# 1. Bevezetés

## A Java kialakulása és felépítése

A Java programozási nyelvet a 90-es évek elején kezdték fejleszteni James Gosling vezetésével a Sun Microsystems cégnél, amelyet 2009-ben felvásárolt az Oracle.

A legelterjedtebb programozási nyelv, a Java programok több milliárd eszközön futnak.

A Java nyelvi szabályai a C és C++ nyelvekből származtathatók. Alkotóinak célja az volt, hogy **széleskörűen alkalmazható** és **hordozható** programokat lehessen fejleszteni a segítségével. Ennek kulcsa a Java platform, amely tartalmazza a Java Virtuális Gépet (Java Virtual Machine – JVM).

A programozási nyelvek egy része **fordítással** állítja elő azt a programot, amelyet azután a felhasználók használhatnak. A fordítás a programban felsorolt utasítások gépi kódra történő átalakítását jelenti, amelyet a számítógép végre tud hajtani. A legtöbb ilyen nyelv a fordítás során közvetlenül gépi kódot állít elő.

A programozási nyelvek másik csoportja nem fordítja le előre a programot, hanem futás közben soronként értelmezi és hajtja végre azt. Ezeket nevezzük **interpretált** nyelveknek.

A Java kombinálja a két módszert. A fordítás során a Java fordító egyfajta köztes kódot, úgynevezett **bájtkódot** állít elő, amely a gépi kódhoz áll közelebb. A **Java Virtuális Gép** egy olyan programcsomag, mely egy adott eszközre telepítve képes a Java fordító által előállított közteskódot, a bájtkódot értelmezni, futtatni.

Addig nem sok értelme van ennek az egésznek, amíg csak egyetlen eszközben gondolkodunk. A Java azonban rengeteg eszközben megtalálható, nyomtatókban, médialejátszókban, telefonokban stb. Ezekre az eszközökre ugyanúgy megírták az adott hardverhez köthető Java Virtuális Gépet. Ez a nagy előny, mivel ugyanaz a programkód ettől kezdve különböző hardvereszközökön is futtatható az eredeti kód módosítása nélkül is. A kész programok rendszerint adott hardverhez kötöttek, a JVM viszont gépfüggetlenné teszi a megírt Java forráskódokat, sőt, még a lefordított bájtkód is **hordozható**.

## Telepítés

Ahhoz, hogy Java programjainkat **megírhassuk**, szükséges, hogy a számítógépen rendelkezésre álljon a Java Fejlesztői Környezet (Java Development Kit) valamelyik változata. Mi a többféle változat közül a Java Standard Editiont (Java SE) fogjuk használni.

Ezen kívül szükségünk van egy fejlesztői felületre. Ebből is többféle van. Mi ezek közül a NetBeans felületet fogjuk használni.

A NetBeans-t és a Java SE fejlesztő csomagot (JDK) egyben innen lehet letölteni:

<http://www.oracle.com/technetwork/articles/javase/jdk-netbeans-jsp-142931.html>

A telepítés a letöltött .exe fájl futtatásával történik.

A Java programok **futtatásához** nem szükséges a teljes JDK, elég annak egy része, a Java Runtime Environment (JRE) is. Azokra a gépekre, amelyeken fejlesztés nem folyik, csak futtatni kell a kész Java-ban írt programokat, elég ezt feltelepíteni. A letöltés erről az oldalról történhet:

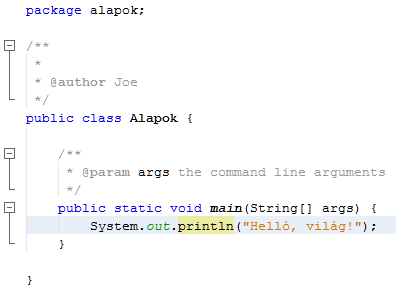
<http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html>

Utána természetesen le kell futtatni a letöltött programot.

## Első programunk

A Java programnyelv objektumorientált, ami azt jelenti, hogy maga a program is egy osztály, amelyet a Java Virtuális Gép futtat.

Első Java programunk legyen ez:



Nézzük meg, milyen részekből áll ez a program!

