



MODELLALAPÚ SZOFTVERFEJLESZTÉS

I. ELŐADÁS

BEVEZETÉS

DR MEZEI GERGELY

A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

III. fejezet Hogyan?



A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

III. fejezet Hogyan?



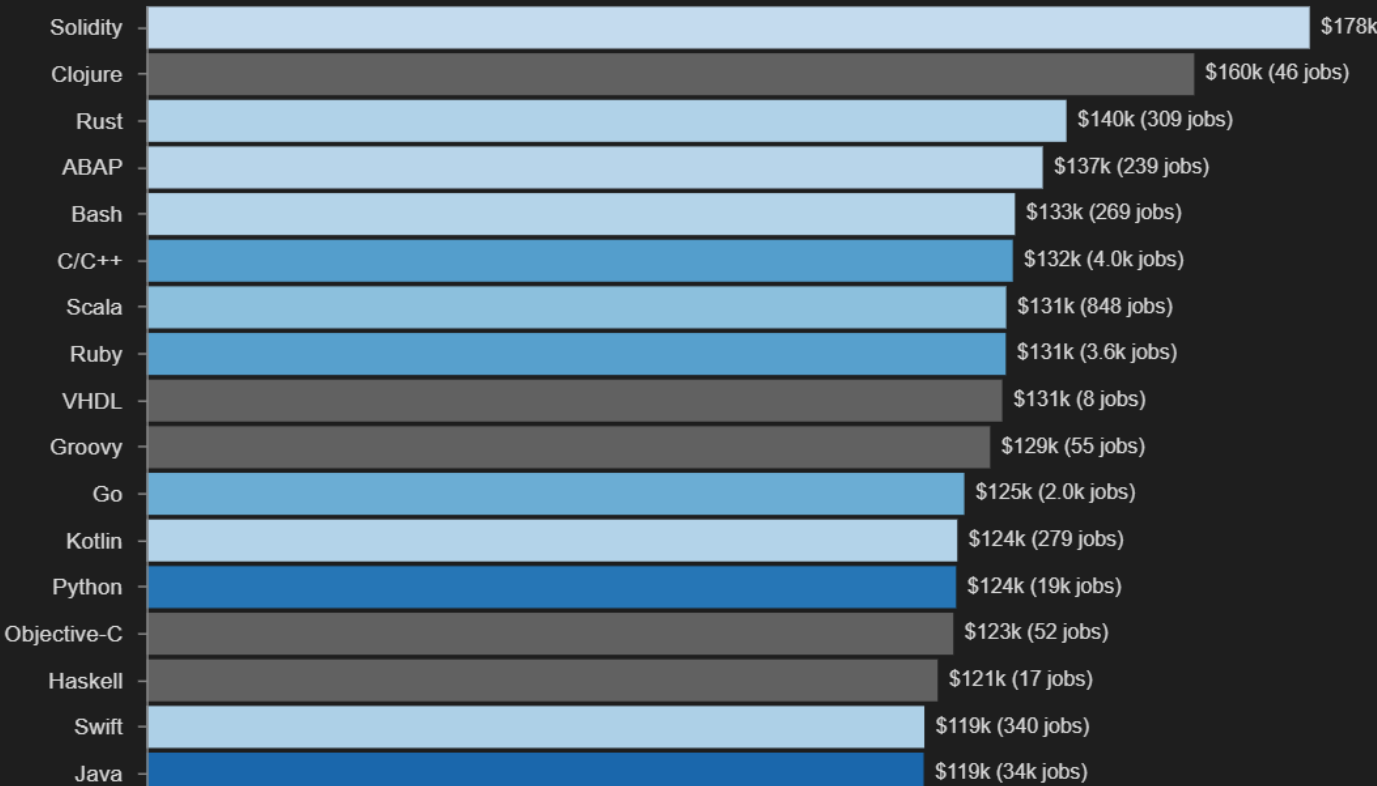
SZOFTVERFEJLESZTÉS – HOGYAN KEZDŐDÖTT?

- Tom Kilburn – Manchester Small-Scale Experimental Machine – 1948
 - Első programkód
 - Program = lyukkártya
- Fortran – 1957
 - Első magas szintű programozási nyelv + fordítóprogram
- C nyelv – 1972
- Objektorientált nyelvek: Simula (1967), Smalltalk (1970)
- C++ - 1984
- Java – 1995
 - Java Virtual Machine – hordozhatóbb kód
- C# - 2000
 - Intermediate Language – nyelvek közti átjárhatóság
- Python, Swift, Go, Kotlin, ...

SZOFTVERFEJLESZTÉS - MA

Top Paid Programming Languages in 2024 (US)

From 01-Sep-2023 to 01-Dec-2024



Feb 2025

Feb 2024

Change

Programming Language

1



Python



C++



Java

LANGUAGE

Rank	Programming Language	Salary in the USA	Salary in India	Salary in Europe
1	Python	\$130,000–\$180,000	₹15,00,000–₹25,00,000	€70,000–€120,000
2	Go (Golang)	\$140,000–\$200,000	₹18,00,000–₹30,00,000	€80,000–€130,000
3	Rust	\$150,000–\$210,000	₹20,00,000–₹35,00,000	€90,000–€140,000
4	Scala	\$120,000–\$175,000	₹18,00,000–₹30,00,000	€75,000–€120,000
5	Kotlin	\$120,000–\$160,000	₹12,00,000–₹25,00,000	€60,000–€100,000
6	Java	\$100,000–\$160,000	₹12,00,000–₹22,00,000	€60,000–€110,000
7	Swift	\$120,000–\$160,000	₹15,00,000–₹25,00,000	€70,000–€110,000
8	TypeScript	\$110,000–\$160,000	₹10,00,000–	€60,000–€100,000

HOVA TART A SZOFTVERFEJLESZTÉS?

- Mik az elvárások?
 - Növekvő alkalmazás méret
 - Csökkenő idő
 - Kevesebb hiba
 - Magasabb minőség
- Megoldás?

A MEGOLDÁS: AI?

- Mottó: “ Természetes nyelven megfogalmazott specifikációból a mesterséges intelligencia működőképes alkalmazást gyárt ”
- Nehézségek
 - A specifikáció
 - A működőképes alkalmazás
 - Biztonsági garanciák, minősegbiztosítás



MEGOLDÁS: LOW CODE – NO CODE?

- Mottó: “Termékfejlesztés programozók nélkül”
 - Grafikus felület, “összekattintgató” alkalmazáslogika
 - Gyors fejlesztés
 - Limitált felhasználási terület
 - Üzleti fogalmak, ellenőrzések, folyamatok

LOW CODE – NO CODE

Low code

- Alkalmazásfejlesztés minimális kódolással
- Gyors betanulás és fejlesztés
- Általában grafikus szerkesztő
- Részben limitált kifejezőerő
- Fejlesztőknek és üzletembereknek

No code

- Alkalmazásfejlesztés kódolás nélkül
- Szinte nulla betanulás, azonnali fejlesztés
- Általában grafikus szerkesztő
- Limitált kifejezőerő
- Üzletembereknek

LOW CODE – NO CODE

- A siker titka
 - Beszéljünk a probléma nyelvén!
 - Koncentráljuk a tényleges feladatra!
 - Hagyjuk el a repetitív részeket!
 - Legyen tömör, átlátható!
 - Ne kelljen tudni hozzá programozni, csak ha muszáj!

MERRE HALADUNK?

- Ne írd meg azt...
- ... amit mások már megírtak!
 - Mindenre van (fél)kész megoldás (library, komponens)
 - Telepíts és konfigurálj kódolás helyett!
- ... amit egy olcsóbb munkaerő is megírhat!
 - Code monkey-k, vagy méginkább: automatizálás és kódgenerálás
 - Miért nem írja meg a megrendelő?

