История

Python	1991
R	1993
C#	2005
Java	2014

- Java 8
- Lambda Expressions
- Stream API

Конструкция

```
(Object obj) -> { return something; }

Class::method

Class::new
```

Сравнение

```
(Object obj) -> { return “lambda”; }
```

```
public String retString(Object obj)  {
        return “lambda”; }
```

```
() -> { return “lambda”; }
```

```
public String retString()  {
        return “lambda”; }
```

```
() -> System.out.println(“lambda)
```

```
public void printString()  {
        System.out.println( “lambda”); }
```

Примеры

```
List<Integer> values = Arrays.asList(new Integer[]{0,1,2,3,4,5,6,7,8});

// Full notation

values.forEach((Integer v) -> { System.out.println(v); });

// short input

values.forEach(v -> { System.out.println(v); });

// short notation

values.forEach(v -> System.out.println(v));

// link to method

values.forEach(System.out::println);
```

Примеры

```
values.sort(new Comparator<Integer>() {
        @Override
        public int compare(Integer arg0, Integer arg1) { return arg0 - arg1; }
});
for (Integer v : values)
        System.out.print(v + " ");


values.sort((v0, v1) -> v0 - v1);
System.out.println(values.stream().map(String::valueOf).collect(Collectors.joining(" ")));
```

@FunctionalInterface

```
@FunctionalInterface

interface IFuncTest {

    public boolean test(Integer v);

}

IFuncTest ift = i -> i > 5;

for (Integer v: values)

    if (ift.test(v))

        System.out.print(v + " ");
```

Predicate

```
@FunctionalInterface
public interface Predicate<T> {
    boolean test(T arg0);
…


Predicate<Integer> pft = i -> i > 5;
values.removeIf(pft);
```

removeIf

```
default boolean removeIf(Predicate<? super E> arg0) {

        Objects.requireNonNull(arg0); boolean arg1 = false; Iterator arg2 =
this.iterator();

        while (arg2.hasNext()) {

            if (arg0.test(arg2.next())) {

                arg2.remove();

                arg1 = true;

        }}

        return arg1; }
```

java.util.*

public interface Predicate<T>

public interface Function<T,R>

public interface BiPredicate<T,U>

public interface BiFunction<T,U,R>

public interface IntPredicate

public interface DoublePredicate

public interface DoubleToIntFunction

public interface DoubleToLongFunction

Области видимости

```
  public static void testF(String a, String b) {
//        a = "NOPE";
      Runnable r = () -> {
//            b = "NOPE";
          System.out.println(a);
          System.out.println(b);
      };
      new Thread(r).start();
  }
```

Stream API

```
sValues.stream().min(Integer::compareTo).get()

sValues.stream().max(Integer::compareTo).get()



sValues.parallelStream().filter(v -> v % 2 == 0 ).collect(Collectors.toList())

sValues.stream().map(String::valueOf).collect(Collectors.joining(" "))
```

Ссылки

- https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html#syntax
- https://docs.oracle.com/javase/8/docs/api/java/util/function/Predicate.html
- https://docs.oracle.com/javase/8/docs/api/java/util/function/package-summary.html
- https://docs.oracle.com/javase/tutorial/collections/streams/
- https://docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html