**Строки** — представляют собой последовательность символов. Строки в Java широко используются и являются объектами.

Платформа Java предоставляет класс строк (class String) для создания и работы со строками.

**Содержание**

[скрыть]

1. [Создание строк](http://proglang.su/java/strings#sozdanie-strok)
   1. [Пример](http://proglang.su/java/strings#primer)
2. [Длина строки](http://proglang.su/java/strings#dlina-stroki)
   1. [Пример](http://proglang.su/java/strings#primer)
3. [Объединение строк в Java](http://proglang.su/java/strings#obedinenie-strok-v-java)
   1. [Пример](http://proglang.su/java/strings#primer)
4. [Создание формата строк](http://proglang.su/java/strings#sozdanie-formata-strok)
5. [Методы класса строк](http://proglang.su/java/strings#metody-klassa-strok)

**Создание строк**

Наиболее простой способ создать строку:

String greeting = "Здравствуй, мир!";

После каждого раза, когда компилятор встречает строковый литерал в коде, он создает строковый объект (String) со значение, в данном случае «Здравствуй, мир!».

Как и в случае с другими объектами, можно создавать строковые объекты, используя ключевое слово new и конструктор. Класс строк имеет одиннадцать конструкторов, которые позволяют обеспечить начальное значение строки, используя различные источники, таких как массив символов.

**Пример**

public class Test {

public static void main(String args[]){

char[] helloArray = { 'П', 'р', 'и', 'в', 'е', 'т', '.'};

String helloString = new String(helloArray);

System.out.println(helloString);

}

}

Получим следующий результат:

Привет.

**Примечание:** Класс строк является неизменяемым, так что как только он будет создан, строковый объект не может быть изменен. Если есть необходимость сделать много изменений в строке символов, следует использовать классы строки буфера (String Buffer) и построитель строки (String Builder Classes).

**Длина строки**

Методы, используемые для получения информации об объекте, известны как методы доступа. Один из методов доступа, который можно использовать со строками является метод length(), он возвращает количество символов, содержащихся в строковом объекте.

Ниже представлен пример метода length(), который поможет определить длину строки.

**Пример**

public class Test {

public static void main(String args[]) {

String s = "Я стану отличным программистом!";

int len = s.length();

System.out.println( "Длина строки: " + len + " символ.");

}

}

Получим следующий результат:

Длина строки: 31 символ.

**Объединение строк в Java**

Класс строк включает метод для объединения двух строк:

string1.concat(string2);

Возвращает новую строку string1, с добавленной строкой string2 в конце. Вы также можете использовать метод concat() со строковыми литералами:

"Меня зовут ".concat("Олег");

Чаще строки объединяют оператором +:

"Здравствуй," + " мир" + "!"

Что приводит к:

Здравствуй мир!

Давайте посмотрим на следующем примере.

**Пример**

public class Test {

public static void main(String args[]) {

String string1 = "отличным ";

System.out.println("Я стану " + string1 + "программистом!");

}

}

Получим следующий результат:

Я стану отличным программистом!

**Создание формата строк**

Мы уже изучали методы printf() и format() для вывода на печать с отформатированными числами. Класс строк в Java обладает эквивалентным методом format(), который возвращает строковый объект, а не потоковый объект (PrintStream).

Использование строкового статического метода format() позволяет создавать строку нужного формата, который можно использовать повторно, в отличие от одноразовых операторов print. Например вместо:

System.out.printf("Значение переменной float = " +

"%f, пока значение integer " +

"переменная = %d, и string " +

"= %s", floatVar, intVar, stringVar);

Вы также можете написать:

String fs;

fs = String.format("Значение переменной float = " +

"%f, пока значение integer " +

"переменная = %d, и string " +

"= %s", floatVar, intVar, stringVar);

System.out.println(fs);

**Методы класса строк**

Ниже приведен список методов, поддерживаемых классом строк.

