СПб НИУ ИТМО

Кафедра ИПМ

Программирование интернет-приложений

Лабораторная работа №1

Простейшая консольная программа на языке Java

Работу выполнил

студент II курса

группы № 2120

Журавлев Виталий

Санкт-Петербург

2013 г.

**Цель работы:**

На языке Java написать простейшую консольную программу, выводящую в поток System.out список переданных ей аргументов командной строки. Дополнить эту программу использованием любого из классов стандартной библиотеки или любой синтаксической конструкции языка Java. .

**Исходный код программы:**

package javalab\_1;

import java.text.DateFormat;

import java.util.Date;

/\*\*

 \* @author Vitaly Zhuravlev

 \* @version 1.0

 \*/

public class JavaLab {

 /\*\*

 \* @param args the command line arguments

 \*/

 public static void main(String[] args) {

 /\*\*

 \* @deprecated application greets the user and tells him the current time

 \* @param Time is sets the parameters of DateFormat

 \* @param sTime the string from the current time without seconds

 \*/

 DateFormat Time = DateFormat.getTimeInstance(DateFormat.SHORT);

 String sTime = Time.format(new Date() {});

 System.out.println("Hello, world!");

 System.out.print("Current time: " + sTime);

 }

}

**Описание базовых утилит в составе JDK:**

**java - интерпретатор Java**

**Синтаксис вызова**

*Java [опции интерпретатора] имя класса [аргументы программы]*

**Описание**

Программа *Java* представляет собой интерпретатор байт-кода Java, который запускает Java-программы. Программа, которую надо выполнить, - это класс, указанный в аргументе *имя\_класса.* Имя должно быть полным и включать имя пакета, но не содержать расширение *class.*

Любые аргументы, следующие за именем класса в командной строке *Java,*помещаются в массив и передаются методу main () при запуске *Java.*Если в методе main () создаются какие-либо потоки,*то Java* выполняется то тех пор, пока не завершится последний поток. В противном случае интерпретатор выполняет тело main () и завершает работу. Хотя при вызове *Java* указывается имя только одного класса, интерпретатор автоматически загружает все дополнительные классы, необходимые для выполнения программы.

**Основные параметры**

-classpath *путь*
Определяет путь, который *Java* использует для поиска указанного имени класса и всех остальных загружаемых им классов.

-cs,(-checksourse)
Оба параметра указывают интерпретатору *Java* на необходимость проверки времени модификации указанного файла класса и соответствующего ему файла с исходным текстом. Если файл класса не найден или устарел, он автоматически перекомпилируется из исходного.

-D имясвойства=значение
Присваивает свойству из списка свойств значение, равное указанному.

-debug
Заставляет *Java* выводить при запуске пароль, который используется для того, чтобы разрешать отладчику *jdb* участвовать в данном сеансе работы интерпретатора.

-noverify
Никогда не проводит проверку байт-кода.

-verify
Запускает верификатор байт-кода для всех загруженных классов.

-v, (-verbose)
Выводит на экран сообщение всякий раз, когда *Java* загружает класс.

## javac - компилятор Java

#### Синтаксис вызова

*javac [параметры] файлы*

#### Описание

Программа *javac* компилирует исходные тексты Java (из файлов с расширением java) в байт-код Java (в файлы с расширением *class).*

 Компилятор *javac* может передать в командной строке любое количество файлов исходных текстов Java*.*Компилятор *javac* генерирует отдельный файл для каждого класса, определенного в файле исходного текста.

По умолчанию *javac* располагает созданные файлы классов в том же каталоге, где находились соответствующие файлы с исходными текстами. Это можно переопределить с помощью параметра -d.

Когда файл исходного текста ссылается на класс, не определенный в командной строке какого-либо другого файла исходного текста, *javac* ищет этот класс, используя параметр -classpath.

#### Основные параметры

-classpath *путь*
Определяет путь, используемый *javac* для поиска классов, на которые имеются ссылки в исходных текстах.

-d *каталог*
Определяет каталог, в котором должны храниться файлы классов.

-depend
Сообщает *javaс* о необходимости перекомпилировать любой встретившийся ему устаревший файл.

-deprecation
Сообщает *javac* о необходимости выдавать предупреждение при каждом использовании устаревшего API.

-nowarn
Отключает вывод предупреждений. Сообщения об ошибках выводятся.

