

Multiagentové systémy

Dr. Andrej Lúčny

KAI FMFI UK

lucny@fmph.uniba.sk

<http://www.agentspace.org/mas>

- **opakowanie języka Java**

Tvar programu

Java = triedno-inštančné OOP

Program = objekt so statickou metódou main

```
// MyProgram.java
public class MyProgram {
    public static void main (String[] args) {
        System.out.println( "Hello world!" );
    }
}
```

Ako editovať

Je vhodné mať editor čo zobrazí syntax

- V H3 je k dispozícii Notepad++
- NetBeans IDE 8.0.2

Ako skompilovať a pustiť

- `c:\Progra~1\Java\jdk1.8.0_144\bin\javac MyProgram.java`
z `cmd.exe` alebo z odpálením pripraveného
`CompileMyProgram.bat` (kompilujú sa aj závislé súbory)
- Bud' hodí chyby alebo vytvorí `MyProgram.class`
- `c:\Progra~1\Java\jdk1.8.0_144\bin\java MyProgram`
z `cmd.exe` alebo z odpálením pripraveného
`RunMyProgram.bat`
- Bud' hodí exception alebo sa pustí

Syntax

- záleží na veľkosti písmen, názov triedy musí byť veľkým písmenom
- v jednom súbore môže byť viacej tried ale iba jedna public a jej meno sa musí zhodovať s názvom súboru
- názov súbory .java má teda začínať veľkým písmenom

Typy

- Java rozlišuje medzi základnými typmi a objektami
- základné typy: int, float, double, char, byte, short, long, void
- Objekty: Integer, Float, Double, Character, Byte, Short, Long, Void, String, Random, MyClass
- konštanty majú svoj typ 1, 1L, 1.0, 1.0f, 'a'

Objekty

- každý objekt vzniká inštanciou určitej triedy
`MyObject myobject = new MyObject();`
object má konštruktor – metódu bez typu
- objekt zaniká zánikom odkazov naň (Garbidge collection), nemá deštruktor
- objekt má premenné a metódy
`myobject.premenna = 12;`
`x = myobject.metoda(12);`

Triedy

- definície premenných a metód

```
public class MyClass {  
    int myAttribute;  
    int myMethod (int myArgument) {  
        return (2 * myArgument);  
    }  
}
```

Modifikátory

- public
- protected
-
- private

static

final

Odovzdávanie parametrov

- metóda (okrem konštruktoru) vždy vracia jeden objekt alebo hodnotu
- ak je parametrom objekt odovzdáva sa do metódy referenciou
- ak je parametrom primitívny typ, odovzdáva sa hodnotou

Dedenie

```
public class VybornyStudent extends Student {  
}
```

- **prekrytie**
- **pret'aženie**

Balíky

- `java.lang`
- `java.util`
- `import java.util.Random;`

Polia

- `int [] a = new int[12];`
- indexované 0, 1, ..., n-1
- nedajú sa realokovať
- ak to potrebujeme – `java.util.ArrayList` a spol (kolekcie, mapy)
- ALE! tie sa už nedajú indexovať, pracujeme s nimi ako s objektami
- `for (Typ o : collection)`

Pretypovanie

- `float b = 1.2; int a = (int) b;`

Práca s reťazcami

- Objekty String a StringBuffer
- Operátor +
- String nemožno zväčšovať, zmenšovať, nie je typické ich modifikovať “a”+”b”
- Pri každej zmene sa vytvára nový String (likvidácia starých je ponechaná na garbage collector)
- StringBuffer sa dá zväčšovať (append)

Rozhrania

- vytvárajú sa keď potrebujeme volať objekt triedy, ktorá ešte neexistuje (tú si zafinuje až užívateľ toho objektu ktorý vytvárame a uvedie, že táto trieda toto rozhranie implementuje). Ide o realizáciu tzv. dynamickej vazby

```
interface B {  
    int met();  
}  
class A {  
    B b; ...  
    b.met();  
}
```

```
class C implements B {  
    int met() {  
        return 2;  
    }  
}
```

Polymorfizmus

- referencia objektov rôznych typov
pomocou všeobecnejšieho typu alebo
pomocou rozhrania

Výnimky

- okrem návratovej hodnoty môžu metódy hádzať výnimky

```
void A () throws FileNotFoundException {  
    ... throw new FileNotFoundException("chyba"); ...  
}
```

```
try {  
    A();  
} catch (Exception e) {  
    e.printStackTrace();  
} finally {  
}
```

Anonymné triedy

- Definícia odvodenej triedy priamo v kóde pri new
- Bez uvedenia mena odvodenej triedy

```
// ked mame klasu A urcenu na to aby sme z nej  
odvodili ...
```

```
class A {  
    public A () {  
    }  
    public void forOverRiding() {  
    }  
    ...  
}
```

```
// ... nasu myA  
class myA extends A {  
    public void forOverRiding() {  
        System.out.println("Ahoj Jelka");  
    }  
}
```

```
// a pouzili ju takto:  
new myA();
```

// mozeme ju definovat rovno pri pouziti ako tzv.
anonymnu triedu

```
new A() {  
    public void forOverRiding() {  
        System.out.println("Ahoj Jelka");  
    }  
};
```

// javac vtedy vytvori vlastnu klasu:

```
class A$12948238746 extends A {  
    public void forOverRiding() {  
        System.out.println("Ahoj Jelka");  
    }  
}
```

// skompiluje ju a zavola

```
new A$12948238746();
```

Objekty pre objekty

- Objekty rozširujúce jazyk (Thread, Timer, System, ...)
- Objekty slúžiace na prekonanie bariér jazyka – napr. vzor status
- Objekty podporujúce znovupoužiteľnosť, modifikovateľnosť a iné vlastnosti, ktoré objekty normálne nemajú, napr. vzory factory, adaptor, singleton, iterator, composition, builder, ...

Delay

```
int ms = 1000; // 1 second
try {
    Thread.sleep(ms);
} catch (InterruptedException ee) {
}
```


Timered task

```
int period = 1000; // 1 second
Timer timer = new Timer(true);
timer.scheduleAtFixedRate(
    new TimerTask() {
        public void run() {
            // do something
        }
    }, period, period
);
```

Thread

```
public class MyThread extends Thread {  
    public MyThread () {  
        start();  
    }  
    public void run () {  
        // do something  
    }  
}
```

Ked' neviem ako d'alej

Do vyhľadávača:

```
java apidocs <class-name>
```