LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Collections, FileReader und Kommandozeilenparameter

Bioinformatik Bachelor
Propädeutikum WS 2019/2020
12. November 2019
Leopold Endres

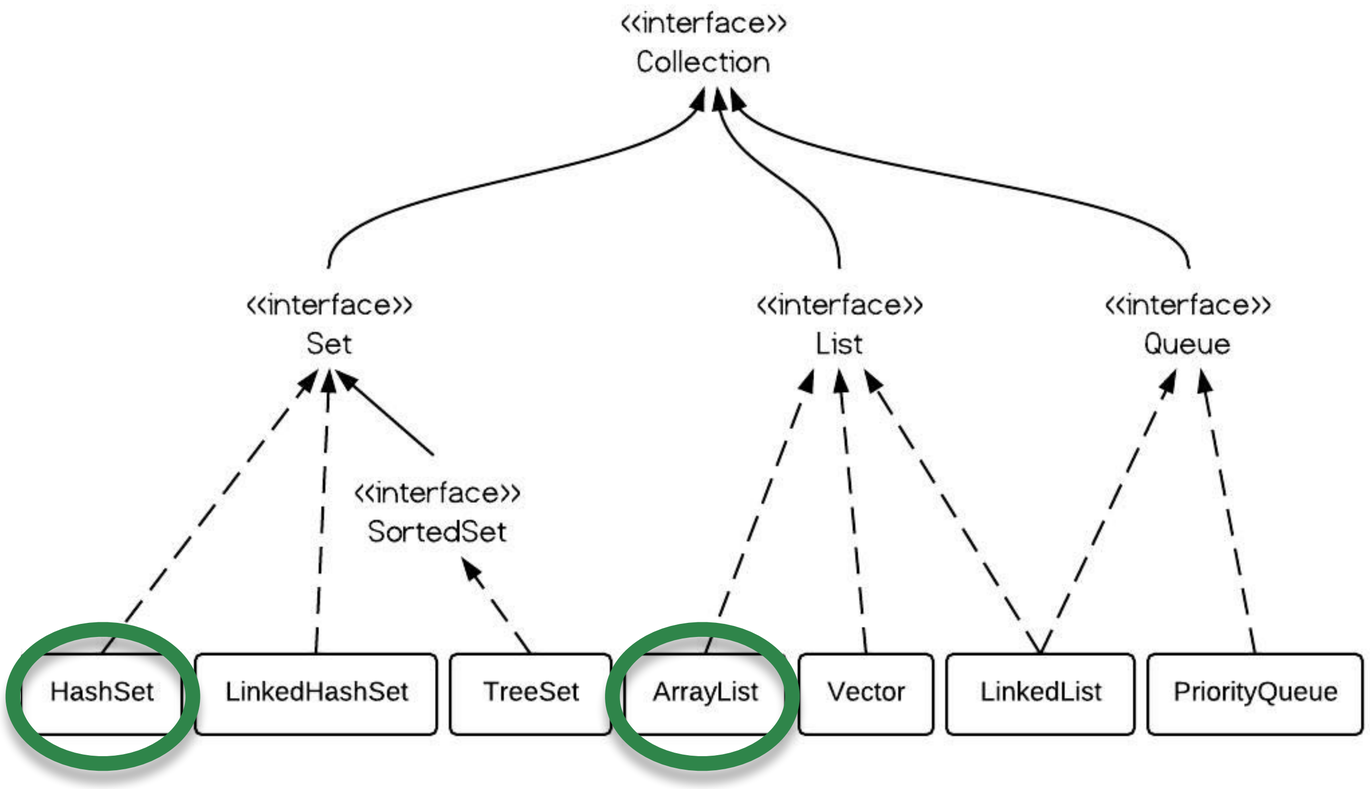
Outline

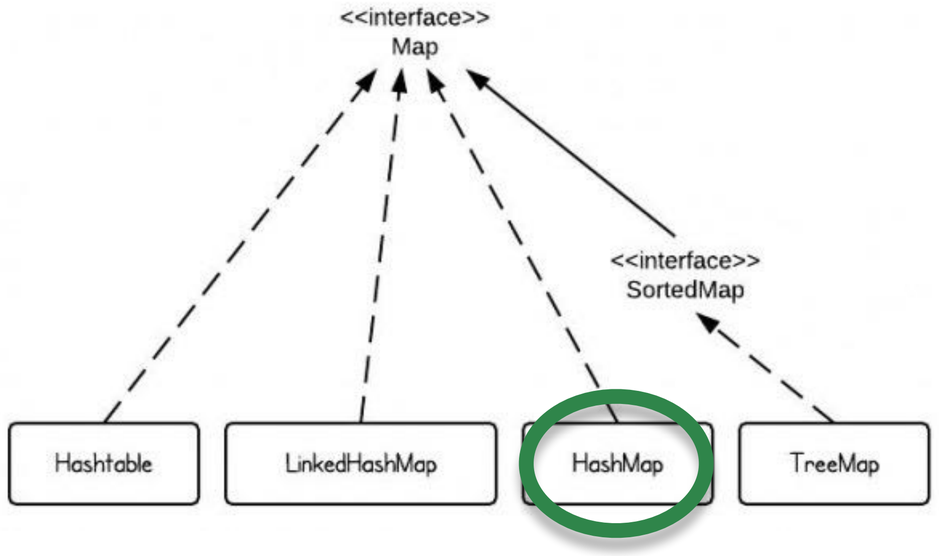
1. Collections
    a) ArrayList
    b) HashSet
    c) HashMap
2. FileReader
3. Kommandozeilenparameter

1. Collections

- Framework in Java
- Sammlung von Klassen und Interfaces
- Reduziert den Programmieraufwand
- Aufgeteilt in *lists*, *maps* und *sets*

1. Collection Hierarchie





1. a) ArrayList

- Wiederholung: Array
  ```
  String[ ] cars = {"Volvo", "BMW", "Ford", “Mazda”};
  ```

- Nachteil: Ein Array hat eine festgelegte Länge (hier 4). Es kann also kein weiteres Element hinzugefügt oder entfernt werden.

- Lösung ArrayList: ArrayList wachsen und schrumpfen dynamisch, so dass Elemente entfernt und hinzugefügt werden können.

1. a) ArrayList

Beispiel

```
ArrayList<String> cars = new ArrayList<String>();

cars.add("Volvo");
cars.add("BMW");
cars.add("Ford");
cars.add("Mazda");
cars.add("Audi");

System.out.println(cars);
```

Um eine ArrayList zu erstellen, müssen wir den Datentyp `<String>` spezifizieren und der ArrayList einen Namen `cars` zuweisen.

1. a) ArrayList

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Beispiel

