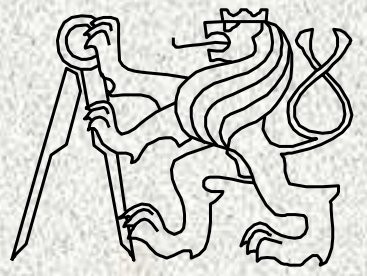
Java – základy



A0B36PR1-Programování 1

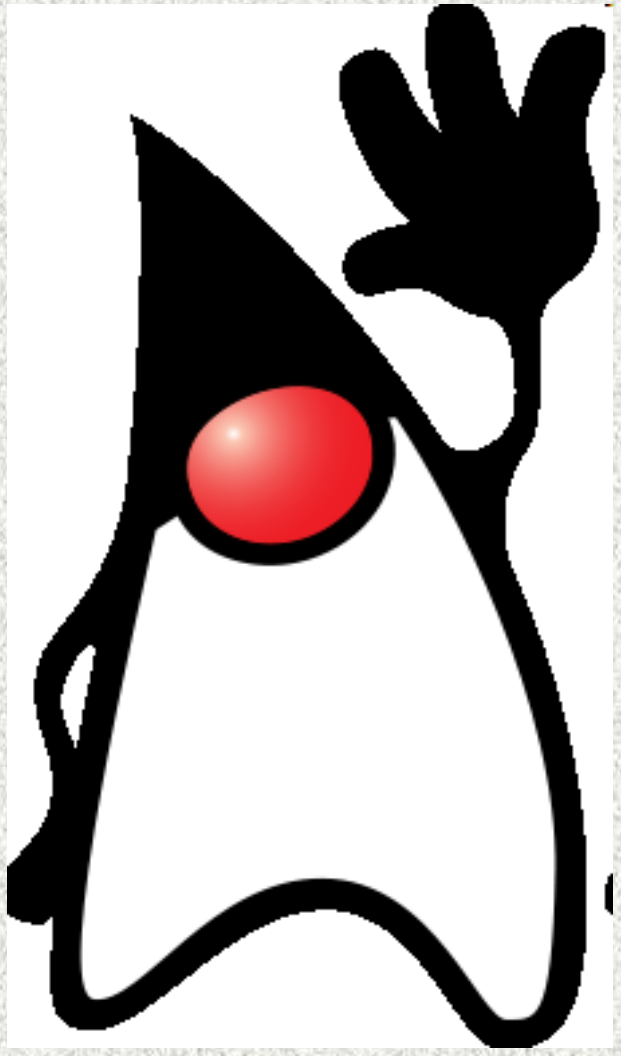
Fakulta elektrotechnická
České vysoké učení technické

Proč jazyk Java?

- jde o vyšší, obecně použitelný programovací jazyk s vysokým stupněm zabezpečení
- je objektově orientovaný, umožňuje však i klasické procedurální programování
- vytvořené programy jsou zcela portabilní (program vytvořený pod MS Windows bez problémů funguje pod Unixem a naopak)
- syntaxe výrazů a příkazů vychází z jazyka C; přechod z Javy na C nebo C++ je tedy jednodušší, než odjinud
- základní implementaci (JDK – Java Development Kit) firmy Sun lze pro prostředí Windows i Unix stáhnout ze stránek firmy Sun: *http://java.sun.com*
- My používáme vývojové prostředí NetBeans 6.7, fy. Sun Microsystem. *http://www.netbeans.org/*

Java - historie

- v roce 1990 vznikl Green Team vedený Jamesem Goslingem – jazyk Oak (dub)
- Java představena firmou Sun Microsystems 23. května 1995
- 1996 vydán první Java Developement Kit 1.0 (pro aplety)
- 2008 - JDK 6 Update 13 s JavaFX SDK



Duke, maskot Javy

Java - edice

- každá edice je určena pro specifické účely, pro specifická zařízení
  - Java Card - smart (chytrý) karty - platební a kreditní karty
  - J2ME Java 2 Micro Edition – midlety, Kilobyte VM, mobilní zařízení, PDA, set-top boxy, vyžaduje 160 kb ROM a 32 kB RAM (CLDC, MIDP)
  - J2SE standard edition – stolní počítače
  - J2EE enterprise edition – podnikové aplikace

