Strings

Problem

Большая часть информации, которая используется в ПО является текстовой.
Как работать с текстом в Java?

Class `String`

Class `String`

- `String` из пакета `java.lang`
- `String` **неизменяемый** (**immutable**), т.е. после создания, его содержимое не может изменяться.

```
String str = "Hello Java!";
```

How create `String` objects?

```
String str1 = "Java";
String str2 = new String();
String str3 = new String(new char[] {'h', 'e', 'l', 'l', 'o'});
String str4 = new String(new char[] {'w', 'e', 'l', 'c', 'o', 'm', 'e'}, 3,

System.out.println(str1); // Java
System.out.println(str2); //
System.out.println(str3); // hello
System.out.println(str4); // come
```

Operator + for strings

Для *объектов* и *литералов* типа `String` определена одна операция `+` (**конкатенация**), которая объединяет две строки.

```
String str1 = "Java";
String str2 = "Hello";
String str3 = str1 + " " + str2;

System.out.println(str3); // Hello Java
```

`String`: main methods

lenght()

```
String str1 = "Java";
System.out.println(str1.length()); // 4
```

toCharArray()

```
String str1 = new String(new char[] {'h', 'e', 'l', 'l', 'o'});
char[] helloArray = str1.toCharArray();

String s = ""; // строка не указывает на объект
if (s.length() == 0) {
    System.out.println("String is empty");
}
```

isEmpty()

```
String s = ""; // строка не указывает на объект
if (s.length() == 0) {
    System.out.println("String is empty");
}

String s = null; // строка не указывает на объект
if(s == null) {
    System.out.println("String is null");
}
```

isEmpty()

```
String s = null; // строка не указывает на объект
if (s.length() == 0) { // NullPointerException
    System.out.println("String is empty");
}

String s = null; // строка не указывает на объект
if (s != null && s.length() == 0) {
    System.out.println("String is empty");
}
```

concat()

```
String str1 = "Java";
String str2 = "Hello";
str2 = str2.concat(str1); // HelloJava
```

join()

```
String str1 = "Java";
String str2 = "Hello";
String str3 = String.join(" ", str2, str1); // Hello Java
```

charAt()

```java
String str = "Java";
char c = str.charAt(2);

System.out.println(c); // v
```

getChars()

```
String str = "Hello world!";
int start = 6;
int end = 11;
char[] dst=new char[end - start];
str.getChars(start, end, dst, 0);

System.out.println(dst); // world
```

`equals()` и `equalsIgnoreCase()`

```
String str1 = "Hello";
String str2 = "hello";

System.out.println(str1.equals(str2)); // false
System.out.println(str1.equalsIgnoreCase(str2)); // true
```

regionMatches()

```
boolean regionMatches(int toffset, String other, int oofset, int len)
boolean regionMatches(boolean ignoreCase, int toffset, String other, int oo
```

regionMatches()

```
String str1 = "Hello world";
String str2 = "I work";
boolean result = str1.regionMatches(6, str2, 2, 3);

System.out.println(result); // true
```

compareTo() и compareToIgnoreCase()

```
String str1 = "hello";
String str2 = "world";
String str3 = "hell";

System.out.println(str1.compareTo(str2)); // -15 -> str1 меньше чем strt2
System.out.println(str1.compareTo(str3)); // 1 -> str1 больше чем str3
```

`indexOf()` и `lastIndexOf()`

```
String str = "Hello world";
int index1 = str.indexOf('l'); // 2
int index2 = str.indexOf("wo"); // 6
int index3 = str.lastIndexOf('l'); // 9
```

`startsWith()` и `endsWith()`

```
String str = "myfile.exe";
boolean start = str.startsWith("my"); // true
boolean end = str.endsWith("exe"); // true
```

replace()

```
String str = "Hello world";
String replStr1 = str.replace('l', 'd'); // Heddo wordd
String replStr2 = str.replace("Hello", "Bye"); // Bye world
```

trim()

```
String str = "  hello world  ";
str = str.trim(); // "hello world"
```

substring()

```
String str = "Hello world";
String substr1 = str.substring(6); // "world"
String substr2 = str.substring(3,5); // "lo"
```

`toLowerCase()` и `toUpperCase()`

```
String str = "Hello World";
System.out.println(str.toLowerCase()); // hello world
System.out.println(str.toUpperCase()); // HELLO WORLD
```

split()

```
String text = "FIFA will never regret it";
String[] words = text.split(" ");
for (String word : words) {
    System.out.println(word);
}
```

