Eclipse IDE

Óbudai Egyetem, Java Standard Edition
Mérnök Informatikus szak, BSc
Labor 2

Bedők Dávid
2016.09.29.
v0.7

Fejlesztői környezet

A fejlesztői környezet legtöbbször egy **szöveges állományok** szerkesztését támogató alkalmazás, mely minimálisan **"project" fá**val rendelkezik, vagyis az összetartozó szöveges állományokat rendszerezni tudja.

Akár a **Notepad++** is nevezhető fejlesztői környezetnek, de egy **jEdit**, egy **Visual Studio**, egy **Eclipse**, egy **IntelliJ IDEA** vagy egy **NetBeans** lényegesen kényelmesebb munkát tud biztosítani. Népszerű az https://atom.io/ is manapság.

Alapvető előnyök

Szintaktikai ellenőrzés: már a begépelés pillanatában ellenőrzi, hogy a begépelt utasítás az adott kontextusban értelmezhető-e, avagy sem

Fordítás, futtatás, tesztelés integrálása: nem kell külön alkalmazás a fejlesztés különböző fázisaihoz

Minden kéznél van: **ezernyi apró kiegészítő** (plugin) segítségével a mindennapi munkát meggyorsítják, hogy az embernek ***"csak"*** a gondolkodással kelljen foglalkoznia!

Figyelmeztetés

Egy IDE csak akkor segíti a munkát, ha a nekünk szükséges funkcióit **megtanuljuk**. Idővel olyan érzésünk lehet, hogy a fejlesztői környezet* kitalálja mit szeretnénk (nyílván csak a motorikus dolgokat), és alkalmazkodik szokásiankhoz.

*: nem minden fejlesztői környezet egyforma, vannak jobbak, rosszabbak, és itt is van "vallási" kérdés (ki mit szeret). Az is fontos lehet, hogy nem minden nyelv rendelkezik elég szigorú szintaktikai szabályokkal ahhoz, hogy jól lehessen automatizálni (a Javanak szerencsére ez az egyik nagy előnye!).

Design environment

A fejlesztői környezetnek van egy sokkal tudományosabb formája, ezt **design környezet**nek nevezhetjük. Lényeges különbség, hogy a design environment nem kizárólag egy **alkalmazás**t tartalmaz, hanem akár **több**et is (fejlesztői környezet, alkalmazás szerver, teszt rendszer, szimulátorok, scriptek, stb.), valamint az egyes alkalmazás komponensek **beállítás**át is magában foglalja, foglalhatja!

Eclipse

A laboron választott fejlesztő eszközünk az **Eclipse IDE**, mely egy széles körben elterjedt rendszer. Nem hibátlan, de folyamatosan javuló, élő projekt. A laborfoglalkozáson csak alapszolgáltatásaival fogunk dolgozni.

Nem csupán Java-hoz használható, létezik variációja C++-ra, PHP-re is.

http://www.eclipse.org/

Verzió: Eclipse Neon

Eclipse IDE for Java EE Developers

Perspektívák és Nézetek I

Az Eclipse fejlesztői felülete ún. **Perspektívák**ra épül. Számos beépített Perspektíva létezik alapértelmezés szerint, de mi magunk is létrehozhatunk újakat. A Perspektíva valójában nem más, mint az adott munkafelületen látható ablakok és azok elhelyezkedésének összessége (ennél kicsit több is). Más gyermek ablakokra van általában szükségünk akkor, mikor fejlesztünk (és azt sem mindegy hogy mit..), és másra mikor pl. adatbázis lekérdezéseket hajtunk végre, vagy debugolunk. A gyermekablakokat az Eclipse-ben Viewnak, **Nézet**nek hívjuk.

Perspektívák és Nézetek II

Perspektívák megnyitása:

Window | Open Perspective

Aktuális perspektíván belül View-k megnyitása:

Window | Show View

Számunkra két perspektíva lehet fontos: a **Java** és a **Debug**. A Debug Perspektívára automatikusan átvált(hat) az Eclipse, ha debugolás során megszakítás történik.