A SZOFTVERFEJLESZTÉS – MA

- Igény: gyorsan, jól és sokat
- Megoldás:
 - Absztrakciós szintet kell növelni
 - *Assembly → C → C++ → Java/C# → ...*
 - “Konfiguráció” programozás helyett
 - *Mindenre van félkész megoldás*
 - Generálni, amit csak lehet
 - *C++ template, generált constructor + destructor, property*
 - Legyen tesztelhető!
 - *Nem csak line coverage, formális tesztek*

.... pontosan ezt adja a **Modellezés**

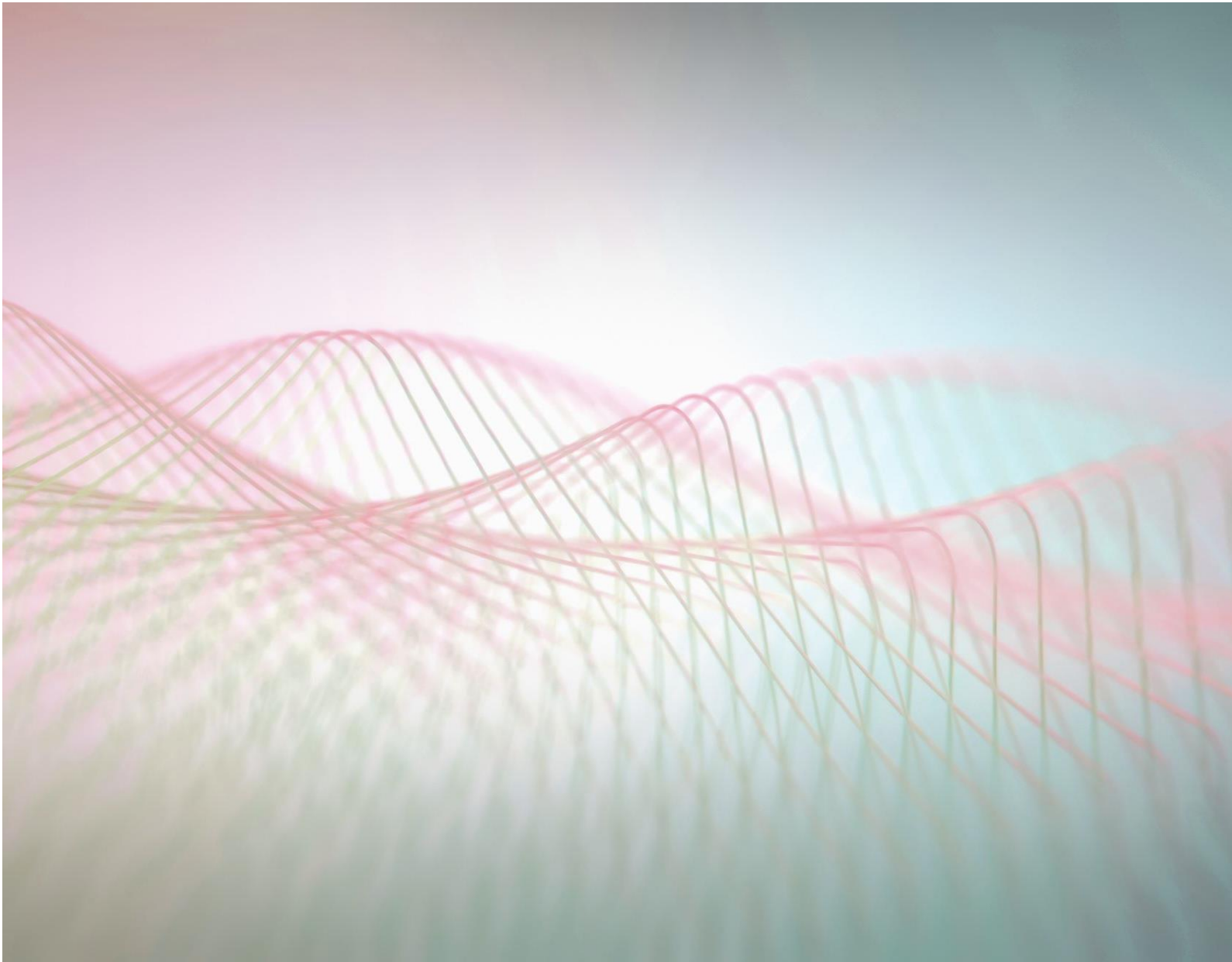
A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

III. fejezet Hogyan?

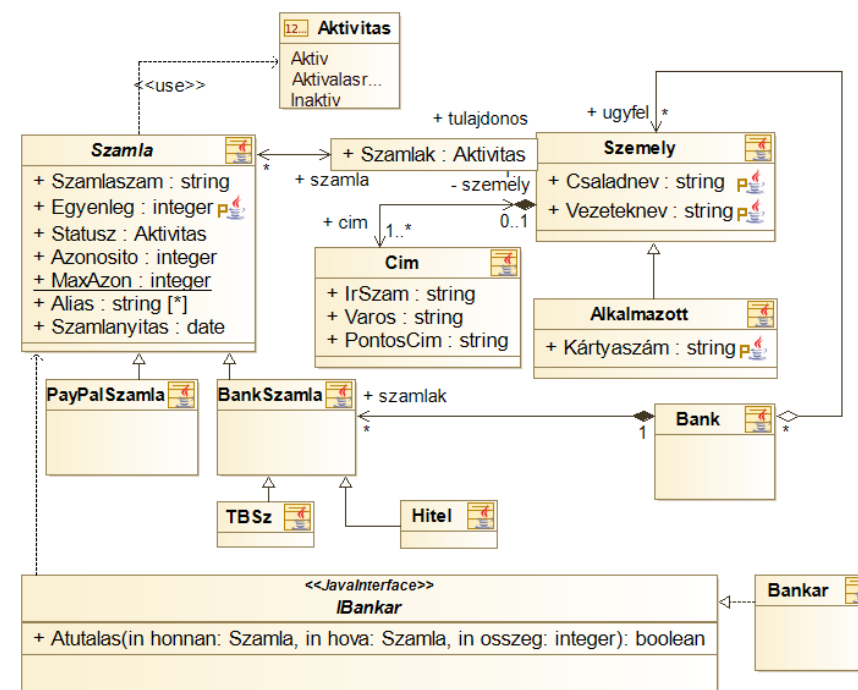




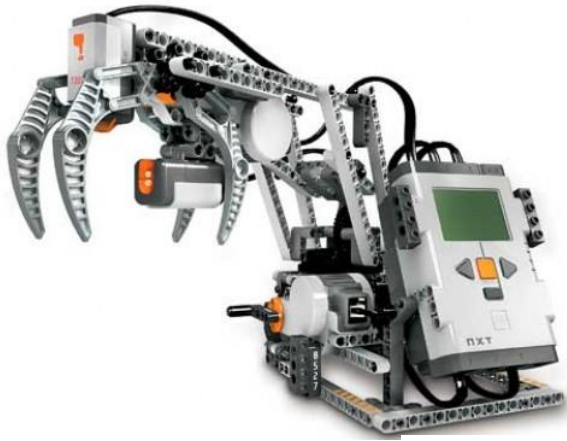
UNIVERZÁLIS, VAGY EGYEDI?

