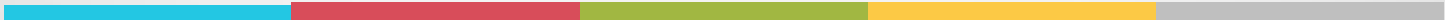


Debugging



Arten von Fehlern

Compiler Fehler

```
for (int i=1;i<=10;i++ ) {  
    System.out.println("Compiler error")  
}
```

→ ; is missing

Laufzeit Fehler

```
for (int i=1;i>=10; i++ ) {  
    System.out.println("Runtime error"+i/0);  
}
```

→ Exception in thread "main"
java.lang.ArithmeticException / by zero at test.main(test.java:4)

Logische Fehler

```
for (int i=1;i>=10; i++ ) {  
    System.out.println("Logical error")  
}
```

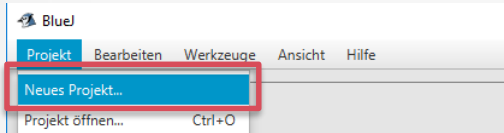
→



Erstellen eines neuen Projektes

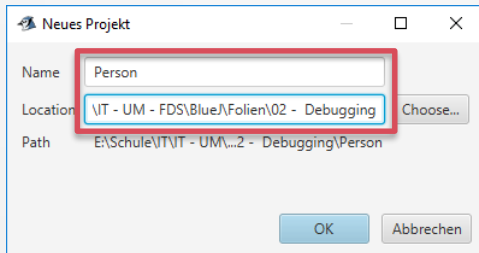
1.

Menüpunkt „Neues Projekt“ wählen



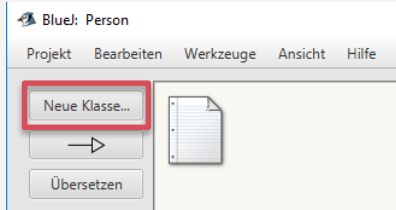
2.

Name(ohne „“) und neuen Pfad(der darf NICHT schreibgeschützt sein) eingeben:



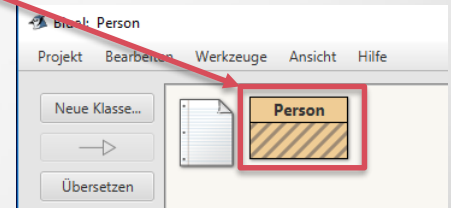
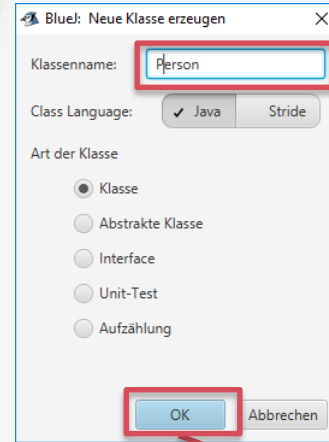
Erstellen einer neuen Klasse I

1.



2.

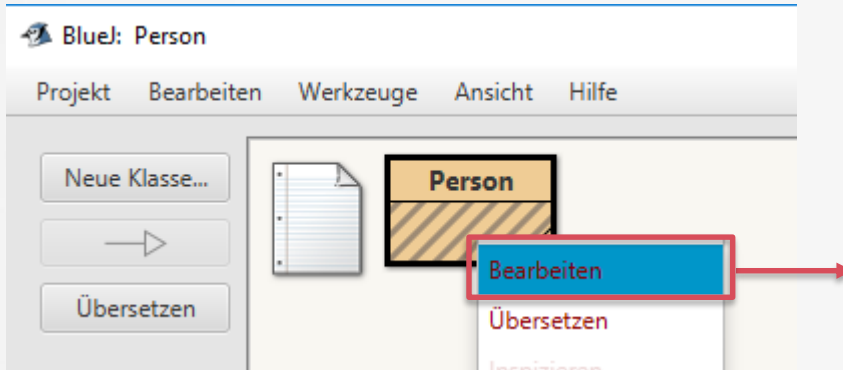
Namen der Klasse eingeben



Erstellen einer neuen Klasse II

3.

Rechte Maustaste betätigen:



4.

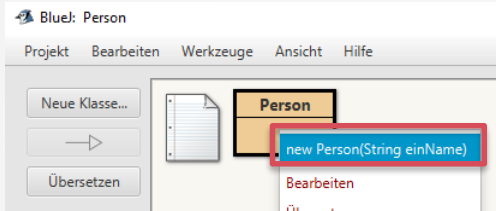
Quelltext rechts eingeben:

```
class Person {  
    private String name;  
    private int alter;  
    public Person( String einName) {  
        name = einName;  
    }  
    public void setAlter(int lebensjahre) {  
        alter = lebensjahre;  
    }  
    public String getName() {  
        return name;  
    }  
    public int getAlter() {  
        return alter;  
    }  
}
```

