Wigner GPU Lab

5. fejezet

A C++ fordítási modellje

Grafikus Processzorok Tudományos Célú Programozása

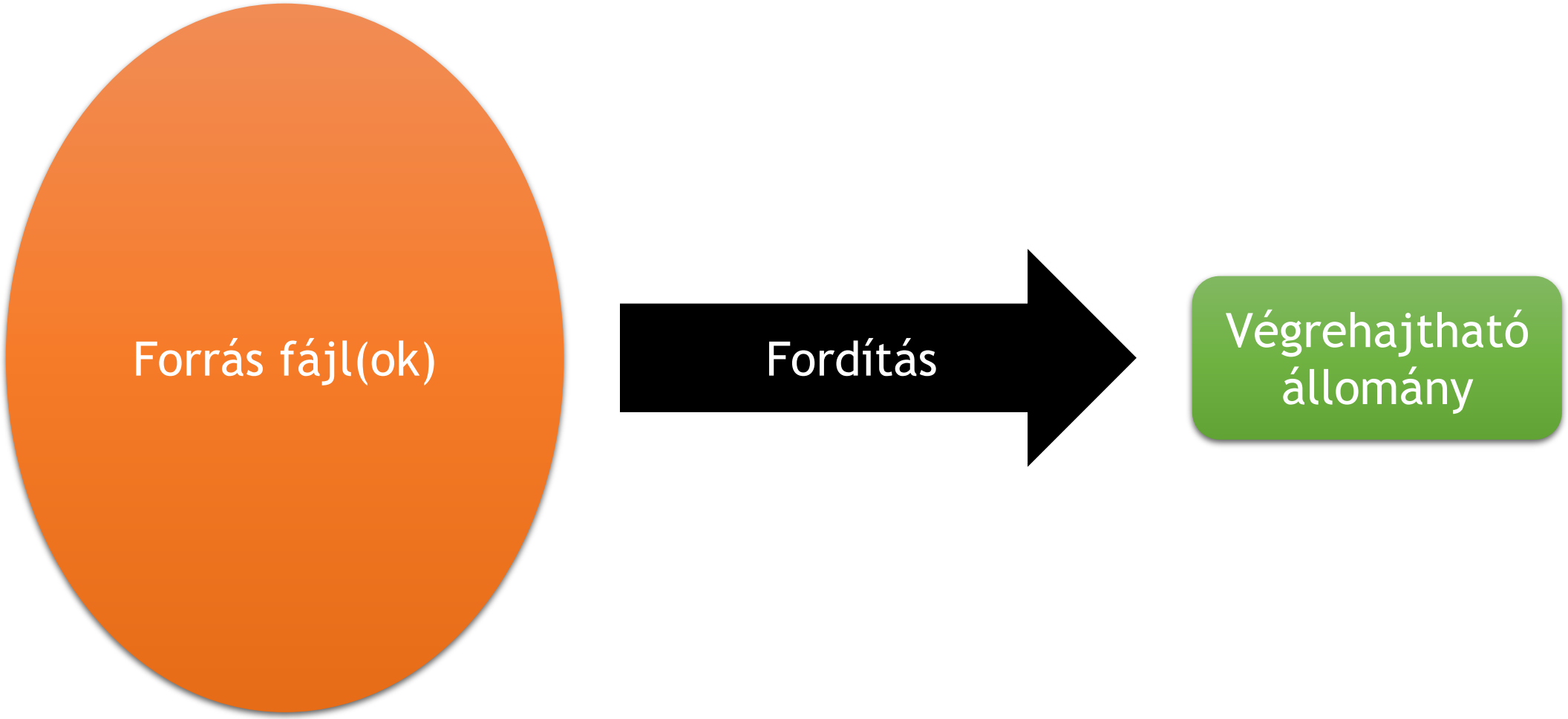
Berényi Dániel – Nagy-Egri Máté Ferenc

Kódtól a végrehajtásig

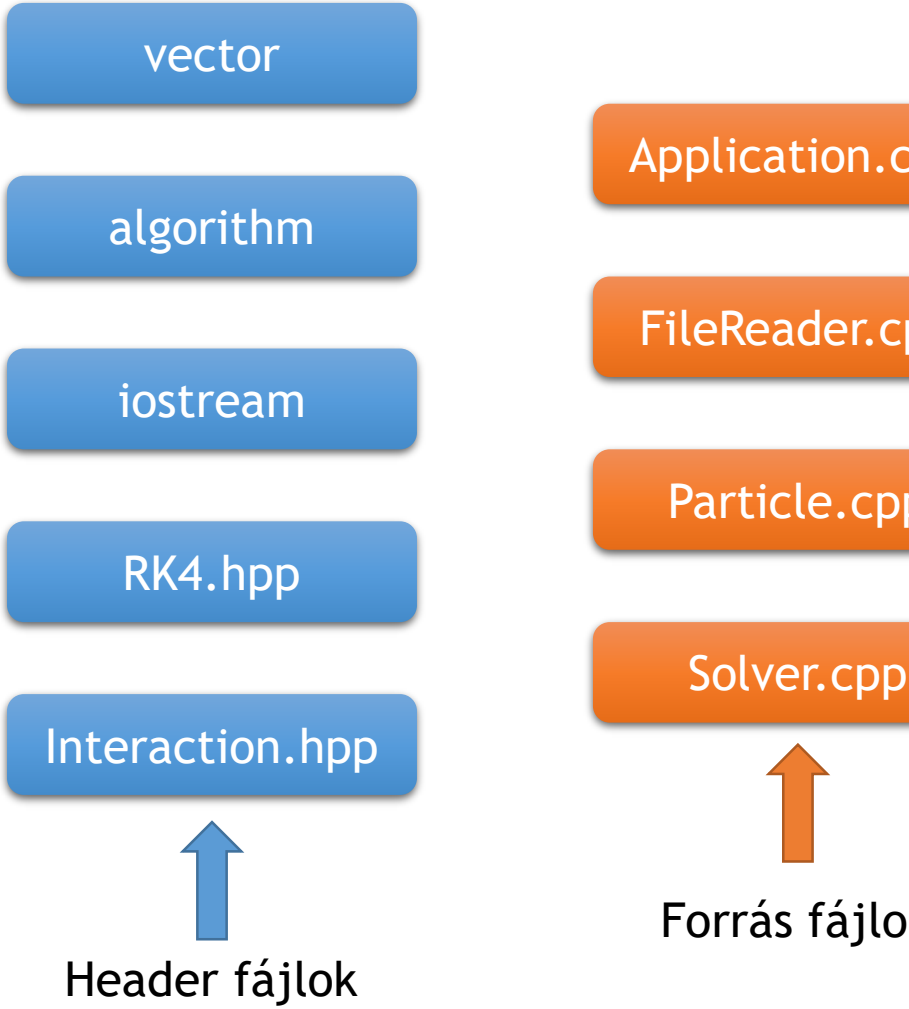
Végrehajtás előtt valamikor létre kell jönnie az adott architektúrára jellemző bináris utasításoknak.

- Fortran, C, C++, stb.
  - Kód + fordító → bináris
- C++AMP, DirectX, Vulkan, stb.
  - Kód + fordítók → bináris + köztes → bináris + köztes + meghajtó
- Java, C#, stb.
  - Kód + fordító → köztes → köztes + értelmező/futásidejű fordító
- Python, PowerShell, MatLab, Mathematica, stb.
  - Kód → értelmező

Hogyan fordul egy C++ program?