UML

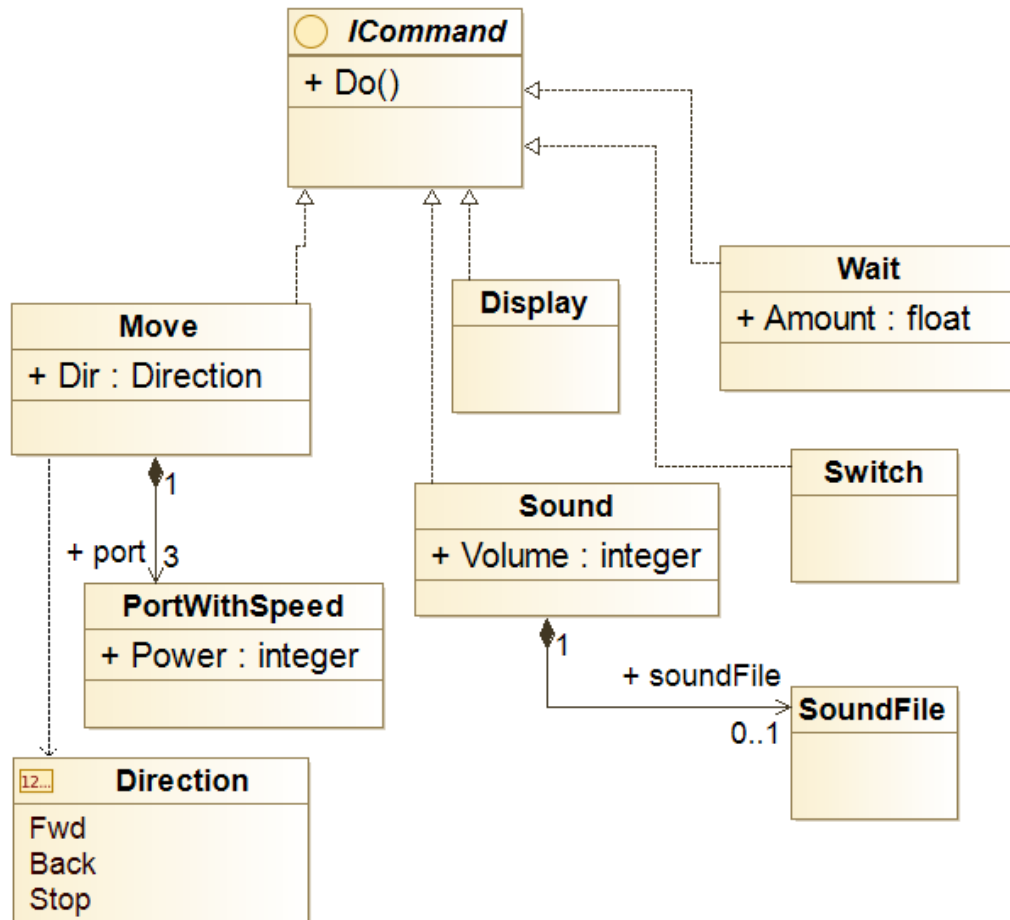
- Modellezés – ahogy már ismeritek: UML
 - Szoftvermérnökök közös modellező nyelve
 - Magas absztrakciós szint
 - Szabványos jelölésrendszer
 - Gazdag eszköztámogatás
 - Limitált testreszabhatóság
 - Részleges kódgenerálás



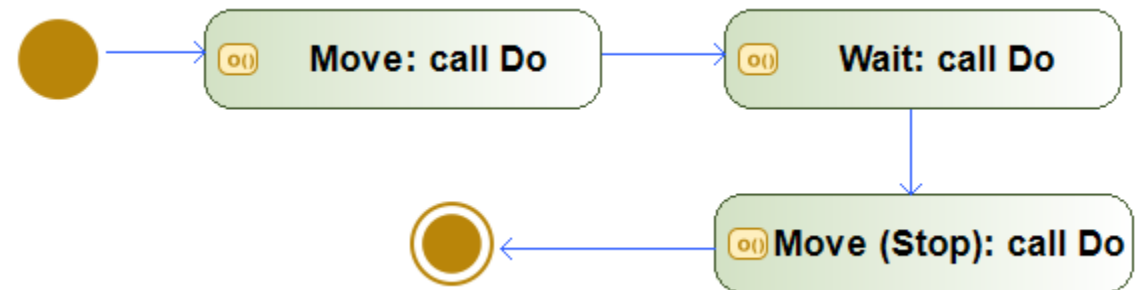
EGY PÉLDA: LEGO MINDSTORMS



MINDSTORMS – UML



- Menjen előre Imp-ig, majd álljon meg!



AZ UML-EN TÚL

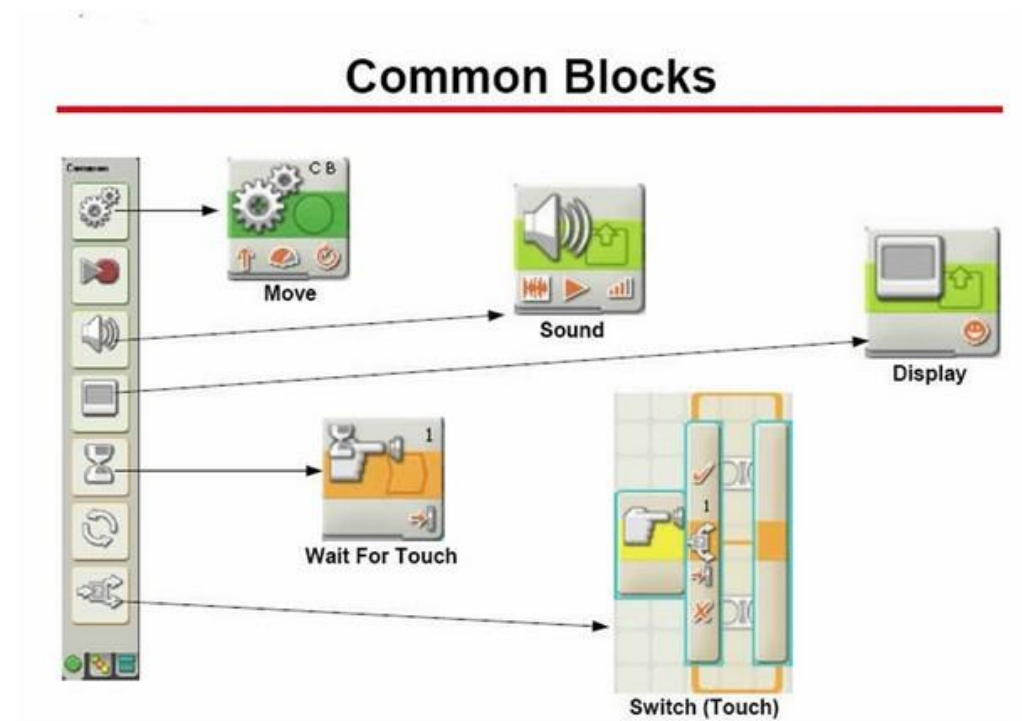
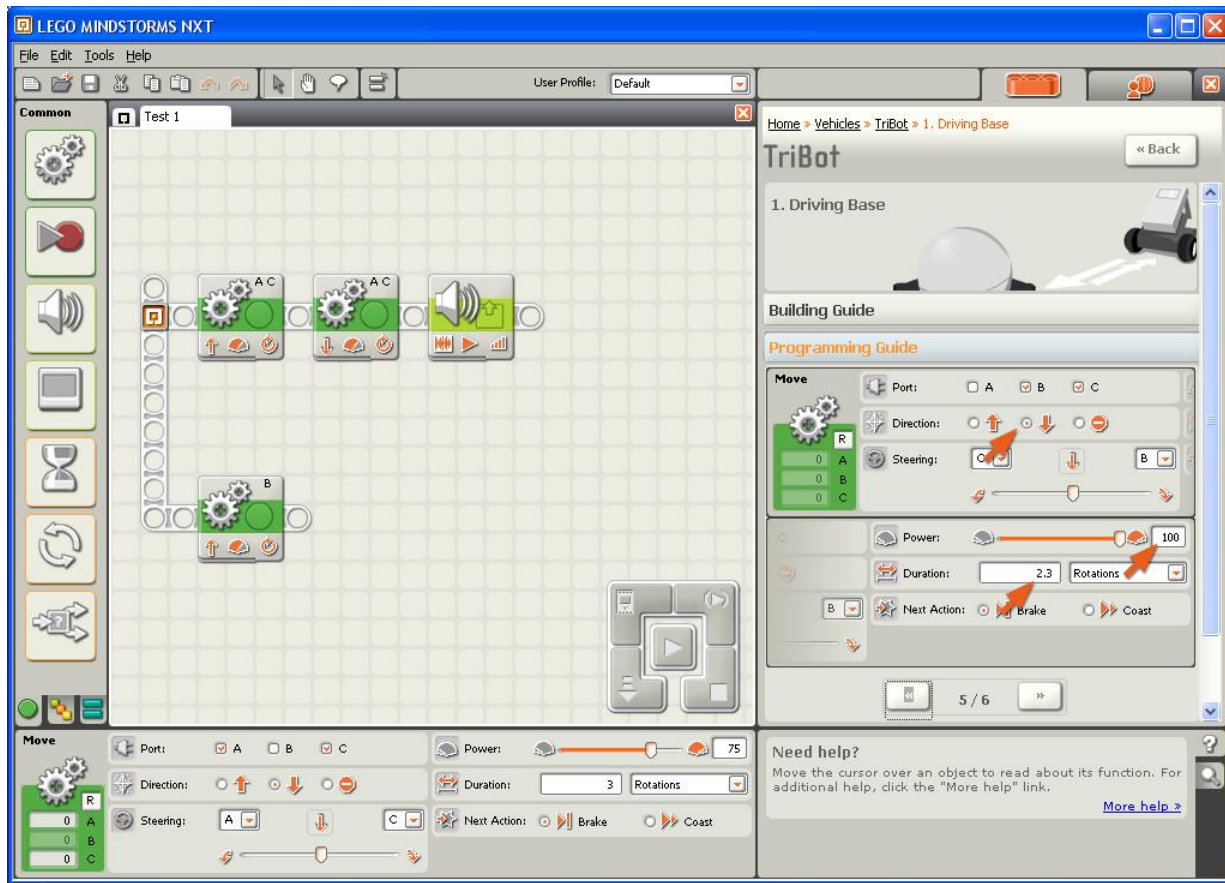
- UML – “svájci bicska”: mindenre jó... egy kicsit