```
ArrayList<String> cars = new ArrayList<String>();

cars.add("Volvo");
cars.add("BMW");
cars.add("Ford");
cars.add("Mazda");
cars.add("Audi");

System.out.println(cars);
```

Fügt der ArrayList cars das Element “Volvo” hinzu.

1. a) ArrayList

Beispiel

```
ArrayList<String> cars = new ArrayList<String>();

cars.add("Volvo");
cars.add("BMW");
cars.add("Ford");
cars.add("Mazda");
cars.add("Audi");

System.out.println(cars);
```

Gibt cars auf der Konsole aus

output:
[Volvo, BMW, Ford, Mazda, Audi]

1. a) ArrayList

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

java.util.ArrayList Methods	
l.add(itm)	Add itm to list
l.get(i)	Return ith item
l.size()	Return number of items
l.remove(i)	Remove ith item
l.set(i, val)	Put val at position i

Mehr Methoden hier: https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html

1. b) HashSet

- deutsch: Menge
- HashSet ist eine Menge von Elemente ohne festgelegte Reihenfolge.
- Kann keine Duplikate enthalten.

1. b) HashSet

Beispiel

```
HashSet<String> cars = new HashSet<String>();

cars.add("Volvo");
cars.add("BMW");
cars.add(“Ford");
cars.add("Mazda");
cars.add("BMW");

System.out.println(cars);

for (String car : cars)
    System.out.println(car);
}
```

HashSet<String> cars = new HashSet<String>();

In den <> Klammern wird der Datentyp der ArrayList definiert.

