

JAVA CHEAT SHEET

Импортиране на необходими библиотеки

За сигурно просто импорт на всичко

```
import java.io.*;
```

```
import java.net.*;
```

```
import java.util.*;
```

Видове променливи

- byte, short, int, long – целочислени типове;
- float, double – реални типове с плаваща запетая;
- boolean – булев (логически) тип;
- char – символен тип;
- Object – обектен тип ;
- String – символни низове.

```
public/private/protected double balance;
```

```
public static final double INTEREST_RATE = 0.02;
```

Масиви

```
arr2 = new int[10];
```

```
arr2[6] = 0;
```

```
String[] stringArray = new String[10];
```

```
int sixth = arr2[6];
```

```
String[] stringArray = new String[10];
```

```
for(int i = 0; i < stringArray.length; i++) {
```

```
    stringArray[i] = "String no " + i;
```

```
}
```

```
for(int i = 0; i < stringArray.length; i++) {
```

```
    System.out.println(stringArray[i]);
```

```
}
```

Стандартен вход и изход

```
Scanner scanner = new Scanner(System.in);           System.out.println("Some text.");

String name = scanner.nextLine();                   System.out.println("text 1 " + "text 2");

int age = scanner.nextInt();                         System.out.println("x = " + x);

double value = scanner.nextDouble();                System.out.format("%s", "Hello World\n");
```

Методи

[**public**] [**static**] [**return_type**] [**method_name**] ([**param_list**])

```
private          void          printInfo          (String inf)          { System.out.println(inf); }
```

```
public class ServiceCenter {
```

```
    public int registerOrder(String carRegNumber, String description) {

        for (RepairOrder order : orders) {

            if (order.getCarRegNumber().equalsIgnoreCase(carRegNumber) &&

                (order.getStatus().equals("NEW") || order.getStatus().equals("IN_PROGRESS"))) {

                return -1;

            }

        }

        int id = nextId++;

        orders.add(new RepairOrder(id, carRegNumber, description, "NEW"));

        return id;

    }

}
```

Класове

Декларация

```
[abstract] public class Cheese  
  
                / extends Food  
  
                / implements Eatable
```

Пренаписване на метод

```
@Override  
  
public String toString() {  
  
    return "RepairOrder{id=" + id + ", car=" + carRegNumber + ", status=" + status + "}";  
  
}
```

Абстрактни класове

```
public abstract class AbstractClassExample {  
  
    public void someMethod() {}  
  
    public abstract void firstAbstractMethod();  
  
}
```

Интерфейси

```
public interface FirstInterface {  
  
    void method1();  
  
    void method2();  
  
}
```

!!! Всички методи са по подразбиране `public abstract`

!!! Променливите в интерфейс са винаги `public static final` и изискват инициализация

Полиморфизъм

```
Figure f1 = new Triangle();

Figure f2 = new Rectangle();

System.out.println(f1.area(4, 5)); // 10.0

System.out.println(f2.area(4, 5)); // 20.0
```

Изключения

```
try {

    int[] arr = new int[3];

    System.out.println(arr[5]);

} catch (ArrayIndexOutOfBoundsException e) {

    System.out.println("Invalid index.");

} finally {

    System.out.println("This always executes.");

}
```

!!! `catch` блоковете трябва да са подредени от по-конкретен към по-общ.

!!! `finally` се изпълнява винаги, независимо дали е хвърлено изключение.

```
class PriceErrorException extends Exception {

    @Override

    public String getMessage() { return "Invalid price!". }

}
```

Стрингове

```
String a = "hello";
```

- `charAt(i)`, `equals()`, `equalsIgnoreCase()`, `compareTo()`
- `startsWith()`, `endsWith()`, `indexOf()`, `lastIndexOf()`
- `substring()`, `concat()`, `replace()`, `replaceAll()`
- `split()`, `toLowerCase()`, `toUpperCase()`, `trim()`

