I. Bevezetés

1. A Java nyelv története

A Java nyelvet 1991-ben a Sun MicroSystems egy csoportja fejlesztette ki. Eredetileg kommunikációs eszközök programozására tervezték, de sikere messze meghaladta az eredeti elképzeléseket. A készítők a C++ programozási nyelv leegyszerűsítésével és módosításával alkották meg az eredetileg Oak nevű programozási nyelvet. Később jöttek rá, hogy ilyen nevű programozási nyelv már létezett, ezért egy új nevet kellett kitalálniuk. A készítők a nyelv új nevén egy gőzölgő kávé mellett gondolkoztak. A kávé neve Java volt, ami a származási helyére utalt. Ekkor döntöttek úgy, hogy a programozási nyelvüknek a ***Java*** nevet adják (a Java gőzölgő kávé emblémája is innen ered).

1. ábra A Java "logója"

2. A Java nyelv jelentősége

A 90-es évek közepétől a Java nyelv meghódította az egész világot, és az Internet programozás nyelvévé vált. A Java szerzői - látván a Web dinamikus fejlődését - felismerték, hogy az általuk készített nyelv kiválóan alkalmas lenne a böngészők kibővítésére, hogy azok képesek legyenek alkalmazások futtatására is.

Ahhoz, hogy megismerhessük a Java működési elvét, először tisztáznunk kell néhány fogalmat. A ***gépi kód*** a számítógépek számára végrehajtható utasítások sorozata (a gépi kódot szokás még *natív kódnak* is nevezni). A gépi kód azonban az ember számára átláthatatlan ezért kifejlesztettek egy gépközeli (alacsony szintű) programozási nyelvet, az ***Assembly***-t, ahol a gépi kódú utasításoknak egy-egy *mnemonik* felel meg. Ezt az assembly programot az ún. *assembler* fordítja le gépi kóddá. Az assembly már kezelhető volt az ember számára, de még mindig nagyon távol állt a mi gondolkozásunktól. Igazi áttörést a magas szintű nyelvek jelentettek, amelyek már igazi segítséget jelentettek az embernek. A magas szintű nyelven írt programot a *fordító* (***compiler***) alakítja át gépi kóddá. Manapság magas szintű nyelvek egész sora áll a programozók rendelkezésére (C, C++, Pascal, Basic, Prolog..stb) és ezek közé tartozik a Java is.

A Java fordítókkal kapcsolatban meg kell állnunk egy kicsit, és körül kell járnunk a bájtkód fogalmát. Egyes fordítók (ilyen a Java compiler is) nem fordítják le teljesen gépi kódúvá a magas szintű programot, hanem egy közbülső ún. bájtkóddá alakítják azt. A bájtkód egy gépközeli kód, de a gépi kóddal ellentétben platformfüggetlen. A bájtkódot pedig sokkal könnyebb natív kóddá alakítani, mint a magas szintű programot, ezért a fordító illetve futtató rendszer sokkal egyszerűbb program lehet.

A Java fordító által előállított bájtkódot az adott számítógépen a JVM (Java Virtual Machine) futtatórendszer futtatja úgy, hogy azt utasításonként értelmezi és natív kóddá alakítja.

3. A Java nyelv jellemzői

A Java nyelv fejlesztői hivatalos kiadványban tették közzé a Java nyelv jellemzőit. Ez a kiadvány a Java nyelvet 11 szóval jellemzi:

- egyszerű
- objektumorientált
- elosztott
- robusztus
- biztonságos
- architektúrasemleges
- hordozható
- interpretált
- nagy teljesítményű
- többszálú
- dinamikus

A Javaban a C++-hoz képest a legfontosabb változások az alábbiak:

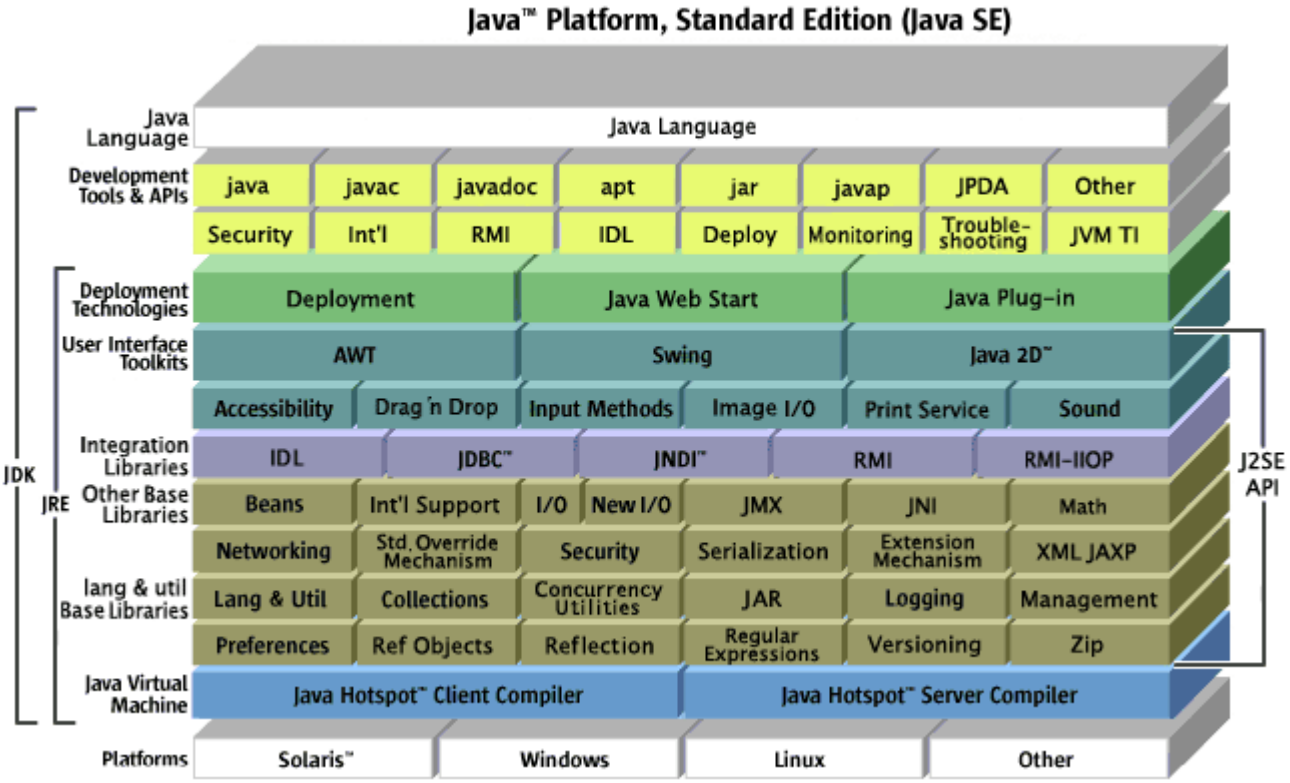
- eltűntek a mutatók és a mutatókkal kapcsolatos műveletek
- kiszedték a statikus változókat és függvényeket
- nincs többszörös öröklődés
- nincs makrózási lehetőség

4. Java telepítése

A Java szoftvereket 2 részre kell bontanunk.

a) A szoftverek egy része ahhoz szükséges, hogy lehetőségünk legyen a Java programok futtatására. Ezt a környezetet hívják **JRE**-nek (***Java Runtime Environment***) azaz Java futtatói környezetnek. Ez tartalmazza a JVM-et, ami a későbbiekben a programok futtatását végzi. Az aktuális verzió ingyenesen letölthető a www.java.com/getjava weboldalról.

b) Amennyiben Java programokat akarunk fejleszteni, úgy szükségünk lesz a **JDK**-ra (***Java Developement Kit***) , azaz a Java fejlesztői eszközre. Ez a JRE-n túl tartalmazza a fejlesztéshez szükséges alapvető eszközöket is (compiler, debugger, decompiler…stb).



2. ábra A JRE és a JDK kapcsolata

Fontos megjegyezni, hogy a JDK által biztosított eszközökkel csak „fapados” módszerrel tudnánk programokat fejleszteni. A kényelmes programfejlesztéshez szükségünk lesz ún. **IDE** (Integrated Development Environment) integrált fejlesztői környezetekre, amelyek rengeteg lehetőséggel teszik könnyebbé és gyorsabbá a programozók munkáját. Ezek a fejlesztői környezetek általában az alábbiakat tartalmazzák:

- ***szövegszerkesztő***, amely segíti a kód strukturálását
- ***fordító***
- ***futtató***
- ***szintaktikai kiemelés*** (syntax highlighting)
- ***nyomkövetés*** (debugger)
- ***stb…***

Néhány példa a Java fejlesztői környezetekre:

- JCreator (Xinox Software)
- NetBeans (Sun Microsystems)
- JBuilder (Borland)
- TextPad (Helios Software)
- JDeveloper (Oracle)

5. Java környezet kialakítása a gyakorlathoz

Ahogy azt már említettük, a JDK (újabban SDK-nak, azaz Software Development Kit-nek is nevezik) tartalmazza a JRE lehetőségeit, ezért a fejlesztéshez elegendő a JDK letöltése és annak telepítése. A gyakorlatok során kezdetben a Xinox cég IDE-jét a **JCreatort** fogjuk használni, majd a későbbiek során a modern grafikus fejlesztésre is lehetőséget biztosító **NetBeans** IDE-t fogjuk használni.
A laborgépek természetesen kialakítottuk a megfelelő környezetet, de a tárgy elsajátításához nélkülözhetetlen, hogy az otthoni gépekre is megfelelően telepítve legyenek ezek a szoftverek a gyakorláshoz.