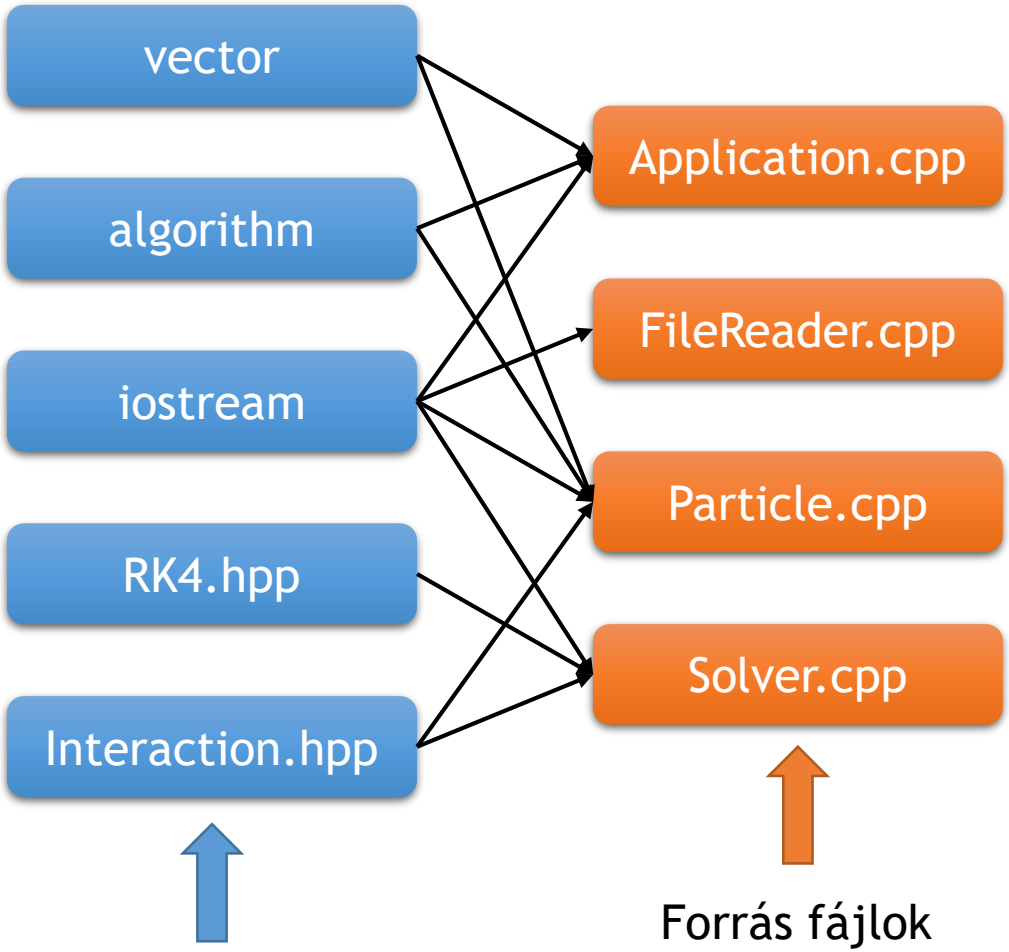


Hogyan fordul egy C++ program?



- Header fájlok
  - Függvény deklarációk
    - Név és szignatúra $func(\mathbb{M}, \mathbb{V}) \rightarrow \mathbb{V}$
  - Típus deklarációk és definíciók
  - Sablon (template) kód
- Forrás fájlok (sources)
  - Függvény definíciók
    - Név, szignatúra, megvalósítás

Hogyan fordul egy C++ program?



- Minden forrás fájl egy külön fordítási egység
  - A fordító mindent elfelejt két forrás között
- A források n headerre hivatkozhatnak (include-olják)
- A headerek egymásra is hivatkozhatnak
- C/C++: One Defintion Rule
  - Ha többször hivatkozunk ugyanarra a headerre, akkor megszegjük az ODR-t
  - Ezt megkerülendő: Include Guard
- Miért ilyen bonyolult a fordítás?
  - Elválik a függvény a megvalósítástól
  - Gyorsabb fordítás (lásd később)
  - C++20-ban jobb lesz (Modulok)

Hogyan fordul egy C++ program?

