DIGITAL
INNOVATION
ONE

Development Environment

Daniel Hatanaka
Especialista de Software

DIGITAL INNOVATION ONE

Mais sobre mim

- Técnico em informática – IFPA
- Bacharel em Ciências de Computação – ICMC – USP – São Carlos
- Engenheiro de Software Especialista – PagSeguro PagBank
- Amante da tecnologia
- Tucuruí - Pará

DIGITAL INNOVATION ONE

Mais sobre mim

facebook.com/hatanakadaniel

github.com/hatanakadaniel

instagram.com/hatanakadaniel

linkedin.com/in/hatanakadaniel

Percurso

Aula 1	Instalando e executando Java
Aula 2	**Utilizando Java 11**
Aula 3	Usando Intellij para desenvolvimento Java
Aula 4	Empacotando aplicações Java

DIGITAL INNOVATION ONE

Requisitos

- ✓ Distribuição Linux (Ubuntu, Mint, etc)
- ✓ Conhecimentos comandos básicos terminal Linux
- ✓ Conhecimentos em lógica de programação
- ✓ Conhecimento básico de linguagem de programação
- ✓ Programação Orientada a Objetos

DIGITAL
INNOVATION
ONE

Aula 2: Utilizando Java 11

Development environment

Objetivos

1. Recursos presentes no Java 11

2. Novidades do Java 11

DIGITAL
INNOVATION
ONE

Aula 2| Etapa 1:
Recursos presentes no Java 11

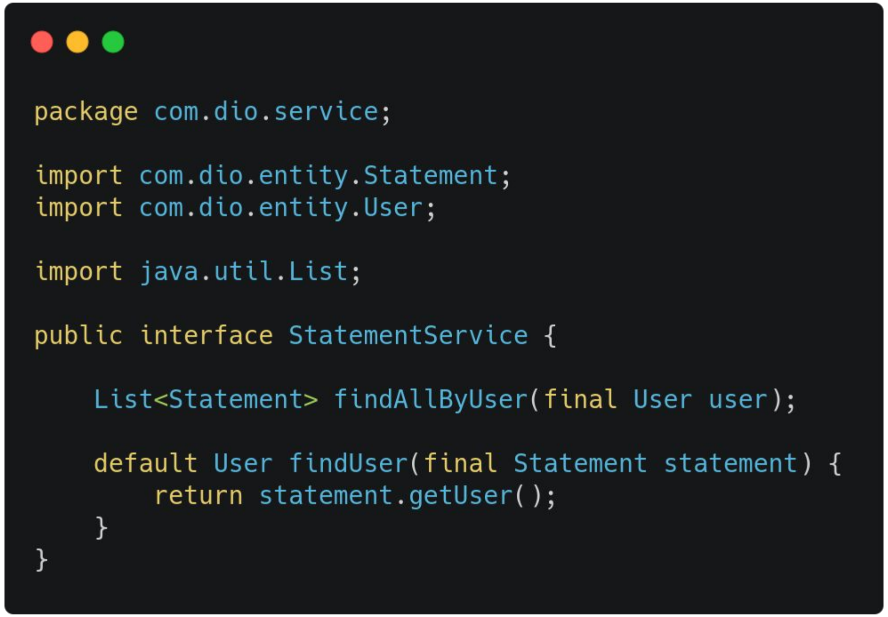
Development environment

Recursos presentes no Java 11

- Default methods
- Lambdas
- Method references
- Streams
- Datas

Recursos presentes no Java 11

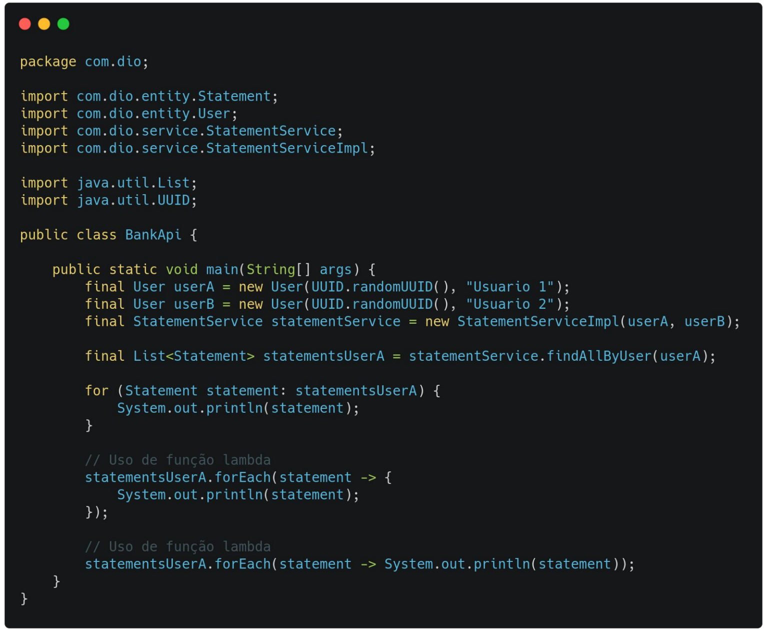
Default methods