Erstellen eines neuen Objektes

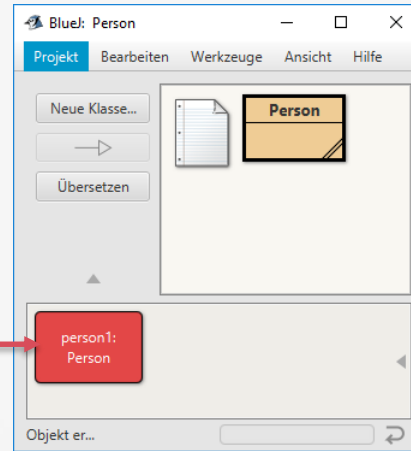
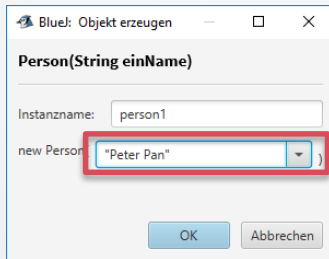
1.

Rechte Maustaste betätigen:



2.

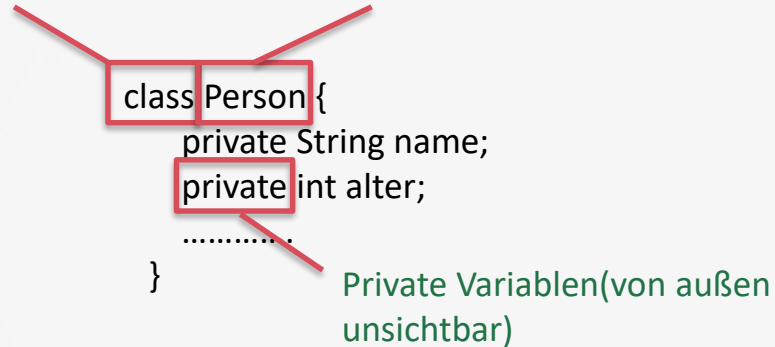
Namen der Person eingeben: Mit „“-Zeichen, da Datentyp String



Erläuterung der Klasse – Rahmen und private Variablen

Schlüsselwort
für Klassendefinition

Name der
Klasse(Großschreibung)



Erläuterung der Klasse - Konstruktor

Konstruktor: Wird beim Erstellen eines Objektes aufgerufen

Beim Erstellen eines Objektes wird 1 Parameter vom Typ String übergeben

```
class Person {  
    .....  
    public Person( String einName) {  
        name = einName;  
    }  
    .....  
}
```

Speichern des eingegebenen Wertes in der privaten Variable

Erläuterung der Klasse - Konstruktor

Zu jeder privaten Variable gehört ein Getter und Setter

Beim Erstellen eines Objektes wird 1 Parameter vom Typ String übergeben

Kein Rückgabewert!
Aufruf:
peter.setAlter(3);

```
.....  
private int alter;  
.....  
}  
public void setAlter(int lebensjahre) {  
    alter = lebensjahre;  
}
```

Immer EIN Parameter; muss vom gleichen Datentyp wie alter sein!

Rückgabewert „int“:
Der gleiche Datentyp wie die Variable „alter“

```
public int getAlter() {  
    return alter;  
}
```

Gibt „alter“ zurück:
Aufruf:
x=peter.getAlter();

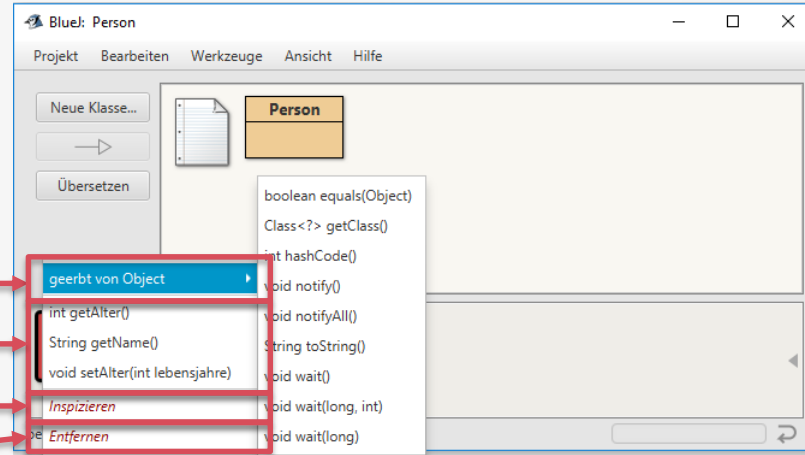
Kontextmenü (rechte Maustaste)