Vývoj programů v Javě

- JRE - běhové prostředí, JRE = JVM +API

  pro běh programů

  – JVM – Java Virtual Machine – virtuální stroj

  – API – Appliacation Programming Interface - knihovny

- JDK - Software Development Kit pro Javu

  - sada základních nástrojů pro vývoj programů v Javě

    - JRE, překladač javac, debugger, javadoc, nástroje pro vytváření jar archivů, mnohé další

  - ke stažení na http://java.sun.com

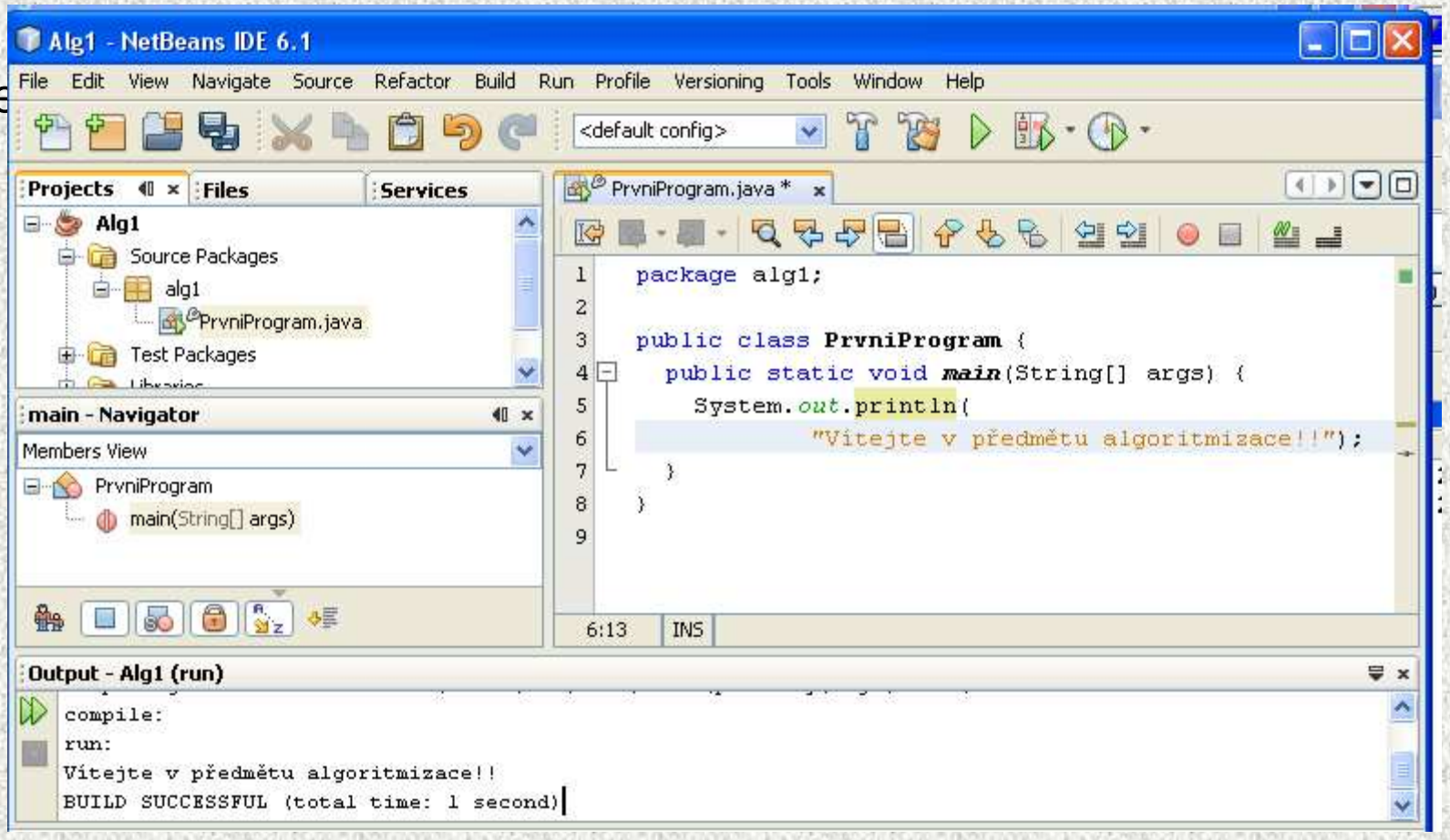
IDE - vývojová prostředí

IDE – integrované vývojové prostředí

- NetBeans – opensource -zdarma, původně Xelfi, vyvíjeno v Praze firmou SUN Microsystems
- Eclipse - opensource - zdarma, fa IBM
- IDEA - komerční (30tidenní zkušební verze zdarma)
- JBuilder - základní verze zdarma pro nekomerční využití, fa Borland
- JDeveloper - vývojové prostředí firmy Oracle, freeware
- BlueJ – volně šiřitelné multiplatformní vývojové prostředí

Vývojový systém

- Programy v jazyku Java budeme vytvářet pomocí vývojového prostředí NetBeans, který přípravu programu, jeho překlad a provedení zjednodušuje
- Se
uče



Úvod do jazyka Java

Jazyk Java je implementován interpretačním způsobem

- program je tvořen jedním nebo několika zdrojovými soubory s příponou **`.java:`**

  `Program.java`

- zdrojové soubory se přeloží překladačem(*) ***javac*** do vnitřní formy (byte code, bajt-kód) s příponou **`.class`**:

  `Program.java >` ***javac*** `> Program.class`

- interpretaci vnitřní formy provede program ***java*** *(*JVM – Java Virtual Machine v balíčku JRE Java Runtime Environment) a provede výpočet:

  `Program.class >` ***java*** `> „výpočet“`

(*) v terminologii firmy Sun to je kompilátor

Poznámka:

- **program obvykle využívá řadu knihoven, které je třeba mít k dispozici jak při překladu, tak při interpretaci!!!**

Zpracování programu v jazyku JAVA

Program.java

```
public class Program {
