Mobil programozási alapismeretek

Mobil készülékek

- Egyre több ember zsebében és táskájában a legkülönfélébb mobileszközök megtalálhatóak
- Mobiltelefonok, PDA-k, PalmTopok és intelligens multimédiás eszközök (mit pl. iPod-ok)
- A készülékek legjelentősebb része különböző márkájú okostelefonokból áll
- Ezek szoftver- és hardverkapacitása dinamikusan fejlődik
- Képességeikben és lehetőségeikben az asztali számítógépekhez hasonlítanak
- Felhasználói részről az izgalmas programok és játékok iránti kereslet
- Szoftverfejlesztői részről pedig az intelligens készülékek programozása utáni vágy

Mobil készülékek programozása

- Komoly problémákkal kell szembesülni, mivel megszámlálhatatlanul sokféle mobil készülék létezik
- A hardver- és szoftverkörnyezet változatossága és egymással való inkompatibilitása olyan mértékű, hogy az nagymértékben megnehezíti az alkalmazás-fejlesztést
- Probléma a számítógépekénél lényegesen gyengébb hardveres képességek
- Ezt hivatott kompenzálni azonban az általános célú operációs rendszer

Az okostelefonok

- Fejlett számítógépszerű mobiltelefonok
- Ezek az eszközök uralják a hordozhatókészülék-ipart
- Szabványosított interfészeket és platformot nyújt az alkalmazásfejlesztők számára
- Hardveres problémák
  - körültekintőnek kell lennie hatékonyság terén
  - alkalmazás-fejlesztőnek fokozottan figyelnie kell az áramtakarékosságra
- Szoftveres problémák
  - számos mobiltelefon-gyártó az idők során különböző technológiákat fejlesztett ki eszközeik szoftveres támogatására
  - azonos szoftverek eltérő verziónak együttes programozása is nehéz

Fejlődő hardver

- Az okostelefonok követhetetlenül gyorsan fejlődő és átalakuló hardverével nehéz lépést tartani
- Folyamatosan jelennek meg az újabb, erősebb készülékek
- A szoftverrel nem lehet ezzel lépést tartani, tehát a szoftvernek nem szabad olyan gyorsan változnia, azért hogy egy programozási nyelv használható legyen az újabb készülékeken is és követhetőek/tanulhatóak legyenek az új funkciók/változtatások a szoftver-fejlesztőknek
- Külön megfontolásokat érdemel a kompatibilitás

Szoftveres megoldások

- Platformfüggetlen programozás
  - Több platformon, architektúrán (eszközön) is futtatható. Olyan nyelvi elemeket és megoldásokat (könyvtárakat) használ, amik a rendszerek nagy részében (vagy a nyelv specifikációjában) rendelkezésre állnak, így számos platformon változtatás nélkül lefordítható és futtatható. (pl.: Java)
  - Ennek segítségével ugyanaz a lefordított programkód Windowson és pl. Linuxon is (nagyjából) ugyanúgy fut
- Platformfüggő programozás
  - Egyes szabványosított nyelvek (C/C++) platform-/oprendszerfüggő megoldásokat tartalmaznak, ezért máshogy néz ki egy program forráskódja ugyanazon a nyelven Windowsra, mint pl. Linuxra (sőt még ugyanarra a platformra is léteznek különböző fordítók)

Platformfüggetlen programozás

- A platformfüggetlen programozási nyelvek általános célú nyelvek
- Ugyanaz a lefordított programkód több platformon is fut (cross-platform)
- Ezt egy úgynevezett virtuális rendszer/program (virtual machine) segítségével érik el, a virtuális rendszer minden platformon egyedi, de a programozó forráskódja minden platformon ugyanaz (tehát egyszer kell megírni) és mindenhol (kb) ugyanúgy működik
- Ez egy zseniális megoldás, de így a forráskódban platformfüggő optimalizálások nem használhatóak, az erőforrás-kezelés nem hatékony
- Ezért sokkal hatékonyabb, ha a platformfüggetlen nyelveknél is vannak platformfüggő megoldások pl.: Dalvik (Android Java)

Függvénykönyvtárak (megoldások)