Methoden der Elternklasse Object

Methoden der Klasse

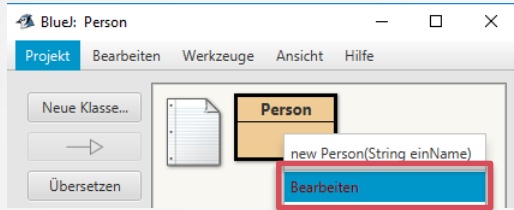
Eigenschaften der Klasse

Löschen der Klasse



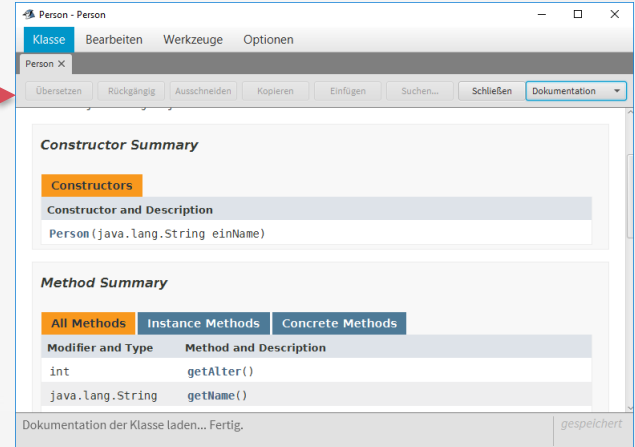
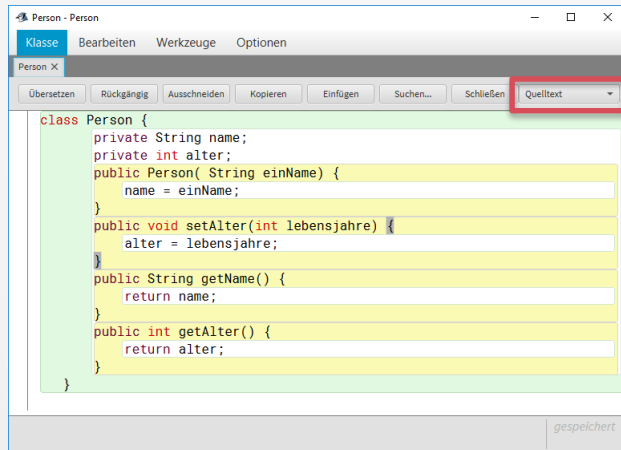
Dokumentation

1.



Im Kombinationsfeld
Dokumentation wählen

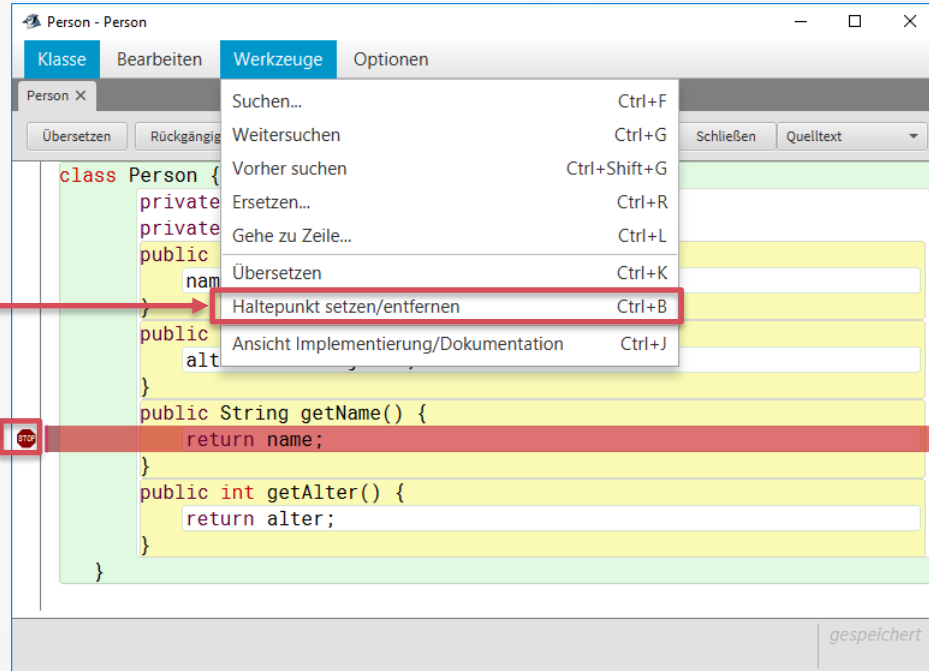
2.



