A Java és a C++ összehasonlítása

Kozsik Tamás

kto@elte.hu
http://kto.web.elte.hu/

Eötvös Loránd Tudományegyetem
Programozási Nyelvek és Fordítóprogramok Tanszék

2008.

Tartalom

Tartalom

Tartalom

Tartalom

Áttekintés

- Olcsó programfejlesztés
- Biztonságra való törekvés
  - Safety
  - Security
- Memóriakezelés
- Statikus szabályok
- Dinamikus szemantikai szabályok
- `http://java.sun.com/`

Típusok

- Primitív típusok
- Referencia típusok
  - Osztályok (konkrét és absztrakt)
    - Tömb típusok
    - Felsorolási típusok
  - Interfészek
    - Annotációk

Primitív típusok

- 8 db beépített típus
  - `boolean`
  - `char`
  - `byte`, `short`, `int`, `long`
  - `float`, `double`
- Rögzített ábrázolás
- Nincs előjel nélküli egész típus
- A `char` típus 2 bájtos Unicode
- A logikai típus önálló (vezérlési szerk., relációk)
- Altípusosság; konverzió: automatikusan csak bővítő (!)
- Osztályosítás: csomagoló osztályokkal
  - Auto-(un)boxing

Referenciák

- A vermen csak primitív típusú adatok és referenciák
- Az összetett adatok mindig dinamikusak
- Felszabadító utasítás nincs
- Automatikus szemétgyűjtés
- Destruktor – `finalize`

Tömbök

- Referenciák
  - Nem lehet deklarációval létrehozni
  - Minden tömb a heap-en van
- Speciális osztályok, öröklődés
- Biztonságos használat
  - `length` attribútum
  - Futási idejű indexellenőrzés

Használat

```
int[] t = new int[100];
for( int i = 0; i<t.length; ++i ){ t[i] = i; }
```

Szövegek

- Nincs `char*`
- Elsődlegesen a `String` osztály
  - Pl. idézőjelek
  - Konkatenáció: túlterhelt `+`
- Emellett `StringBuffer` és `StringBuilder`
- Ritkán `char[]`

Felsorolási típusok

A legegyszerűbb eset

```
enum Color { RED, GREEN, BLUE, YELLOW }
```

- Nem `int`
- Speciális osztályok
- Definiálhatók hozzájuk mezők és műveletek

Vezérlési szerkezetek

- Elágazás: `if` és `switch–case`
- Ciklus: `for`, `while` és `do–while`
  - Iterálás adatszerkezeten: enhanced for-loop
- Blokk utasítás
- Nem strukturált
  - Címkézhető `break` és `continue`
  - Nincs `goto`

Iterálás tömbön

```
int[] t = new int[100]; ...
int sum = 0;
for( int elem: t ){ sum += elem; }
```

Választás

- `if` és `while` esetén: logikai típusú feltétel
- `switch` esetén diszkrét vagy felsorolási típusú diszkrimináns

Deklaráció utasítások

- Típus-, alprogram- és változódeklarációk
- Konvenció: azonosítók neve
- A léptetős ciklusnak lehet lokális változója, de a többi utasításnak nem!
- Konvenció: használjuk a kapcsos zárójeleket, tördelés
- Típuskifejezésekhez nem deklarálható név (nincs `typedef`)

Blokkok

- Egyszerűsített hatóköri/láthatósági szabályok
- Egymásba ágyazott blokkok lokális változói
- Elfedés: csak tagok esetében
- Lokális változók inicializálása

Megjegyzések

- Egysoros és többsoros
- Dokumentációs megjegyzés
  - `/**` és `*/` között
  - Kódegység elé írható
  - `javadoc`
  - Tipikusan HTML, de programozható (doclet)
  - Kiegészítő információk (`@return`, `@param`, `@see` stb.)

Kifejezések

- A kiértékelési sorrend pontosan definiált
- Operátorokat nem terhelhet túl a programozó
- Lusta és mohó logikai operátorok
- Nincsenek explicit mutatók (dereferencing, címképzés, aritmetika)
  - Hivatkozás tagokra: `.`
- Konstansok nincsenek
  - `final` van
  - Üres konstans
  - Fordítási idejű konstans
  - Feltételes fordítás
- Az operátorok picit mások
  - Nincs `sizeof` és `typeid`
  - Van `instanceof`, valamint `>>>` és `>>>=`

