VDAB

samen sterk voor werk

Java

() -> { LAMBDA }
TAKENBUNDEL

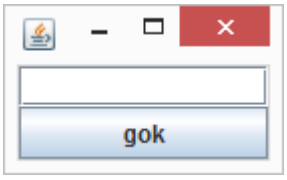
Peoplesoftcode:
Wettelijk depot:
versie: 17/03/2015

INHOUD

1 Taken

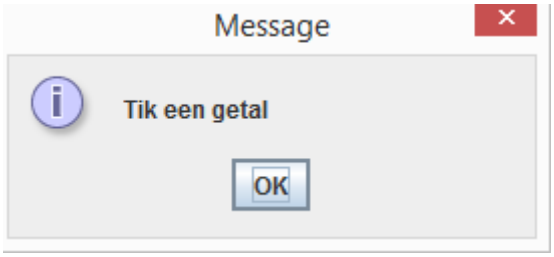
1.1 Hoger lager

De gebruiker raadt in een Swing applicatie een geheel getal (tussen 1 en 10) dat de computer heeft verzonnen.

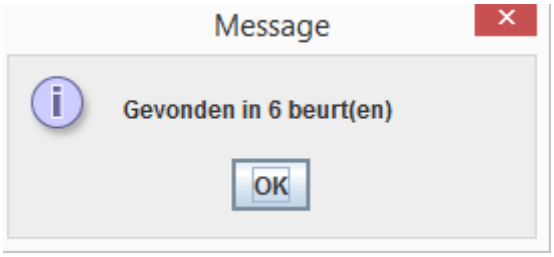


De gebruiker tikt het getal in de textbox en kiest de button gok

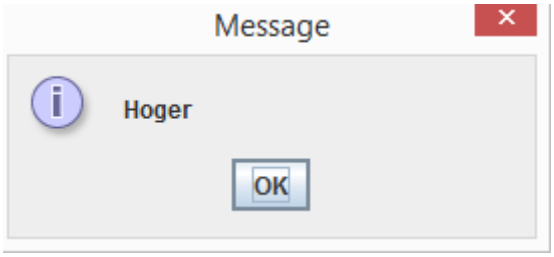
Als de gebruiker geen getal tikte, toon je een foutmelding:



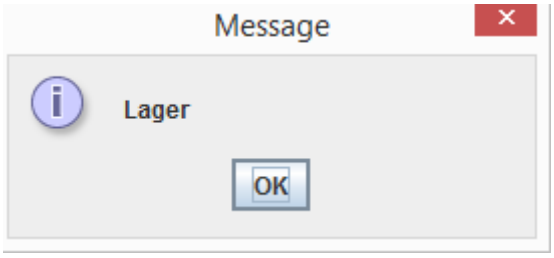
Als de gebruiker het getal geraden heeft, toon je volgende melding



Als het getal van de gebruiker kleiner is dan het te raden getal, toon je volgende melding:



Als het getal van de gebruiker groter is dan het te raden getal, toon je volgende melding:



1.2 Landcodes

Je maakt een method die een `String` met een landcode als parameter aanvaardt.

De method zoekt deze code in `landcodes.txt` en geeft een `String` met de bijbehorende landnaam terug. De code in de parameter van de method kan ook niet voorkomen in het bestand.

1.3 Sterrenbeelden

Je leest de regels in `sterrenbeelden.txt` en je toont die in hoofdlettervorm op het scherm.

1.4 Zoek een sterrenbeeld

De gebruiker tikt een woord. Jij toont de regels in `sterrenbeelden.txt` die dit woord bevatten.

1.5 Oneven getallen

De gebruiker tikt getallen tot hij 0 tikt. Jij toont *daarna* de oneven getallen uit de ingetikte reeks, in volgorde van groot naar klein.

1.6 Landnamen sorteren

Je toont een gesorteerde lijst van de landnamen in `landcodes.txt`

```
Afghanistan
Albanië
Algerije
Amerikaans-Samoa
Amerikaanse Maagdeneilanden
Andorra
...
```

De volgorde zal verschillen van de volgorde van de regels in het bestand:
je moet bijvoorbeeld `Bangladesh` voor `Barbados` tonen.

1.7 Artiestennamen

Je toont een gesorteerde lijst van de artiestennamen in `albumsartists.txt`

1.8 Artiesten en hun albums

Je toont een gesorteerde lijst van de artiestennamen in `albumsartists.txt`

Je toont per artiest een gesorteerde lijst van zijn albumtitels.

1.9 Kleinste oppervlakte

Je vult een `List` met enkele willekeurige `Rechthoek` objecten.

