Seminář Java

I

Program

- Organizace semináře
- Základní principy OOP
- Úvod do programovacího jazyka Java
- Distribuce
- Demonstrační příklad

Organizace semináře

- 12 seminářů po 3 blocích
  - Java – OO jazyk, architektura
  - Základní knihovny, vlákna, GUI
  - Pokročilé techniky
- Projekt
  - 30 bodů
  - společný s IPP (Principy programovacích jazyků a OOP)
  - podmínky projektu a odevzdání v rámci IPP
  - GUI pro překladač
- Klasifikovaný zápočet
  - Závěrečný test – 70 bodů
  - 13. seminář

Informace, studijní materiály

- Garant: Vladimír Janoušek
  - janousek@fit.vutbr.cz
- 1. blok: Radek Kočí
  - koci@fit.vutbr.cz
- 2. blok: Pavel Slavíček
  - slavicek@fit.vutbr.cz
- 3. blok: Ivan Šmarda
  - smardaiv@fit.vutbr.cz
- Stránky předmětu (http://www.fit.vutbr.cz/study/courses/IJA/)
- Prezentace semináře (1. - 4. seminář)
  - Vytvořeno na základě prezentací T. Pitnera (http://www.fi.muni.cz/ tomp/java/)

Proces Objektově orientované tvorby

- Objektově orientovaná analýza
  - Porozumění řešené doméně
- Objektově orientovaný návrh
  - Návrh řešení, model domény (struktura, aktivity)
- Objektově orientované programování
  - Implementace řešení
- Dobrý návrh tvoří 2/3 práce ...
- Je to proces, ne vodopád ...
- OOA je jazykově nezávislá

Objektově orientované programování – I

- Objektově orientované systémy:
  - kompozice doménově specifických objektových abstrakcí
  - sloučení dat a funkčnosti do objektu
  - objekty komunikují zasíláním zpráv
- Objekt je doménový koncept mající:
  - stav
  - chování
  - identitu
- Vlastnosti OOP
  - Abstrakce (abstraction)
  - Zapouzdření (encapsulation)
  - Polymorfismus (polymorphism)
  - Hierarchie (hierarchy) /dědičnost/

