Quick Tour

Java Language Features 8 - 17

Sunil Ghargaonkar

Java 8

- Stream API
- Functional Interface
- Lambda Expressions
- Interface Default & Static Method
- Optional Class
- Method Reference

Stream API

```java
public List<Account> getAccountDetailsFromAccountNumber(final long accountNumber) throws FileNotFoundException {
    final List<Account> accountData = new ArrayList<>();

    for (Account accountRecord : readData.read()) {
        if (accountNumber == accountRecord.getAccountNumber()) {
            accountData.add(accountRecord);
        }
    }

    return accountData;
}
```

```java
public List<Account> getAccountDetailsFromAccountNumber(final long accountNumber) throws FileNotFoundException {

    return readData.read() List<Account>
            .stream() Stream<Account>
            .filter(a -> a.getAccountNumber() == accountNumber)
            .collect(Collectors.toList());
}
```

Functional Interface & Lambda Expression

```java
@FunctionalInterface
public interface Square {
    int calculate(int n);
}


@Test
void calculateSquareOfGivenNumber() {

    final Square square = new Square() {
        @Override
        public int calculate(int n) {
            return n * n;
        }
    };

    System.out.println(square.calculate( n: 5));
}
```

```java
@FunctionalInterface
public interface Square {
    int calculate(int n);
}


@Test
void calculateSquareOfGivenNumber() {
    final Square square = (int n) -> n*n;

    System.out.println(square.calculate( n: 5));
}
```

Interface Default & Static Method

Default

- It is a method with default keyword
- We can provide the implementation in interface or let class provide it
- Class can override this method

Static

- It is a static method which belongs to the interface only.
- We have to provide implementation of this method in interface itself
- Class can not override this method

Interface Static & Default Method

```java
@Test
void defaultInterfaceMethodTest() {
    final Welcome welcome = new InviteGuest();

    Welcome.invite();
    welcome.greet();
    welcome.serve();
}

interface Welcome {
    static void invite() {
        System.out.println("Let's invite guests for the barbeque party"

    default void greet() {
        System.out.println("Hello Everyone!");
    }

    void serve();
}

static class InviteGuest implements Welcome {

    @Override
    public void serve() { System.out.println("Would you like to have so

    @Override
    public void greet() {
        System.out.println("Hello Java how is world?");
    }
}
}
```

```
est.defaultInterfaceMethodTest
Tests passed: 1 of 1 test – 19 ms
/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java ...
Let's invite guests for the barbeque party
Hello Java how is world?
Would you like to have some drinks

Process finished with exit code 0
```

Optional Class

```java
public class EmailStoreHeaderName implements DisplayNameProvider {
    private final EmailStore emailStore;
    private DisplayUserName displayUserName;

    public EmailStoreHeaderName(final EmailStore emailStore) {
        this.emailStore = emailStore;

    }

    @Override
    public DisplayUserName getDisplayName(final long type) {
        if(emailStore != null){
            final String headerName = emailStore.getFromHdrName();

            if(headerName != null && !(headerName.isEmpty())){
                displayUserName = new DisplayUserName(headerName, DisplayUserName.HeaderFromType.HEADER, isSuspicious: false, type);
            }
        }
        return displayUserName;
    }
}
```

```java
public class EmailStoreHeaderName implements DisplayNameProvider {
    private final EmailStore emailStore;

    public EmailStoreHeaderName(final EmailStore emailStore) {
        this.emailStore = emailStore;

    }

    @Override
    public Optional<DisplayUserName> getDisplayName(final long type) {

        return Optional.ofNullable(emailStore).map(EmailStore::getFromHdrName)
                .filter(headerName -> !headerName.isEmpty())
                .map(displayName -> new DisplayUserName(displayName, DisplayUserName.HeaderFromType.HEADER, isSuspicious: false, type));
    }
}
```