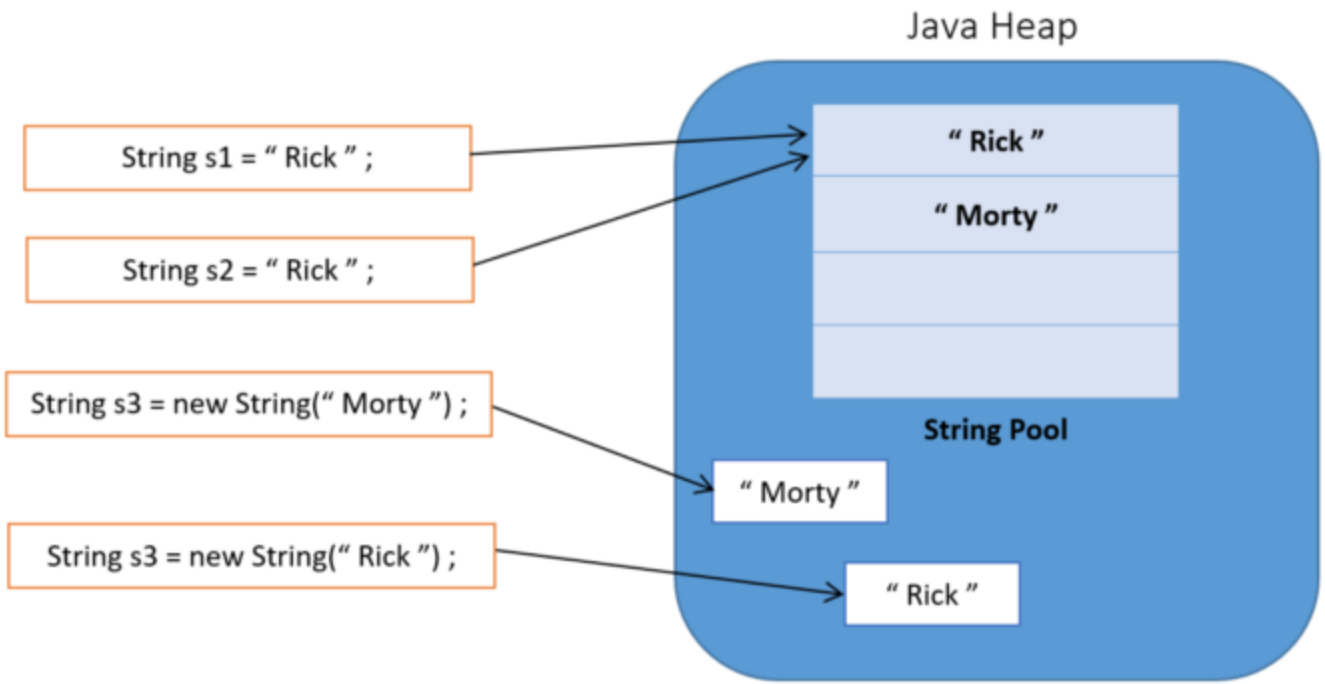
split()

```
FIFA
will
never
regret
it
```

String Pool

String Pool

String Poll (**Пул строк**) - это набор строк, который хранится в **Heap**.



String Pool

- При создании объекта через оператор `new` строка не помещается в **String Pool**.
- Для того чтобы поместить строку в **String Pool** используется метод `intern()`.

StringBuffer и StringBuilder

Immutable `String`

Класс `String` **immutable** (неизменяемый).

```
String str = "Hello";
str += " Java";
```

Код приведенный выше, приводит к тому, что создается новый объект, и содержимое обеих исходных строк в него копируется.

`StringBuffer` и `StringBuilder`

Эту проблему решают объекты типа `StringBuilder` или `StringBuffer`. Оба эти класса позволяют менять содержимое находящихся в них строк.

```
String str = "Hello";
StringBuilder strBuilder = new StringBuilder(str);
strBuilder.append(" Java");
```

`StringBuffer` и `StringBuilder`

- Класс `StringBuilder` - *NOT thread safe* (потокоНЕбезопасный), но быстрый
- Класс `StringBuffer` - *thread safe* (потокобезопасный), но медленный

`StringBuffer`

Constructors:

- `StringBuffer()`
- `StringBuffer(int capacity)`
- `StringBuffer(String str)`
- `StringBuffer(CharSequence chars)`

`StringBuilder`

Constructors:

- `StringBuilder()`
- `StringBuilder(int capacity)`
- `StringBuilder(String str)`
- `StringBuilder(CharSequence chars)`