Hello World project elkészítése

Új projekt létrehozása:

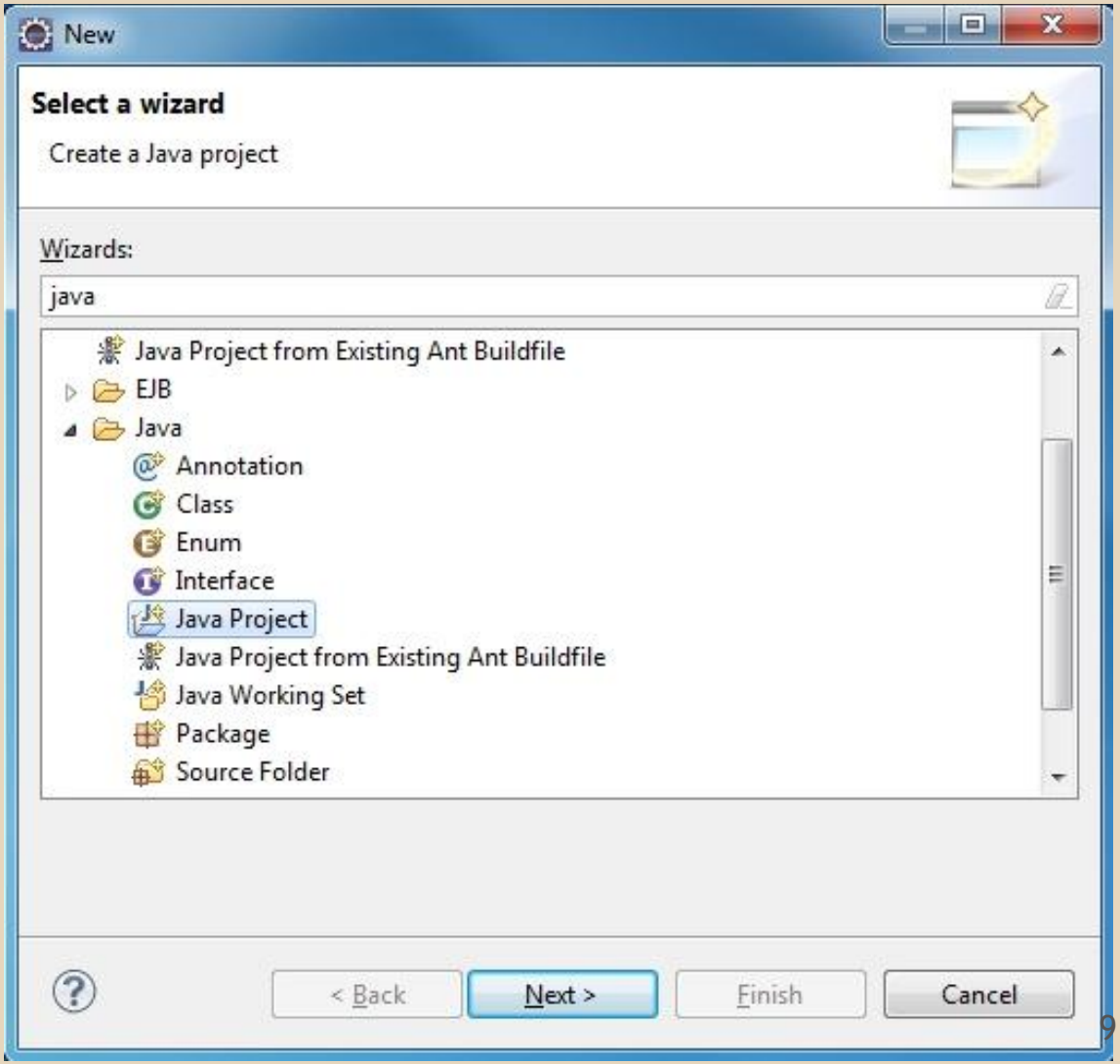
File | New | Other... (Ctrl + N)

[Next >]