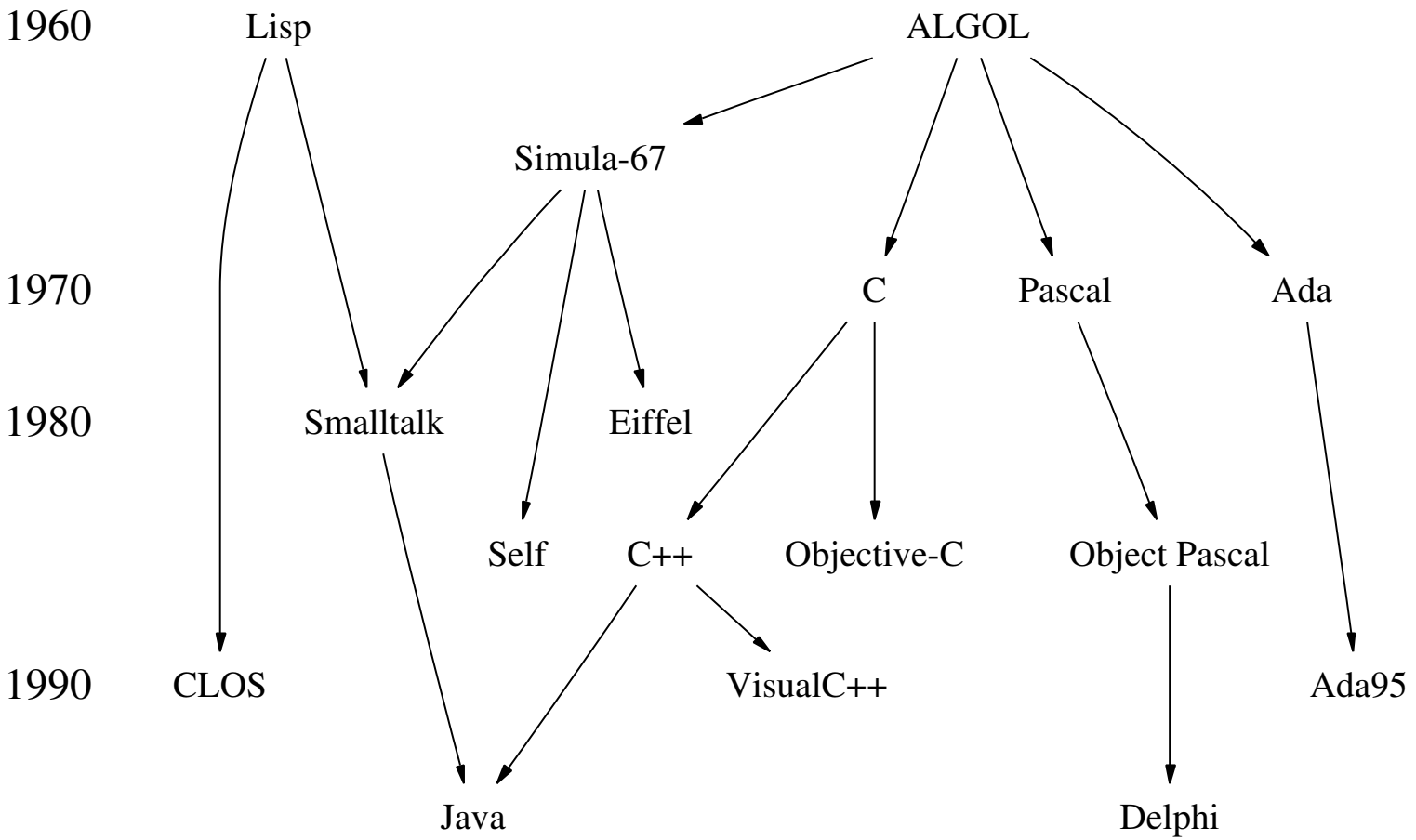
Objektově orientované programování – II

- Abstrakce
  - rozpoznávání podobností
  - zjednodušený pohled na reálný objekt
  - relativní
- Zapouzdření
  - ukrývání detailů
  - zaručené rozhraní
- Polymorfismus
  - logický vztah podobných operací (aplikace operací na podobné, ale technicky různé situace)
  - časná vazba / pozdní vazba
- Hierarchie
  - Klasifikace pořadí abstrakcí
  - Dědičnost (inheritance)
  - Agregace (aggregation), kompozice

Objektově orientované programování – III

- Typy
  - třída je chápána jako komplexní typ
  - statická kontrola typů
  - dynamická kontrola typů
- Souběžnost
  - objekty mohou konat ve stejném čase
  - procesy, vlákna
- Perzistence
  - Uložení stavu / dat během evoluce
  - Serializace