Method Reference

```
@Override
public Optional<DisplayUserName> getDisplayName(final long type) {

    return Optional.ofNullable(emailStore).map(name -> name.getFromHdrName())
            .filter(headerName -> !headerName.isEmpty())
            .map(displayName -> new DisplayUserName(displayName, DisplayUserName.HeaderFromType.HEADER, isSuspicious: false, type));
}
```

```
@Override
public Optional<DisplayUserName> getDisplayName(final long type) {

    return Optional.ofNullable(emailStore).map(EmailStore::getFromHdrName)
            .filter(headerName -> !headerName.isEmpty())
            .map(displayName -> new DisplayUserName(displayName, DisplayUserName.HeaderFromType.HEADER, isSuspicious: false, type));
}
```

Java 9

- Module System
- Interface Private Methods
- Immutable Collection Methods
- Stream API Improvement
- Jshell

Immutable Collection Methods

```java
@Test
void listSetAndMapOf() {
    List<String> names = new ArrayList<>();
    Set<String> state = new HashSet<>();
    Map<String, String> personDetails = new HashMap<>();

    names.add("Sunil");
    names.add("Simran");

    state.add("Maharashtra");
    state.add("Punjab");

    personDetails.put("Sunil", "Maharashtra");
    personDetails.put("Simran", "Punjab");

    names.remove( index: 0);

    System.out.println(names);
}
```

```java
@Test
void listSetAndMapOf2() {
    List<String> names2 = List.of("Sunil", "Simran");
    Set<String> state2 = Set.of("Maharashtra", "Punjab");
    Map<String, String> personDetails2 = Map.of( k1: "Sunil", v1: "Maharashtra", k2: "Simran", v2: "Punjab");

    names2.remove( index: 0);
    System.out.println(names2);
}
}
```

```
tAndMapOf2
) Tests failed: 1 of 1 test – 16 ms

java.lang.UnsupportedOperationException Create breakpoint
    at java.base/java.util.ImmutableCollections.uoe(ImmutableCollections.java:142)
    at java.base/java.util.ImmutableCollections$AbstractImmutableList.remove(ImmutableCollections.java:258)
    at com.java.test.programs.CollectionFactoryMethodTest.listSetAndMapOf2(CollectionFactoryMethodTest.java:35) <31 int
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 internal lines>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <21 internal lines>
```

Stream API Improvement

- takeWhile
- dropWhile
- ofNullable
- Iterate

takeWhile

```java
    @Test
    void takeWhileVsFilter() {
        System.out.println("-----takewhile-----");
        IntStream.of(1, 10, 100, 1000, 10000, 1000, 100, 10, 1, 0, 10000)
                .takeWhile(i -> i < 5000)
                .forEach(System.out::println);

        System.out.println("-----filter-----");
        IntStream.of(1, 10, 100, 1000, 10000, 1000, 100, 10, 1, 0, 10000)
                .filter(i -> i < 5000)
                .forEach(System.out::println);

    }
}
```

Tests passed: 1 of 1 test – 19 ms

```
-----takewhile-----
1
10
100
1000
-----filter-----
1
10
100
1000
1000
100
10
1
0
```

dropWhile

```java
    @Test
    void dropWhileVsFilter() {
        System.out.println("-----takewhile-----");
        IntStream.of(2, 4, 6, 8, 9, 10, 12)
                .dropWhile(n -> n % 2 == 0)
                .forEach(System.out::println);

        System.out.println("-----filter-----");
        IntStream.of(2, 4, 6, 8, 9, 10, 12)
                .filter(n -> n % 2 == 0)
                .forEach(System.out::println);

    }
}
```

Tests passed: 1 of 1 test – 25 ms

```
/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java ...
-----takewhile-----
9
10
12
-----filter-----
2
4
6
8
10
12
```

jshell

```
sghargaonkar@mc-lon-nb15243 bin % jshell
|  Welcome to JShell -- Version 17.0.1
|  For an introduction type: /help intro

jshell> String program = "--Hello World--";
program ==> "--Hello World--"

jshell> program.replace('-','*');
$2 ==> "**Hello World**"

jshell>
```