Je toont de kleinste oppervlakte van deze rechthoeken.
Je toont daaronder de lengte en de breedte van de rechthoek(en) die deze oppervlakte hebben.

1.10 Laatste land

Je toont de landnaam die alfabetisch laatst komt (`Åland`) in `landcodes.txt`

1.11 Sterrenbeelden 2

Je leest de regels in `sterrenbeelden.txt` en je toont die in hoofdlettervorm op het scherm.
Je gebruikt geen lambda's, maar method references.

2 Voorbeeldoplossingen

2.1 Hoger lager

```
import java.awt.BorderLayout;
import java.util.Random;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
class Venster extends JFrame {
  private static final long serialVersionUID = 1L;
  private final int teRadenGetal = new Random().nextInt(10) + 1;
  private final JTextField textField = new JTextField();
  private final JButton button = new JButton("gok");
  private int beurten;
  Venster() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    add(textField, BorderLayout.NORTH);
    add(button, BorderLayout.SOUTH);
    button.addActionListener(event -> gok());
    pack();
  }
  private void gok() {
    beurten++;
    try {
      int getal = Integer.parseInt(textField.getText());
      if (getal == teRadenGetal) {
        JOptionPane.showMessageDialog(this,
          "Gevonden in " + beurten + " beurt(en)");
      } else {
        JOptionPane.showMessageDialog(this,
          getal > teRadenGetal ? "Lager" : "Hoger");
      }
    } catch (NumberFormatException ex) {
      JOptionPane.showMessageDialog(this, "Tik een getal");
    }
  }
}
class Main {
  public static void main(String[] args) {
    new Venster().setVisible(true);
  }
}
```

2.2 Landcodes

```
import java.io.BufferedReader;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.Optional;
class Main {
  private static final Path LANDCODES_PATH = Paths.get("/data/landcodes.txt");
  private static Optional<String> landNaam(String landcode) {
    try (BufferedReader reader = Files.newBufferedReader(LANDCODES_PATH)) {
      for (String regel; (regel = reader.readLine()) != null;) {
        int spatieIndex = regel.indexOf(' ');
```

```
        if (landcode.equals(regel.substring(0, spatieIndex))) {
          return Optional.of(regel.substring(spatieIndex+1));
        }
      }
    } catch (Exception ex) {
      ex.printStackTrace();
    }
    return Optional.empty();
  }
  public static void main(String[] args) {
    landNaam("BE").ifPresent(landNaam -> System.out.println(landNaam));
  }
}
```

2.3 Sterrenbeelden

```
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.stream.Stream;
class Main {
  private static final Path STERRENBEELDEN_PATH =
    Paths.get("/data/sterrenbeelden.txt");
  public static void main(String[] args) {
    try (Stream<String> stream = Files.lines(STERRENBEELDEN_PATH)) {
      stream.forEach(
        sterrenbeeld -> System.out.println(sterrenbeeld.toUpperCase()));
    } catch (Exception ex) {
      ex.printStackTrace();
    }
  }
}
```

2.4 Zoek een sterrenbeeld

```
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.Scanner;
import java.util.stream.Stream;
class Main {
  private static final Path STERRENBEELDEN_PATH = Paths
    .get("/data/sterrenbeelden.txt");
  public static void main(String[] args) {
    System.out.print("woord:");
    try (Scanner scanner = new Scanner(System.in)) {
      String woord = scanner.next().toUpperCase();
      try (Stream<String> stream = Files.lines(STERRENBEELDEN_PATH)) {
        stream.filter(sterrenbeeld -> sterrenbeeld.toUpperCase().contains(woord))
              .forEach(sterrenbeeld -> System.out.println(sterrenbeeld));
      } catch (Exception ex) {
        ex.printStackTrace();
      }
    }
  }
}
```

2.5 Oneven getallen

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
class Main {
  public static void main(String[] args) {
    List<Integer> getallen = new ArrayList<>();
    try (Scanner scanner = new Scanner(System.in)) {
    for (int getal; (getal = scanner.nextInt()) != 0;) {
      getallen.add(getal);
    }
    getallen.stream()
      .filter(getal -> getal % 2 == 1)
      .sorted((getal1, getal2) -> getal2 - getal1)
      .forEach(getal -> System.out.println(getal));
    }
  }
}
```

2.6 Landnamen sorteren

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.stream.Stream;
class Main {
  private static final Path LANDCODES_PATH = Paths.get("/data/landcodes.txt");
  public static void main(String[] args) {
    try (Stream<String> stream = Files.lines(LANDCODES_PATH)) {
      stream.map(regel -> regel.substring(regel.indexOf(' ') + 1))
            .sorted()
            .forEach(naam -> System.out.println(naam));
    } catch (IOException ex) {
      ex.printStackTrace();
    }
  }
}
```