Project name: *HelloWorld*

JRE: *Use default JRE* (checked)

[FINISH]

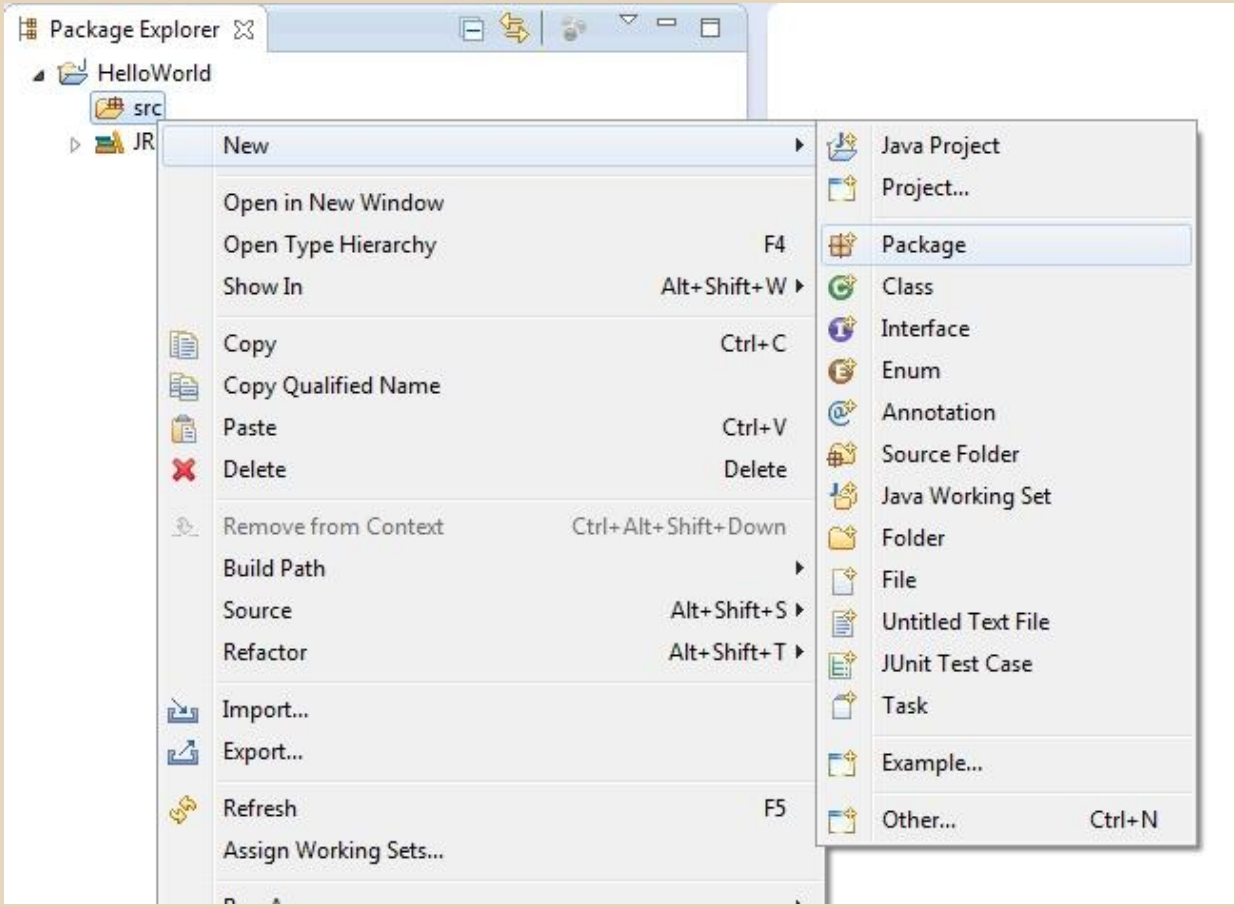


Package(ek) létrehozása I

A Java forrás állományokat csomagokba szervezzük. A csomagok "névterek", melyek segítik az alkalmazásunk integrálását más rendszerekkel, illetve fontos szervezési szerepük is van.