Java 10

Unmodifiable Collection Copy

Optional orElseThrow()

Var for local variables

```
var length = str.length();
```

Time-Based Versioning

Unmodifiable Collection Copy

```
@Test
void listSetAndMapOf2() {
    List<String> names = List.of("Sunil", "Simran");
    Set<String> states = Set.of("Maharashtra", "Punjab");
    Map<String, String> personDetails2 = Map.of( k1: "Sunil",  v1: "Maharashtra",  k2: "Simran",  v2: "Punjab");

    Set<String> newStates = Set.copyOf(states);

    List<String> state = newStates.stream()
            .filter(s -> s.equals("Punjab"))
            .collect(Collectors.toUnmodifiableList());

}
```

orElseThrow()

```java
    @Test
    void listSetAndMapOf2() {
        List<String> names2 = List.of("Sunil", "Simran");
        Set<String> state2 = Set.of("Maharashtra", "Punjab");
        Map<String, String> personDetails2 = Map.of( k1: "Sunil",  v1: "Maharashtra",  k2: "Simran",  v2: "Punjab");

        state2.stream()
                .filter(n -> n.equals("Delhi"))
                .findFirst()
                .orElseThrow();
    }
}
```

```
stSetAndMapOf2 ×
Tests failed: 1 of 1 test – 22 ms
/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java ...

java.util.NoSuchElementException: No value present

    at java.base/java.util.Optional.orElseThrow(Optional.java:377)
    at com.java.test.programs.CollectionFactoryMethodTest.listSetAndMapOf2(CollectionFactoryMethodTest.java:38) <31
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 internal lines>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <21 internal lines>
```

Java -version

```
sghargaonkar@mc-lon-nb15243 ~ % java -version
openjdk version "17.0.1" 2021-10-19
OpenJDK Runtime Environment (build 17.0.1+12-39)
OpenJDK 64-Bit Server VM (build 17.0.1+12-39, mixed mode, sharing)
sghargaonkar@mc-lon-nb15243 ~ %
```

Java 11

TLS V1.3
New HTTP Client API
New String methods (repeat, isBlank , strip , lines)
New File Methods
Not Predicate
Collection to Array

```
var length = str.length();
```

Not Predicate

```
List<String> names = List.of("Sunil", "Simran");
Set<String> states = Set.of("Maharashtra", "Punjab");
Map<String, String> personDetails2 = Map.of( k1: "Sunil", v1: "Maharashtra", k2: "Simran", v2: "Punjab");

Set<String> newStates = Set.copyOf(states);

List<String> state = newStates.stream()
        .filter(Predicate.not(s -> s.equals("Punjab")))
        .collect(Collectors.toUnmodifiableList());
```

Collection to Array

```
@Test
void listSetAndMapOf2() {
    List<String> names = List.of("Sunil", "Simran");
    Set<String> states = Set.of("Maharashtra", "Punjab");
    Map<String, String> personDetails2 = Map.of( k1: "Sunil", v1: "Maharashtra", k2: "Simran", v2: "Punjab");

    Set<String> newStates = Set.copyOf(states);

    String [] state = newStates.stream()
            .filter(Predicate.not(s -> s.equals("Punjab")))
            .toArray(String[]::new);

}
```

Java 12

File::mismatch method

Switch Expressions (Preview)

File:: mismatch

```
@Test
void tempFileCreationAndMatchTest() throws IOException {
    final Path filePath1 = Files.writeString(Files.createTempFile( prefix: "demo1", suffix: ".txt"), csq: "Test JAVA 12 feature");

    final Path filePath2 = Files.writeString(Files.createTempFile( prefix: "demo2", suffix: ".txt"), csq: "Test JAVA 12 feature");


    long mismatch = Files.mismatch(filePath1, filePath2);
    assertEquals( expected: -1, mismatch);
}
```

Java 13

Switch Expressions (preview)