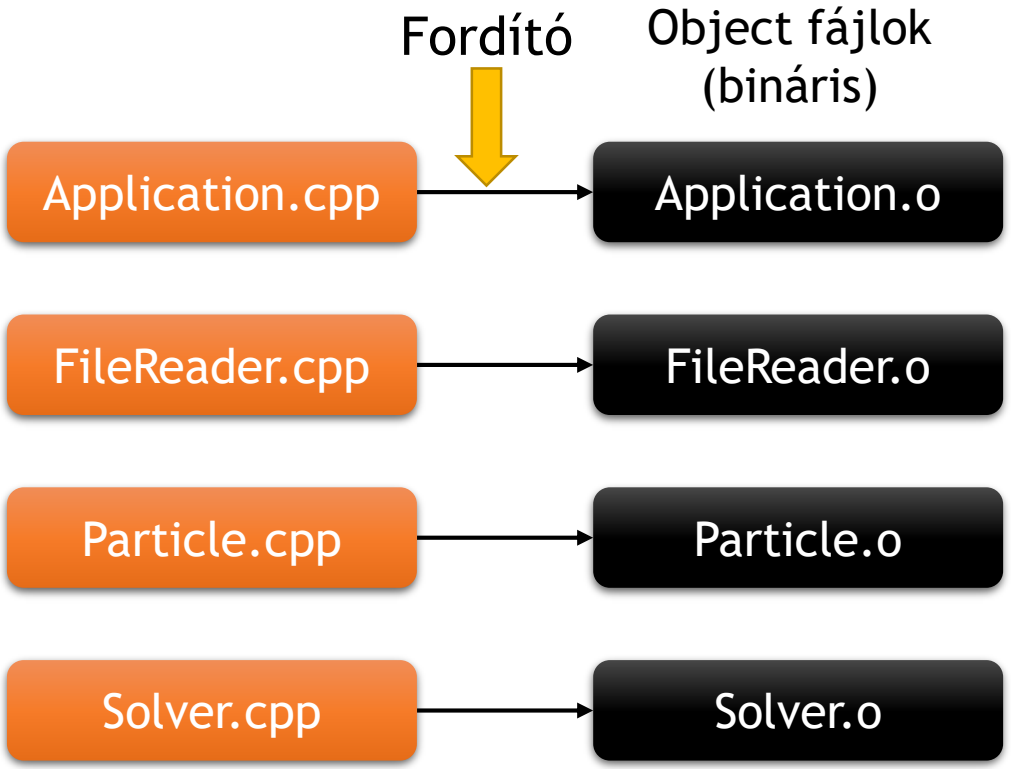
vector

algorithm

iostream

RK4.hpp

Interaction.hpp



Object fájlokban dekorált binárisok

Szimbólumok és a hozzájuk tartozó binárisok vannak bennük

A szimbólum egy a függvény nevéből és szignatúrájából generált név, pl.:

```
__operator*(classMat,classVec)->classVec
```

Hogyan fordul egy C++ program?