Minden forrásállományt **kötelező** csomagokba szervezni (ha nem tesszük, kapunk egy generált *default* csomagot, de ezt kerüljük el).

Package(ek) létrehozása II



A project **src** könyvtárának helyi menüjéből (jobb klikk) válasszuk ki:

New | Package

vagy

New | Other... | Java | Package

Name:
hu.uniobuda.hello

Osztály létrehozása I

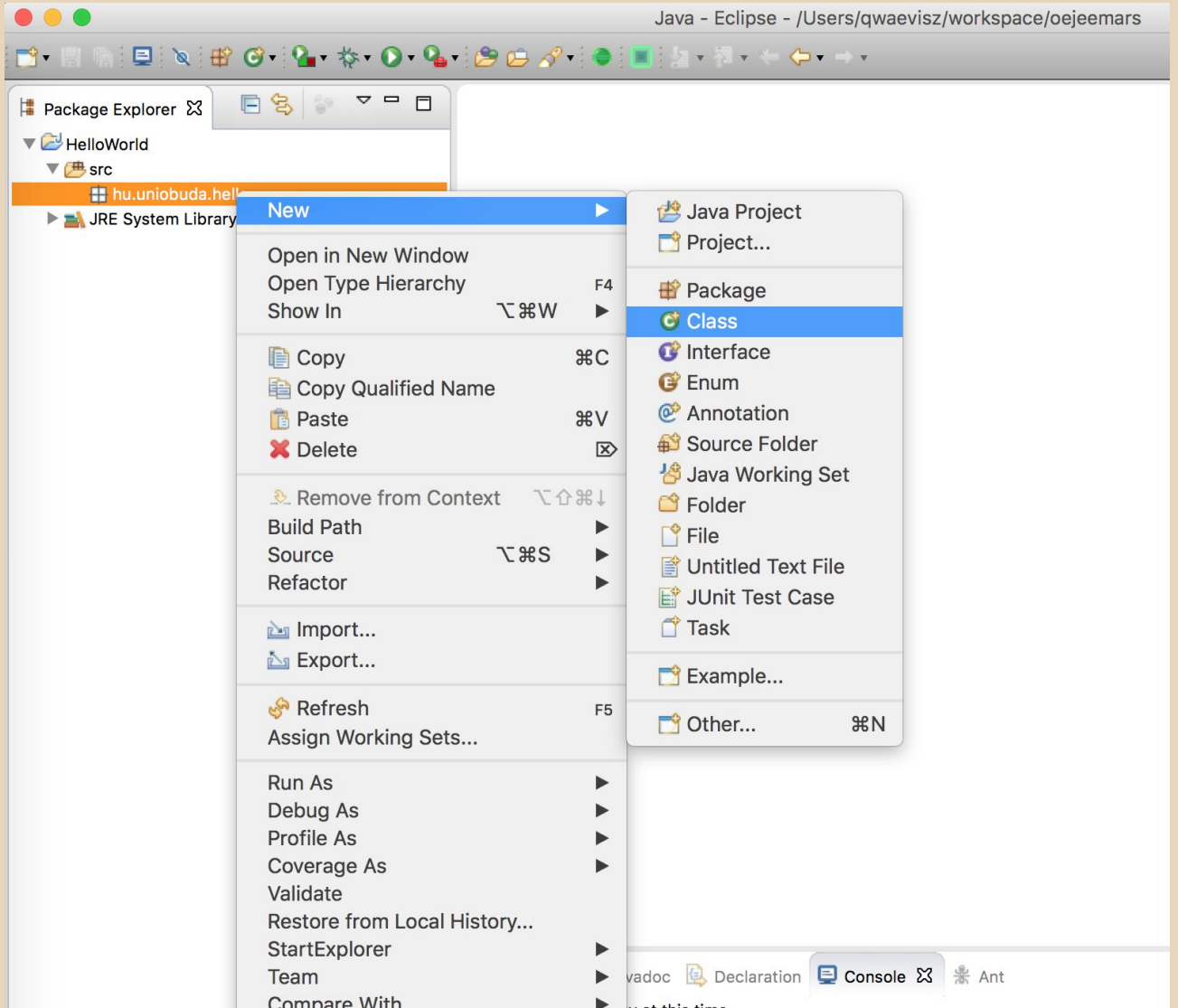
A most létrehozott **hu.uniobuda.hello** package helyi menüjén (jobb klikk), majd:

New | Class

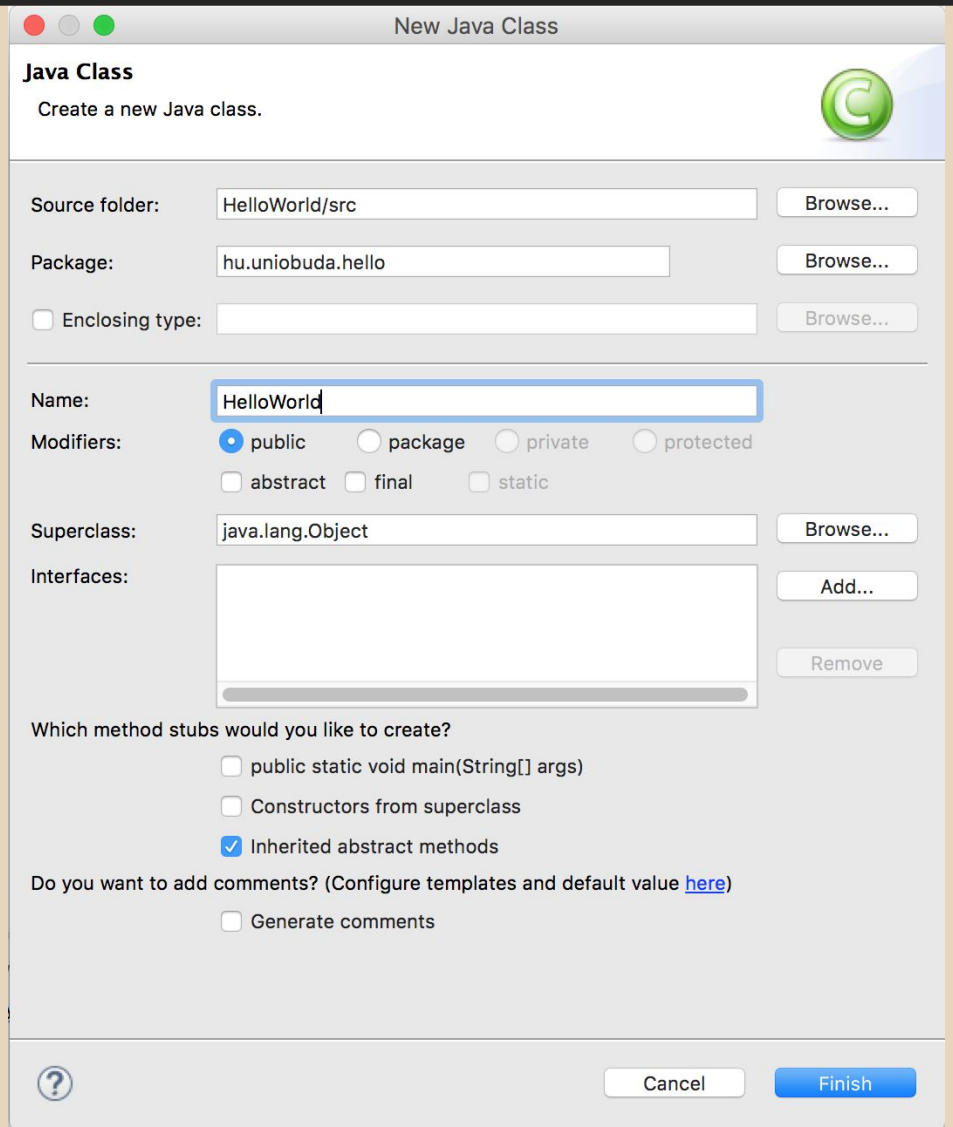
vagy

New | Other... | Java | Class

Name: **HelloWorld**



Osztály létrehozása II



Which method stubs would you like to create?

