# Understand Java Functional Interfaces: Overview

**Douglas C. Schmidt**
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
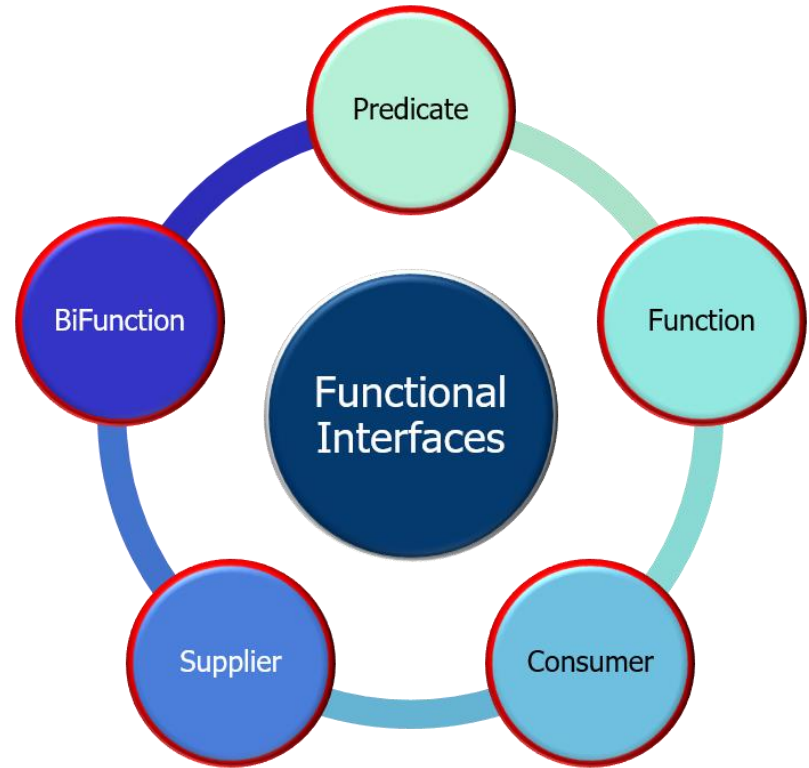
Institute for Software
Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

# Learning Objectives in this Part of the Lesson

- Understand foundational functional programming features in Java 8, e.g.,
  - Lambda expressions
  - Method & constructor references
  - Key functional interfaces

# Learning Objectives in this Part of the Lesson

- Understand foundational functional programming features in Java 8, e.g.,

  - Lambda expressions

  - Method & constructor references
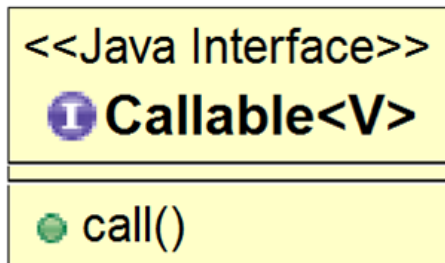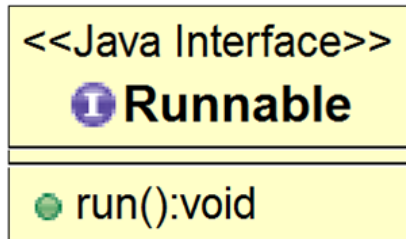
  - Key functional interfaces

These features form the basis for Java streams & concurrency/parallelism frameworks

# Overview of Functional Interfaces

# Overview of Functional Interfaces

- A *functional interface* is an interface that contains only one abstract method

```
<<Java Interface>>
  Runnable

  run():void
```

```
<<Java Interface>>
  Callable<V>

  call()
```

THERE CAN BE
ONLY ONE

# Overview of Functional Interfaces

- A functional interface is the type used for a parameter when a lambda expression or method reference is passed as an argument to a method

```java
<T> void runTest(Function<T, T> fact, T n) {
  long startTime = System.nanoTime();
  T result = fact.apply(n);
  long stopTime = (System.nanoTime() - startTime) / 1_000_000;
  ...
}
runTest(ParallelStreamFactorial::factorial, n);
runTest(SequentialStreamFactorial::factorial, n);
...
```