- A különféle szabványosított programozási nyelvek általános szintaxisa rögzített (kulcsszavak, szabályok)
- Amikben definiálhatunk saját függvényeket, eljárásokat, amiket meghívunk a programkódunk különböző részein
- De a különféle problémák, platformok konkrét megoldásait segítik az előre elkészített és megírt függvénycsomagok és könyvtárak
- Ezeknek a programozási interfészek (API) használatával történik a platformfüggő programozás
- A különböző rendszerekre való programozáshoz fejlesztői csomagokat adnak ki (SDK) amit a számítógépünk oprendszerén (pl Windows, ahol fejlesztünk) használunk
- A gépünkön fejlesztői környezet (IDE) segítségével történik a forráskódunk szerkesztése, amiben rengeteg segítséget kapunk a kényelmesebb programozáshoz

Dalvik: Android Java

- A Java egy általános célú platformfüggetlen Objektum Orientál nyelv (OOP) (lásd később)
- A Java virtuális rendszere a Java Virtual Machine (JVM)
- A Java nyelv a szintaxisát főleg a C és a C++ nyelvektől örökölte, viszont sokkal egyszerűbb objektummodellel rendelkezik, mint a C++
- A Dalvik virtuális gép a Google által létrehozott Android operációs rendszer virtuális számítógépe (VM). Ezen a készülékeken futó programok legtöbbször Java nyelven íródnak és első lépésben Java bájtkódra fordulnak
- A fejlesztés most már a hivatalosan kiadott Android Studio segítségével történik

Windows Phone

- Windows Phone programozás Visual Studioban történik c++/c# nyelven
- A felület definiálása XAML (Extensible Application Markup Language) segítségével történik
- A fejlesztői eszközök (SDK) a Windows Phone SDK-ban megtalálhatóak
- Visual Studioban a Blend segítségével vizuális szerkesztő segítségével hozhatjuk létre az UI-t
- https://msdn.microsoft.com/en-us/library/windows/apps/ff402529(v=vs.105).aspx

IOS

- Apple IOS programozás Xcode IDE segítségével történik Mac gépen
- Swift, Objective-C programozási nyelven, ez a szabványos C kibővítve Objektum Orientált (OO) elemekkel
- Az Objective-C nyelv mellett az Apple által létrehozott Swift (2014.jún) nyelv is fontos szerepet kap, azért jött létre, hogy a C nyelv nehézségeit könnyítse és hogy a programozók kevesebbet hibázzanak
- https://developer.apple.com/library/ios/referencelibrary/GettingStarted/DevelopiOSAppsSwift/

Mobilprogramozás

- Hasonlóságok is vannak a különböző platformok fejlesztési lépéseiben
- A programozás fejlesztői környezetben (IDE) történik, amiben különböző elemeket (gombokat, szövegdobozokat, képeket) behúzhatunk egy virtuális telefonképernyőre
- Általános leírónyelv (xml,xaml) segítségével az alkalmazásunk kinézetének a leírását adhatjuk meg (HTML-hez hasonló)
- Majd ezeknek a leírási tageknek az attribútumában megadott függvénynévhez írunk egy függvénydefiníciót, ami akkor fut le amikor a megadott esemény (attribútum név) bekövetkezik

Eseményvezérelt alkalmazás

- Rengetegféle eseményt definiálhatunk, amik különböző feltételek mellett bekövetkeznek (pl gomb nyomásra)
- Minden eseményhez tartozik egy függvény a forráskódban, ami akkor fut le, ha az esemény bekövetkezett
- Grafikus felületű GUI alkalmazásoknál gyakori ez a megközelítés
- Itt a programkód értelmezése a szokásostól (sorról sorra) eltérő, ugyanis a felhasználó interakciójától és nem a programozási definíciók sorrendjétől függ a programnak a futása

XML

- Az XML (Extensible Markup Language) általános leírónyelv különböző adattípusok leírására képes
- Elsődleges célja strukturált szöveg és információ megosztása
- Fontos különbség a HTML nyelvhez képest, hogy itt a tagek nevét és alkalmazhatóságát az adott szoftverfejlesztő alkalmazás szabja meg
- Rendkívül egyszerűen érthető és feldolgozható