```java
package com.dio.service;

import com.dio.entity.Statement;
import com.dio.entity.User;

import java.util.List;

public interface StatementService {

    List<Statement> findAllByUser(final User user);

    default User findUser(final Statement statement) {
        return statement.getUser();
    }
}
```

Recursos presentes no Java 11

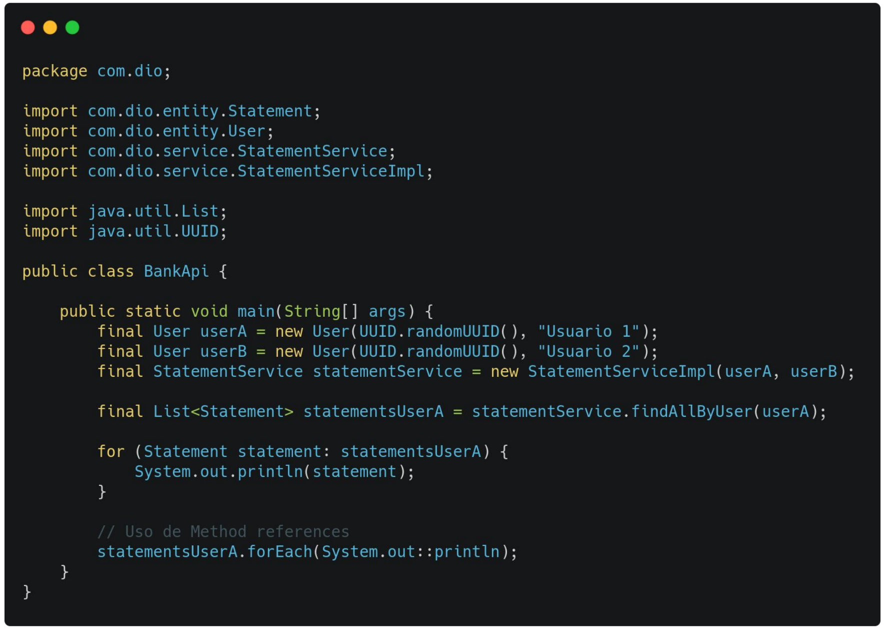
Lambdas



```java
package com.dio;

import com.dio.entity.Statement;
import com.dio.entity.User;
import com.dio.service.StatementService;
import com.dio.service.StatementServiceImpl;

import java.util.List;
import java.util.UUID;

public class BankApi {

    public static void main(String[] args) {
        final User userA = new User(UUID.randomUUID(), "Usuario 1");
        final User userB = new User(UUID.randomUUID(), "Usuario 2");
        final StatementService statementService = new StatementServiceImpl(userA, userB);

        final List<Statement> statementsUserA = statementService.findAllByUser(userA);

        for (Statement statement: statementsUserA) {
            System.out.println(statement);
        }

        // Uso de função lambda
        statementsUserA.forEach(statement -> {
            System.out.println(statement);
        });

        // Uso de função lambda
        statementsUserA.forEach(statement -> System.out.println(statement));
    }
}
```