A /\*\* és \*/ közötti részek az egyes programrészek **dokumentációját** tartalmazzák. Ha nem szeretnénk látni őket, összecsukhatók a sor elején lévő mínusz jelekre kattintva. A // kezdetű részek **megjegyzések**.

A Java-ban az osztályokat **csomagokba** szokták szervezni. Ez főleg sok osztályból álló programoknál fontos, most elég, ha a NetBeans által létrehozott alapok csomagot használjuk. A package alapok; sor azt jelzi, hogy az Alapok osztály az alapok csomag része.

A következő sor is magyarázatra szorul: public class Alapok

A public kulcsszó azt jelenti, hogy az adott osztály publikus, vagyis bármely másik osztály láthatja, használhatja.

A class kulcsszó azt jelenti, hogy ez a kód egy osztályt ír le, aminek a neve Alapok (nagybetűvel!). Az osztály neve meg kell, hogy egyezzen a fájlnévvel, ahova a forráskódot mentjük, jelen esetben a forráskódunk neve a következő lesz: Alapok.java   
Ezt a NetBeans automatikusan létrehozza.

A sor végén van egy { jel. Ez egy blokk nyitását jelenti. A Java nyelvben a blokkokat (utasítások sorozatát) { } jelek közé tesszük. Mint a matematikában, úgy a programozásban is nagyon fontos, hogy a jelekhez, legyen az { [ vagy ( jel, mindig tartozzon nyitó és záró pár is. Ha ezek közül akár csak egy is hiányzik, akkor a programunk nem fordítható le, fordítási hibát eredményez.

A fordítási hiba egy programkód esetén azt jelenti, hogy nem felel meg az adott programnyelv “nyelvtani” szabályainak. Ezek jellemzően zárójel hibákat, pontosvessző hibákat jelentenek, amelyeknek mindnek a helyén kell lennie, hogy nyelvtani szempontból hibátlan kódot kapjunk. Ez persze nem azt jelenti, hogy a programunk tökéletes, csak a fordító fogadja el. Az már csak megszokás kérdése, hogy ezeket a blokk jeleket ki hogyan használja. A NetBeans-ben az itt látható formát fogjuk alkalmazni.

A következő sor a legbonyolultabb, de programozási szempontból most elég, ha csak így megtanuljuk és alkalmazzuk. Később majd megértjük, hogy mit jelent. Elég annyit tudni, hogy a main() metódus a programunk úgynevezett belépési pontja, vagyis itt kezdődik a programunk végrehajtása.

A program olvashatósága szempontjából fontos, hogy a blokk utasításait behúzással írjuk be. A blokk behúzását a tabulátor billentyű megnyomásával oldjuk meg. A tabulátor megnyomásakor 4 szóköz íródik be.

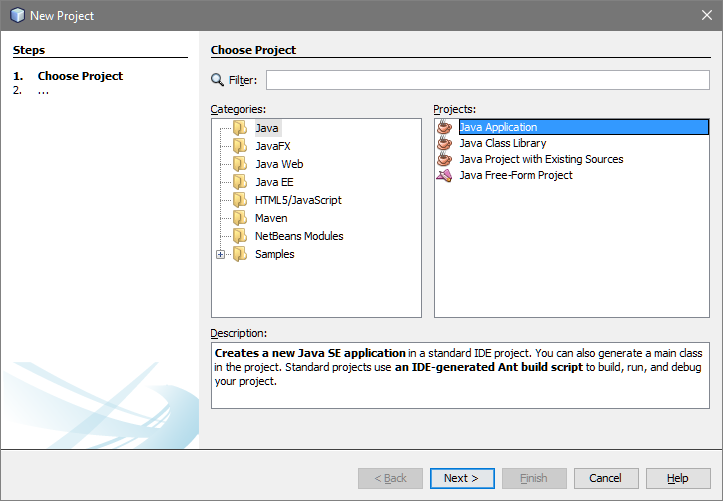
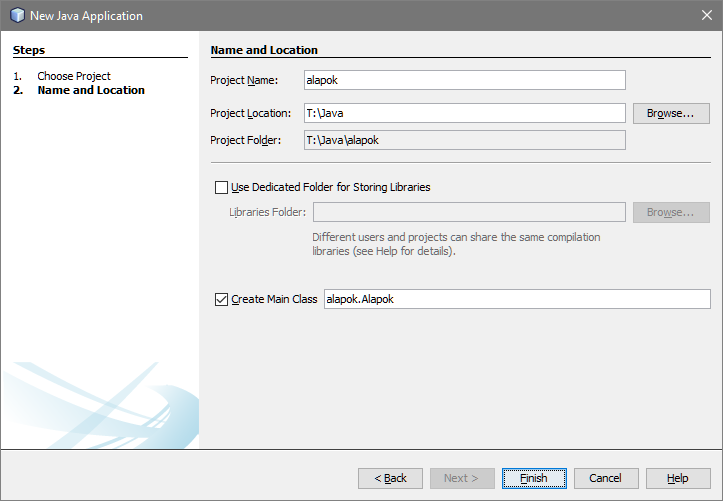
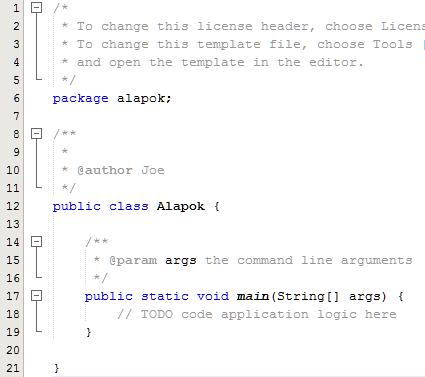
A következő sor már egy tényleges program utasítás. A System.out.println() egy metódus, amivel a konzolra lehet kiíratni egy szöveget. Ez egy több részből álló metódus hívás. A System osztályon belüli out nevű objektum println() metódusát hívja meg.