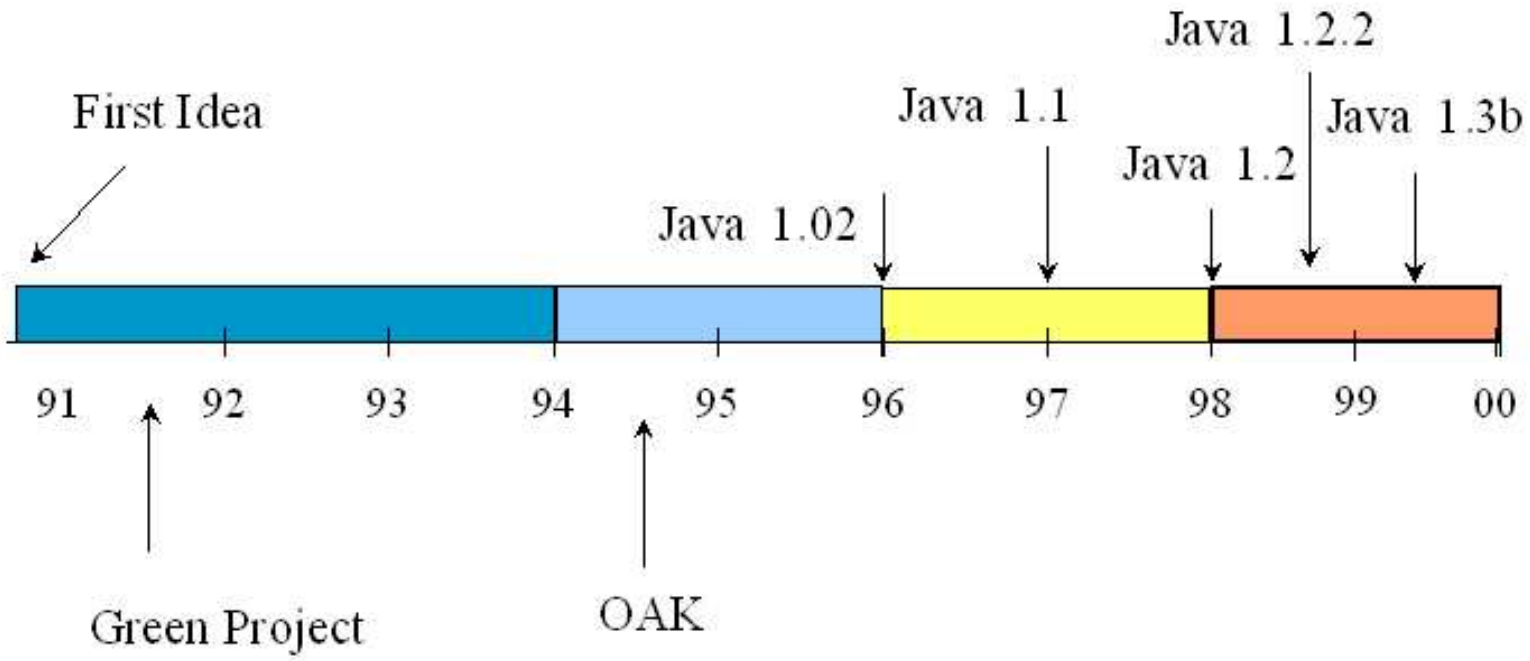
OOP – historie



Programovací jazyk Java

- univerzální (není určen výhradně pro specifickou aplikační oblast)
- objektově-orientovaný (výpočet je realizován jako volání metod/zasílání zpráv objektů)
- jednodušší než C++
- reálným soupeřem je (Microsoft) C# (zatím převážně na platf. Windows)
- program v Javě je meziplatformně přenositelný na úrovni zdrojového i přeloženého kódu
- je to umožněno tím, že přeložený javový program běží v tzv. Java Virtual Machine (JVM)
- zdrojový i přeložený kód je tedy přenositelný mezi všemi obvyklými platformami (UNIX, Windows, MAC OS X, ale také sálové počítače, minipočítače typu IBM AS/400 apod.)
- tedy všude tam, kde existuje příslušná JVM

Java – vývoj


First Idea
Java 1.02
Java 1.1
Java 1.2
Java 1.2.2
Java 1.3b
91
92
93
94
95
96
97
98
99
00
Green Project
OAK

Využití Javy – I

- jazyk vhodný pro efektivní (rychlé) psaní přehledných programů (mj. také díky dokumentačním možnostem)
- Java je jednodušší než C++ (méně syntaktických konstrukcí, méně nejednoznačností v návrhu)
- v průměru vyšší produktivita programátorské práce v Javě než v C++
- zdarma dostupné nezměrné množství knihoven pro různorodé aplikační oblasti, např. na SourceForge a tisících dalších místech
- k dispozici je řada kvalitních vývojových prostředí (i zdarma) - NetBeans, JBuilder, Visual Age for Java, Eclipse, IDEA
- v Javě se dobře píší vícevláknové aplikace (multithreaded applications)
- Java má automatické odklizení nepoužitelných objektů (automatic garbage collection)