- Nekünk inkább egy ládányi célszerszám kellene

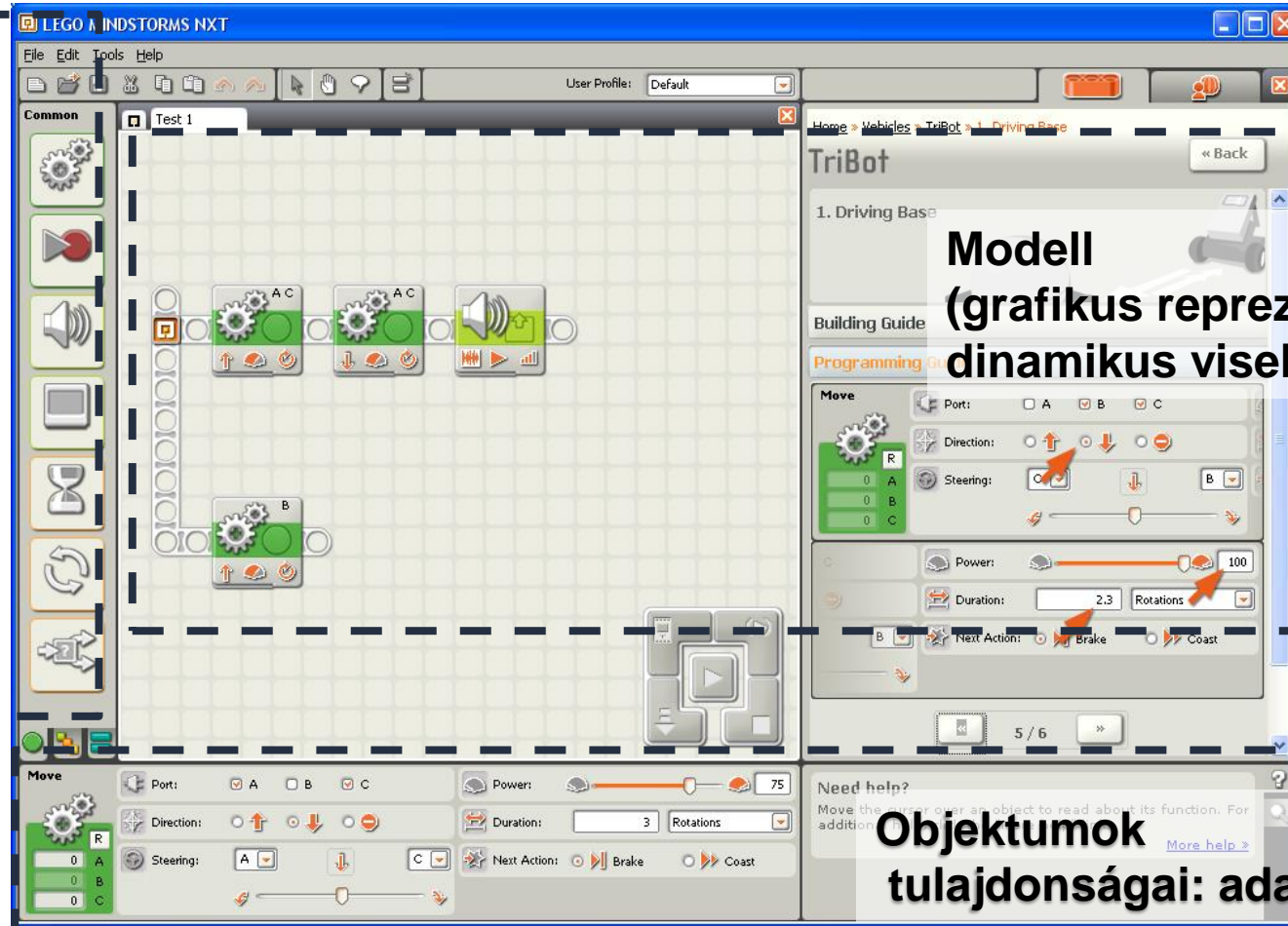


MINDSTORMS – NO CODE GRAFIKUS SZERKESZTŐ



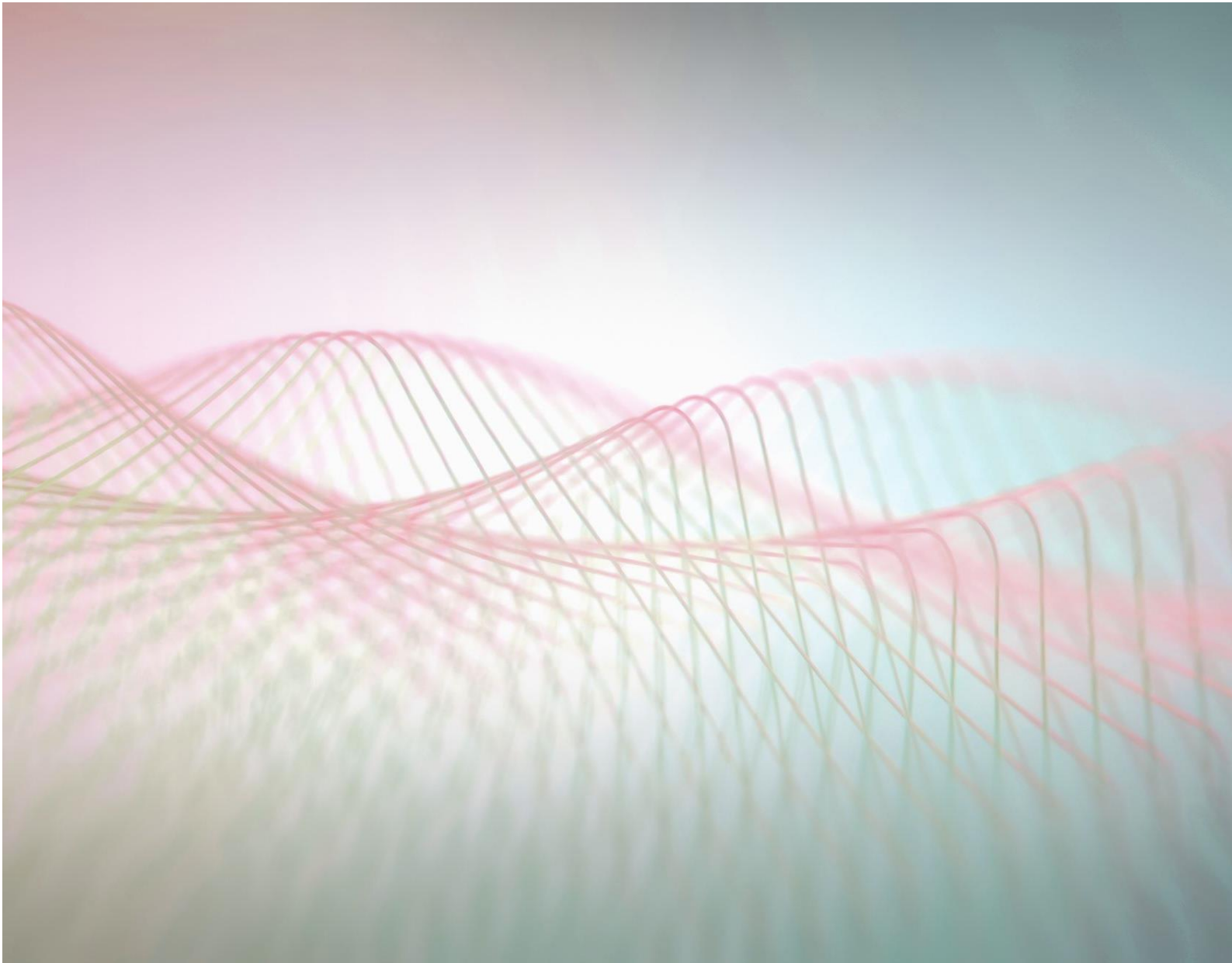
MINDSTORMS – PROGRAMOZÁSI KÖRNYEZET

**Modellezési
Primitívek**



