**Package java.util.function**

Functional interfaces provide target types for lambda expressions and method references.

**Basic ones:**

# Supplier<T>

Represents a supplier of results.

(void) -> T

# Consumer<T>

Represents an operation that accepts a single input argument and returns no result.

(T) -> void

# BiConsumer<T,U>

Represents an operation that accepts two input arguments and returns no result.

(T, U) -> void

# Function<T,R>

Represents a function that accepts one argument and produces a result.

(T) -> R

# BiFunction<T,U,R>

Represents a function that accepts two arguments and produces a result.

(T, U) -> R

# Predicate<T>

Represents a predicate (boolean-valued function) of one argument.

(T) -> Boolean

# BiPredicate<T,U>

Represents a predicate (boolean-valued function) of two arguments.

(T, U) -> boolean

# UnaryOperator<T>

Represents an operation on a single operand that produces a result of the same type as its operand.

(T) -> T

# BinaryOperator<T>

Represents an operation upon two operands of the same type, producing a result of the same type as the operands.

(T, T) -> T

**Extended ones:**

# BooleanSupplier

Represents a supplier of boolean-valued results.

(void) -> Boolean

# DoubleBinaryOperator

Represents an operation upon two double-valued operands and producing a double-valued result.

(double, double) -> double

# DoubleConsumer

Represents an operation that accepts a single double-valued argument and returns no result.

(double) -> void

# DoubleFunction<R>

Represents a function that accepts a double-valued argument and produces a result.

(double) -> R

# DoublePredicate

Represents a predicate (boolean-valued function) of one double-valued argument.

(double) -> boolean

# DoubleSupplier

Represents a supplier of double-valued results.

(void) -> double

# DoubleToIntFunction

Represents a function that accepts a double-valued argument and produces an int-valued result.

(double) -> int

# DoubleToLongFunction

Represents a function that accepts a double-valued argument and produces a long-valued result.

(double) -> long

# DoubleUnaryOperator

Represents an operation on a single double-valued operand that produces a double-valued result.

(double) -> double

# IntBinaryOperator

Represents an operation upon two int-valued operands and producing an int-valued result.

(int, int) -> int

# IntConsumer

Represents an operation that accepts a single int-valued argument and returns no result.

(int) -> void

# IntFunction<R>

Represents a function that accepts an int-valued argument and produces a result.

(int) -> R

# IntPredicate

Represents a predicate (boolean-valued function) of one int-valued argument.

(int) -> boolean

# IntSupplier

Represents a supplier of int-valued results.

(void) -> int

# IntToDoubleFunction

Represents a function that accepts an int-valued argument and produces a double-valued result.

(int) -> double

# IntToLongFunction

Represents a function that accepts an int-valued argument and produces a long-valued result.

(int) -> long

# IntUnaryOperator

Represents an operation on a single int-valued operand that produces an int-valued result.

(int) -> int

# LongBinaryOperator

Represents an operation upon two long-valued operands and producing a long-valued result.

(long, long) -> long

# LongConsumer

Represents an operation that accepts a single long-valued argument and returns no result.

(long) -> void

# LongFunction<R>

Represents a function that accepts a long-valued argument and produces a result.

(long) -> R

# LongPredicate

Represents a predicate (boolean-valued function) of one long-valued argument.

(long) -> Boolean

# LongSupplier

Represents a supplier of long-valued results.

(void) -> long

# LongToDoubleFunction

Represents a function that accepts a long-valued argument and produces a double-valued result.

(long) -> double

# LongToIntFunction

Represents a function that accepts a long-valued argument and produces an int-valued result.

(long) -> int

# LongUnaryOperator

Represents an operation on a single long-valued operand that produces a long-valued result.

(long) -> long

# ObjDoubleConsumer<T>

Represents an operation that accepts an object-valued and a double-valued argument, and returns no result.

(T, double) -> void

# ObjIntConsumer<T>

Represents an operation that accepts an object-valued and a int-valued argument, and returns no result.

(T, int) -> void

# ObjLongConsumer<T>

Represents an operation that accepts an object-valued and a long-valued argument, and returns no result.

(T, long) -> void

# ToDoubleBiFunction<T,U>

Represents a function that accepts two arguments and produces a double-valued result.

(T, U) -> double

# ToDoubleFunction<T>

Represents a function that produces a double-valued result.

(T) -> double

# ToIntBiFunction<T,U>

Represents a function that accepts two arguments and produces an int-valued result.

(T, U) -> int

# ToIntFunction<T>

Represents a function that produces an int-valued result.

(T) -> int

# ToLongBiFunction<T,U>

Represents a function that accepts two arguments and produces a long-valued result.

(T, U) -> long

# ToLongFunction<T>

Represents a function that produces a long-valued result.

(T) -> long