`public static void main (String[] args)`

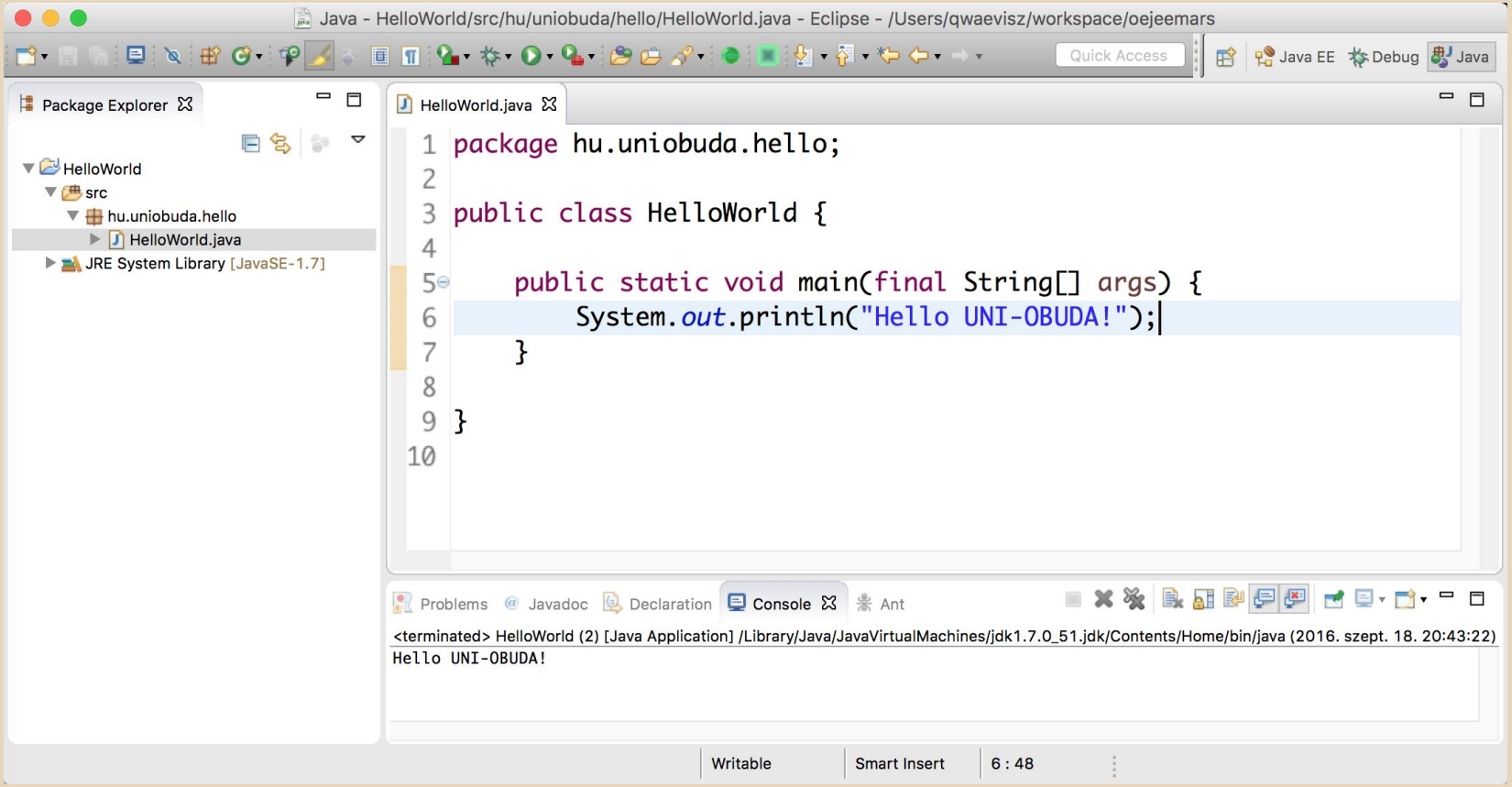
(checked)

[FINISH]

Üdvözöljük az egyetemet!

