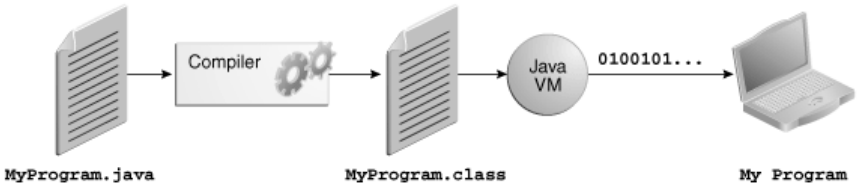
Programozás III

JAVA ALAPOK

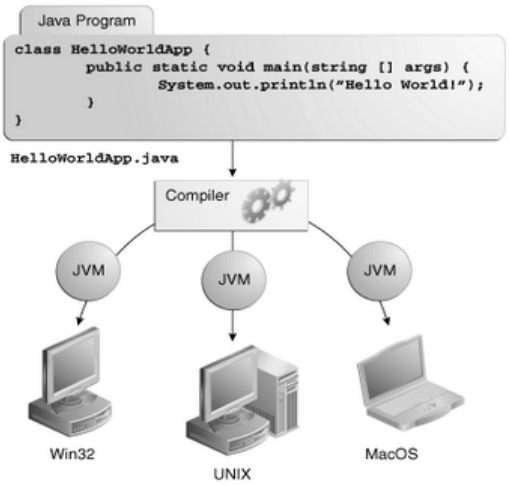
A JAVA TECHNOLÓGIA LÉNYEGE

Többlépcsős fordítás



A JAVA TECHNOLÓGIA LÉNYEGE

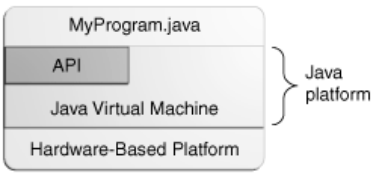
Platformfüggetlenség



JAVA PLATFORM

Két komponense:

- Java Virtual Machine (JVM)
- Java Application Programming Interface (API)



Kicsit lassúbb, mint a natív kód futtatása.

KITÉRŐ: JAVA vs C++ vs C#

http://www.developer.com/java/article.php/3856906/Java-vs-C-The-Performance-Showdown.htm

http://www.25hoursaday.com/CsharpVsJava.html

http://www.harding.edu/fmccown/java_csharp_comparison.html

http://slashdot.org/topic/cloud/java-vs-c-which-performs-better-in-the-real-world/

http://shootout.alioth.debian.org/

http://stackoverflow.com/questions/1049004/java-vs-c-are-there-any-studies-that-compare-their-execution-speed

Google

A JAVA TECHNOLÓGIA LÉNYEGE

Néhány tutoriál:

http://docs.oracle.com/javase/tutorial/

http://www.oracle.com/technetwork/java/langenv-140151.html

http://www.java2s.com/Tutorial/Java/CatalogJava.htm

+ Google és Java fórumok.

JAVA ALAPOK (ISMÉTLÉS)

Ismétlés??!! ⟶ ugyanaz, mint C-ben

• **Típusok:**
int, float, char, double, boolean, stb. ← *egyszerű*
String típus (fontos a nagy „S”) ← *referencia*

Egyszerű típus: azonosítójával közvetlenül hivatkozunk a változó memóriahelyére. Ezt a helyet a rendszer a deklaráló utasítás végrehajtásakor foglalja le.

Referencia típus: A referencia típusú változók objektumokra mutatnak. Egy referencia típusú változó azonosítójával közvetve hivatkozunk az objektum memóriahelyére. (Maga a hivatkozás rejtve marad.) Deklaráláskor csak a referencia részére foglalunk tárterületet, maga az objektum a példányosítás során jön létre.

JAVA ALAPOK (ISMÉTLÉS)

• **Kifejezések:**
Aritmetikai kifejezések (+,-,*,/,%)
Inkrementálás, dekrementálás (prefix és postfix alakok)
Összehasonlító operátorok (==, <, >, != … stb.)
Logikai operátotok (ÉS: && VAGY: || NEM: !)

• **Programszervező utasítások:**
Szelekciók (if…else, switch…case)
Iterációk (elől tesztelő ciklus (while);
hátul tesztelő (do…while);
növekményes (for))
Egyebek (break, continue)

JAVA ALAPOK (ISMÉTLÉS)

• **Metódusok:**
Általános alak:

visszatérési_típus *metódusnév* *(paraméterlista)* {
// törzs
}

• **Tömbök:**
Igazi referencia típusok – indexelés 0-tól.

Deklarálás: **tipus tomb[ ] = new tipus[méret];**

Hivatkozás: **elem=tomb[index];**

tomb.length; (a deklarált méretet jelenti)

JAVA ALAPOK (ISMÉTLÉS)

• **Osztályok:**
Általános alak:

[módosító] class *osztálynév* *(paraméterlista)* {
// törzs
}

• **Konstruktor:**
public *osztálynév* *(paraméterlista)* {
// törzs
}

• **Példányosítás:**
Tipus valtozo = new Tipuskonstruktor *(paraméterek)*;

PÉLDA – KIÍRATÁS

```
System.out.printf("a= %5s \n" , a);
System.out.printf("b= %5d \n" , b);
System.out.printf("c= %5.2f \n" , c);
System.out.printf("d= %5.4f\n" , Math.PI);
System.out.printf("d= %5.2f" , d);
```

```
/*A printf olyan szintaktika szerint máködik, mint c-ben:
        * printf("formátumstring",változólista);
        *
        *Pl. a formátumstringben szereplő %5.2f formátumvezérlő hatása:
        *
        *         5 <- minimális mezőszélesség
        *         2 <- pontosság (tizedesjegyek száma)
        *         f <- lebegőpontos szám
        *
        *         s <- string
        *         d <- egész szám
*/
```

FONTOS

MEGJEGYZÉS:

Minden programnak lehet több **jó** megoldása is!!!

Egy programot sohase tanuljon **be**,
de tanuljon **meg** programozni ! ☺