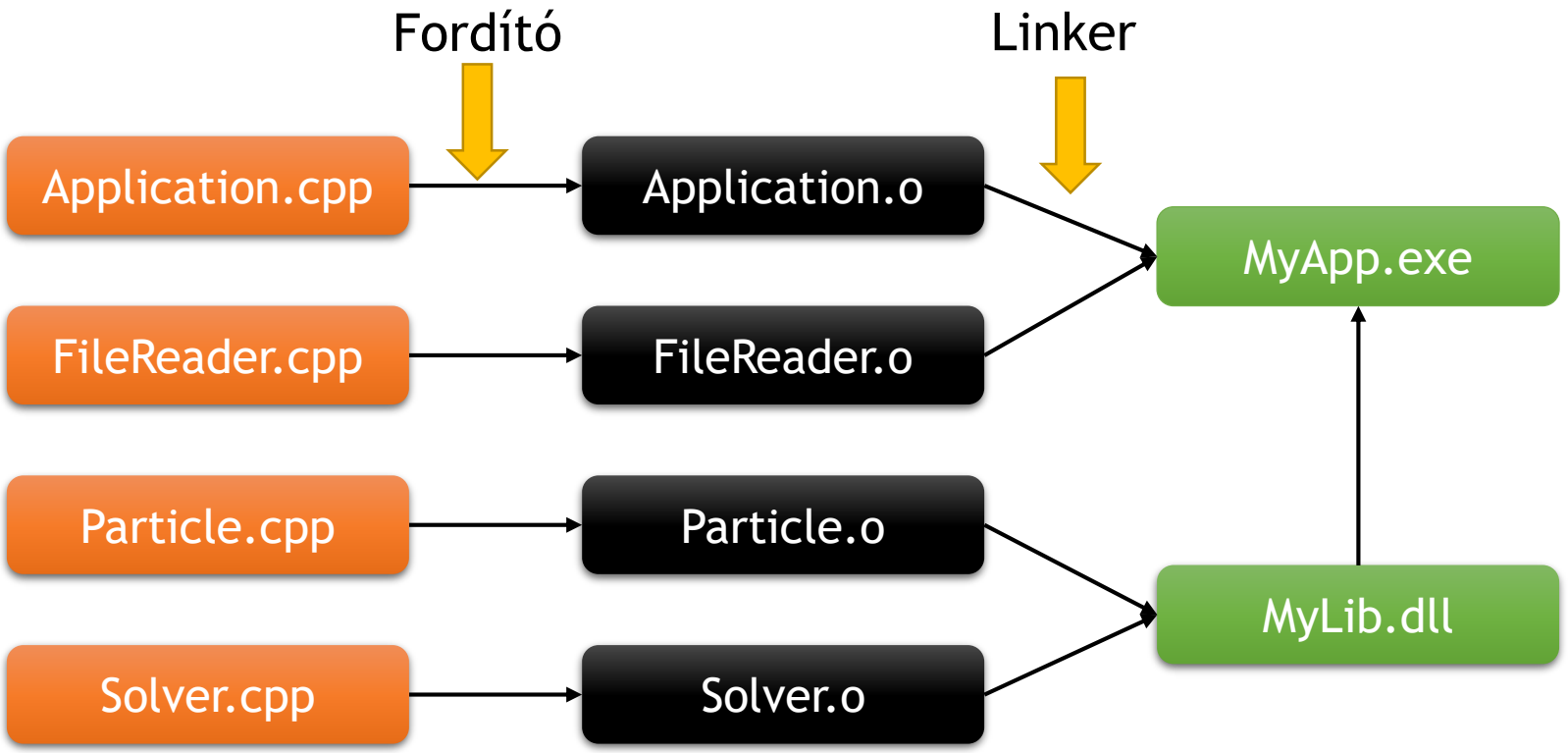
vector

algorithm

iostream

RK4.hpp

Interaction.hpp



Hogyan fordul egy C++ program?

Linkelési fázis:

A linker megpróbálja az összes szükséges* függvény bináris reprezentációját előállítani/megtalálni. Ami nem szükséges, azt eldobja.

*szükséges:

- futtatható esetén, ami a belépési pontból elérhető/kellő,
- könyvtár esetén, ami exportált függvény, vagy abból elérhető.

Ha futtatható állományt fordítunk, akkor a linker keres még egy:

- ”main” függvényt, ha parancssori alkalmazást fordítunk
- grafikus alkalmazásoknál esetleg spec névű fv-t (pl. ”WinMain”)

Hogyan fordul egy C++ program?

A linker továbbá:

- Ha van külső könyvtár függőség, amihez linkelni kell, akkor az abban található szimbólumokat/binárist is átveszi
- A hiányzó szimbólumok hibát eredményeznek
  (unresolved external symbol hibák)
- Lehetnek függvények, amelyek többször lettek lefordítva
  - Ha a binárisok megegyeznek, akkor egy kivételével az összeset eldobja
  - Ha különböznek, a linker hibát jelez
    (tipikus hiba, amikor olyan dolgokat akarunk összelinkelni, amik:
    - eltérő fordítóval
    - más architektúrára (x86-x64)
    - vagy más standard library-val készültek)

Hogyan fordul egy C++ program?

A forráskód header/source részekre bontásával minimalizáljuk annak az esélyét, hogy egy függvényt többször fordítsunk le

Különben egy nagyobb projektnél sokáig tarthat a linkelés (is)

Könyvtárak linkelése

Statikus könyvtárak

- A szimbólumokat egy az egyben átveszi a könyvtárból a futtathatóba
- Gyorsabb kódot eredményez
- Ha sok futtatható hivatkozik ugyanarra a szimbólumra, azok többször szerepelnek a lemezen/memóriában
- Kényelmesebb használni

Dinamikus könyvtárak

- Csak egy hivatkozást illeszt a kódba a könyvtárban található szimbólumra
- Kisebb binárist eredményez
- Ha sok futtatható hivatkozik ugyanarra a szimbólumra, azok csak egyszer szerepelnek a lemezen/memóriában
- Futásidőben csak ez működik