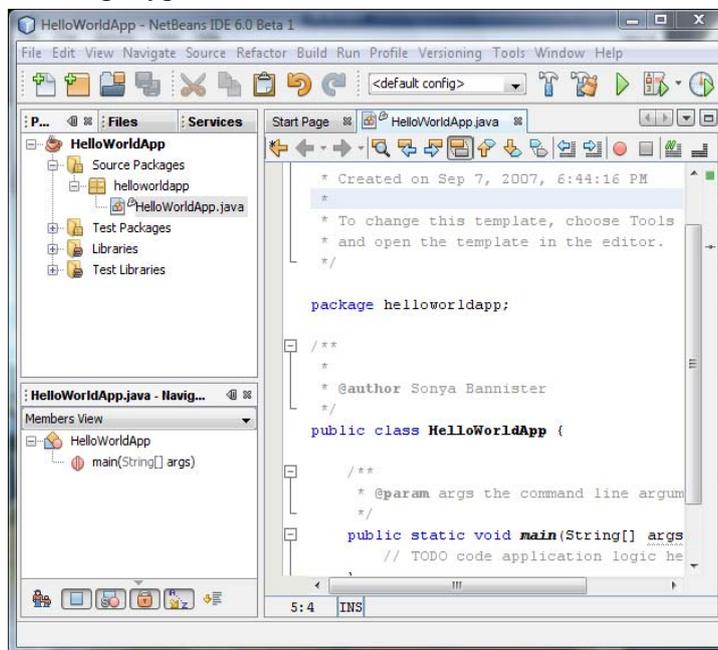
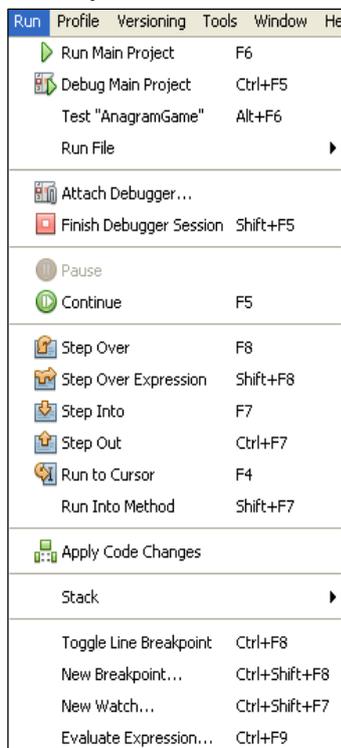
### 3.8 Компиляция



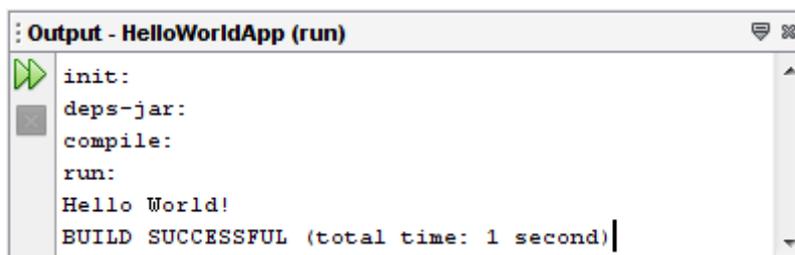
- Ошибки компиляции подсвечены как гиперссылки (на место ошибки в исходном коде) в окне для вывода.
- Файлы проекта можно упаковать в дистрибутив (JAR, WAR) в указанный каталог (по умолчанию: projectdir/dist).
- Для собственных нужд можно редактировать файл build.xml.

### 3.9 Запуск

- Для запуска можно использовать различные конфигурации.

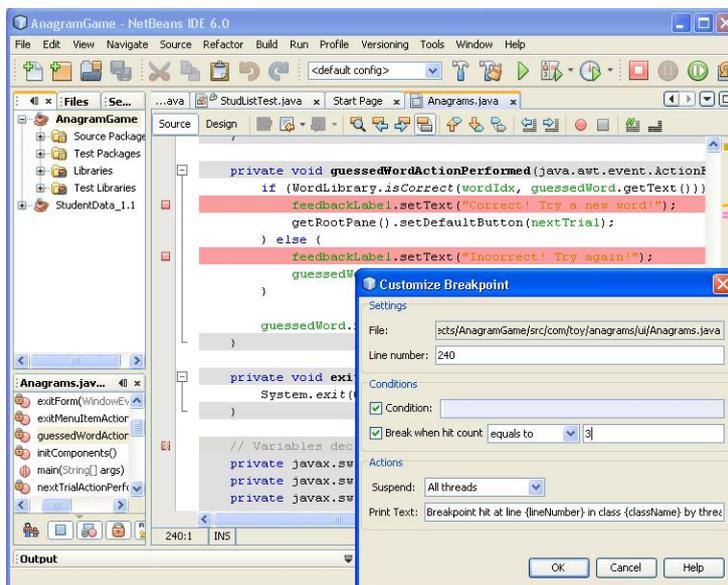


- Результат работы приложения отображается в окне.