1. b) HashSet

Beispiel

```
HashSet<String> cars = new HashSet<String>();

cars.add("Volvo");
cars.add("BMW");
cars.add(“Ford");
cars.add("Mazda");
cars.add("BMW");

System.out.println(cars);

for (String car : cars)
    System.out.println(car);
}
```

cars.add(“Volvo");

Fügt dem HashSet cars das Element “Volvo” hinzu.

1. b) HashSet

Beispiel

```
HashSet<String> cars = new HashSet<String>();

cars.add("Volvo");
cars.add("BMW");
cars.add(“Ford");
cars.add("Mazda");
cars.add("BMW");

System.out.println(cars);

for (String car : cars)
    System.out.println(car);
}
```

Gibt cars auf der Konsole aus

output:

[Volvo, Mazda, Ford, BMW]

Neue Reihenfolge und keine Duplikate!

1. b) HashSet

Beispiel

```
HashSet<String> cars = new HashSet<String>();

cars.add("Volvo");
cars.add("BMW");
cars.add(“Ford");
cars.add("Mazda");
cars.add("BMW");

System.out.println(cars);

for (String car : cars)
    System.out.println(car);
}
```

For each führt den Code innerhalb der Klammern für jedes Element in cars aus.

output:

Volvo
Mazda
Ford
BMW

Neue Reihenfolge und keine Duplikate!

1. b) HashSet

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

java.util.HashSet Methods	
l.add(itm)	Add itm to list
l.clear()	Clear HashSet
l.size()	Return number of items

Mehr Methoden hier: https://docs.oracle.com/javase/8/docs/api/java/util/HashSet.html

1. c) HashMap

- Wiederholung: Array
  ```
  String[ ] cars = {"Volvo", "BMW", "Ford", “Mazda”};
  ```

- Zugriff auf Elemente mit einer Indexnummer. Zum Beispiel cars[2]

- HashMap hingegen speichert Elemente in "Schlüssel/Wert"-Paaren und Sie können über einen Index eines anderen Typs (z.B. einen String) darauf zugreifen.

1. c) HashMap

Beispiel

```
HashMap<String, String> capitalCities = new HashMap<String, String>();

capitalCities.put("England", "London");
capitalCities.put("Germany", "Berlin");
capitalCities.put("Norway", "Oslo");
capitalCities.put("Austria", "Vienna");
capitalCities.put("USA", "Washington DC");

System.out.println(capitalCities);
```

HashMap<String, String> capitalCities = new HashMap<String, String>();

In den <> Klammern werden die Datentypen von Key und Value definiert.

1. c) HashMap

Beispiel

```
HashMap<String, String> capitalCities = new HashMap<String, String>();

capitalCities.put("England", "London");
capitalCities.put("Germany", "Berlin");
capitalCities.put("Norway", "Oslo");
capitalCities.put("Austria", "Vienna");
capitalCities.put("USA", "Washington DC");

System.out.println(capitalCities);
```

Mit dem Befehl capitalCities.put("England", “London"); wird der HashMap das key/value - Paar ("England", “London”) hinzugefügt.

1. c) HashMap

Beispiel

```
HashMap<String, String> capitalCities = new HashMap<String, String>();

capitalCities.put("England", "London");
capitalCities.put("Germany", "Berlin");
capitalCities.put("Norway", "Oslo");
capitalCities.put("Austria", "Vienna");
capitalCities.put("USA", "Washington DC");

System.out.println(capitalCities);
```

Gibt capitalCities auf der Konsole aus.

output:
{Austria=Vienna, USA=Washington DC, Norway=Oslo, England=London, Germany=Berlin}

1. c) HashMap

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

java.util.HashMap Methods	
m.put(key,value)	Inserts value with key
m.get(key)	Retrieves value with key
m.containsKey(key)	true if contains key

Mehr Methoden hier: https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html

1. Collection Methoden

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

java.util.HashSet Methods	
Collections.sort(cars);	Sorts cars
Collections.shuffle(cars);	Shuffles cars
Collections.reverse(cars);	Reverses the order of elements in cars

Mehr Methoden hier: https://docs.oracle.com/javase/7/docs/api/java/util/Collections.html

2. Kommandozeilenparameter

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

- Mithilfe von Kommandozeilenparametern können dem Programm Parameter übergeben werden.