See www.baeldung.com/java-8-functional-interfaces

# Overview of Functional Interfaces

- A functional interface is the type used for a parameter when a lambda expression or method reference is passed as an argument to a method

```
<T> void runTest(Function<T, T> fact, T ...
  long startTime = System.nanoTime();
  T result = fact.apply(n);
  long stopTime = (System.nanoTime() - startTime) / 1_000_000;
  ...
}
runTest(ParallelStreamFactorial::factorial, n);
runTest(SequentialStreamFactorial::factorial, n);
...
```

*Record & print time taken to compute 'n' factorial*

See github.com/douglascraigschmidt/LiveLessons/tree/master/Java8/ex16

# Overview of Functional Interfaces

- A functional interface is the type used for a parameter when a lambda expression or method reference is passed as an argument to a method

```
<T> void runTest(Function<T, T> fact, T n)
    long startTime = System.nanoTime();
    T result = fact.apply(n);
    long stopTime = (System.nanoTime() - startTime) / 1_000_000;
    ...
}
runTest(ParallelStreamFactorial::factorial, n);
runTest(SequentialStreamFactorial::factorial, n);
...
```

*'fact' parameterizes the factorial implementation*

# Overview of Functional Interfaces

- A functional interface is the type used for a parameter when a lambda expression or method reference is passed as an argument to a method

```
<T> void runTest(Function<T, T> fact, T n) {
  long startTime = System.nanoTime();
  T result = fact.apply(n);
  long stopTime = (System.nanoTime() - startTime) / 1_000_000;
  ...
}
runTest(ParallelStreamFactorial::factorial, n);
runTest(SequentialStreamFactorial::factorial, n);
...
```

*Different factorial implementations can be passed as method reference params to the runTest() method*

This is an example of behavior parameterization

# Overview of Functional Interfaces

- A functional interface is the type used for a parameter when a lambda expression or method reference is passed as an argument to a method

```java
<T> void runTest(Function<T, T> fact, T n) {
  long startTime = System.nanoTime();
  T result = fact.apply(n);
  long stopTime = (System.nanoTime() - startTime) / 1_000_000;
  ...
}
runTest(ParallelStreamFactorial::factorial, n);
```

```java
static BigInteger factorial(BigInteger n) {
  return LongStream.rangeClosed(1, n)
                .parallel()
                .mapToObj(BigInteger::valueOf)
                .reduce(BigInteger.ONE, BigInteger::multiply);
}
```

# Summary of Common Functional Interfaces

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces

**Package java.util.function**

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

See: Description

**Interface Summary**

| Interface | Description |
|---|---|
| BiConsumer<T,U> | Represents an operation that accepts two input arguments and returns no result. |
| BiFunction<T,U,R> | Represents a function that accepts two arguments and produces a result. |
| BinaryOperator<T> | Represents an operation upon two operands of the same type, producing a result of the same type as the operands. |
| BiPredicate<T,U> | Represents a predicate (boolean-valued function) of two arguments. |
| BooleanSupplier | Represents a supplier of boolean-valued results. |
| Consumer<T> | Represents an operation that accepts a single input argument and returns no result. |
| DoubleBinaryOperator | Represents an operation upon two double-valued operands and producing a double-valued result. |
| DoubleConsumer | Represents an operation that accepts a single double-valued argument and returns no result. |
| DoubleFunction<R> | Represents a function that accepts a double-valued argument and produces a result. |
| DoublePredicate | Represents a predicate (boolean-valued function) of one double-valued argument. |
| DoubleSupplier | Represents a supplier of double-valued results. |
| DoubleToIntFunction | Represents a function that accepts a double-valued argument and produces an int-valued result. |
| DoubleToLongFunction | Represents a function that accepts a double-valued argument and produces a long-valued result. |
| DoubleUnaryOperator | Represents an operation on a single double-valued operand that produces a double-valued result. |
| Function<T,R> | Represents a function that accepts one argument and produces a result. |