Využití Javy – II

- Škálovatelné výkonné aplikace běžící na serverech (Java Enterprise Edition)
- Aplikace na přenosných a vestavěných zařízeních (Java Micro Edition)
- webové aplikace (servlety, JSP) - konkurence proprietárním ASP, SSI, CGI
- zpracování semistrukturovaných dat (XML)
- přenositelné aplikace s GUI
- aplikace distribuované po síti (applety nebo Java Web Start)

Typy aplikací

- Konzolové aplikace
  - jednoduchá textová konzole
- GUI aplikace
- Applety
  - běží v HTML prohlížečích
  - mají silná bezpečnostní omezení

Java – platforma

Java platfromu tvoří:

- Java Virtual Machine (JVM)
- překladač (přístupný např. příkazem javac) a další vývojové nástroje
- Java Core API (základní knihovna tříd)

Java je tedy dána...

- definicí jazyka (Java Language Definition) - syntaxe a sémantika jazyka
- popisem chování JVM
- popisem Java Core API

Specifikace a implementace Javy

- Specifikace Javy (tzv. "Editions") - např.: Java 2 Standard Edition, v1.4
- Implementace Javy ("Development Kits" nebo "Runtime Environments") - např.: Java 2 Software Development Kit, v1.4.2 - obsahuje vývojové nástroje
- Java 2 Runtime Enviroment, v1.4 - obsahuje jen běhové prostředí pro spouštění hotových přeložených pg.

Verze Javy

- hrubé členění - na verze "Java (před Java 2)" a "Java 2"
- číslování verzí:
  - tzv. major číslo, např. Java 2, v1.4
  - tzv. minor číslo, např. Java 2, v1.4.2
- změnu minor (třetího) čísla doprovází jen odstraňování chyb
- při změně major (druhého) čísla se může měnit Core API a někdy i jazyk
- ke změně prvního čísla zatím nedošlo...
- Aktuální verze
  - Java 2 Standard Edition v1.4.2 pro všechny platformy
  - aktuálně vždy na webu http://java.sun.com

Získání distribuce Javy

- používání Javy pro běžný vývoj (i komerční) je zdarma
- redistribuce javového vývojového prostředí je dovolena pouze s licencí od Sunu
- redistribuce javového běhového prostředí je možná zdarma
- distribuce vyvíjí Sun Microsystems Inc. (Javasoft) i další výrobci (např. IBM) a tvůrci Open Source

Stažení distribuce Sun

- http://java.sun.com (pro Windows, Solaris, Linux)
- dokumentace se stahuje z téhož místa, ale samostatně (nebo lze číst z WWW)
- celkově vývojové prostředí J2SDK 1.4.2 vč. dokumentace zabere cca 220 MB na disku
- potřebná velikost operační paměti - min 64 MB, doporučeno 128 MB (i více :-))

Obsah vývojové distribuce Javy

- Vývojové nástroje (Development Tools) v bin – určené k vývoji, spouštění, ladění a dokumentování programů v Javě.
- Běhové prostředí Javy (Java Runtime Environment) se nalézá v jre. Obsahuje Java Virtual Machine (JVM), knihovnu tříd Java Core API a další soubory potřebné pro běh programů v Javě.
- Přídavné knihovny (Additional libraries) v podadresáři lib jsou další knihovny nutné pro běh vývojových nástrojů.
- Ukázkové applety a aplikace (Demo Applets and Applications) v demo. Příklady zahrnují i zdrojový kód.