- Diese Parameter werden der main-Methode des Programms im String-Array args übergeben. (startet wie immer bei 0)

2. Kommandozeilenparameter in der Konsole

- Mithilfe von Kommandozeilenparametern können dem Programm Parameter übergeben werden.

- Diese Parameter werden der main-Methode des Programms im String-Array args übergeben. (startet wie immer bei 0)

Beispiel

```
public class Main {

    public static void main(String[] args) {

        System.out.println(args[0]);

    }
}
```

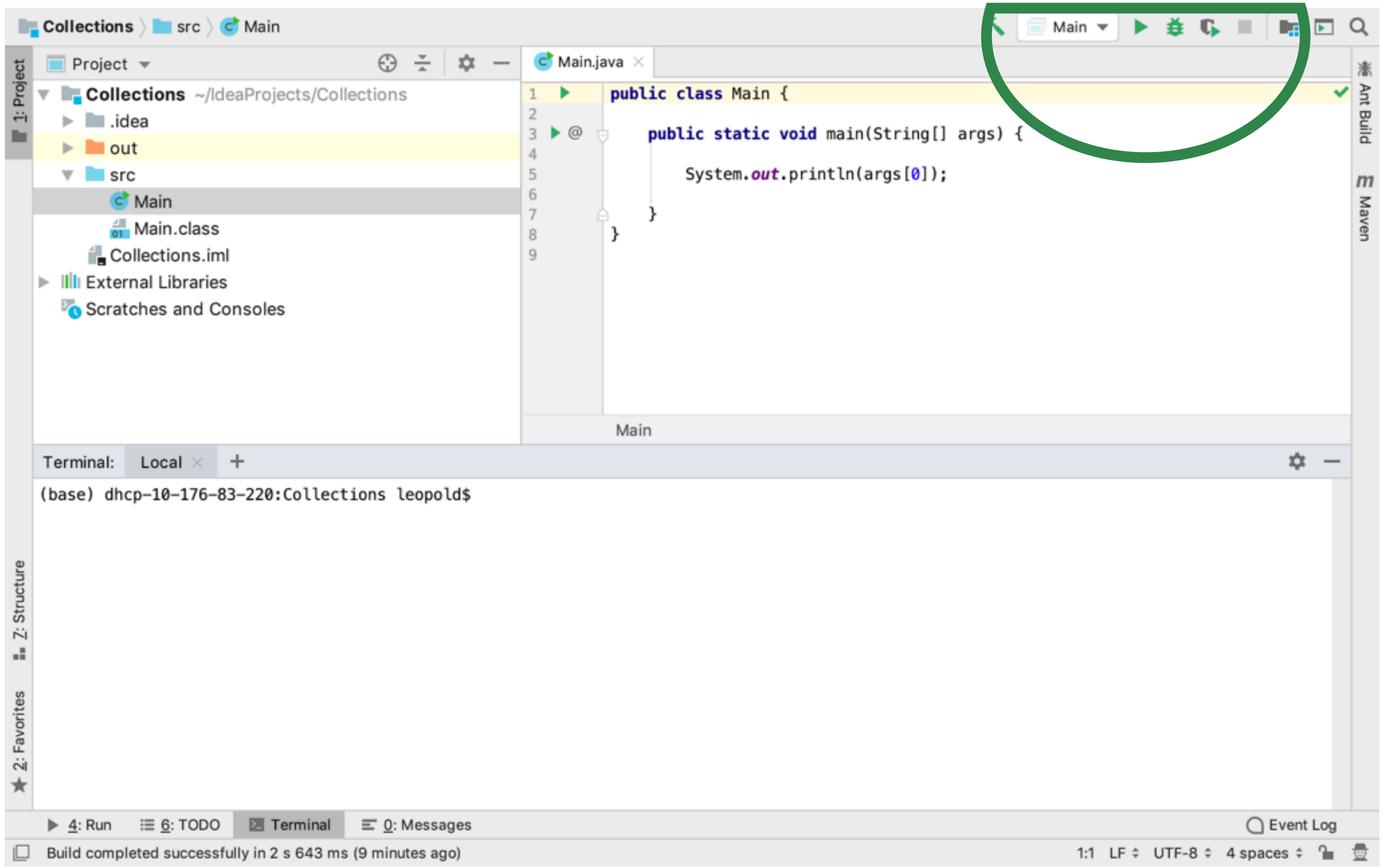
Im Terminal

```
dhcp-10-176-83-220:src leopold$ javac Main.java
dhcp-10-176-83-220:src leopold$ java Main Parameter1 Parameter2
Parameter1
```