HelloWorld.java

```
System.out.println("Hello UNI-OBUDA!");
```



Eclipse: Configuration - Code Style

- Window (win) / Eclipse (mac) | Preferences (type: formatter)
  - Java | Code Style | Formatter
    - New…
      - Profile name: **uni-obuda-java-formatter**
      - Initialize: Eclipse [build-in]
      - Indentation | Indent | Statement within ‘switch’ body
      - Line Wrapping | General | Maximum line width: 160
      - Line Wrapping | Enum declaration
        - Policy: Wrap all elements, every element on a new line
        - Constants policy: Wrap all elements, every element on a new line + Force split
      - Comments | Line width | Maximum: 120

Eclipse: Configuration

- Window (win) / Eclipse (mac) | Preferences (type: save actions)
  - Java | Editor | Save Actions
    - Perform the selected actions on save
      - Format source code (all lines)
      - Organize imports
      - Additional actions - Configure
        - Code Organaizing: Remove trailing whitespaces
        - Code Style: Use blocks in if/while/for/do statements
        - Member Accesses: Use ‘this’ qualifier for field accesses: Always
        - Member Accesses: Use ‘this’ qualifier for method accesses: Always
        - Unnecessary Code: Remove unused imports

Gyorsbillentyűk

Számos billentyűkombináció létezik, mindet megtanulni értelmetlen, de párat kezdésnek érdemes memorizálni.

Az alapértelmezett billentyűzet kiosztást megváltoztatni csak nagyon indokolt esetben ajánlott (más géphez kerül a fejlesztő, gyakorlatilag nem tud dolgozni).

A gyorsbillenytűk alapértelmezett beállítása egy dolog, fontos figyelembe venni a számítógép billentyűzetének kiosztását is. Sok fejlesztő pontosan ezért kizárólag angol kiosztáson tud hatékonyan dolgozni, mások viszont a magyar ékezetes gyors gépelés szükségessége miatt megtanultak együtt élni a magyar billentyűzetkiosztással.

Magyar billentyűzet + Eclipse

- Window (win) / Eclipse (mac) | Preferences (type: key)
  - General | Keys

WinOS:

- { → **ALTGR + b** → `Skip All Breakpoints` (CTRL + ALT + b jelöléssel szerepel, mert itt az angol fizikai kiosztásra épít) Törölni kell az összerendelést (vagy kitalálni neki valamilyen egyedit)

MacOS:

- `Run on Server`: Törölni kell az összerendelést néhány Eclipse verzióban.

Billentyű kombinációk - Általános

CTRL + Space

Adott kurzor pozícióban hivatkozható változók, referenciák listája a valószínűsíthető szükséglet szerint rendezve! A már beírt karakterek szűrik a listát!

CTRL + 1 (win) / **cmd + 1** (mac)

Adott kurzor pozíció alapján javasolt akciók listája, a valószínűsíthető szükséglet szerint rendezve!

Billentyű kombinációk - File

CTRL + s (win) / **cmd + s** (mac) [save]

mentés (ha *Save Action* be van állítva, akkor a mentés során számos post action is lefut!)

CTRL + n (win) / **cmd + n** (mac) [new]

új elem hozzáadása (project, package és class is ilyen volt az eddigi példában, de ezernyi más is létrehozható).

F11 (win) / **SHIFT + cmd + F11** (mac)

fordítás és futtatás

Billentyű kombinációk - Megnyitás

CTRL + SHIFT + t (win) / **SHIFT + cmd + t** (mac)

[type] osztály (típus) megnyitása (pl. egy `ShortBundleIdGenerator` osztály megkereshető bármely rész karakterlánc begépelésével (bundl.., gen..), de ha az SBIG-t, vagy ennek részeit gépeljük be, akkor is sikerrel járunk (BIG..), de ezeket még kombinálhatjuk is (SBGen..)

