

Abstraktní třídy

Tomáš Pitner, upravil Marek Šabo

Motivace

- Java disponuje rozhraními.
- Pak máme třídu(y) implementující určité rozhraní.
- Někdy je vhodné určité rozhraní implementovat pouze *částečně*:
 - *Rozhraní* = Specifikace
 - *Abstraktní třída* = Částečná implementace rozhraní (stačí mít hotové některé metody) a současně předek konkrétních tříd, tedy plných implementací
 - *Neabstraktní třída* = Úplná implementace rozhraní (musí mít hotové všechny metody)

Zápis abstraktní třídy

- Abstraktní třída je označena klíčovým slovem **abstract** v hlavičce, např.:

```
public abstract class AbstractSearcher
```

- Název začínající na **Abstract** není povinný ani nutný.
- Abstraktní třída má obvykle alespoň jednu *abstraktní metodu*, deklarovanou např.:

```
public abstract int indexOf(double d);
```

- Od abstraktní třídy *nelze vytvořit instanci*, nelze napsat např.:

```
Searcher ch = new AbstractSearcher(...);
```

Reálný příklad: rozhraní → abstraktní třída → neabstraktní třída

- Viz demo **searching** pro BlueJ:
 - **Searcher** = rozhraní — specifikuje, co má prohledávač umět
 - **AbstractSearcher** = abstraktní třída — předek konkrétních plných implementací prohledávače
 - **LinearSearcher** = konkrétní třída — plná implementace prohledávače

Searcher

- **Searcher** je rozhraní = specifikuje, co má prohledávač umět

```
public interface Searcher {
    // Set the array for later searching
    void setData(double[] a);
    // Detect where the array contains d
    boolean contains(double d);
    // Return the position of d in the array (or -1)
    int indexOf(double d);
}
```

AbstractSearcher

- **AbstractSearcher** je abstraktní třída = předek konkrétních plných implementací prohledávače

```
public abstract class AbstractSearcher implements Searcher {
    // the class AbstractSearcher implements Searcher but not fully
    // the array is implemented
    private double[] array;
    // setting the array is implemented
    public void setData(double[] a) { array = a; }
    public double[] getData() { return array; }
    // decide whether d is contained in array
    public boolean contains(double d) {
        return indexOf(d) >= 0;
    }
    // finding the position of d is NOT implemented, i.e. is abstract
    public abstract int indexOf(double d);
}
```

LinearSearcher

- **LinearSearcher** je konkrétní třída = plná implementace prohledávače, tentokrát pomocí lineárního prohledání

```
public class LinearSearcher extends AbstractSearcher {
    // finding the position of d is now finally implemented
    public int indexOf(double d) {
        double[] data = getData();
        for(int i = 0; i < data.length; i++) {
            if(data[i] == d) {
                return i;
            }
        }
        return -1;
    }
}
```