args = {“Parameter1”, “Parameter2”}

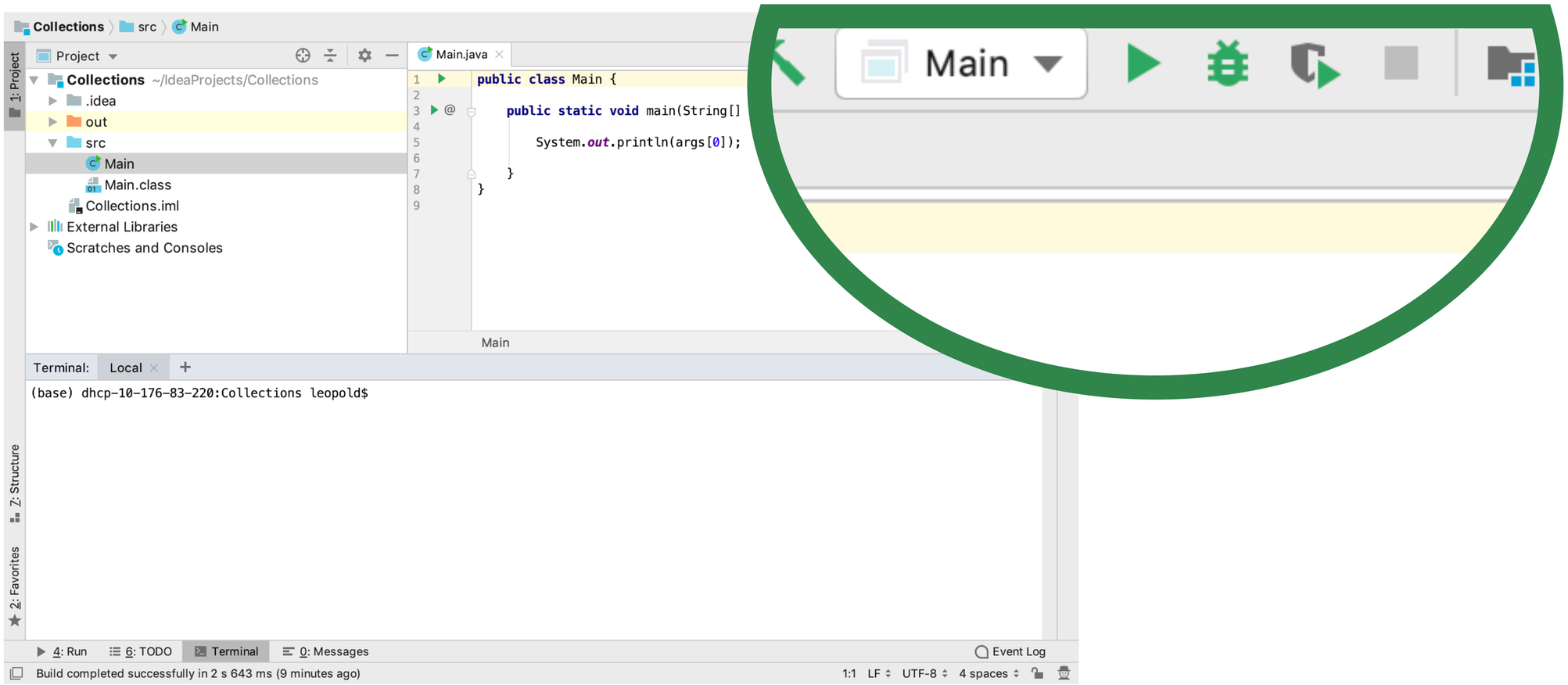
2. Kommandozeilenparameter in IntelliJ

- Zum Testen können die Parameter auch direkt in IntelliJ übergeben werden.