Programegységek

- Típusdefiníciók
  - Metódusok és mezők
  - Deklarációk sorrendje
- Nincsenek globális alprogramok
  - Példány- vagy osztálymetódusok
  - Procedurális programozás imitálása
- Nincsenek globális változók sem!
- Csomagok
- Típusdefiníciók egymásba ágyazása

Alprogramok

- Paraméterátadás
- Nem adható formális paraméternek alapértelmezett érték
- Túlterhelés
- Inline-osítás
- Felüldefiniálás
- Hívási lehetőség: invokevirtual, invokestatic, invokespecial
- Változó hosszúságú paraméterlista
  - `printf`

Kivételek

- Sokkal intenzívebben használt
- `try–catch–finally` szerkezet
  - Nincs automatikus ábrázolású objektum
- A `Throwable` leszármazottai
- Ellenőrzött kivételek, `throws` kulcsszó
- Altípusosság, kovariancia
- Előfeltételek, `assert`

Forrásszöveg

- Forrásfájl
  - Karakterkódolás
- Fordítási egységek
  - Csomagok
  - `import` utasítás
- Nincs előfordító
- Nincs szükség fejállományokra
  - Kompatíbilitási kérdések

Fordítás és futtatás

- Fordítók jellemzően JVM bájtkódot csinálnak
  - Esetleg platformfüggő kódot: gcj
  - Nem csak Java forrás fordítható JVM-re
- Dinamikus szerkesztés
- Bájtkód interpretálása
- JIT-compilation
- Futtatás
  - Betöltés
    - Programozható
    - Dinamikus/mobil kód
  - Ellenőrzés
  - Inicializáció
  - Megszűnés
- Kódtranszformáció
- Futási idejű információk
  - Önelemzés
  - Visszafejtés

Programkönyvtárak

- Szabványos könyvtárak
  - Változatok
    - Standard Edition
    - Enterprise Edition
    - Micro Edition
  - Csomagok: `java.*`, `javax.*` stb.
  - Legalapvetőbb: `java.lang`
- Saját könyvtárak
  - Csomagolási konvenciók

Referencia típusok definiálhatók

- Osztályok (konkrét és absztrakt)
  - Felsorolási típusok
- Interfészek
  - Annotációk

- Definíció
  - Egy egység, nincs szétbontva
  - Nincs pontosvessző
  - `extends`

Öröklődés

- Osztályok: egyszeres kódöröklés
  - Minden osztály őse: `java.lang.Object`
  - Felüldefiniálás
  - Dinamikus kötés
  - Megvalósítás
- Interfészek: többszörös
  - Absztrakt osztályra hasonlít
  - Csak absztrakt metódusok és végleges értékű osztálymezők
  - Minden tagja publikus
- Osztály megvalósíthat interfészeket

Objektumok élettartama

- Dinamikus ábrázolás
- Allokációtól szemétgyűjtésig
- Inicializáció menete
  - Alapértékek
  - Inicializáló kifejezések és blokkok
  - Konstruktorok végrehajtása
- Megszűnés menete
  - Szemétgyűjtés, erőltetése
  - Gyenge referenciák
  - `finalize`

Metódusok túlterhelése

- Ugyanolyan névvel, különböző szignatúrával
- Örökölt metódus is túlterhelhető
- Túlterhelés és altípusosság
- Statikus és dinamikus kötés, pl. `equals`

Példa

```
class A { void m(){} }
class B extends A { void m(int i){} }
```

Láthatósági kategóriák

- Publikus (`public`)
- `protected`
- Félnyilvános (package-private)
- `private`

Beágyazott típusdefiníciók

- Nem csak statikus tagosztályok
- Belső osztályok is vannak
  - Példányszintű tagosztályok
  - Lokális osztályok
  - Névtelen osztályok

Objektumok összehasonlítása és másolása

- Az == referenciát hasonlít
- `public boolean Object.equals(Object)`

- Nem copy-konstruktorral és értékadó operátorral másolunk
- `protected Object Object.clone()`
  `throws CloneNotSupportedException`

Funktorok

- Alprogramok átadása paraméterként
  - Nincs függvényre mutató pointer
  - Objektumosítás
- Interfész és megvalósítása
  - Nincs függvényhívás operátor
  - Pl. `apply` vagy `execute` nevű metódus

Generikus programozás

- Típussal való paraméterezés, sablonok
- Parametrikus polimorfizmus
- Megkötések a típusparaméterről
- Adatszerkezetek
  - `java.util`
  - Collection Framework

Generatív programozás?

- Nincs template metaprogramming
- Önelemzés
- Aspektus-elvű programozás, AspectJ
- Annotációk