Recursos presentes no Java 11

Method references

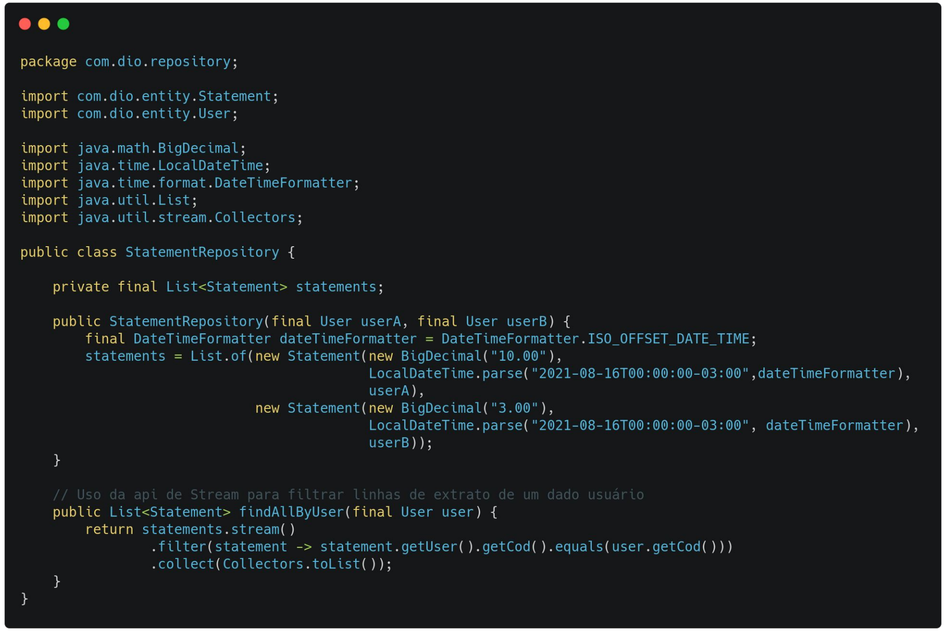


```java
package com.dio;

import com.dio.entity.Statement;
import com.dio.entity.User;
import com.dio.service.StatementService;
import com.dio.service.StatementServiceImpl;

import java.util.List;
import java.util.UUID;

public class BankApi {

    public static void main(String[] args) {
        final User userA = new User(UUID.randomUUID(), "Usuario 1");
        final User userB = new User(UUID.randomUUID(), "Usuario 2");
        final StatementService statementService = new StatementServiceImpl(userA, userB);

        final List<Statement> statementsUserA = statementService.findAllByUser(userA);

        for (Statement statement: statementsUserA) {
            System.out.println(statement);
        }

        // Uso de Method references
        statementsUserA.forEach(System.out::println);
    }
}
```