See docs.oracle.com/javase/8/docs/api/java/util/function/package-summary.html

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces

  - Some of these interfaces handle reference types

## Package java.util.function

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

See: Description

### Interface Summary

| Interface | Description |
|---|---|
| BiConsumer<T,U> | Represents an operation that accepts two input arguments and returns no result. |
| BiFunction<T,U,R> | Represents a function that accepts two arguments and produces a result. |
| BinaryOperator<T> | Represents an operation upon two operands of the same type, producing a result of the same type as the operands. |
| BiPredicate<T,U> | Represents a predicate (boolean-valued function) of two arguments. |
| BooleanSupplier | Represents a supplier of boolean-valued results. |
| Consumer<T> | Represents an operation that accepts a single input argument and returns no result. |
| DoubleBinaryOperator | Represents an operation upon two double-valued operands and producing a double-valued result. |
| DoubleConsumer | Represents an operation that accepts a single double-valued argument and returns no result. |
| DoubleFunction<R> | Represents a function that accepts a double-valued argument and produces a result. |
| DoublePredicate | Represents a predicate (boolean-valued function) of one double-valued argument. |
| DoubleSupplier | Represents a supplier of double-valued results. |
| DoubleToIntFunction | Represents a function that accepts a double-valued argument and produces an int-valued result. |
| DoubleToLongFunction | Represents a function that accepts a double-valued argument and produces a long-valued result. |
| DoubleUnaryOperator | Represents an operation on a single double-valued operand that produces a double-valued result. |
| Function<T,R> | Represents a function that accepts one argument and produces a result. |

See www.oreilly.com/library/view/java-8-pocket/9781491901083/ch04.html

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces
  - Some of these interfaces handle reference types
  - Other interfaces support primitive types

**Package java.util.function**

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

See: Description

**Interface Summary**

| Interface | Description |
|---|---|
| IntConsumer | Represents an operation that accepts a single int-valued argument and returns no result. |
| IntFunction<R> | Represents a function that accepts an int-valued argument and produces a result. |
| IntPredicate | Represents a predicate (boolean-valued function) of one int-valued argument. |
| IntSupplier | Represents a supplier of int-valued results. |
| IntToDoubleFunction | Represents a function that accepts an int-valued argument and produces a double-valued result. |
| IntToLongFunction | Represents a function that accepts an int-valued argument and produces a long-valued result. |
| IntUnaryOperator | Represents an operation on a single int-valued operand that produces an int-valued result. |
| LongBinaryOperator | Represents an operation upon two long-valued operands and producing a long-valued result. |
| LongConsumer | Represents an operation that accepts a single long-valued argument and returns no result. |
| LongFunction<R> | Represents a function that accepts a long-valued argument and produces a result. |
| LongPredicate | Represents a predicate (boolean-valued function) of one long-valued argument. |
| LongSupplier | Represents a supplier of long-valued results. |
| LongToDoubleFunction | Represents a function that accepts a long-valued argument and produces a double-valued result. |
| LongToIntFunction | Represents a function that accepts a long-valued argument and produces an int-valued result. |
| LongUnaryOperator | Represents an operation on a single long-valued operand that produces a long-valued result. |
| ObjDoubleConsumer<T> | Represents an operation that accepts an object-valued and a double-valued argument, and returns no result. |
| ObjIntConsumer<T> | Represents an operation that accepts an object-valued and a int-valued argument, and returns no result. |

See docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces

  - Some of these interfaces handle reference types

  - Other interfaces support primitive types

    - Avoids "auto-boxing" overhead



**Package java.util.function**

*Functional interfaces* provide target types for lambda expressions and method references.
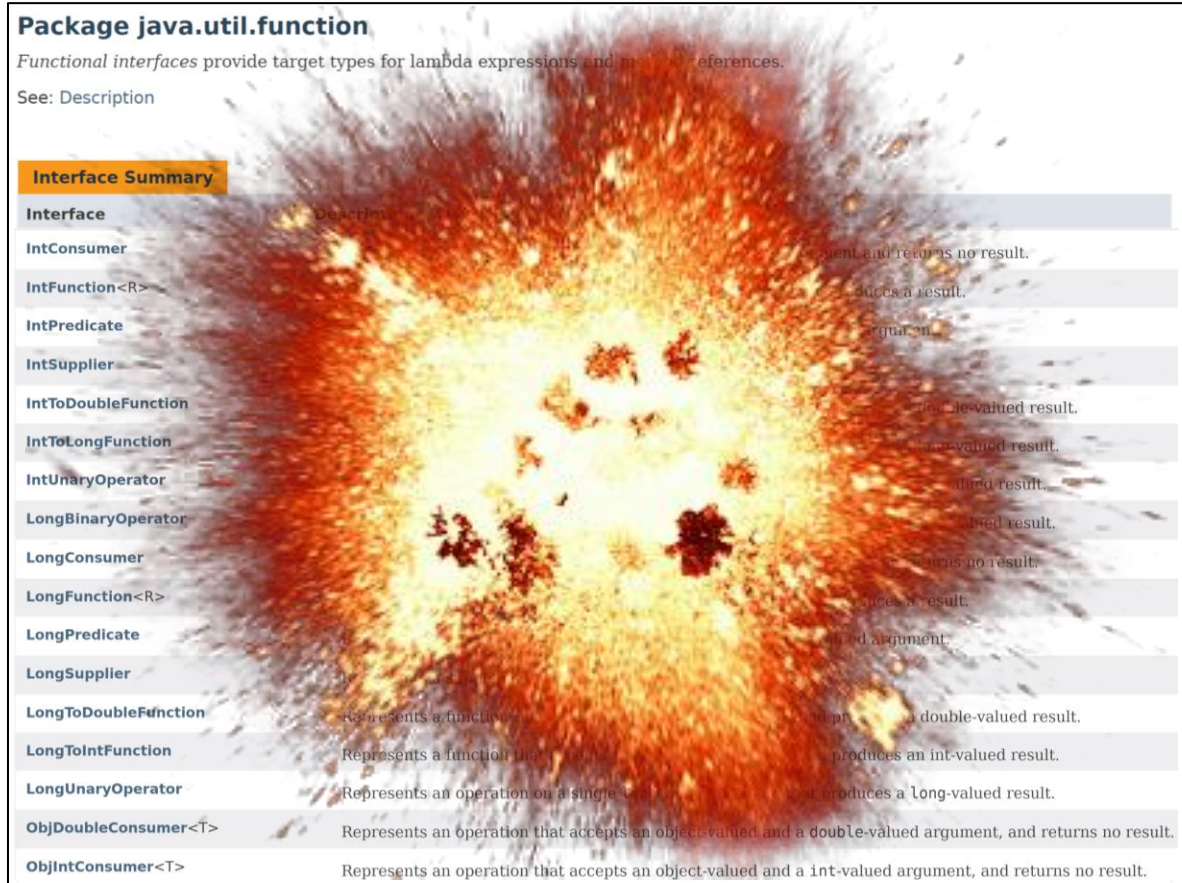
See: Description

**Interface Summary**

| Interface | Description |
| --- | --- |
| IntConsumer | Represents an operation that accepts a single int-valued argument and returns no result. |
| IntFunction<R> | Represents a function that accepts an int-valued argument and produces a result. |
| IntPredicate | Represents a predicate (boolean-valued function) of one int-valued argument. |
| IntSupplier | Represents a supplier of int-valued results. |
| IntToDoubleFunction | Represents a function that accepts an int-valued argument and produces a double-valued result. |
| IntToLongFunction | Represents a function that accepts an int-valued argument and produces a long-valued result. |
| IntUnaryOperator | Represents an operation on a single int-valued operand that produces an int-valued result. |
| LongBinaryOperator | Represents an operation upon two long-valued operands and producing a long-valued result. |
| LongConsumer | Represents an operation that accepts a single long-valued argument and returns no result. |
| LongFunction<R> | Represents a function that accepts a long-valued argument and produces a result. |
| LongPredicate | Represents a predicate (boolean-valued function) of one long-valued argument. |
| LongSupplier | Represents a supplier of long-valued results. |
| LongToDoubleFunction | Represents a function that accepts a long-valued argument and produces a double-valued result. |
| LongToIntFunction | Represents a function that accepts a long-valued argument and produces an int-valued result. |
| LongUnaryOperator | Represents an operation on a single long-valued operand that produces a long-valued result. |
| ObjDoubleConsumer<T> | Represents an operation that accepts an object-valued and a double-valued argument, and returns no result. |
| ObjIntConsumer<T> | Represents an operation that accepts an object-valued and a int-valued argument, and returns no result. |

See rules.sonarsource.com/java/tag/performance/RSPEC-4276

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces

  - Some of these interfaces handle reference types

  - Other interfaces support primitive types.

- There's an explosion of Java functional interfaces!