Debugging – Haltepunkte setzen

Klick auf Zeile
oder Menüpunkt

Neues Objekt;
Aufruf von
getName();



Debugging – Haltepunkte verwenden

The screenshot shows an IDE window titled "Person - Person" with a menu bar containing "Klasse", "Bearbeiten", "Werkzeuge", and "Optionen". Below the menu bar is a toolbar with buttons for "Übersetzen", "Rückgängig", "Ausschneiden", "Kopieren", and "Einfügen". The main editor displays the following Java code:

```
class Person {  
    private String name;  
    private int alter;  
    public Person( String einName)  
        name = einName;  
    }  
    public void setAlter(int lebensjahre)  
        alter = lebensjahre;  
    }  
    public String getName() {  
        return name;  
    }  
    public int getAlter() {  
        return alter;  
    }  
}
```

A green arrow on the left margin points to the line `return name;`, indicating a breakpoint. The "BlueJ: Debugger" window is open, showing the following details:

- Optionen:** Threads: main (an Haltepunkt); Aufruffolge: Person.getName
- Statische Variablen:** (Empty)
- Instanzvariablen:** private String **name** = "Peter Pan"; private int **alter** = 0
- Lokale Variablen:** (Empty)

At the bottom of the debugger window is a toolbar with five buttons: "Anhalten" (stop icon), "Schritt über" (down arrow icon), "Schritt hinein" (up arrow icon), "Fortsetzen" (play icon), and "Beenden" (red X icon).

Debugging mit Arrays I

```
public class CStone{
    public int size, weight;
    public CStone(){}
    public void roll(){
        System.out.println("Stone ("+weight+"/"+"size+") is rolling!");
    }
}
```

Debugging mit Arrays II

```
import java.util.*;
public class CHill{
    private ArrayList<CStone> manyStones= new ArrayList<CStone>();
    public CHill(){
        for (int i=0;i<(int)(Math.random()*49+1);i++){
            CStone stone=new CStone();
            stone.size=(int)(Math.random()*49+1);
            stone.weight=stone.size*3;
            manyStones.add(stone);
        }
    }
    public void manyStonesAreRolling(){
        for (CStone stone: manyStones){
            stone.roll();
        }
    }
}
```

Debugging mit Arrays III

The screenshot shows the BlueJ IDE interface. At the top, the window title is "BlueJ: Arrays". Below the title bar is a menu bar with "Project", "Edit", "Tools", "View", and "Help". On the left side, there are buttons for "New Class...", "Compile", and "Share...". The main workspace displays a class diagram with two classes: "CHill" and "CStone". A dashed arrow points from "CHill" to "CStone", indicating inheritance. Below the workspace, there is a "Teamwork" section with a "Share..." button. In the bottom section, a variable named "myHighestHill" is shown with the value "CHill". A context menu is open over this variable, listing options: "inherited from Object", "void manyStonesAreRolling()", "Inspect", and "Remove". The status bar at the bottom indicates "myHighestHill : CHill".

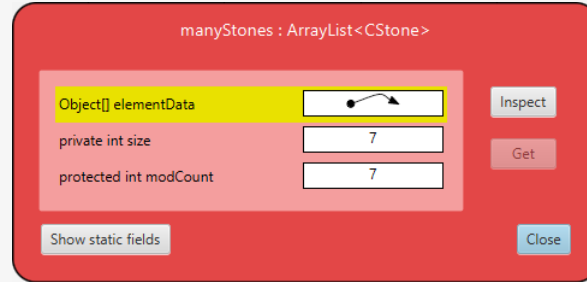
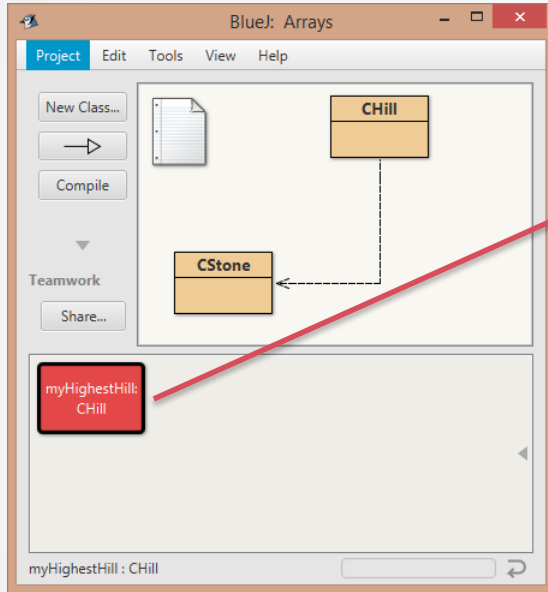


The screenshot shows a terminal window titled "BlueJ: Terminal Window - Arrays". The window has an "Options" button at the top left. The terminal displays the following output:

```
Stone (117/39) is rolling!  
Stone (96/32) is rolling!  
Stone (138/46) is rolling!  
Stone (144/48) is rolling!  
Stone (111/37) is rolling!  
Stone (135/45) is rolling!  
Stone (99/33) is rolling!
```

At the bottom of the terminal, there is a prompt: "Can only enter input while your prog".

Debugging mit Arrays IV



Debugging mit Arrays V