DIGITAL INNOVATION ONE

Recursos presentes no Java 11

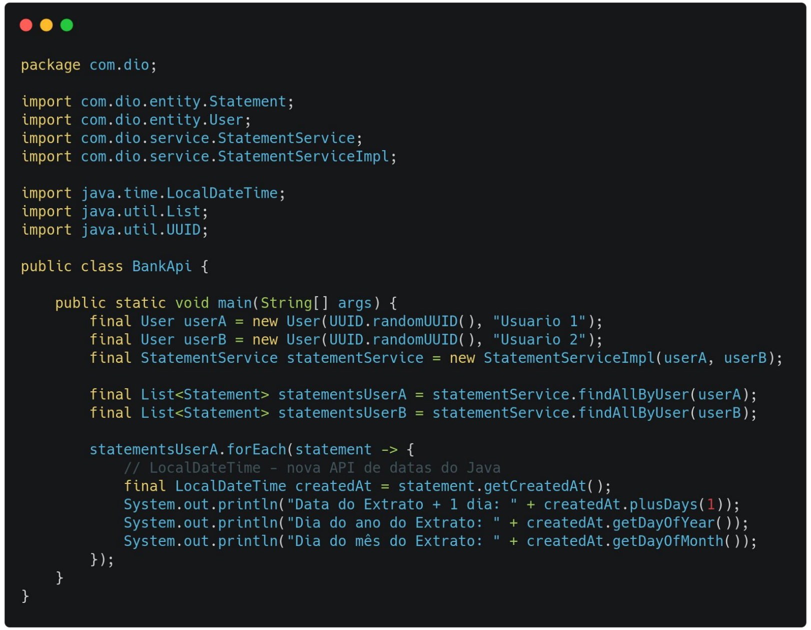
Streams



```java
package com.dio.repository;

import com.dio.entity.Statement;
import com.dio.entity.User;

import java.math.BigDecimal;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.List;
import java.util.stream.Collectors;

public class StatementRepository {

    private final List<Statement> statements;

    public StatementRepository(final User userA, final User userB) {
        final DateTimeFormatter dateTimeFormatter = DateTimeFormatter.ISO_OFFSET_DATE_TIME;
        statements = List.of(new Statement(new BigDecimal("10.00"),
                                           LocalDateTime.parse("2021-08-16T00:00:00-03:00",dateTimeFormatter),
                                           userA),
                             new Statement(new BigDecimal("3.00"),
                                           LocalDateTime.parse("2021-08-16T00:00:00-03:00", dateTimeFormatter),
                                           userB));
    }

    // Uso da api de Stream para filtrar linhas de extrato de um dado usuário
    public List<Statement> findAllByUser(final User user) {
        return statements.stream()
                .filter(statement -> statement.getUser().getCod().equals(user.getCod()))
                .collect(Collectors.toList());
    }
}
```

Recursos presentes no Java 11

Datas



```java
package com.dio;

import com.dio.entity.Statement;
import com.dio.entity.User;
import com.dio.service.StatementService;
import com.dio.service.StatementServiceImpl;

import java.time.LocalDateTime;
import java.util.List;
import java.util.UUID;

public class BankApi {

    public static void main(String[] args) {
        final User userA = new User(UUID.randomUUID(), "Usuario 1");
        final User userB = new User(UUID.randomUUID(), "Usuario 2");
        final StatementService statementService = new StatementServiceImpl(userA, userB);

        final List<Statement> statementsUserA = statementService.findAllByUser(userA);
        final List<Statement> statementsUserB = statementService.findAllByUser(userB);

        statementsUserA.forEach(statement -> {
            // LocalDateTime - nova API de datas do Java
            final LocalDateTime createdAt = statement.getCreatedAt();
            System.out.println("Data do Extrato + 1 dia: " + createdAt.plusDays(1));
            System.out.println("Dia do ano do Extrato: " + createdAt.getDayOfYear());
            System.out.println("Dia do mês do Extrato: " + createdAt.getDayOfMonth());
        });
    }
}
```

DIGITAL
INNOVATION
ONE

Aula 2 | Etapa 2:

Novidades do Java 11

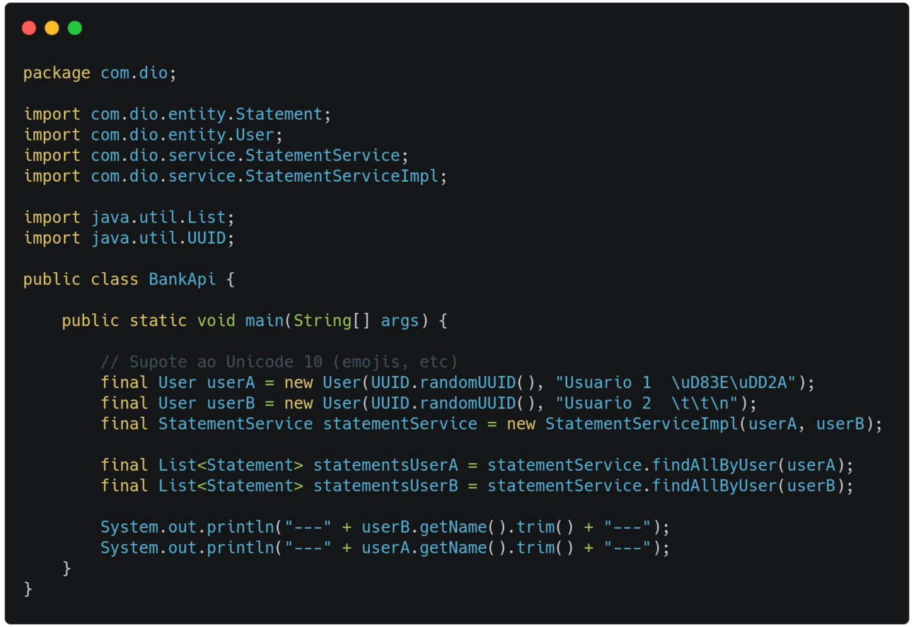
Development environment

Novidades do Java 11

- Manipulação de Strings
- Variáveis por inferência
- Manipulação de arquivos
- Novidades no Garbage Collector (GC)
- Http Client padronizado

Novidades do Java 11

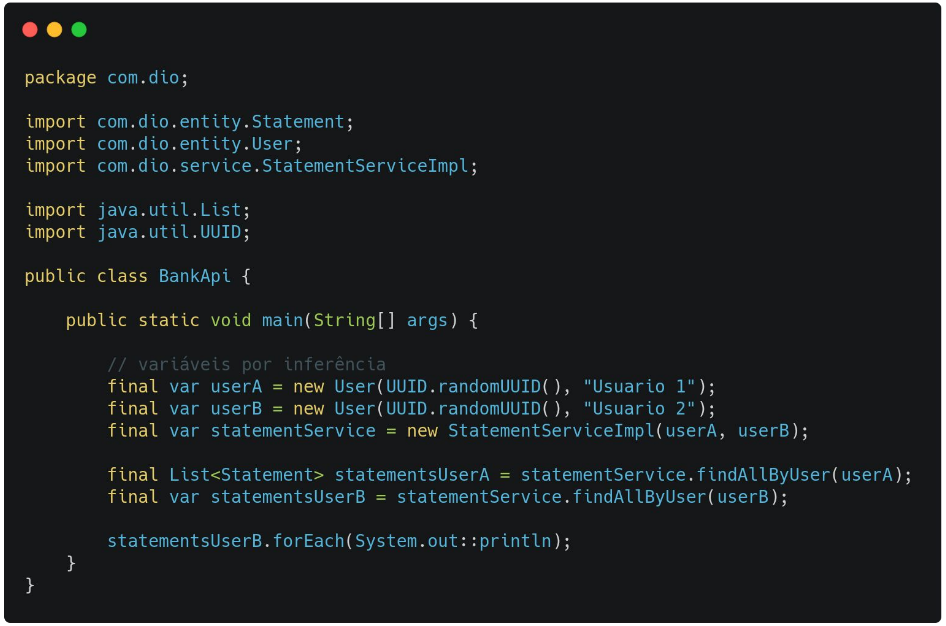
Manipulação de Strings



```java
package com.dio;

import com.dio.entity.Statement;
import com.dio.entity.User;
import com.dio.service.StatementService;
import com.dio.service.StatementServiceImpl;

import java.util.List;
import java.util.UUID;

public class BankApi {

    public static void main(String[] args) {

        // Supote ao Unicode 10 (emojis, etc)
        final User userA = new User(UUID.randomUUID(), "Usuario 1  \uD83E\uDD2A");
        final User userB = new User(UUID.randomUUID(), "Usuario 2  \t\t\n");
        final StatementService statementService = new StatementServiceImpl(userA, userB);

        final List<Statement> statementsUserA = statementService.findAllByUser(userA);
        final List<Statement> statementsUserB = statementService.findAllByUser(userB);

        System.out.println("---" + userB.getName().trim() + "---");
        System.out.println("---" + userA.getName().trim() + "---");
    }
}
```