public static void main(String[] args) {
System.out.println(“Nazdar,toto je první program”);
   }
}
```

Spuštění překladače do byte-code:

```
javac Program.java
```

vznikne:

```
Program.class
```

spuštění interpretru:

```
java Program
```

Nazdar, toto je prvni program

První program v jazyku Java

- Příklad programu, který vypíše daný text na obrazovku:

```
public class PrvniProgram {
   public static void main(String[] args) {
     System.out.println("Nazdar, toto je prvni program");
   }
}
```

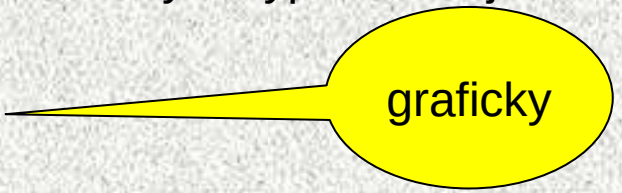
- Po překladu a spuštění se na obrazovku vypíše
  - Nazdar, toto je prvni program
- Nejjednodušší zdrojový program - jeden soubor
  - deklarace veřejné třídy (`public class`),
  - hlavní funkce `main` (veřejná statická metoda, public static method)
- Soubor musí mít jméno shodné se jménem veřejné třídy a příponu *.java*
- Hlavička funkce funkce `main` ():
  - klíčová slova `public static void` (`void` - procedura, nic nevrací)
  - `(String[] args)` specifikace parametrů zadané při spuštění
- Konvence: jména tříd se píší s prvním velkým písmenem

Jména a konvence

- **balíček - package -** jen malá písmena, i několik jmen oddělených tečkou, např: **`prog1, java.util`**
- **třída, abstraktní třída, jejich konstruktory, rozhraní - class, abstract class, interface** podstatné(interface přídavné) jméno začínající velkým písmenem, např: `String, MyFirstClass, Serializable, Comparable`
  Výjimky by měly mít sufix Exception, např: `MySpecialException`
- **metoda** -**method -** sloveso začínající malým písmenem, další slovo začíná velkým písmenem, např: `setBorder, isEmpty, getNumber`
- **proměnná - variable -** začínající vždy malým písmenem další slovo začíná velkým písmenem, např: `diskriminant, totalCount`
- **konstanta - final variable a návěští -** jen velká písmena, jednotlivá slova oddělena podtržítkem, např. `MAX_COUNT, RED`