**Modell
(grafikus reprezentáció):
dinamikus viselkedés**

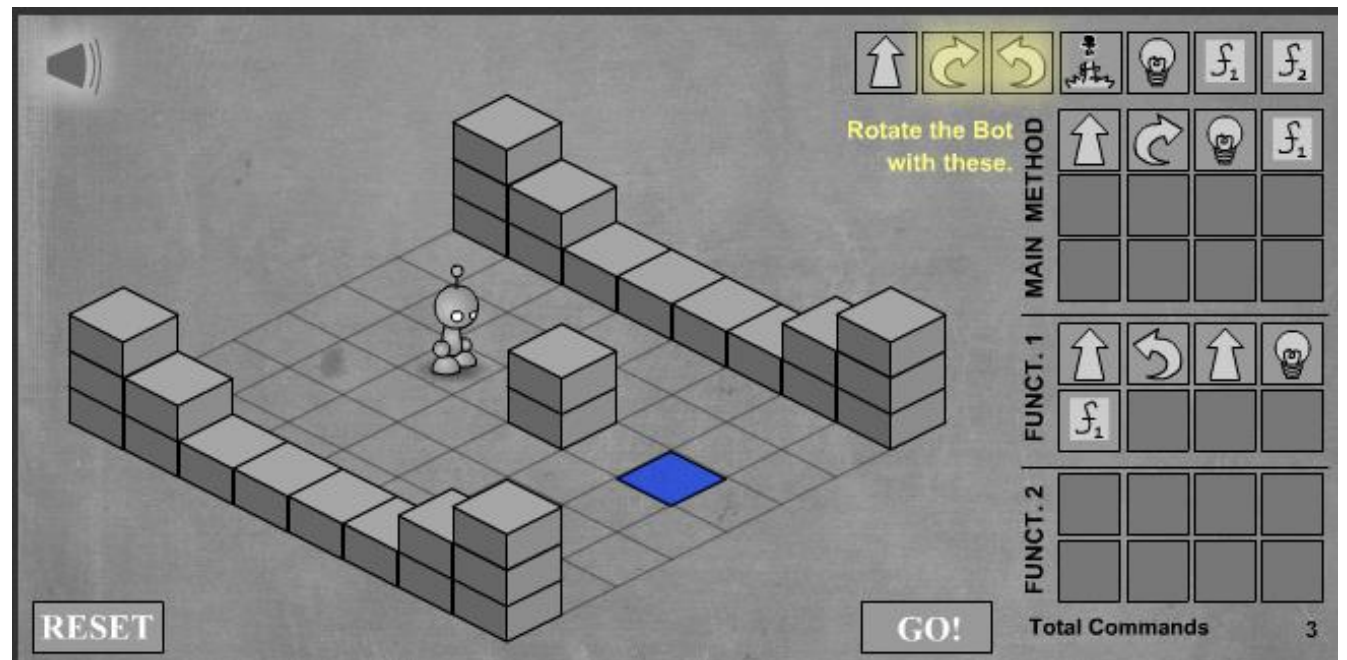
**Objektumok
tulajdonságai: adat**



MODELLEK A NAGYVILÁGBÓL

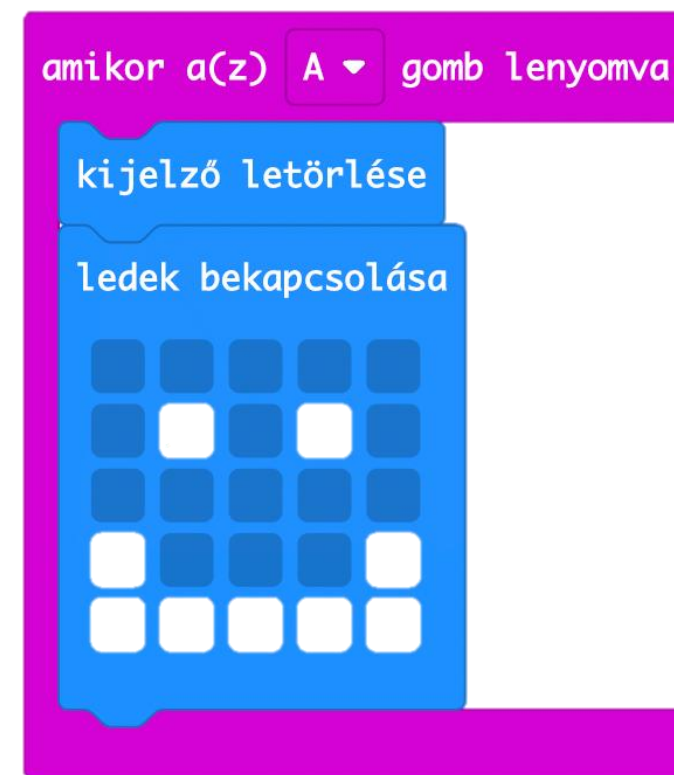
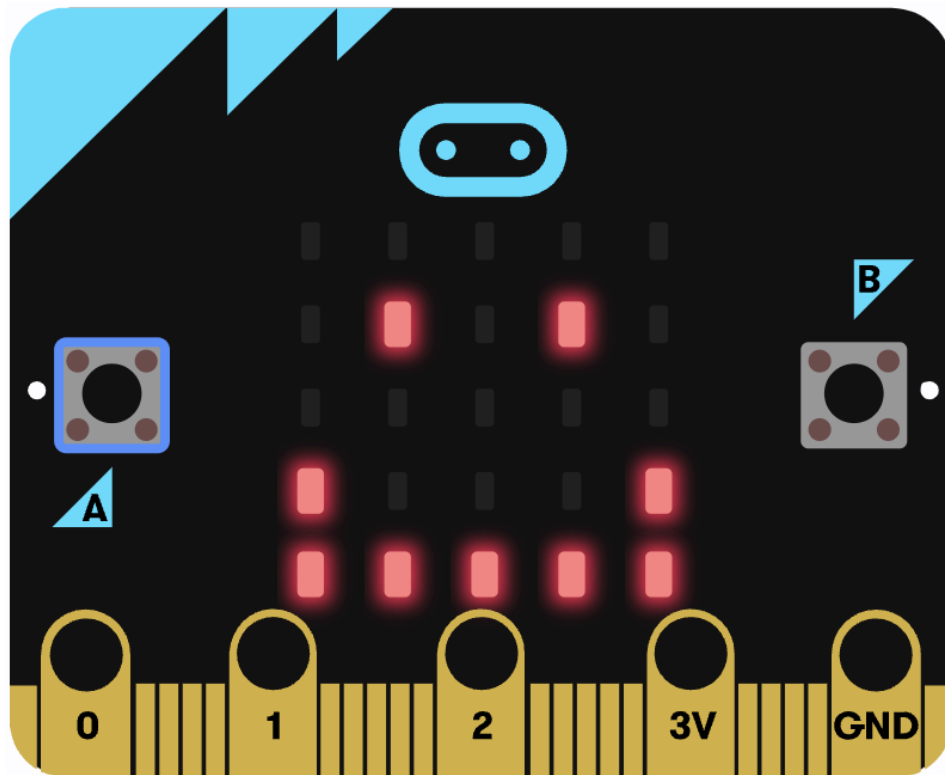
TOVÁBBI PÉLDÁK – LIGHTBOT