Text Blocks (preview)

Java 14

- NullPointerExceptions with StackTrace
- Switch Expressions
- Text Blocks (preview)

NPE StackTrace

```
    @Test
    @DisplayName("Throws exception when the passed configuration is null")
    void throwsExceptionWhenConfigIsNull() {
        new DomainCategory( config: null, domainScanClient);
//        assertThrows(NullPointerException.class, () -> new DomainCategory(null, domainScanClient));
    }
```

```
ceptionWhenConfigIsNull ×
Tests failed: 1 of 1 test – 3 sec 971 ms

java.lang.NullPointerException Create breakpoint
    at com.mimecast.mta.scanner.domain.DomainCategory.<init>(DomainCategory.java:44)
    at com.mimecast.mta.scanner.domain.DomainCategory.<init>(DomainCategory.java:52)
    at com.mimecast.mta.scanner.domain.DomainCategoryTest.throwsExceptionWhenConfigIsNull(DomainCategoryTest.java:72)
  <31 internal lines>
    at java.util.ArrayList.forEach(ArrayList.java:1257) <9 internal lines>
    at java.util.ArrayList.forEach(ArrayList.java:1257) <21 internal lines>
```

```
class NPEStackTraceTest {
    DomainScanClientV2 domainScanClient = new DomainScanClientV2();

    static class DomainCategory{
        public DomainCategory(final String config, DomainScanClientV2 client) {
            config.equals("Test");
        }
    }

    static class DomainScanClientV2{
    }

    @Test
    void testNPEStackTrace() {
        new DomainCategory( config: null, domainScanClient);
    }
}
```

```
Trace ×
Tests failed: 1 of 1 test – 24 ms
/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java ...

java.lang.NullPointerException: Cannot invoke "String.equals(Object)" because "config" is null

    at com.java.test.programs.NPEStackTraceTest$DomainCategory.<init>(NPEStackTraceTest.java:10)
    at com.java.test.programs.NPEStackTraceTest.testNPEStackTrace(NPEStackTraceTest.java:19) <31
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 internal lines>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <21 internal lines>


Process finished with exit code 255
```

Sunil Ghargaonkar

Switch Expression

```
switch (choice) {
    case 1:
        final AccountBalance accountBalance = new AccountBalance(accountNumber);
        System.out.println(accountBalance.formatAccountBalance());
        break;
    case 2:
        final Withdraw withdraw = new Withdraw(accountNumber);
        withdraw.withdrawAmount(amount);
        break;
    case 3:
        final Deposit deposit = new Deposit(accountNumber);
        deposit.depositMoney(amount);
        break;
    default:
        System.out.println("Unable to find the choice");
}
```

```
switch (choice) {
    case 1 -> {
        final AccountBalance accountBalance = new AccountBalance(accountNumber);
        System.out.println(accountBalance.formatAccountBalance());
    }
    case 2 -> {
        final Withdraw withdraw = new Withdraw(accountNumber);
        withdraw.withdrawAmount(amount);
    }
    case 3 -> {
        final Deposit deposit = new Deposit(accountNumber);
        deposit.depositMoney(amount);
    }
    default -> System.out.println("Unable to find the choice");
}
```