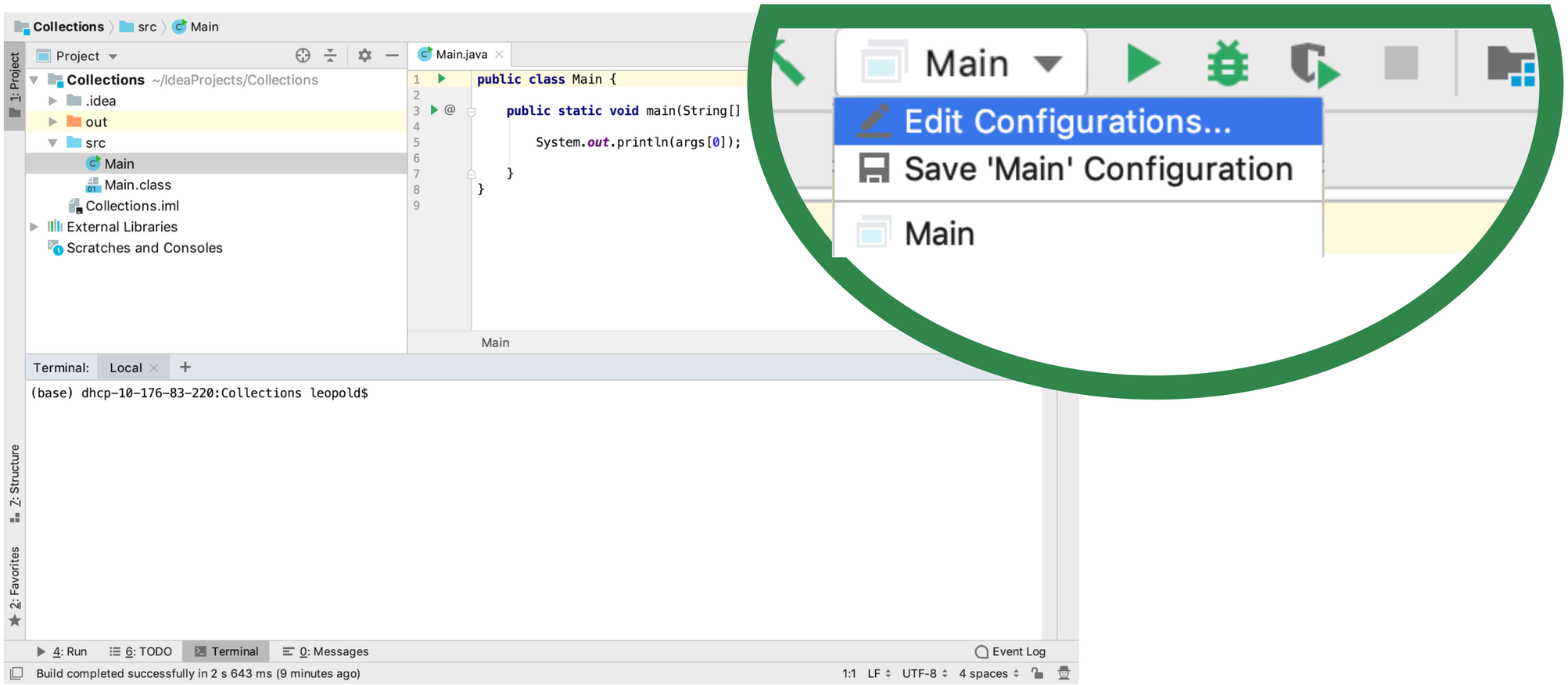
2. Kommandozeilenparameter in IntelliJ

- Zum Testen können die Parameter auch direkt in IntelliJ übergeben werden.