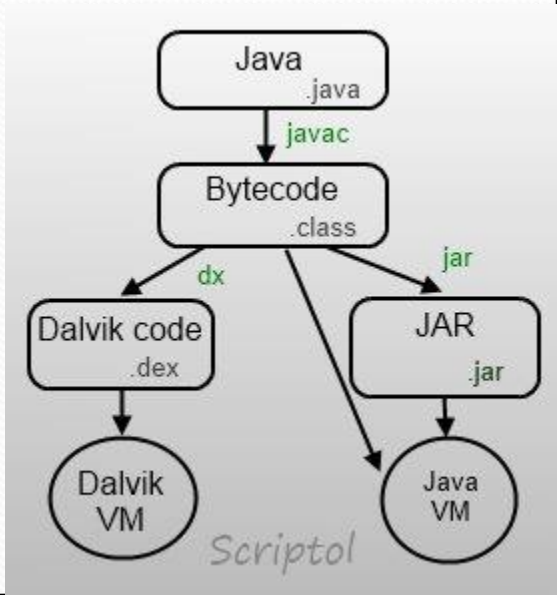
```
<?xml version="1.0" encoding="UTF-8"?>
<xypair>
    <xaxis axistype="independent">
        <property>Wavelength</property>
        <value>254.0</value>
        <unit>nm</unit>
    </xaxis>
    <yaxis axistype="dependent">
        <property>Absorbance</property>
        <value>0.1234</value>
        <unit>none</unit>
    </yaxis>
</xypair>
```

Android Studio

- Ezzel a fejlesztői környezettel (IDE) programozhatunk Android okostelefonunkra
- Ajánlásokkal segíti a kódkiegészítést
- Van beépített kódanalízise
- Kódolás közben észreveszi a hibákat
- Appokat fejleszthetünk Android telefonra, tabletra, Android Wearre, Android TV-re, Android Autora és Google Glassra
- Virtuális Android kijelzőn akár a számítógépünkön is megnézhetjük az Appunkat (Android programunkat)
- http://developer.android.com/sdk/index.html

Forráskód fordítás programmá

- Megírjuk az Android Appunknak a forráskódját Android Studioban (IDE) az Android függvénykönyvtárak segítségével (API) Java programozási nyelven
- Ebből kapunk egy(pár) .java kiterjesztésű forrásfájlt
- Ezt a fejlesztői környezetünk először Java bájkóddá fordítja => .class
- Majd, ha Androidra fordítunk, akkor létrejön a Dalvik code (.dex), a programunk, amit a telefonunkon levő Dalvik VM képes értelmezni
- Vagy, ha PC-re fordítunk, akkor létrejön a .jar futtatható fájl, amit a Java VM értelmez (futtat)