Java 15

- Record Class
- Sealed Classes
- Text Blocks

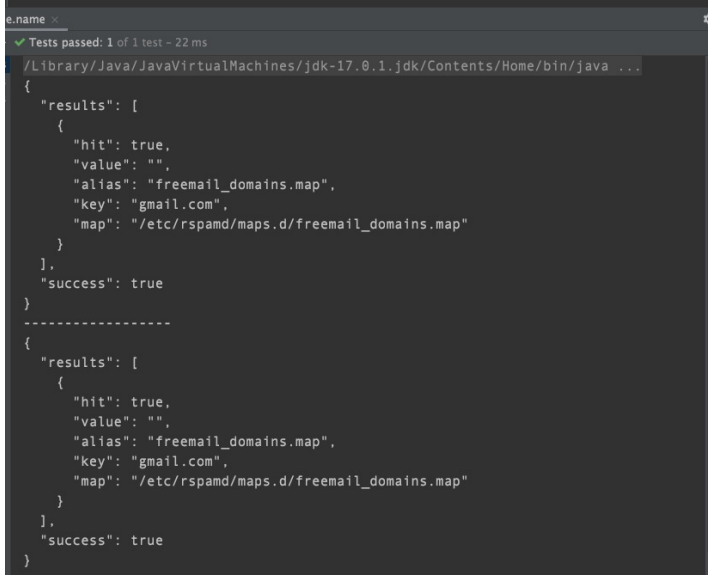
Text Blocks

```java
@Test
void name() {
    String JSON_RESPONSE = "{\n" +
            "  \"results\": [\n" +
            "    {\n" +
            "      \"hit\": true,\n" +
            "      \"value\": \"\",\n" +
            "      \"alias\": \"freemail_domains.map\",\n" +
            "      \"key\": \"gmail.com\",\n" +
            "      \"map\": \"/etc/rspamd/maps.d/freemail_domains.map\"\n" +
            "    }\n" +
            "  ],\n" +
            "  \"success\": true\n" +
            "}";

    String JSON_RESPONSE2 = """
            {
              "results": [
                {
                  "hit": true,
                  "value": "",
                  "alias": "freemail_domains.map",
                  "key": "gmail.com",
                  "map": "/etc/rspamd/maps.d/freemail_domains.map"
                }
              ],
              "success": true
            }""";

    System.out.println(JSON_RESPONSE);
    System.out.println("-----------------");
    System.out.println(JSON_RESPONSE2);

}
}
```



Java 16

- Record Class
- Collection convertor method
- Sealed Class support for Inner Class

Record Class

```java
@Test
void recordClassTest() {
    BooksWithRecord booksWithRecord = new BooksWithRecord( title: "Can't hurt me",  author: "David Goggins");

    System.out.printf(" Title: %s%n Author: %s%n", booksWithRecord.title(), booksWithRecord.author());

    BooksWithoutRecord booksWithoutRecord = new BooksWithoutRecord( title: "Can't hurt me",  author: "David Goggins");

    System.out.printf(" Title: %s%n Author: %s", booksWithoutRecord.getTitle(), booksWithoutRecord.getAuthor());
}

public record BooksWithRecord(String title, String author) {
}

public class BooksWithoutRecord {
    private final String title;
    private final String author;

    public BooksWithoutRecord(String title, String author) {
        this.title = title;
        this.author = author;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }
}
```

Collection convertor methods

```java
@Test
void listSetAndMapOf2() {
    List<String> names = List.of("Sunil", "Simran");
    Set<String> states = Set.of("Maharashtra", "Punjab");
    Map<String, String> personDetails2 = Map.of( k1: "Sunil",  v1: "Maharashtra",  k2: "Simran",  v2: "Punjab");

    states.stream()
            .filter(Predicate.not(s -> s.equals("Punjab")))
            .collect(Collectors.toList());

    states.stream()
            .filter(Predicate.not(s -> s.equals("Punjab")))
            .toList();

}
```

Java 17

Sealed Classes with strict extend rules

Enhanced pseudo-random number generator

Sealed Classes

```java
@Test
void defaultInterfaceMethodTest() {
    final Welcome welcome = new InviteGuest();

    Welcome.invite();
    welcome.greet();
    welcome.serve();
}

sealed interface Welcome permits InviteGuest{
    static void invite() { System.out.println("Let's invite guests for the barbeque party"); }

    default void greet() { System.out.println("Hello Everyone!"); }

    void serve();
}

static final class InviteGuest implements Welcome {

    @Override
    public void serve() { System.out.println("Would you like to have some drinks"); }

    @Override
    public void greet() { System.out.println("Hello Java how is world?"); }
}
```

Questions?