**Package java.util.function**

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

See: Description

**Interface Summary**

| Interface | Description |
| --- | --- |
| IntConsumer | ...ment and returns no result. |
| IntFunction<R> | ...duces a result. |
| IntPredicate | ...rgumen... |
| IntSupplier | |
| IntToDoubleFunction | ...le-valued result. |
| IntToLongFunction | ...ar-valued result. |
| IntUnaryOperator | ...lued result. |
| LongBinaryOperator | ...lued result. |
| LongConsumer | ...rns no result. |
| LongFunction<R> | ...uces a result. |
| LongPredicate | ...d argument. |
| LongSupplier | |
| LongToDoubleFunction | Represents a functio... ...d pr... ...double-valued result. |
| LongToIntFunction | Represents a function tha... ...produces an int-valued result. |
| LongUnaryOperator | Represents an operation on a single ... ...t produces a long-valued result. |
| ObjDoubleConsumer<T> | Represents an operation that accepts an object-valued and a double-valued argument, and returns no result. |
| ObjIntConsumer<T> | Represents an operation that accepts an object-valued and a int-valued argument, and returns no result. |

See dzone.com/articles/whats-wrong-java-8-part-ii

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces

  - Some of these interfaces handle reference types

  - Other interfaces support primitive types.

- There's an explosion of Java functional interfaces!

  - However, learn these interfaces before trying to customize your own

**Package java.util.function**

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

See: Description

**Interface Summary**

| Interface | Description |
|---|---|
| IntConsumer | Represents an operation that accepts a single int-valued argument and returns no result. |
| IntFunction<R> | Represents a function that accepts an int-valued argument and produces a result. |
| IntPredicate | Represents a predicate (boolean-valued function) of one int-valued argument. |
| IntSupplier | Represents a supplier of int-valued results. |
| IntToDoubleFunction | Represents a function that accepts an int-valued argument and produces a double-valued result. |
| IntToLongFunction | Represents a function that accepts an int-valued argument and produces a long-valued result. |
| IntUnaryOperator | Represents an operation on a single int-valued operand that produces an int-valued result. |
| LongBinaryOperator | Represents an operation upon two long-valued operands and producing a long-valued result. |
| LongConsumer | Represents an operation that accepts a single long-valued argument and returns no result. |
| LongFunction<R> | Represents a function that accepts a long-valued argument and produces a result. |
| LongPredicate | Represents a predicate (boolean-valued function) of one long-valued argument. |
| LongSupplier | Represents a supplier of long-valued results. |
| LongToDoubleFunction | Represents a function that accepts a long-valued argument and produces a double-valued result. |
| LongToIntFunction | Represents a function that accepts a long-valued argument and produces an int-valued result. |
| LongUnaryOperator | Represents an operation on a single long-valued operand that produces a long-valued result. |
| ObjDoubleConsumer<T> | Represents an operation that accepts an object-valued and a double-valued argument, and returns no result. |
| ObjIntConsumer<T> | Represents an operation that accepts an object-valued and a int-valued argument, and returns no result. |

See tutorials.jenkov.com/java-functional-programming/functional-interfaces.html

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces.

  - Some of these interfaces handle reference types.

  - Other interfaces support primitive types.

  - There's an explosion of Java functional interfaces!