- Robotirányítás néhány paranccsal
 - Grafikus programozási felület
 - Grafikus “debugger”



TOVÁBBI PÉLDÁK – MICRO:BIT

- Beágyazott programozás néhány blokkal



TOVÁBBI PÉLDÁK – CCG

■ Gyűjtögetős kártyajáték - szabályrendszer



```
CARD {  
  Name: "Kvatch Solder";  
  Type: Creature;  
  Attack: 2;  
  Health: 3;  
  Cost: 3;  
  Guard: true;  
}
```



TOVÁBBI PÉLDÁK – FORM EDITOR

Select

SELECT ▲

Name (?)

Select

Description (?)

Options (?)

☐ Option 1

☐ Option 2

☐ Option 3

+

Add Option

Validation (?)

Required (?)

No

Size (?)

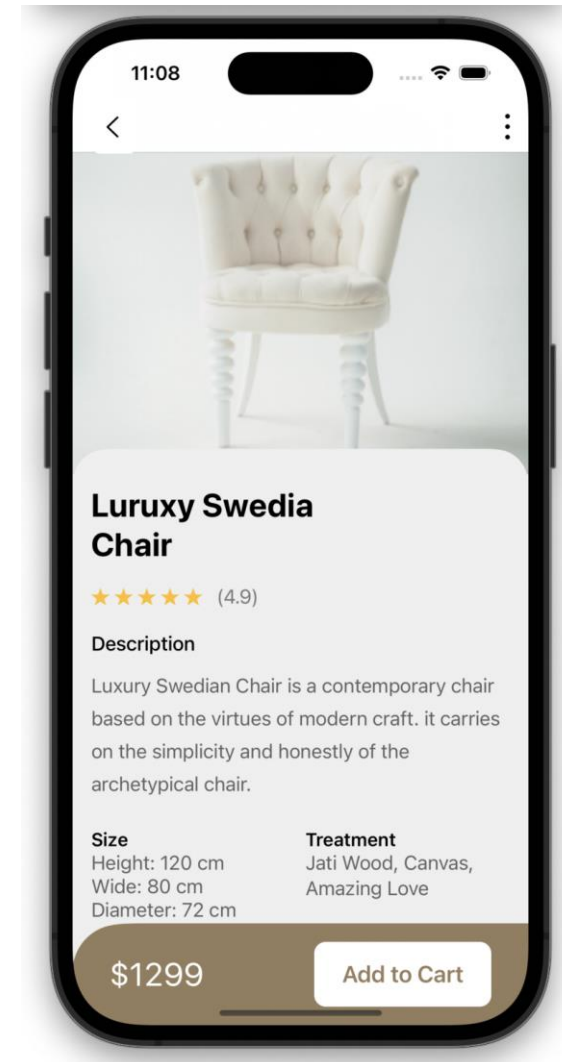
Medium

Field Layout (?)

Default

CSS Classes (?)

Delete



TOVÁBBI PÉLDÁK – SQL / NOSQL

SQL: általános célú relációs adatbázis definíció és lekérdezésre specializált deklaratív **nyelv**

- Adatbázistól megvalósítástól független
- Programozás helyett használható
- Absztrakciós réteg

NoSQL: Specializált adatbázisok és lekérdezési nyelvek

- Új keresőalgoritmusok használatára új nyelvek
- A nyelvek kiemeli az algoritmusok erősségeit

```
SELECT Book.title,  
       count(*) AS Authors  
FROM Book  
      JOIN Book_author ON  
Book.isbn = Book_author.isbn  
      GROUP BY Book.title;  
SQL query
```

```
(:Person {name: string})  
-[:ACTED_IN {roles: [string]}]->  
(:Movie {title: string, released: number})  
Cypher query
```

A Neo4J hatékony **join** algoritmust ígér,
ezért ezt **nyelvi elemként** használja:

$(x) -[]-> (y)$

TOVÁBBI PÉLDÁK

- Markup nyelvek: HTML, CSS, Latex
- Programozás tanulása: Logo, Scratch
- Játék engine programozás: UnrealScript
- Hardver leírás: VHDL, Verilog
- Pénzügyi szoftverek: HR szabályrendszer, Drools
- Beágyazott rendszerek: Yakindu, AUTOSAR

SZAKTERÜLETI NYELVEK VS. ÁLTALÁNOS CÉLÚ NYELVEK

Szakterületi nyelvek	Általános célú nyelvek
Adott szakterület fogalmait használja (pl. bicikli, HTML input form)	Általános fogalmakat használ (pl. osztály, függvény, XML tag)
Szakértők számára készül	Programozók számára készül
Speciálisabb célok	Általánosabb célok
Szabadabb szintaxis	Kötöttebb szintaxis
Egyedi feldolgozás és környezet	Támogatott fejlesztőkörnyezet

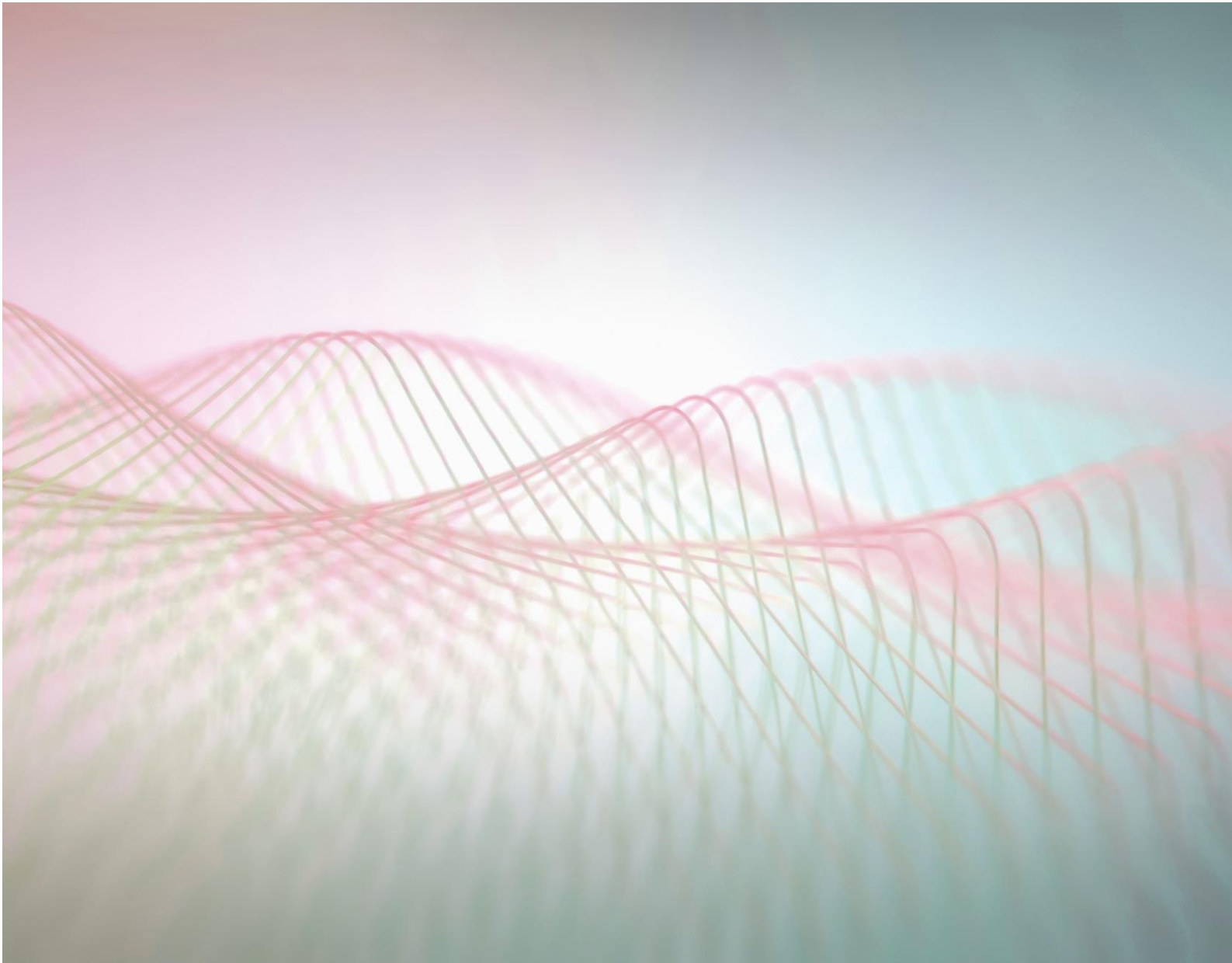
- Szakterületi nyelv = Domain-Specific Language (DSL)
- Általános célú nyelv = General-Purpose (Programming) Language (GPL)
 - Léteznek nem programozási, általános célú nyelvek is – Id. XML, JSON