2.7 Artiestennamen

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.stream.Stream;
class Main {
  private static final Path ALBUMS_ARTISTS_PATH =
    Paths.get("/data/albumsartists.txt");
  public static void main(String[] args) {
    try (Stream<String> stream = Files.lines(ALBUMS_ARTISTS_PATH)) {
      stream.map(regel -> regel.substring(regel.indexOf(',') + 1))
            .distinct()
            .sorted()
            .forEach(naam -> System.out.println(naam));
    } catch (IOException ex) { ex.printStackTrace(); }
  }
}
```

2.8 Artiesten en hun albums

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;
import java.util.stream.Stream;
class Main {
  private static final Path ALBUMS_ARTISTS_PATH =
    Paths.get("/data/albumsartists.txt");
  public static void main(String[] args) {
    try (Stream<String> stream = Files.lines(ALBUMS_ARTISTS_PATH)) {
      Map<String, List<String>> albumsPerArtiest =
        stream.collect(
          Collectors.groupingBy(
            regel -> regel.substring(regel.indexOf(',') + 1)));
      albumsPerArtiest.entrySet().stream()
        .sorted((entry1, entry2) -> entry1.getKey().compareTo(entry2.getKey()))
        .forEach(entry -> {
          System.out.println(entry.getKey());
          entry.getValue().stream()
            .map(regel -> regel.substring(0, regel.indexOf(',')))
            .sorted()
            .forEach(album -> System.out.println("\t" + album));
        });
    } catch (IOException ex) {
      ex.printStackTrace();
    }
  }
}
```

2.9 Kleinste oppervlakte

```
import java.util.Arrays;
import java.util.List;
import java.util.OptionalInt;
class Rechthoek {
  private final int lengte;
  private final int breedte;
  public Rechthoek(int lengte, int breedte) {
    this.lengte = lengte;
    this.breedte = breedte;
  }
  public int getOppervlakte() {
    return lengte * breedte;
  }
  @Override
  public String toString() {
    return lengte + " op " + breedte;
  }
}
```

```
class Main {
  public static void main(String[] args) {
    List<Rechthoek> rechthoeken = Arrays.asList(
      new Rechthoek(3, 2), new Rechthoek(4, 5), new Rechthoek(2, 3));
    OptionalInt kleinsteOppervlakte = rechthoeken.stream()
      .mapToInt(rechthoek -> rechthoek.getOppervlakte())
      .min();
    kleinsteOppervlakte.ifPresent(oppervlakte -> {
      System.out.println(oppervlakte);
      rechthoeken.stream()
        .filter(rechthoek -> rechthoek.getOppervlakte() == oppervlakte)
        .forEach(rechthoek -> System.out.println(rechthoek));
    });
  }
}
```

2.10 Laatste land

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.stream.Stream;
class Main {
  private static final Path LANDCODES_PATH = Paths.get("/data/landcodes.txt");
  public static void main(String[] args) {
    try (Stream<String> stream = Files.lines(LANDCODES_PATH)) {
      stream.map(regel -> regel.substring(regel.indexOf(' ') + 1))
            .max((naam1, naam2) -> naam1.compareToIgnoreCase(naam2))
            .ifPresent(grootsteNaam -> System.out.println(grootsteNaam));
    } catch (IOException ex) {
      ex.printStackTrace();
    }
  }
}
```

2.11 Sterrenbeelden 2

```
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.stream.Stream;
class Main {
  private static final Path STERRENBEELDEN_PATH =
    Paths.get("/data/sterrenbeelden.txt");
  public static void main(String[] args) {
    try (Stream<String> stream = Files.lines(STERRENBEELDEN_PATH)) {
      stream.map(String::toUpperCase)
            .forEach(System.out::println);
    } catch (Exception ex) {
      ex.printStackTrace();
    }
  }
}
```

COLOFON

Domeinexpertisemanager	Jean Smits
Moduleverantwoordelijke	
Auteurs	Hans Desmet
Versie	26/3/2015
Codes	Peoplesoftcode: Wettelijk depot:

Omschrijving module-inhoud

Abstract	Doelgroep	Opleiding Java Ontwikkelaar
	Aanpak	Zelfstudie
	Doelstelling	Lambda's kunnen gebruiken
Trefwoorden		Lambda
Bronnen/meer info		