  *We focus on the most common types of functional interfaces*
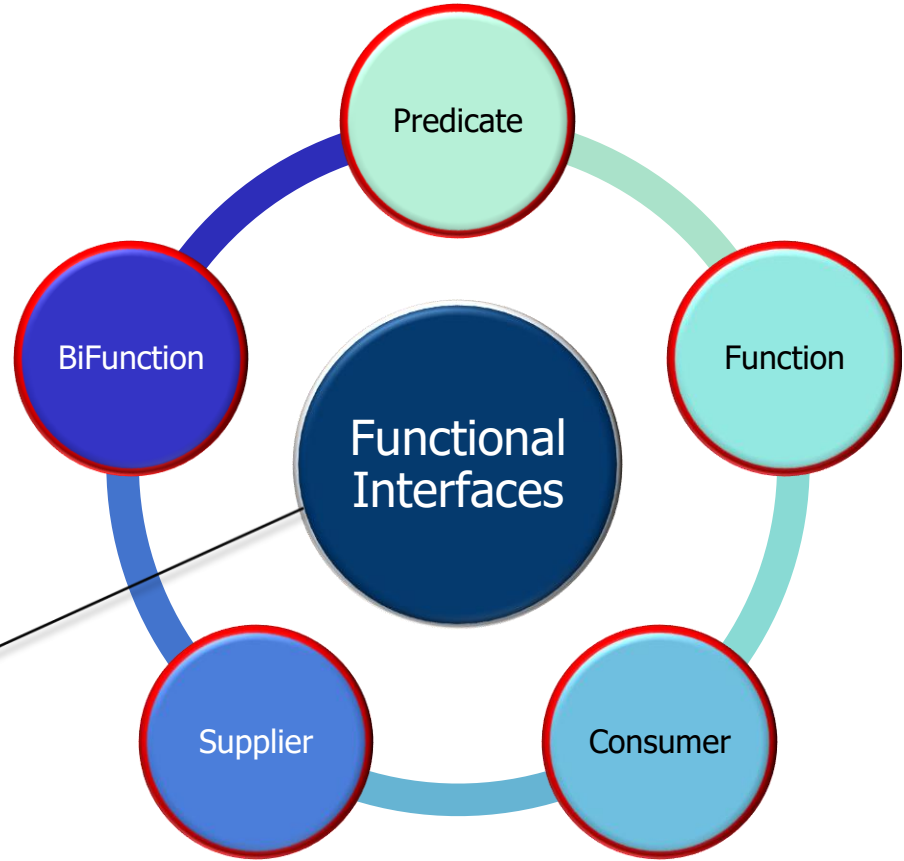
Predicate

BiFunction

Function

Functional Interfaces

Supplier

Consumer

# Summary of Common Functional Interfaces

- Java defines many types of functional interfaces.

  - Some of these interfaces handle reference types.

  - Other interfaces support primitive types.

  - There's an explosion of Java functional interfaces!
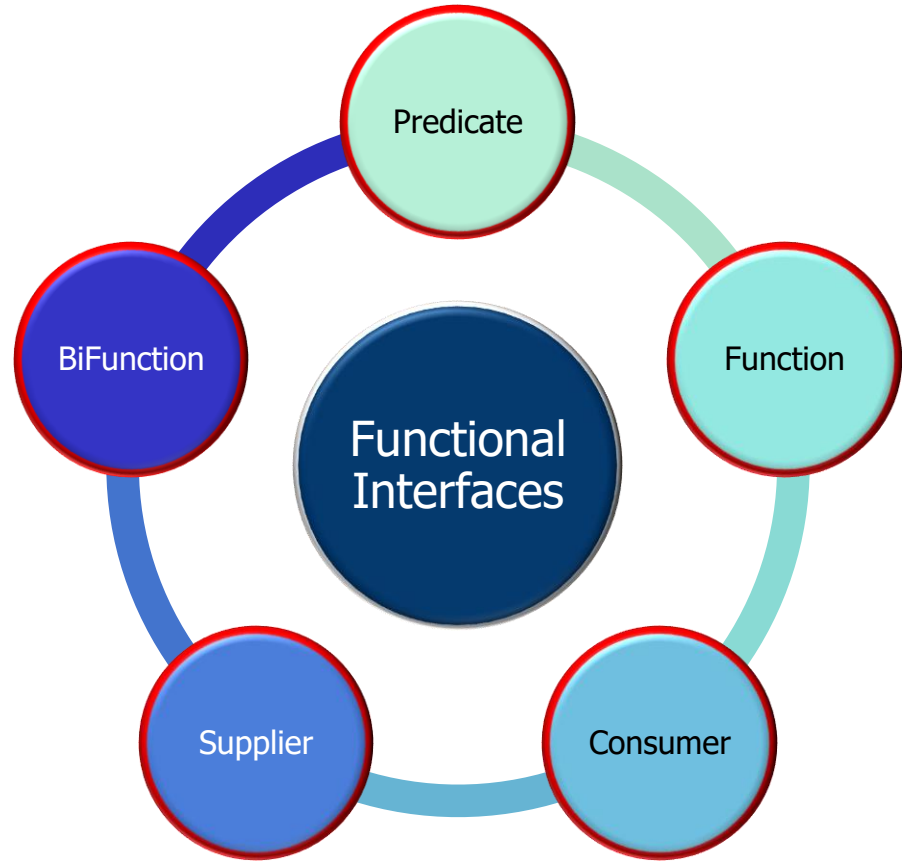


All usages of functional interfaces in the upcoming examples are "stateless"!

# End of Understand Java Functional Interfaces: Overview