SZAKTERÜLETI NYELV

- Szakterületi nyelv (Domain-Specific Language, DSL)
 - Speciális nyelv egy adott szakterületre
 - Korlátozott elemkészlet
 - Erősen specializált szabályok és jelölés
 - Egy adott termék(családhoz) készül
 - Tudunk kódot generálni belőle!
 - Low code – No code felfogás

NYELVEK ÖSSZETEVŐI

- **Nyelv összetevői**
 - Szintaxis (szerkezet + forma)
 - Absztrakt szintaxis (építőelemek, kapcsolatok)
 - Szöveges nyelv: nyelvtan, levezetési szabályok
 - Grafikus nyelvek: metamodellezés
 - Konkrét szintaxis (megjelenés)
 - Szemantika (jelentés)

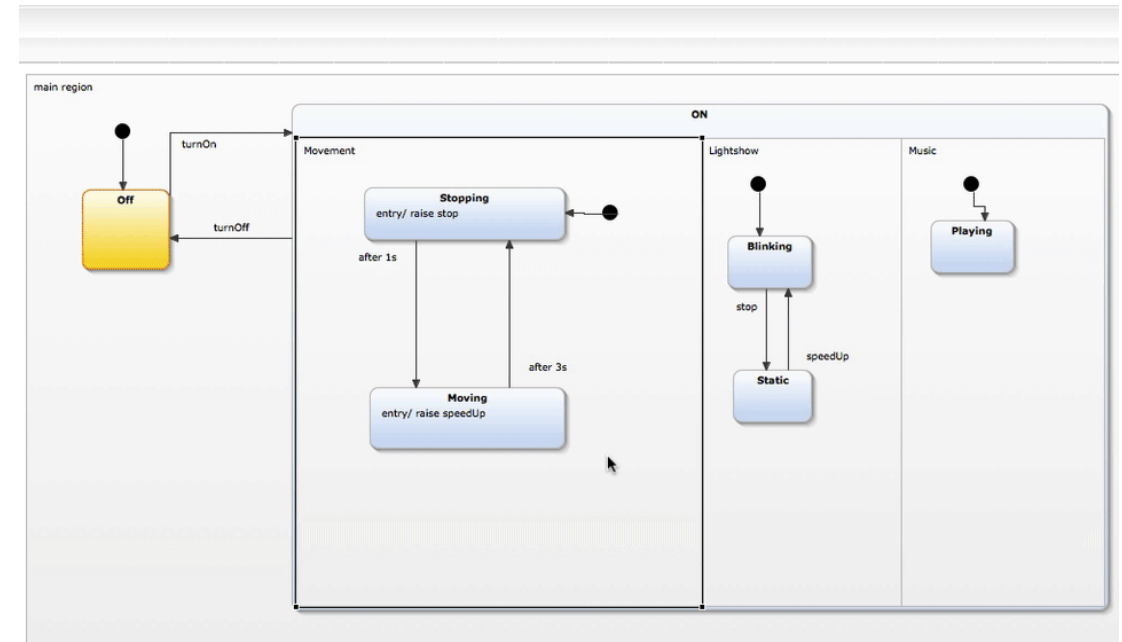


MODELLALAPÚ FEJLESZTÉS

SZAKTERÜLETI NYELV != MODELLEZÉS

- Egy szakterületi nyelv önmagában nem mindig elég!
 - Szerkesztő környezet
 - Debugger / szimulátor
 - Modellfeldolgozó (pl. kódgenerátor)
 - Kiegészítő funkciók (pl. helyesség ellenőrzés)

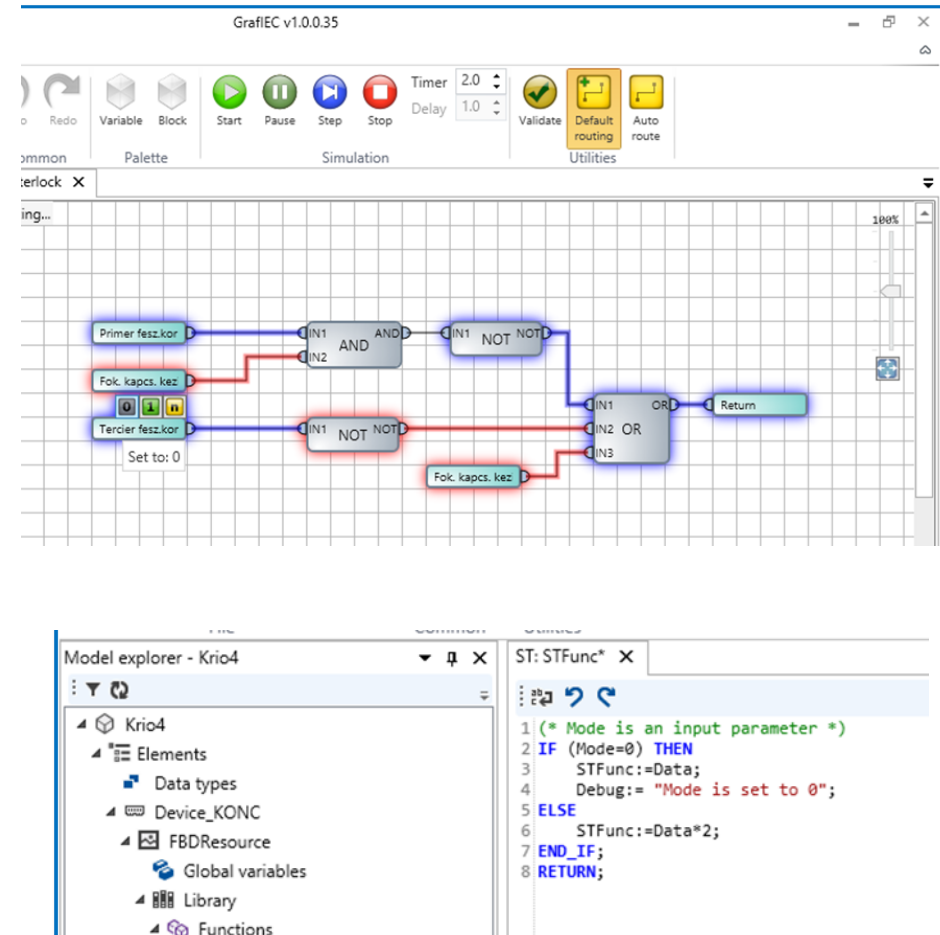
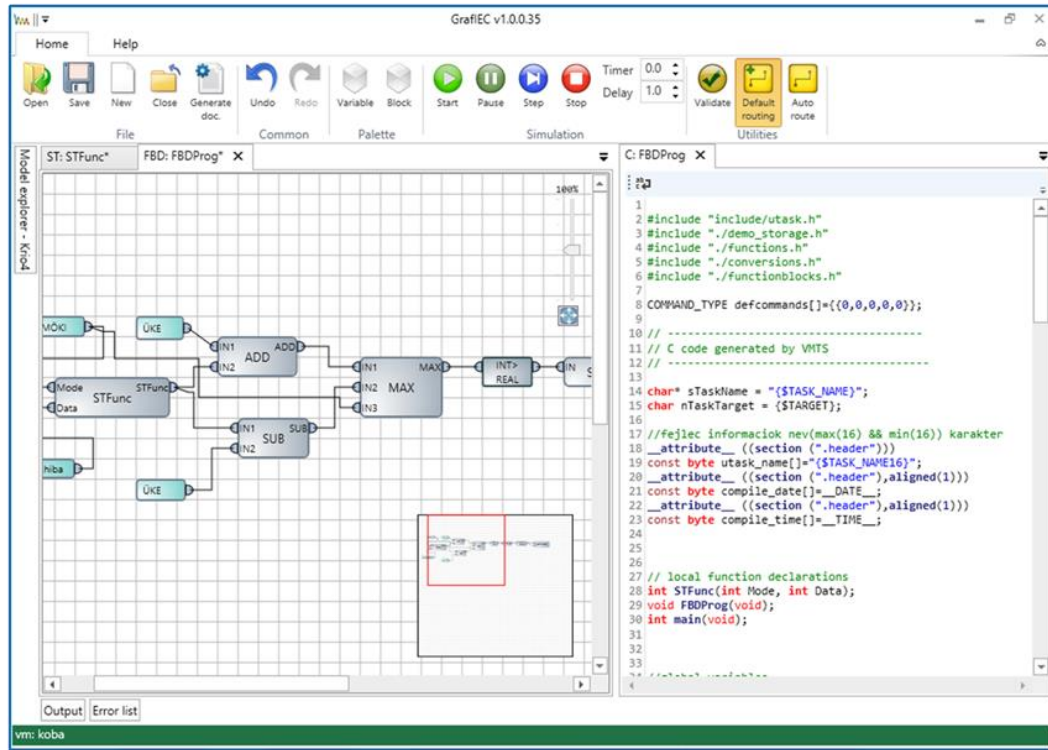
- Modellezőkörnyezet
- Szimulátor
- Kódgenerátor több nyelvre
- Matematikai helyességellenőrzés