Nástroje ve vývojové distribuci

Pod Windows jsou to .exe soubory umístěné v podadresáři bin

- java - spouštěč (přeloženého bajtkódu)
- javac - překladač (.java -> .class)
- javadoc - generátor dokumentace API
- jar - správce archivů JAR (sbalení, rozbalení, výpis)
- jdb - debugger
- appletviewer - referenční prostředí pro spouštění appletů

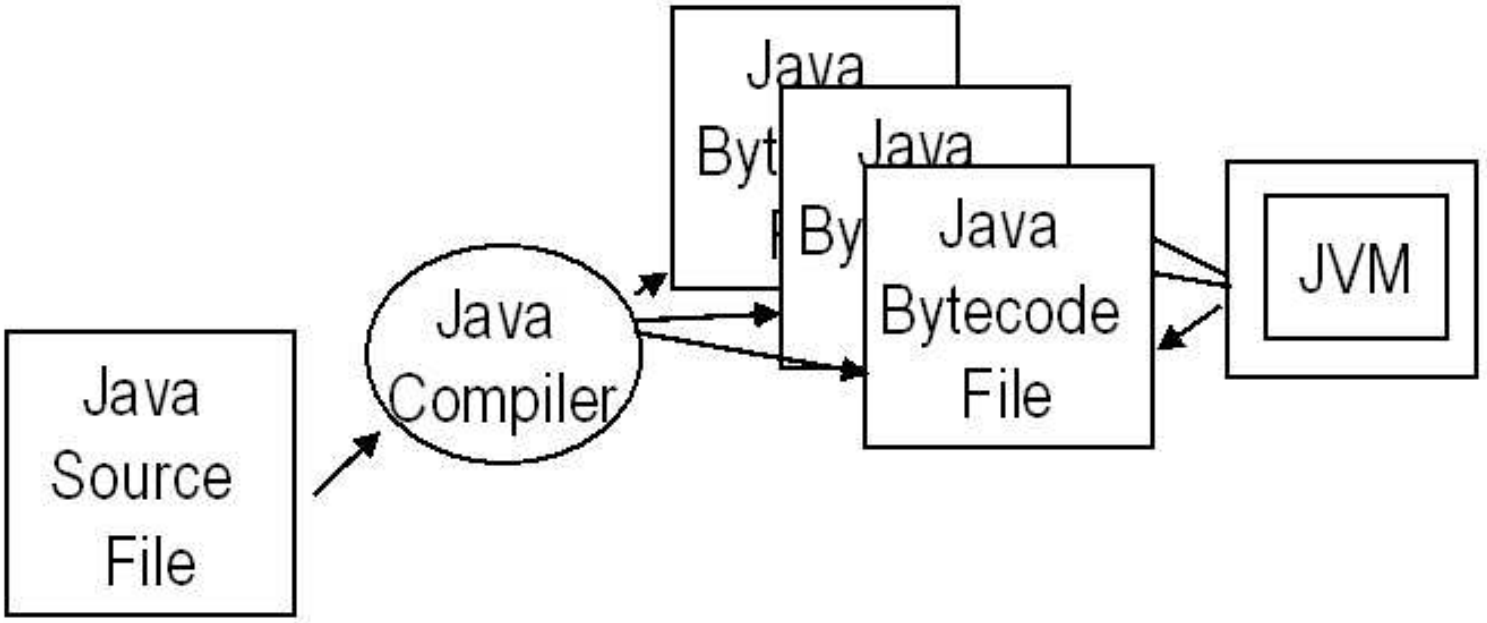
Základní životní cyklus javového programu

- Program sestává z jedné (ale obvykle více) tříd (class)
- Zdrojový kód každé veřejně přístupné třídy je umístěn v jednom souboru (NazevTridy.java)
- Postup:
  - vytvoření zdrojového textu (libovolným editorem čistého textu) -> Pokus.java
  - překlad (nástrojem javac) Pokus.java -> Pokus.class
  - spuštění, např. java Pokus

- překládá se javac název souboru se třídou (včetně přípony .java!!!)
- spouští se vždy udáním java a názvu třídy (bez přípony .class!!!)