Környezeti beállítások

- A JDK környezet megfelelő beállításához hozzátartozik még, hogy a **PATH**-ba beállítsuk a JDK könyvtárunk BIN alkönyvtárát is (pl.. C:\Program Files\Java\jdk1.5.0_07\bin
- Végül be kell állítanunk a **CLASSPATH** környezeti változót is, amely arra a könyvtárra kell hogy mutasson, ahol dolgozunk. Ha a CLASSPATH környezeti változóba egy „.” Kerül megadásra, akkor a futtató az aktuális könyvtárban fogja keresni az osztálydefiníciókat. (Régi JDK telepítése esetén a CLASSPATH környezeti változóhoz hozzá kell adnunk még a JDK\LIB könyvtárt is)

Ezzel elkészítettük a Java fejlesztői környezetet.

6. Első Java programunk elkészítése

A Java programokat két csoportba sorolhatjuk:

- **Application** (alkalmazás)
  Az alkalmazások önálló futtatásra képes programok, amelyeket a JRE-vel futtathatunk

- **Applet** („alkalmazáska”)
  Az appletek önmagukban nem futtathatók, csak egy másik alkalmazásba beágyazva. Ez a másik alkalmazás jellemzően egy Web böngésző vagy a fejlesztő környezet appletek tesztelésére szolgálóappletviewer programja.

Első programunk egy „mezei” alkalmazás lesz, amely a C nyelv óta tipikusan a Hello World program. A program Java kódja a következő:

```
/********************
 *    Java alapok    *
 *        1.         *
 ********************/

public class HelloWorld
{
    public static void main (String args[] )
    {
        System.out.println("Hello World!");
    }
}
```

3. ábra Első Java programunk

A programot gépeljük be egy szövegszerkesztőbe (pl.: NotePad) és mentsük el `HelloWorld.java` néven.

FONTOS!!!

A forrásfájl nevének pontosan meg kell egyeznie az osztály nevével (jelen esetben HelloWorld)

7. Java programok fordítása és futtatása

Java programok fordítását végezhetjük a JDK beépítet fordítójával, melynek szintaktikája a következő:

`javac programneve.java`

pl.: `javac HelloWorld.java`

Sikeres fordítás esetén létrejön annyi *class* kiterjesztésű állomány, ahány osztályt tartalmazott a programunk. Jelen esetben létrejön a `HelloWorld.class` fájl. Ez utóbbi már alkalmas arra, hogy a Virtuális gép lefordítsa és futtassa azt. Vegyük észre, hogy a *class* fájl a bevezetőben említett bájtkód. Mivel a fordítás csak erre a közbülső lépcsőre készült el, ezért ezek a fájlok még nem futtathatóak a C nyelvben megismert módon közvetlenül. Szükség van a futtató indítására, ami a JVM segítségével elkészíti a natív kódot és elindítja az alkalmazást.

A Java program futtatására a JDK-ban a következőképpen történik:

`java programneve`

pl.: `java HelloWorld`

A továbbiakban nem várjuk el, hogy a JDK beépített eszközeit használják programjaik fejlesztéséhez, de egyszer érdemes kipróbálni, mert sose tudhatjuk, hogy mikor kerülünk olyan helyzetbe, hogy egy idegen gépen csak ezek az eszközök fognak rendelkezésünkre állni. (Volt már rá példa, hogy frissen végzett informatikust álláskeresésekor ilyen feladat elé állítottak)

Javaslom a mintapélda kipróbálását JCreator és NetBeans környezetekben is. A példában szereplő nyelvspecifikus elemek részletes tárgyalása a II. gyakorlat anyaga lesz.

8. Rövidítések

- **JVM** : Java Virtual Machine
- **J2SE**: Java2 Platform Standard Edition
- **J2EE**: Java2 Platform Enterprise Edition
- **J2ME**: Java2 Platform Micro Edition
- **JRE**: Java Runtime Environment
- **JDK**: Java Developement Kit
- **SDK**: Software Developement Kit
- **IDE**: Integrated Development Environment
- **API**: Application Program Interface