Datové typy

- Při návrhu algoritmů a psaní programů ve vyšších programovacích jazycích abstrahujeme od binární podoby paměti počítače
- S daty pracujeme jako s hodnotami různých datových typů, které jsou uloženy v datových objektech
- Datový typ (zkráceně jen typ) specifikuje:
  - množinu hodnot
  - množinu operací, které lze s hodnotami daného typu provádět



- Příklad typu: celočíselný typ **`int`** v jazyku Java:
  - množinou hodnot jsou celá čísla z intervalu -2147483648 .. 2147483647
  - množinu operací tvoří
    - aritmetické operace +, -, *, /, jejichž výsledkem je hodnota typu *int*
    - relační operace ==, !=, >, >=, <, <=, jejichž výsledkem je hodnota typu **`boolean`**
    - a další
- Typ **`int`** je jednoduchý typ, jehož hodnoty jsou atomické (z hlediska operací dále nedělitelné)

Primitivní či základní datové typy

Typ	Bitů	Rozsah		Obal.třída
Celočíselný typ				
byte	8	-128 … 127		Integer
short	16	-32768 … 32767		Short
int	32	-2147483648 … 2147483647		Int
long	64	-9223372036854775808 … 9223372036854775807		Long
Reálný typ, IEEE 754 (NaN, infinity)				
float	32	2^{-149} … $(2-2^{-23})*2^{127}$		Float
double	64	2^{-1074} … $(2-2^{-52})*2^{1023}$		Double
Znaky, UCS2				
char	16	'\u0000' to '\uffff' 0 ... 65535		Character
Logický typ				
boolean	1/8	true false		Boolean
Pomocný prázdný typ				
void				

Počítání s primitivními typy

```
byte b1 = 6, b2 =8;
```

- b1 + b2 .... 14 POZOR! typ součtu je int, bez ohledu na hodnotu výsledku (stejně rozdíl, součin i podíl !!)

Tedy:

byte + byte ... int

short + short ... int

int + int ... int (pozor na přetečení a podtečení)

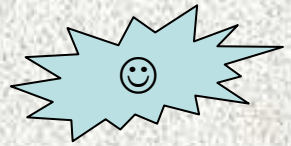
long + long ... long

Poznámky:

- to, co je zde uvedeno pro + platí i pro -,*,/,%
- celočíselné dělení: 17/6 ... 2 (pouze celá část, nezaokrouhluje)
- zbytek po dělení: 17%6 ... 5

Asociativita a priorita operací

- Binární operace na množině *S* **asociativní**, jestliže platí
  - (*x* * *y*) * *z* = *x* * (y * z), pro každé *x*, *y* a *z* v *S*.
- U neasociativních operací je tedy třeba buď důsledně závorkovat, nebo se dohodnout na implicitním pořadí provádění operací – pak se někdy mluví o operacích *asociativních zleva* či *asociativních zprava*.
- **Priorita binárních operací** vyjadřuje pořadí, v jakém se provádějí binární operace
- Příklady:
  - Odčítání levě asociativní,
    - výraz 10 − 5 − 3 se chápe jako (10 − 5) − 3,
  - Umocňování je asociativní zprava $2^{3^4} = 2^{(3^4)}$
  - $3 + 5^2 = 28$ nebo $3 \times 5^2 = 75$???



Vyhodnocení výrazů, příklady

124+4*8 = 124 + (4*8) = 124 + 32 = 156
// násobení má vyšší prioritu

36/2*9 = (36/2)*9 = 18*9 = 162
// násobení i dělení mají stejnou prioritu, asociativita je L - zleva

Operátory a jejich priorita

priorita	operátor	typ operandu	asociativita	operace
1	**++**	aritmetický	P	pre/post inkremetace
	--	aritmetický	P	pre/post dekrementace
	-	aritmetický	P	unarní plus/minus
	~	celočíselný	P	bitová inverze
	!	logický	P	logická negace
	(typ)	libovolný	P	přetypování
2	***, / ,%**	aritmetický	L	násobení, dělení, zbytek
3	**-**	aritmetický	L	odečítání
	+	aritmetický, řetězový	L	sčítání, zřetězení