|  |  |
| --- | --- |
| № | Методы с описанием |
| 1 | [char charAt(int index)](http://proglang.su/java/strings-charat) Возвращает символ по указанному индексу. |
| 2 | [int compareTo(Object o)](http://proglang.su/java/strings-compareto) Сравнивает данную строку с другим объектом. |
| 3 | int compareTo(String anotherString) Сравнивает две строки лексически. |
| 4 | [int compareToIgnoreCase(String str)](http://proglang.su/java/strings-comparetoignorecase) Сравнивает две строки лексически, игнорируя регистр букв. |
| 5 | [String concat(String str)](http://proglang.su/java/strings-concat) Объединяет указанную строку с данной строкой, путем добавления ее в конце. |
| 6 | [boolean contentEquals(StringBuffer sb)](http://proglang.su/java/strings-contentequals) Возвращает значение true только в том случае, если эта строка представляет собой ту же последовательность символов как указанно в буфере строки (StringBuffer). |
| 7 | [static String copyValueOf(char[] data)](http://proglang.su/java/strings-copyvalueof) Возвращает строку, которая представляет собой последовательность символов, в указанный массив. |
| 8 | static String copyValueOf(char[] data, int offset, int count) Возвращает строку, которая представляет собой последовательность символов, в указанный массив. |
| 9 | [boolean endsWith(String suffix)](http://proglang.su/java/strings-endswith) Проверяет заканчивается ли эта строка указанным окончанием. |
| 10 | [boolean equals(Object anObject)](http://proglang.su/java/strings-equals) Сравнивает данную строку с указанным объектом. |
| 11 | [boolean equalsIgnoreCase(String anotherString)](http://proglang.su/java/strings-equalsignorecase) Сравнивает данную строку с другой строкой, игнорируя регистр букв. |
| 12 | [byte getBytes()](http://proglang.su/java/strings-getbytes) Кодирует эту строку в последовательность байтов с помощью платформы charset, сохраняя результат в новый массив байтов. |
| 13 | byte[] getBytes(String charsetName Кодирует эту строку в последовательность байтов с помощью платформы charset, сохраняя результат в новый массив байтов. |
| 14 | [void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin)](http://proglang.su/java/strings-getchars) Копирует символы из этой строки в массив символов назначения. |
| 15 | [int hashCode()](http://proglang.su/java/strings-hashcode) Возвращает хэш-код для этой строки. |
| 16 | [int indexOf(int ch)](http://proglang.su/java/strings-indexof) Возвращает индекс первого вхождения указанного символа в данной строке. |
| 17 | int indexOf(int ch, int fromIndex) Возвращает индекс первого вхождения указанного символа в данной строке, начиная поиск с указанного индекса. |
| 18 | int indexOf(String str) Возвращает индекс первого вхождения указанной подстроки в данной строке. |
| 19 | int indexOf(String str, int fromIndex) Возвращает индекс первого вхождения указанной подстроки в данной строке, начиная с указанного индекса. |
| 20 | [String intern()](http://proglang.su/java/strings-intern) Возвращает каноническое представление для строкового объекта. |
| 21 | [int lastIndexOf(int ch)](http://proglang.su/java/strings-lastindexof) Возвращает индекс последнего вхождения указанного символа в этой строке. |
| 22 | int lastIndexOf(int ch, int fromIndex) Возвращает индекс последнего вхождения указанного символа в этой строке, начиная обратный поиск с указанного индекса. |
| 23 | int lastIndexOf(String str) Возвращает индекс последнего вхождения указанной подстроки в данной строке. |
| 24 | int lastIndexOf(String str, int fromIndex) Возвращает индекс последнего вхождения указанной подстроки в этой строке, начиная обратный поиск с указанного индекса. |
| 25 | [int length()](http://proglang.su/java/strings-length) Возвращает длину строки. |
| 26 | [boolean matches(String regex)](http://proglang.su/java/strings-matches) Сообщает, соответствует ли или нет эта строка заданному регулярному выражению. |
| 27 | [boolean regionMatches(boolean ignoreCase, int toffset, String other, int ooffset, int len)](http://proglang.su/java/strings-regionmatches) Проверяет равны ли две области строки. |
| 28 | boolean regionMatches(int toffset, String other, int ooffset, int len) Проверяет равны ли две области строки. |
| 29 | [String replace(char oldChar, char newChar)](http://proglang.su/java/strings-replace) Возвращает новую строку, в результате, заменив все вхождения oldChar в этой строке на newChar. |
| 30 | [String replaceAll(String regex, String replacement)](http://proglang.su/java/strings-replaceall) Заменяет каждую подстроку строки, соответствующей заданному регулярному выражению с данной заменой. |
| 31 | [String replaceFirst(String regex, String replacement)](http://proglang.su/java/strings-replacefirst) Заменяет первые подстроки данной строки, которая соответствует заданному регулярному выражению с данной заменой. |
| 32 | [String[] split(String regex)](http://proglang.su/java/strings-split) Разделяет эту строку, окружая данным регулярным выражением. |
| 33 | String[] split(String regex, int limit) Разделяет эту строку, окружая данным регулярным выражением. |
| 34 | [boolean startsWith(String prefix)](http://proglang.su/java/strings-startswith) Проверяет, начинается ли эта строка с заданного префикса. |
| 35 | boolean startsWith(String prefix, int toffset) Проверяет, начинается ли эта строка с указанного префикса, начиная с указанного индекса. |
| 36 | [CharSequence subSequence(int beginIndex, int endIndex)](http://proglang.su/java/strings-subsequence) Возвращает новую последовательность символов, которая является подпоследовательностью этой последовательности. |
| 37 | [String substring(int beginIndex)](http://proglang.su/java/strings-substring) Возвращает новую строку, которая является подстрокой данной строки. |
| 38 | String substring(int beginIndex, int endIndex) Возвращает новую строку, которая является подстрокой данной строки. |
| 39 | [char[] toCharArray()](http://proglang.su/java/strings-tochararray) Преобразует эту строку в новый массив символов. |
| 40 | [String toLowerCase()](http://proglang.su/java/strings-tolowercase) Преобразует все символы в данной строке в нижний регистр, используя правила данного языкового стандарта. |
| 41 | String toLowerCase(Locale locale) Преобразует все знаки в данной строке в нижний регистр, используя правила данного языкового стандарта. |
| 42 | [String toString()](http://proglang.su/java/strings-tostring) Этот объект (который уже является строкой!) возвращает себя. |
| 43 | [String toUpperCase()](http://proglang.su/java/strings-touppercase) Преобразует все символы в строке в верхний регистр, используя правила данного языкового стандарта. |
| 44 | String toUpperCase(Locale locale) Преобразует все символы в строке в верхний регистр, используя правила данного языкового стандарта. |
| 45 | [String trim()](http://proglang.su/java/strings-trim) Возвращает копию строки с пропущенными начальными и конечными пробелами. |
| 46 | [static String valueOf(primitive data type x)](http://proglang.su/java/strings-valueof) Возвращает строковое представление переданного типа данных аргумента. |