<https://blogs.itemis.com/en/how-to-simulate-a-statechart-model>
<https://github.com/ftsrg/gamma>

GRAF IEC

■ IEC 61131 ipari szabvány



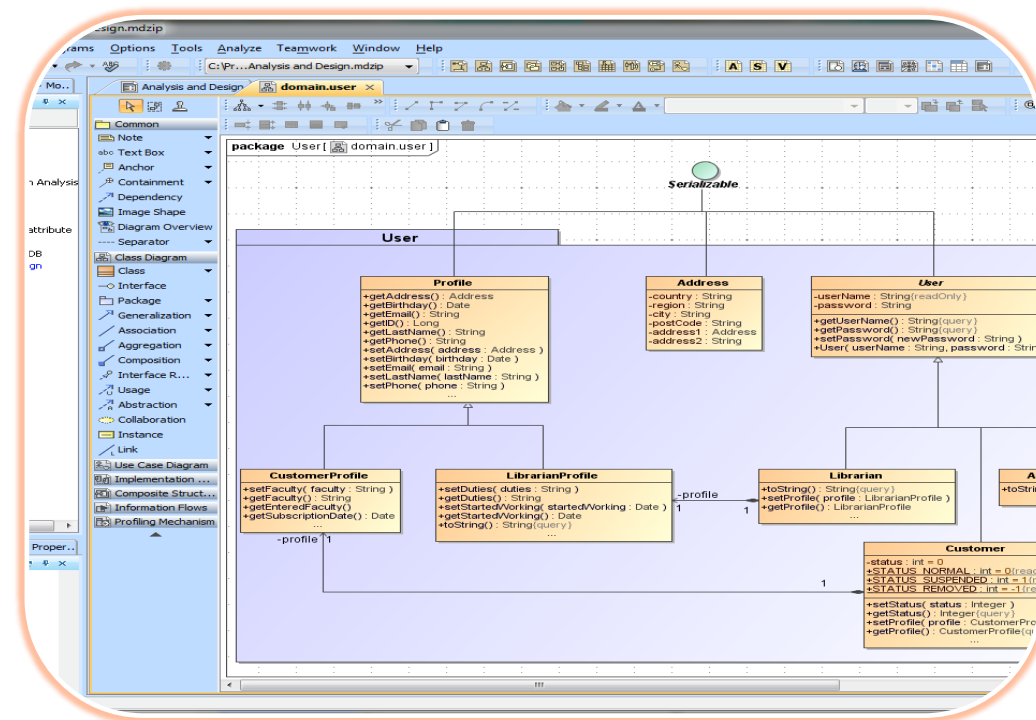
IPARI MODELLEZŐESZKÖZÖK

- Számos területen egyeduralkodók a modellezési nyelvek és modellezőeszközök
 - Fejlesztők kizárólag ezeken az eszközökön dolgoznak
 - Szabvány írja elő a használatukat

DO-178C, Software Considerations in Airborne Systems and Equipment Certification. SG4: Model Based Development and Verification

- Az eszközökhöz illeszkedni kell!
 - SysML, AUTOSAR, MATLAB, ...

Példák:



SysML: MagicDraw

MODELLALAPÚ FEJLESZTÉS

- Produktivitás és magas minőség
 - Ismerős nyelvi elemek és fogalmak a felhasználóknak
 - Kisebb változtatások fejlesztők nélkül
 - Kizárólag szakterületi szabályok mentén
 - Elrejtje a lényegtelen részeket (magas absztrakciós szint)
 - Célzott matematikai analízis
 - Multiplatform fejlesztés
- De: a kezdeti költség nagy lehet!
 - Saját nyelv és eszköz fejlesztése és karbantartása

A MAI ELŐADÁS

I. fejezet Miért?

II. fejezet Miről?

III. fejezet Hogyan?



A TÁRGYRÓL

- Adatlap: <https://portal.vik.bme.hu/kepzes/targyak/VIAUMA22>
- Oktatás – három tanszék kooperációja (AUT – IIT – MIT)
 - AUT: Mezei Gergely, Somogyi Ferenc
 - IIT: Simon Balázs
 - MIT: Semeráth Oszkár
- Előadás – minden héten
 - Elméleti ismeretek
- Gyakorlat – minden második héten
 - Az elmélet szemléltetése a gyakorlatban
 - Demók, gyakorlati példák, esettanulmányok



SZÁMONKÉRÉSEK

- 5 db házi feladat
 - Github
 - A gyakorlatok mentén
 - 3 sikeres házi kell az aláíráshoz
- ZH (2025.04.07.)
- Vizsga
 - Írásbeli
 - A félévközi eredmények beleszámítanak (48p + 10p + 14p)

1

Szöveges modellezés

Fordítóprogramok,
Nyelvfeldolgozás lépései.
Kódgenerálás,
Interpreterk

2

Grafikus modellezés

Szerkezet + megjelenítés,
Blockly, UML Profile,
Metamodellezés,
Szemantika

3

Modellfeldolgozás

Modellfeldolgozás,
Kódgenerálás,
Gráftranszformáció,
Modellalapú fejlesztés

MIRŐL LESZ SZÓ?

A KÖVETKEZŐ RÉSZ TARTALMÁBÓL...

- **Gyakorlat – modellezés az iparban**
- **Theisz Zoltán** (Thyssenkrupp):
Modellezés az autóiparban
- **Kulcsár Géza** (IncQuery Labs):
Modellezéssel Budapestről a Marsig
- **Graics Bence** (MIT):
Komplex rendszerek modellalapú megbízhatóságellenőrzése





KÖSZÖNÖM A FIGYELMET!