2. Kommandozeilenparameter in IntelliJ

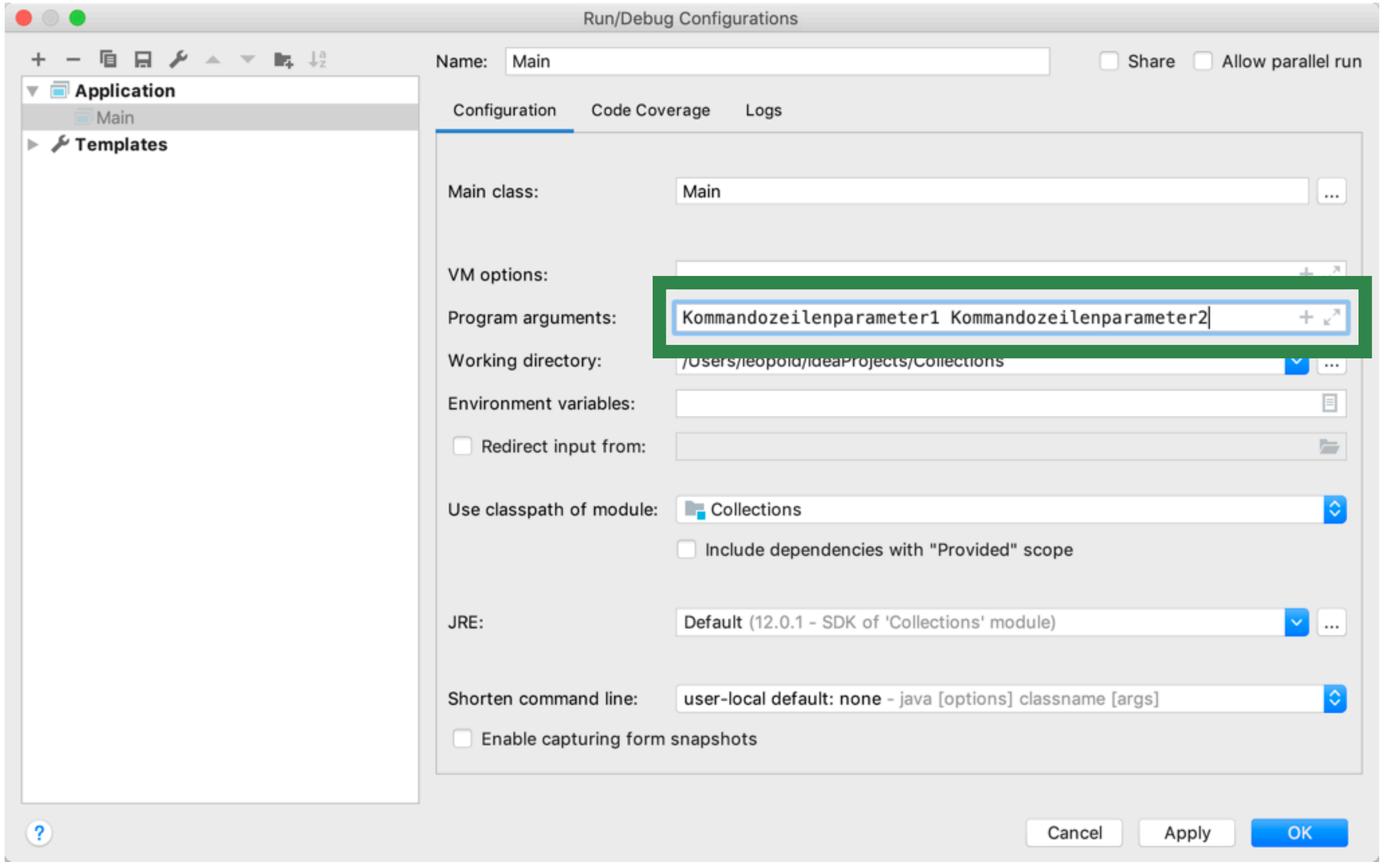
- Zum Testen können die Parameter auch direkt in IntelliJ übergeben werden.



2. Kommandozeilenparameter in IntelliJ

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

- Zum Testen können die Parameter auch direkt in IntelliJ übergeben werden.



3. FileReader

- Mithilfe eines *file-readers* können Dateien in ein Programm eingelesen werden.

3. FileReader

- Mithilfe eines *file-readers* können Dateien in ein Programm eingelesen werden.

Beispiel

Code

```
public class ReaderTest {

    public static void main(String[] args) throws IOException {

        ArrayList<String> lines = new ArrayList<String>();

        String filename = “text.txt";
        Path path = FileSystems.getDefault().getPath(filename);
        ArrayList<String> lines = (ArrayList<String>) Files.readAllLines(path);


    }
}
```

Fragen?