-nowrite
Сообщает *javac* о том, что не следует создавать файл класса. Файлы с исходными текстами обрабатываются обычным путем, но результаты такой обработки не записываются. Этот параметр полезен в том случае, когда требуется, не выполняя фактической компиляции, проверить, может ли компилироваться некоторый файл.

-verbose
Дает компилятору указание выдавать сообщения о том, что он делает.

##

## javadoc - генератор документации Java

#### Синтаксис вызова

*javadoc [параметры] имя пакета
javadoc [параметры] имена файлов*

#### Описание

Программа *javadoc* создает документацию на API в формате HTML для указанных пакетов или отдельных файлов исходных текстов Java, заданных в командной строке. Если в командной строке указано имя пакета, *javadoc* ищет соответствующий каталог пакета относительно параметра -classpath. Затем анализирует все файлы с исходными текстами в этом каталоге и формирует HTML-файл документации для каждого класса и HTML-индекс для классов пакета

Файлы документации классов, создаваемые *javadoc*, описывают и его иерархию наследования, индекс и каждый член класса, объявленный как public или protected. Созданный файл содержит комментарии для документации, которые связаны с классами и их методами, конструкторами и переменными. Комментарий для документации начинаются символами / \* \* и оканчиваются символами \* /. Они могут включать любой HTML-тег, а также значения тегов.

#### Параметры

-author *путь*
Указывает на необходимость вывода информации об авторе, заданной с помощью тега @author. Эта информация по умолчанию не выводится.

-classpath *путь*
Определяет путь, который *javadoc* использует для поиска, как файлов классов, так и файлов с исходными текстами для указанного пакета.

-d *каталог*
Каталог, в который *javadoc* должен записывать создаваемые им HTML-файлы.

-decoding *название\_кодировки*
Задает кодировку символов, которая будет использоваться в документации, создаваемой с помощью *javadoc.*

 -encoding *название\_кодировки*
Задает кодировку символов, которая будет использоваться при чтении файлов с исходными текстами и комментариями, содержащимися в них

-J*аргумент*
Передает аргумент непосредственно интерпретатору.

-nodeprecated
Сообщает, что теги @deprecated в создаваемые документы включать не нужно.

-verbose
Заставляет *javadoc* выдавать сообщения о том, что он делает в данный момент.

## jar - программа создания архивов Java

#### Синтаксис вызова

*jar c|t|x[f][m][v] [jar-файл] [файл описания} [файлы}*

#### Описание

Программа *jar* используется для создания архивных файлов Java (JAR) и работы с ними. JAR-файл представляет собой сжатый ZIP-файл с дополнительным файлом описания.

Параметры командной строки *jar* задаются в виде блока записанных слитно букв, которые передаются в виде одного аргумента, а не через отдельные аргументы командной строки. Первая буква такого аргумента задает необходимое действие, которое должна выполнить программа *jar.*Остальные буквы в этом аргументе являются необязательными. Различные аргументы файлов зависят от того, какие буквы параметров заданы.

**Параметры**

Первым аргументом командной строки *jar* является набор символов, задающих операцию, которая должна быть выполнена. Первый символ определяет основную операцию и является обязательным. Возможны следующие варианты:

|  |  |
| --- | --- |
| **с -** | Создать новый JAR-архив. |
| **t -** | Вывести список файлов, содержащихся в JAR-архиве.  |
| **х -** | Извлечь содержимое JAR-архива.  |

Вслед за идентификатором, определяющим выполняемое действие, могут следовать необязательные параметры:

|  |  |
| --- | --- |
| **f -** | Указывает на то, что имя JAR-файла, который необходимо создать, из которого нужно извлечь файлы или получить список содержащихся файлов, задается в командной строке.  |
| **m -** | Используется только в сочетании с параметром с и указывает на то, что *jar* должна читать файл описания, указанный в командной строке и использовать его в качестве основы для создания описания, которое включается в JAR-файл. |
| **v -** | Если этот параметр задается вместе с параметром с, то выводится имя каждого добавляемого в архив файла со статистикой его сжатия.  |

**Вывод:**

В ходе лабораторной работе я разобрал синтаксические особенности языка Java, а так же воспользовался стандартными классами java.text.DateFormat и java.util.Date из стандартной библиотеки языка Java. Кроме того, я изучил основные средства JDK и их основные параметры.