Operátory a jejich priorita II

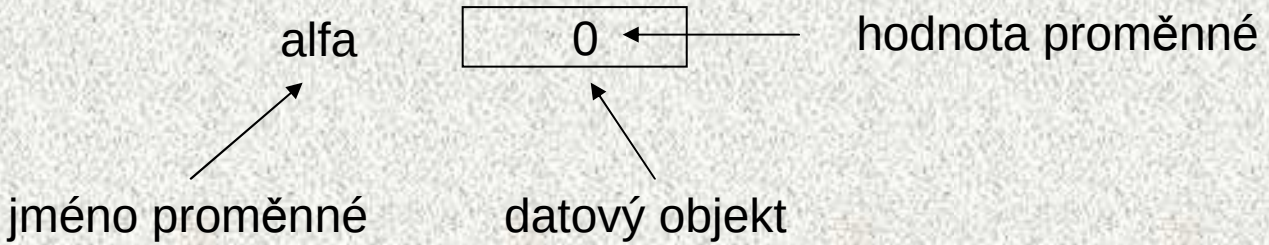
priorita	operátor	typ operandu	asociativita	operace
4	**<<**	celočíselný	L	posun vlevo
	>>	celočíselný	L	posun vpravo
	>>>	celočíselný	L	posun vpravo s doplňováním nuly
	<,>, <=, >=	aritmetický	L	porovnání
	instanceof	objekt	L	test třídy
	==, !=	primitivní	L	rovno, nerovno
7	**&**	celočíselný nebo logický	L	bitové nebo logické AND
8	**^**	-"-	L	bitové nebo logické XOR
9	**|**	-"-	L	bitové nebo logické OR

Operátory a jejich priorita III

priorita	operátor	typ operandu	asociativita	operace
10	**&&**	logický	L	logické AND vyhodnocované zkráceně
11	**||**	logický	L	logické OR vyhodnocované zkráceně
12	**? :**	logický	P	podmíněný operátor
13	**=, -=, *=, /=, %=, ==, &=, ^=, |=**	libovolný	P	přiřazení

Proměnné a přiřazení

- Proměnná je datový objekt, který je označen jménem a je v něm uložena hodnota nějakého typu, která se může měnit



- V jazyku Java zavedeme výše uvedenou proměnnou deklarací

```
int alfa = 0;
```

- Hodnotu proměnné lze změnit přiřazovacím příkazem

```
alfa = 37;
```

Deklarace proměnných

- každá proměnná má definovaný **typ**
- Java zná 8 primitivních datových typů
  - ostatní typy jsou referenční – objekty, pole, řetězce,..)

Příklad

```
int a;
```

- definuje proměnnou typu int
- lze do ní přiřadit pouze hodnoty typu int a hodnoty užší(tedy hodnoty, které se dají rozšířit na int, tedy byte, short a char
- lze s ní provádět operace definované pro int

Jak spolu souvisí typ int a typ „celá čísla“??

Deklarace proměnných

Proměnné se zavádějí deklaracemi

```
int i;          // deklarace proměnné i typu int
double x;       // deklarace proměnné x typu double
```

- Proměnná deklarovaná uvnitř funkce (lokální proměnná) nemá deklarací definovanou hodnotu
- Použití proměnné s nedefinovanou hodnotou v jazyku Java je chyba při překladu

```
int x, y;
x = y + 2;// chyba při překladu, není známa hodnota y
```

- Deklaraci proměnné lze doplnit o inicializaci proměnné:

```
int x = 10;     // deklarovaná proměnná má hodnotu 10
```

- Deklarací lze zavést několik proměnných stejného typu:

```
int x, z;
```

Deklarace proměnných

```
int a, b = 10;
```

- hodnota a není definována, hodnota b je 10

```
a = b;
```

- a je 10, b je 10

```
b = 12 + 5;
```

- a je 10, b je 17

```
a = a+3;
```

- vezmi hodnotu z a (10), zvětši ji o 3 a výsledek ulož zpět do a, a je 13
- Pozor, bude vysvětleno:
-

```
y = x = x + 6;
  // vyhodnotí se jako y = (x = (x + 6));
```

Ukázka vývojového prostředí

toto je nejlepší ukázat v NetBeans .....