Java Virtual Machine

- Překladač generuje byte-kód pro JVM
- JVM interpretuje byte-kód
- Optimalizace (JIT)



Struktura javového programu

- Každý netriviální javový program sestává z více tříd (class)
- Třídy jsou členěny do balíků (package)
- Zařazení do balíků znamená mj. umístění zdrojového souboru do příslušného adresáře!!!
- U běžné "desktopové" aplikace představuje vždy jedna (evt. více) třída vstupní bod do programu - je to třída/y obsahující metodu main.

Ukázka aplikace

Soubor Pozdrav.java je umístěn v balíku IJA.seminar1 (tj. v adresáři IJA/seminar1)

```
package IJA.seminar1;
public class Pozdrav {
    // Program spouštíme aktivací funkce "main"
    public static void main(String[] args) {
        System.out.println("Ahoj!");
    }
}
```

http://java.sun.com/reference/api/index.html

Překlad

1. Máme nainstalován J2SDK 1.4.2
2. Jsme v adresáři $HOME, v něm je podadresář IJA/seminar1, v něm je soubor Pozdrav.java
3. Spustíme překlad
   javac IJA/seminar1/Pozdrav.java
4. Je-li program správně napsán, přeloží se "mlčky"
5. Výsledný .class (Pozdrav.class) soubor bude v témže adresáři jako zdroj

Spuštění

1. Poté spustíme program Pozdrav:
   java -classpath . IJA.seminar1.Pozdrav
2. Volba překladače -classpath adresář zajistí, že (dříve přeložené) třídy používané při spuštění této třídy budou přístupné pod adresářem adresář.
3. -classpath . tedy značí, že třídy (soubory .class) se budou hledat v odpovídajících podadresářích aktuálního adresáře (adresáře .)
4. Je-li program správně napsán a přeložen, vypíše se Ahoj!

Co znamená spustit program?

Spuštění javového programu = **spuštění metody main jedné ze tříd tvořících program**
Tato funkce může mít parametry:

- podobně jako např. v Pascalu nebo v C
- jsou typu String (řetězec)
- předávají se při spuštění z příkazového řádku do pole String[] args

Metoda main nevrací žádnou hodnotu - návratový typ je vždy(!) void

Její hlavička musí vypadat vždy přesně tak, jako ve výše uvedeném příkladu, jinak nebude spuštěna!

Praktické informace

Co je nutné udělat

- Cesty ke spustitelným programům (PATH) musejí obsahovat i adresář $JAVA_HOME/bin

Co je vhodné udělat

Systémové proměnné by měly obsahovat:

- JAVA_HOME=kořenový adresář instalace Javy, např. JAVA_HOME=/usr/local/j2sdk1.4.2
- CLASSPATH=cesty ke třídám (podobně jako v PATH jsou cesty ke spustitelným souborům), např. CLASSPATH=$HOME/java

Distribuce Javy na FIT

- adela.fit.vutbr.cz
  - 1.4.2
  - proměnné jsou nastaveny (kromě CLASSPATH)

Ukázkový příklad

Adresář $HOME:

java
- IJA
  - seminar1
    - Pozdrav.java

Soubor Pozdrav.java:

```
package IJA.seminar1;
public class Pozdrav {
    // Program spouštíme aktivací funkce "main"
    public static void main(String[] args) {
        System.out.println("Ahoj!");
    }
}
```

Ukázkový příklad – II

- Překlad
  - cd $HOME/java
  - javac IJA.seminar1.Pozdrav.java
- Spuštění
  - java -classpath . IJA.seminar1.Pozdrav
- Spuštění
  - cd $HOME
  - java -classpath $HOME/java IJA.seminar1.Pozdrav
- Spuštění
  - export CLASSPATH="$CLASSPATH:$HOME/java"
  - java IJA.seminar1.Pozdrav

Cvičení

- Vyzkoušejte si přeložit a spustit ukázkový příklad.
- Experimentujte s kódem, překladem a spouštěním.