Novidades do Java 11

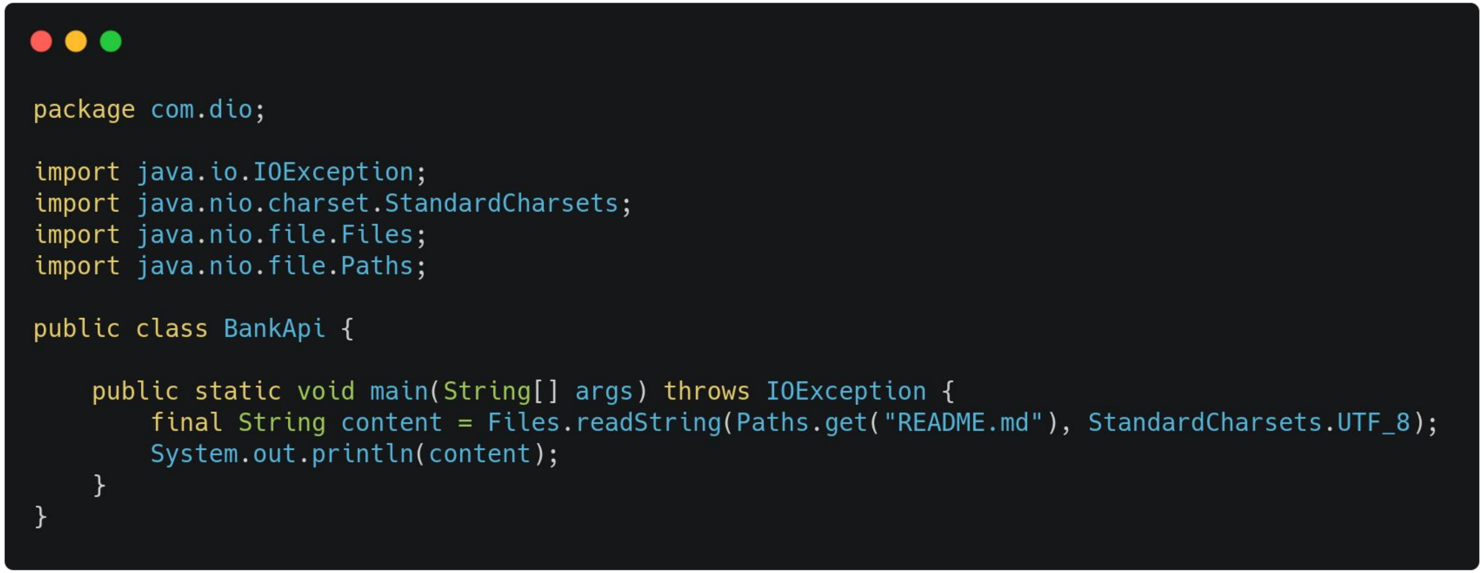
Variáveis por inferência



```java
package com.dio;

import com.dio.entity.Statement;
import com.dio.entity.User;
import com.dio.service.StatementServiceImpl;

import java.util.List;
import java.util.UUID;

public class BankApi {

    public static void main(String[] args) {

        // variáveis por inferência
        final var userA = new User(UUID.randomUUID(), "Usuario 1");
        final var userB = new User(UUID.randomUUID(), "Usuario 2");
        final var statementService = new StatementServiceImpl(userA, userB);

        final List<Statement> statementsUserA = statementService.findAllByUser(userA);
        final var statementsUserB = statementService.findAllByUser(userB);

        statementsUserB.forEach(System.out::println);
    }
}
```

Novidades do Java 11

Manipulação de arquivos



```java
package com.dio;

import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;

public class BankApi {

    public static void main(String[] args) throws IOException {
        final String content = Files.readString(Paths.get("README.md"), StandardCharsets.UTF_8);
        System.out.println(content);
    }
}
```

DIGITAL INNOVATION ONE

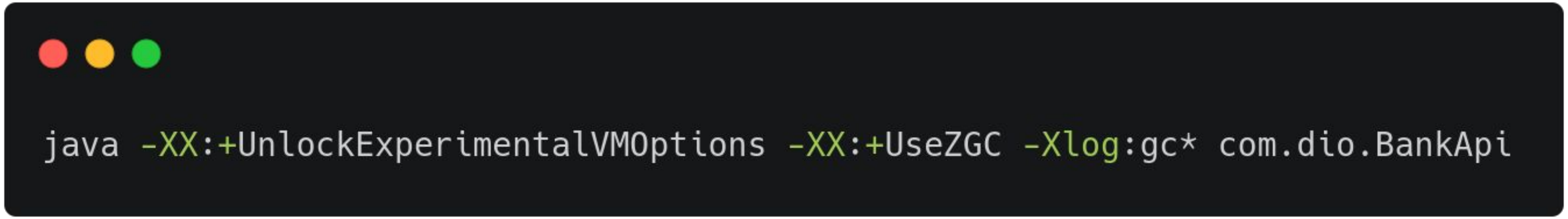
Novidades do Java 11

Novidades no Garbage Collector (GC)