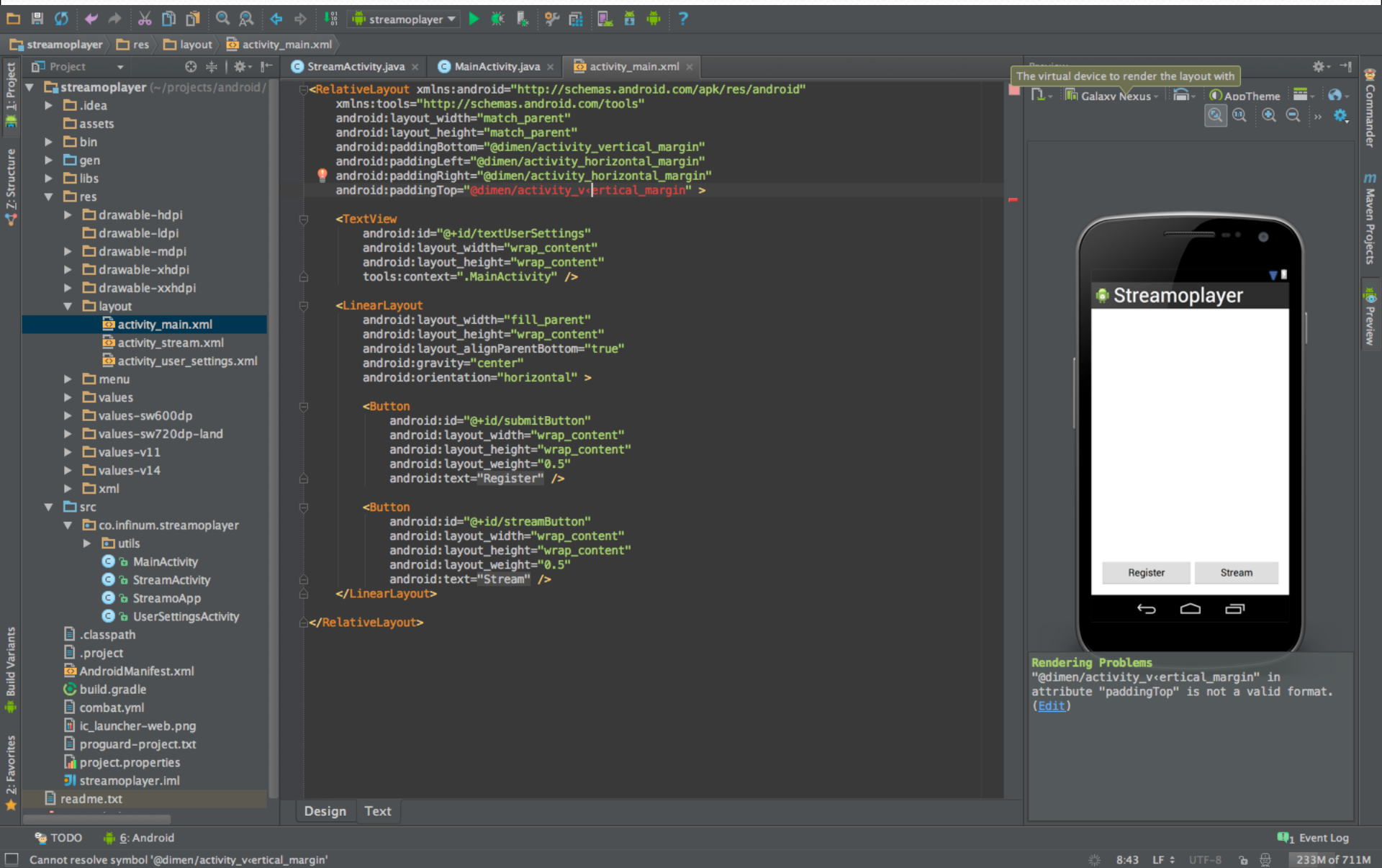
Kompatibilitási megfontolások

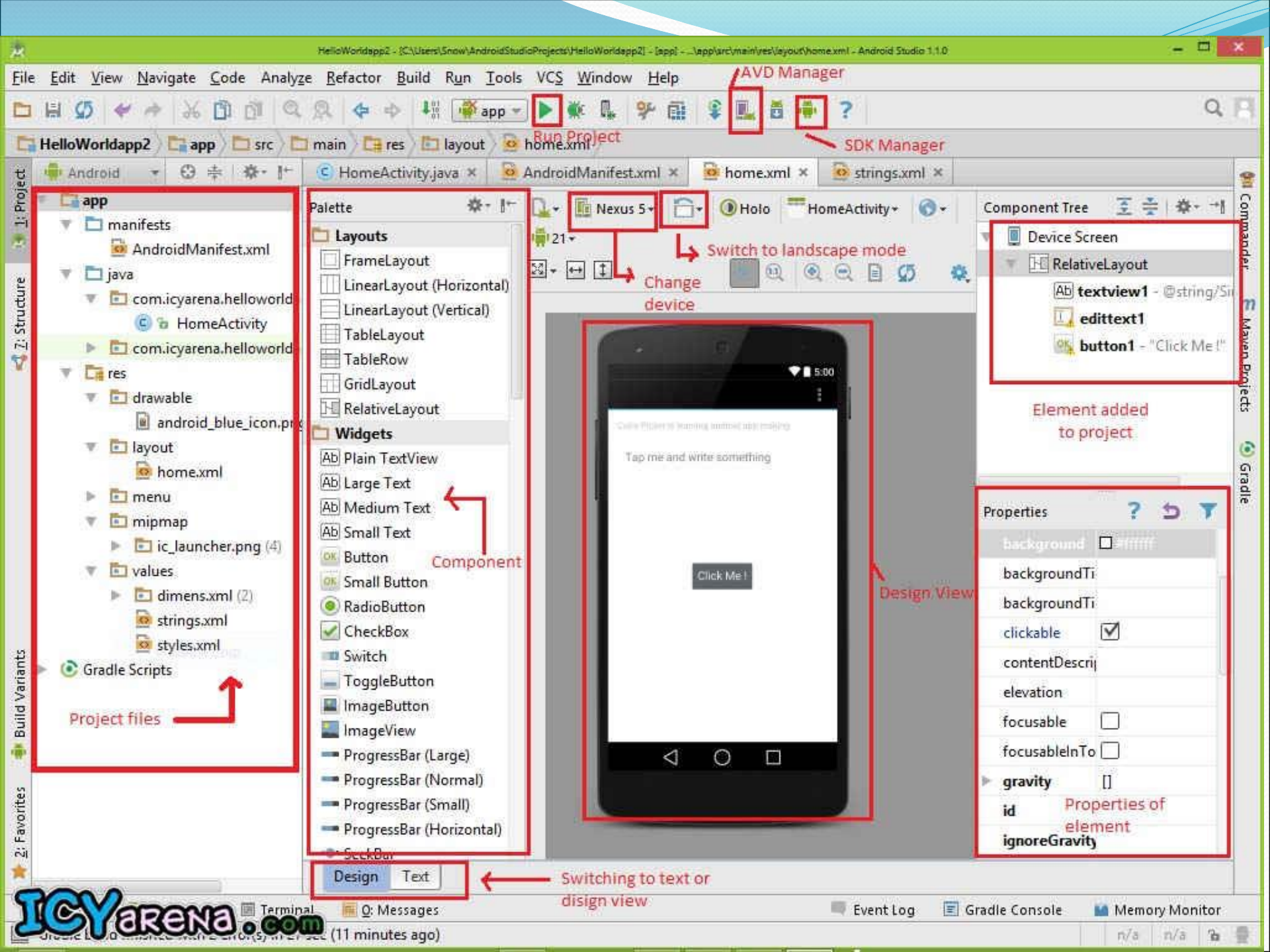
- A telefonunk oprendszere gyorsan fejlődik, sokat frissítik, minden egyes nagy frissítés után újdonságokat építenek bele
- Ezekkel együtt párhuzamosan fejlődik a programozási nyelv is úgy, hogy a régi programkódok ugyanúgy használhatóak újabb rendszeren is, de a legújabb kódok, amik valami különleges verzióhoz köthető definíciót tartalmaznak, azok nem fognak működni régebbi rendszereken
- Meg kell gondolni, hogy az újabb lehetőségekkel akarunk-e élni, vagy inkább minél több verziójú eszközre próbálunk programozni
- Lehetőség van több verzióra akár külön módosított kódot lefordítani

Integrált fejlesztői környezet (IDE)

- IDE segítségével komplex projekteket kezelhetünk, amiben rengeteg segítséget kapunk
- Segítségével láthatunk akár egy virtuális telefonképernyőt is amibe behúzhatunk elemeket (UI), majd leprogramozhatjuk a viselkedésüket
- Az elemek tulajdonságainak a beállításaiban sok értéket egyszerűen átírhatunk
- A projektünkhöz tipikusan sok forrásfájl, multimédia (resource) tartozik, amiket könnyen kezelhetünk
- Kódolás közben javaslatokat tesz és hibákat is jelez

Android Studio (IDE)




AVD Manager
Run Project
SDK Manager
Switch to landscape mode
Change device
Component
Design View
Element added to project
Properties of element
Project files
Switching to text or disign view
Design
Text
Tap me and write something
Click Me!

Android Studio (IDE)

- http://developer.android.com/training/index.html