StringBuffer и StringBuilder

```
String str = "Java";
StringBuffer strBuffer = new StringBuffer(str);
System.out.println("Емкость: " + strBuffer.capacity()); // 20
strBuffer.ensureCapacity(32);
System.out.println("Емкость: " + strBuffer.capacity()); // 42
System.out.println("Длина: " + strBuffer.length()); // 4
```

charAt() и setCharAt()

```
StringBuffer strBuffer = new StringBuffer("Java");
char c = strBuffer.charAt(0); // J
System.out.println(c);
strBuffer.setCharAt(0, 'c');
System.out.println(strBuffer.toString()); // cava
```

getChars()

```
StringBuffer strBuffer = new StringBuffer("world");
int startIndex = 1;
int endIndex = 4;
char[] buffer = new char[endIndex - startIndex];
strBuffer.getChars(startIndex, endIndex, buffer, 0);
System.out.println(buffer); // orl
```

append()

```
StringBuffer strBuffer = new StringBuffer("hello");
strBuffer.append(" world");
System.out.println(strBuffer.toString()); // hello world
```

insert()

```
StringBuffer strBuffer = new StringBuffer("word");

strBuffer.insert(3, 'l');
System.out.println(strBuffer.toString()); // world

strBuffer.insert(0, "s");
System.out.println(strBuffer.toString()); // sworld
```

delete() и deleteCharAt()

```
StringBuffer strBuffer = new StringBuffer("assembler");
strBuffer.delete(0,2);
System.out.println(strBuffer.toString()); // sembler

strBuffer.deleteCharAt(6);
System.out.println(strBuffer.toString()); // semble
```

substring()

```
StringBuffer strBuffer = new StringBuffer("hello java!");
String str1 = strBuffer.substring(6); // обрезка строки с 6 символа до конца
System.out.println(str1); //java!

String str2 = strBuffer.substring(3, 9); // обрезка строки с 3 по 9 символ
System.out.println(str2); //lo jav
```

setLength()

```
StringBuffer strBuffer = new StringBuffer("hello");
strBuffer.setLength(10);
System.out.println(strBuffer.toString()); // "hello     "

strBuffer.setLength(4);
System.out.println(strBuffer.toString()); // "hell"
```

replace()

```
StringBuffer strBuffer = new StringBuffer("hello world!");
strBuffer.replace(6, 11, "java");
System.out.println(strBuffer.toString()); // hello java!
```

reverse()

```
StringBuffer strBuffer = new StringBuffer("assembler");
strBuffer.reverse();
System.out.println(strBuffer.toString()); // relbmessa
```

Regular Expression in `String`

String: split()

```
String text = "FIFA will never regret it";
String[] words = text.split("\\s*(\\s|,|!|\\.)\\s*");
for (String word : words) {
    System.out.println(word);
}
```

String: matches()

```
String input = "+12343454556";
boolean result = input.matches("(\\+*)\\d{11}");
if (result == true) {
    System.out.println("It is a phone number");
} else {
    System.out.println("It is not a phone number!");
}
```

String: replaceAll()

```
String input = "Hello Java! Hello JavaScript! JavaSE 8.";
String myStr =input.replaceAll("Java(\\w*)", "HTML");
System.out.println(myStr); // Hello HTML! Hello HTML! HTML 8.
```

Regular Expression with `Pattern` and `Matcher`

Regular Expression with `Pattern` and `Matcher`

- Более мощные средства, для работы с регулярными выражениями предлагают классы `Pattern` и `Matcher` из пакета `java.util.regex`.
- Класс `Pattern` служит для хранения регулярного выражения
- Класс `Matcher` служит для выполнения операций поиска и сравнения.

Pattern: matches()

```
import java.util.regex.Pattern;

public class StringsApp {
    public static void main(String[] args) {
        String input = "Hello";
        boolean found = Pattern.matches("Hello", input);
        if (found) {
            System.out.println("Найдено");
        } else {
            System.out.println("Не найдено");
        }
    }
}
```

Pattern: split()

```
import java.util.regex.Pattern;

public class StringsApp {
    public static void main(String[] args) {
        String input = "Hello Java! Hello JavaScript! JavaSE 8.";
        Pattern pattern = Pattern.compile("[ ,.!?]");
        String[] words = pattern.split(input);
        for (String word : words) {
            System.out.println(word);
        }
    }
}
```

Pattern: split()

```
Hello
Java

Hello
JavaScript

JavaSE
8
```

Matcher: matches()

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class StringsApp {
    public static void main(String[] args) {
        String input = "Hello";
        Pattern pattern = Pattern.compile("Hello");
        Matcher matcher = pattern.matcher(input);
        boolean found = matcher.matches();
        if (found) {
            System.out.println("Найдено");
        } else {
            System.out.println("Не найдено");
        }
    }
}
```

Matcher: find() и group()

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class StringsApp {
    public static void main(String[] args) {
        String input = "Hello Java! Hello JavaScript! JavaSE 8.";
        Pattern pattern = Pattern.compile("Java(\\w*)");
        Matcher matcher = pattern.matcher(input);
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

Matcher: find() и group()

```
Java
JavaScript
JavaSE
```

Matcher: replaceAll()

```
String input = "Hello Java! Hello JavaScript! JavaSE 8.";
Pattern pattern = Pattern.compile("Java(\\w*)");
Matcher matcher = pattern.matcher(input);
String newStr = matcher.replaceAll("HTML");
System.out.println(newStr); // Hello HTML! Hello HTML! HTML 8.
```