- Z Garbage Collector (ZGC)
- Epsilon Garbage Collector (No-Op GC)

Novidades do Java 11

Z Garbage Collector (ZGC)

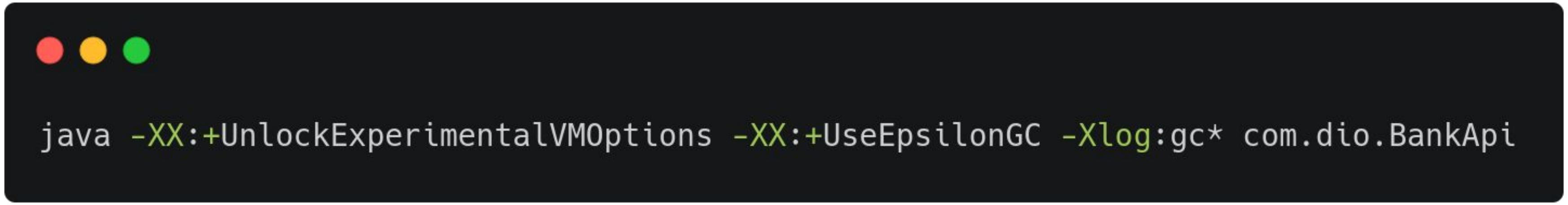


```
java -XX:+UnlockExperimentalVMOptions -XX:+UseZGC -Xlog:gc* com.dio.BankApi
```

DIGITAL INNOVATION ONE

Novidades do Java 11

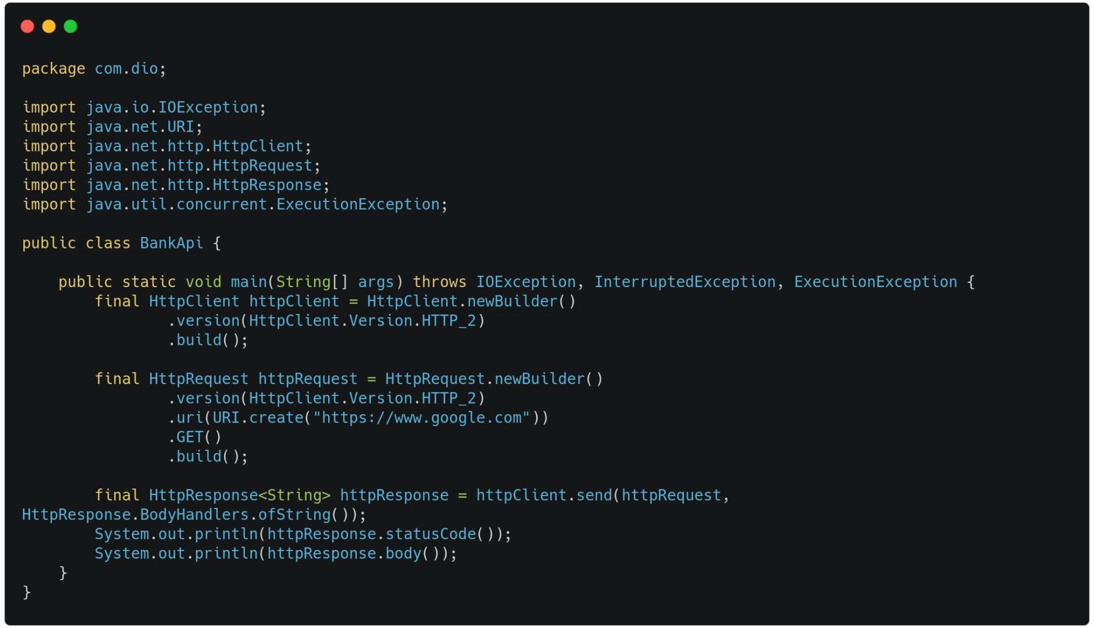
Epsilon Garbage Collector (No-Op GC)



```
java -XX:+UnlockExperimentalVMOptions -XX:+UseEpsilonGC -Xlog:gc* com.dio.BankApi
```