A következő fontos dolog a kiíratást végző utasítás végén lévő pontosvessző. A Java-ban minden egyes utasítást pontosvesszővel zárunk le.

Vége is van a programunknak. Illetve mégsem. Mi maradt ki? A két blokkot lezáró } jel. Az első (belső) blokk a main() metódus a benne lévő kiíratással, a második (külső) blokk pedig maga az Alapok osztály, a tényleges programunk.

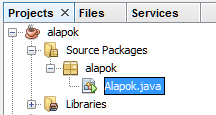
## A program elkészítése

Kezdj egy új projektet az alábbiak szerint:

1. File, New Project
2. Java, Java Application, Next  
   
3. Project Name: alapok, Finish  
     
   (mappa – csomag, fájl – osztály)
4. Nézd meg a kódot:  
   

Mivel a 3. lépésben be volt kapcsolva a Create Main Class kapcsoló, a NetBeans elkészítette a main() metódust.

Nézd meg az ablak bal felső részében, hogyan jelenik meg a csomag és a program a projektben:

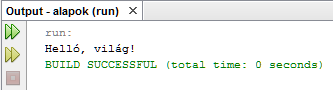


Ezután írd be a System.out.println("Helló világ!"); utasítást a // TODO… megjegyzés helyére (18. sor)! Figyeld meg közben az automatikus kiegészítést!

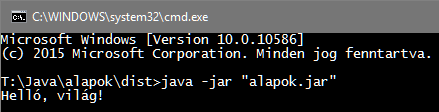
A begépelés közben figyelj arra, hogy a Java programozási nyelv különbséget tesz kis- és nagybetűk között. A "system" és a "System" szavak különbözőnek számítanak!

## Program futtatása

Futtasd az elkészült programot a Run Project gomb vagy az F6 megnyomásával! Utána nézd meg a kimenetet az alsó részen:



Ha csak fordítani szeretnéd a programot, kattints a Clean and Build Project gombra vagy nyomd meg a Shift+F11 billentyűket! Ez a fordítási hibák kiszűrésére is használható, és egy JAR fájlt is előállít. A JAR fájl önállóan (a NetBeans nélkül a Java futtatókörnyezetben) futtatható fájl. Mivel parancssoros programot készítettünk, a JAR fájl is parancssorból futtatható, ezzel a paranccsal:



Próbáld ki!

## Feladatok

1. Töröld a programból az első öt sor dokumentációt, majd a szerzőhöz (@author) írd be a saját neved!
2. Keresd meg, hogy hová mentette a Netbeans az Alapok.java fájlt!  
   Írd ide:

## Házi feladat

1. Telepítsd otthoni számítógépedre a NetBeans programot a JDK-val együtt!