CTRL + SHIFT + r (win) / **SHIFT + cmd + r** (mac)

[resuorce] erőforrás megnyitása

Billentyű kombinációk - Refactor

ALT + SHIFT + r (win) / **ALT + cmd + r** (mac)

[rename] refactor - átnevezés

CTRL + m

[maximize / minimize] aktuális view tejes képernyőre váltása

Billentyű kombinációk - Formázás

SHIFT + CTRL + f (win) / **SHIFT + cmd + f** (mac)

[format] kód újraformázása az aktuális Formatter szabályok szerint (Save action megfelelő beállítása mellett e billentyűkombináció használata felesleges)

CTRL + d (win) /**cmd + d** (mac)

[delete] aktuális sor törlése

kijelölés (vagy aktuális sor) + ALT + fel/le

a kijelölés (vagy az aktuális sor) mozgatása (kontextust felismeri)

Debug keys

Add Breakpoints

- **F5** - Step Into
  - Belelép a soron következő hívásba (metódusba)
- **F6** - Step Over
  - Átlép a következő utasításra az adott blokban (leggyakoribb)
- **F7** - Step Return
  - A következő return-hoz/hívási lánc forrásához ugrik
- **F8** - Resume
  - Továbbengedi a futást (a következő töréspontig)

Kód template-ek

Window (win) / Eclipse (mac) | Preferences | Java | Editor | Templates

main - a belépési pontként szolgáló `public static void main(String[] args)` metódus legenerálása

switch - switch statement generálás

sysout - `System.out.println();` generálás

while, **do**, **else**, **elseif**, **for**, **foreach**, **if**, **ifelse**, stb.

Használat: **template + CTRL + SPACE (+ ENTER)**

Betűméret átállítása

Ez egy olyan opció az Eclipse-ben, mely rendkívüli módon el van rejtve:

Window (win) / Eclipse (mac) | Preferences | General | Appearance | **Colors and Fonts**

Listából kiválasztani: Java | **Java Editor Text Font**

[EDIT] gombra kattintani

Felugró ablakban **Size**-ot átállítani

Fontos! Különbség van magyar és angol kiosztás esetén, mivel a +, - és = jelek máshol vannak a két kiosztáson.

Eclipse Neon-tól betűméret növelése ill. csökkentése:

HU: **CTRL + SHIFT + 3** (fizikai angol billentyűzeten ez #) BR: CTRL + PLUS (win)

HU: nincs kiosztva* BR: cmd + = (fizikai magyar billentyűzete ez **ó**) (mac)

* mac magyar billentyűzet kiosztás esetén a “Zoom In” [cmd + =] helyett a [cmd + .] összerendelést szoktam beállítani

HU: **CTRL + -** (fizikai angol billentyűzeten ez **?**) BR: CTRL + MINUS (win)

HU: **cmd + -** BR: cmd + = (fizikai magyar billentyűzete ez **ü**) (mac)

Plugins

Az Eclipse-hez számos előre telepített és telepíthető Plugin létezik (Help | Eclipse Marketplace). Gyakorlatilag minden fejlesztői feladatra találunk plugin-t, és így az Eclipse felületén is elvégezhetjük a kívánt műveleteket (pl. **verziókezelés** (pl. git), **adatbázis kezelés**, **build rendszerek** kezelése (pl. gradle, maven), **futtatási környezetek** kezelése (servers/runtime environments, pl. apache tomcat, jboss).

Ezen a laborfoglalkozáson külön telepítendő plugin-t nem fogunk használni, és a legtöbb fent említettet sem (pl. nem fogjuk a tomcat-et Eclipse-ből elindítani, arra az IDE-ből deployolni). Sok esetben ezek kényelmes dolgok, de a tematika része az érintett komponensek architektúrális alapjainak megismerése, és nem része egy okos eszköz lehetőségeinek kihasználása (egy mérnök egy eszközt akkor használ, ha már tisztában van annak hatásával).