Novidades do Java 11

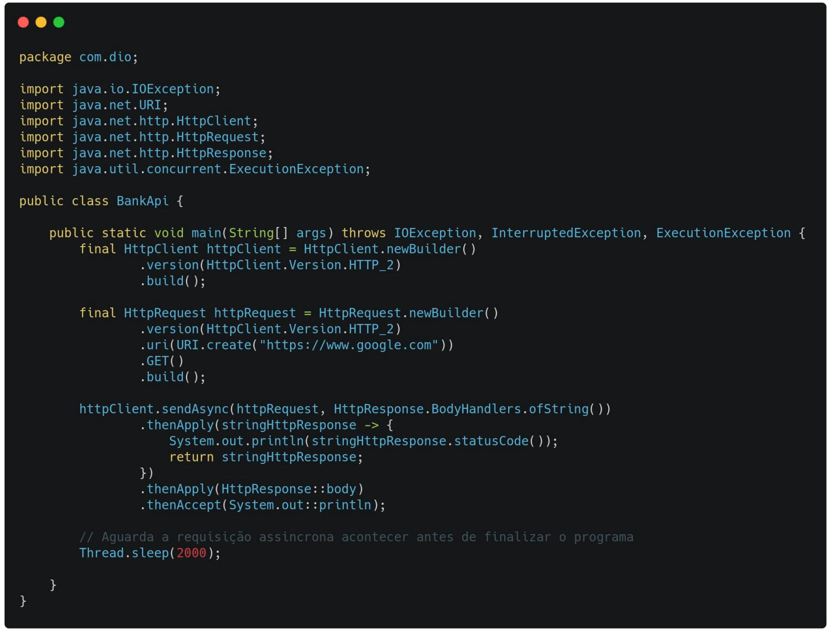
Http Client padronizado (chamada síncrona)



```
package com.dio;

import java.io.IOException;
import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.util.concurrent.ExecutionException;

public class BankApi {

    public static void main(String[] args) throws IOException, InterruptedException, ExecutionException {
        final HttpClient httpClient = HttpClient.newBuilder()
                .version(HttpClient.Version.HTTP_2)
                .build();

        final HttpRequest httpRequest = HttpRequest.newBuilder()
                .version(HttpClient.Version.HTTP_2)
                .uri(URI.create("https://www.google.com"))
                .GET()
                .build();

        final HttpResponse<String> httpResponse = httpClient.send(httpRequest,
HttpResponse.BodyHandlers.ofString());
        System.out.println(httpResponse.statusCode());
        System.out.println(httpResponse.body());
    }
}
```

Novidades do Java 11

Http Client padronizado (chamada assíncrona)



```java
package com.dio;

import java.io.IOException;
import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.util.concurrent.ExecutionException;

public class BankApi {

    public static void main(String[] args) throws IOException, InterruptedException, ExecutionException {
        final HttpClient httpClient = HttpClient.newBuilder()
                .version(HttpClient.Version.HTTP_2)
                .build();

        final HttpRequest httpRequest = HttpRequest.newBuilder()
                .version(HttpClient.Version.HTTP_2)
                .uri(URI.create("https://www.google.com"))
                .GET()
                .build();

        httpClient.sendAsync(httpRequest, HttpResponse.BodyHandlers.ofString())
                .thenApply(stringHttpResponse -> {
                    System.out.println(stringHttpResponse.statusCode());
                    return stringHttpResponse;
                })
                .thenApply(HttpResponse::body)
                .thenAccept(System.out::println);

        // Aguarda a requisição assincrona acontecer antes de finalizar o programa
        Thread.sleep(2000);

    }
}
```

DIGITAL
INNOVATION
ONE

Dúvidas?

> Fórum do curso
> Comunidade online (discord)