**Классы StringBuffer и StringBuilder** в Java используются, когда возникает необходимость сделать много изменений в строке символов.

В отличие от строк, объекты типа StringBuffer и StringBuilder могут быть изменены снова и снова, оставляя после себя множество новых неиспользуемых объектов.

В Java StringBuilder был введен начиная с Java 5. Основное различие между StringBuffer и StringBuilder является то, что методы StringBuilder не являются безопасными для потоков (несинхронизированные).

Рекомендуется использовать StringBuilder всякий раз, когда это возможно, потому что он быстрее, чем StringBuffer в Java. Однако, если необходима безопасность потоков, наилучшим вариантом являются объекты StringBuffer.

**Содержание**

[скрыть]

1. [Пример](http://proglang.su/java/strings-stringbuilder-stringbuffer#primer)
2. [Методы StringBuffer](http://proglang.su/java/strings-stringbuilder-stringbuffer#metody-stringbuffer)

**Пример**

public class Test {

public static void main(String args[]){

StringBuffer sBuffer = new StringBuffer("тест");

sBuffer.append(" StringBuffer");

System.out.println(sBuffer);

}

}

Получим следующий результат:

тест StringBuffer

**Методы StringBuffer**

Список методов, поддерживаемых классом StringBuffer:

|  |  |
| --- | --- |
| № | Описание |
| 1 | [public StringBuffer append(String s)](http://proglang.su/java/strings-stringbuilder-stringbuffer-append) Обновляет значение объекта, который вызывает метод. Этот метод принимает boolean, char, int, long, Strings и т.д. |
| 2 | [public StringBuffer reverse()](http://proglang.su/java/strings-stringbuilder-stringbuffer-reverse) Этот метод изменяет значение объекта StringBuffer, который вызывает метод. |
| 3 | [public delete(int start, int end)](http://proglang.su/java/strings-stringbuilder-stringbuffer-delete) Удаляет строку, начиная с начального индекса до конечного индекса. |
| 4 | [public insert(int offset, int i)](http://proglang.su/java/strings-stringbuilder-stringbuffer-insert) Этот метод вставляет строку s в позицию упомянутую по смещению. |
| 5 | [replace(int start, int end, String str)](http://proglang.su/java/strings-stringbuilder-stringbuffer-replace) Этот метод заменяет символы в подстроке данного StringBuffer символами в указанной строке. |

Список других методов (за исключением методов set), которые очень похожи на [методы класса строк](http://proglang.su/java/strings" \o "Список методов класса строк в Java):

|  |  |
| --- | --- |
| № | Описание |
| 1 | int capacity() Возвращает текущую вместимость буфера String. |
| 2 | char charAt(int index) Возвращается указанный символ последовательности, в настоящее время представленный буфером строки, указанный индексом аргумент. |
| 3 | void ensureCapacity(int minimumCapacity) Гарантирует вместимость буфера, по крайней мере равным указанному минимуму. |
| 4 | void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin) Символы копируются из этого буфера строки в символьный массив назначения dst. |
| 5 | int indexOf(String str) Возвращает индекс в данной строке первого вхождения указанной подстроки. |
| 6 | int indexOf(String str, int fromIndex) Возвращает индекс в данной строке первого вхождения указанной подстроки, начиная с указанного индекса. |
| 7 | int lastIndexOf(String str) Возвращает индекс в данной строке последнего вхождения указанной подстроки. |
| 8 | int lastIndexOf(String str, int fromIndex) Возвращает индекс в данной строке последнего вхождения указанной подстроки, начиная с указанного индекса. |
| 9 | int length() Возвращает длину строкового буфера (количество символов). |
| 10 | void setCharAt(int index, char ch) Символ с указанным индексом этого буфера строки имеет значение ch. |
| 11 | void setLength(int newLength) Устанавливает длину буфера строки (Stringbuffer). |
| 12 | CharSequence subSequence(int start, int end) Возвращает новую последовательность символов, которая является подпоследовательностью этой последовательности. |
| 13 | String substring(int start) Возвращает новую строку, которая содержит подпоследовательность символов в данный момент содержащихся в StringBuffer. Подстрока начинается с указанного индекса и продолжается до конца StringBuffer. |
| 14 | String substring(int start, int end) Возвращает новую строку, которая содержит подпоследовательность символов в данный момент содержащихся в этом StringBuffer. |
| 15 | String toString() Преобразование в строку, представляющую данные в этой строке буфера. |