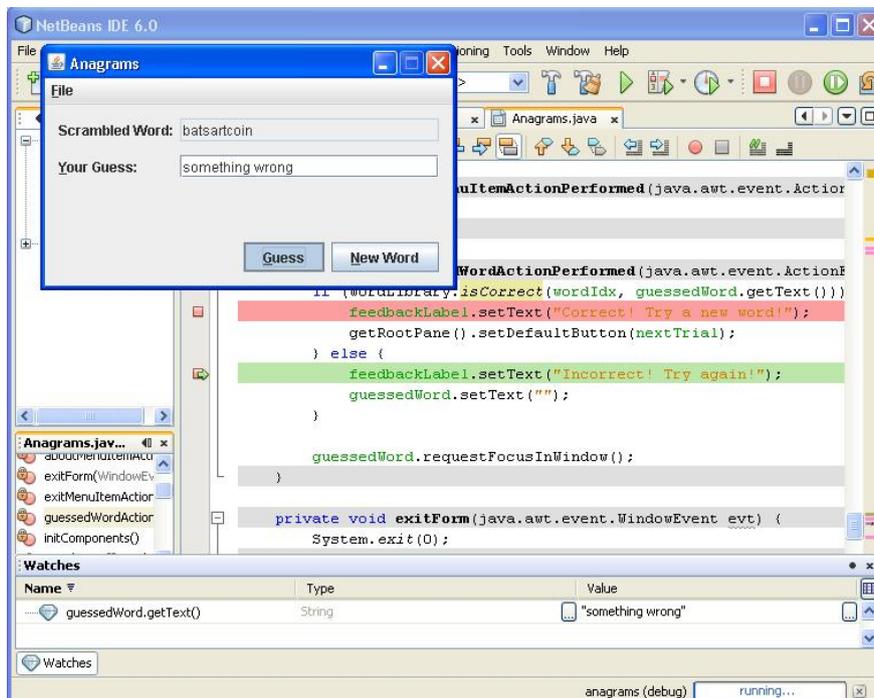


### 3.10 Отладка

- Установка точек прерывания исполнения.
- Выполнение программы по шагам.
- Отслеживание значений переменных.



Отладка (прерывание выполнения)



Отладка (отслеживание значений)



## Примеры проектов

Дистрибутив среды включает примеры типовых приложений, оформленных в виде проекта.

<b>Anagram Game</b>	Простое игровое приложение, использующее такие компоненты Swing, как JFrame, JLabel, и JTextField. Также содержит тестовый класс JUnit.
<b>GUI Form Examples</b>	Демонстрирует возможности работы NetBeans GUI Builder с тремя типичными схемами размещения (layouts).
<b>Document Editor</b>	Простой текстовый редактор; демонстрирует использование действий, карт ресурсов и других функциональных возможностей <i>Swing Application Framework</i> .
<b>Mars Rover Viewer</b>	Простой просмотрщик изображений; демонстрирует использование действий, заданий в фоновом потоке, карт ресурсов и т.д.
<b>Client Editor</b>	Простой редактор клиентских данных; демонстрирует использование связывания Bean компонентов. Включает примеры конвертации и валидации.

## Ссылки

- Сайт NetBeans:  
> <http://netbeans.org/>
- Онлайн-курсы:  
> <http://javapassion.com/>
- NetBeans IDE Field Guide by Patrick Keegan, Ludovic Champenois, etc.



Список заданий для самостоятельного выполнения по курсу  
«Язык программирования Java и платформа JavaME»

А.В.Дмитриев 2010

1. Скачать и установить среду разработки NetBeans: <http://netbeans.org/>  
На данный момент последняя версия 6.9.1.

Используя среду разработки, создавайте новый Java проект для каждого из следующих заданий.

2. По введенному пользователем значению радиуса рассчитать:
  - длину окружности,
  - площадь поверхности,
  - объем сферы соответствующего радиуса.
3. Определите время, за которое программа выполняет операцию по суммированию 10000000 единиц, вычислению факториала целого числа и т.д. В реализации можно использовать цикл:
  - for,
  - while,
  - do-while.В работе вы также можете использовать статический метод `System.currentTimeMillis()` или `System.nanoTime()`.
4. Распечатайте последовательность чисел Фибоначчи до числа 100.
5. Посчитайте факториал числа, вводимого пользователем. Пользователь может ввести отрицательное число или буквенные символы.
6. Подсчитать количество слов в строке, введенной пользователем. Не используйте встроенные функции разбора строк.
7. Установить правильность расстановки скобок в математическом выражении. Расстановку знаков операции и операндов во внимание не принимать.
8. Во введенной строке:
  - удалить все символы «а».
  - удалить все серии символов «bb».

Не используйте дополнительные хранилища соразмерного с введенной строкой размера.

9. Продемонстрируйте недетерминированность работы потоков исполнения (Threads). Минимум их нужно создать два. Их состояние отображать на экране с помощью компонента `JProgressBar`.

