Többszálúság

Csorba Kristóf

Process, thread, Task

- Oprendszer szinten, shared memory, hozzáférési jogosultságok

async-await: főzéses példa

- https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/
- (Főzési példa… blokkolva várás rossz lenne)
- Taskok, de mindig await: sorban, de legalább válaszolsz, ha valaki szól hozzád.

async-await: főzéses példa (1)

```
static void Main(string[] args)
{
    Coffee cup = PourCoffee();
    Console.WriteLine("coffee is ready");
    Egg eggs = FryEggs(2);
    Console.WriteLine("eggs are ready");
    Bacon bacon = FryBacon(3);
    Console.WriteLine("bacon is ready");
    Toast toast = ToastBread(2);
    ApplyButter(toast);
    ApplyJam(toast);
    Console.WriteLine("toast is ready");
    Juice oj = PourOJ();
    Console.WriteLine("oj is ready");

    Console.WriteLine("Breakfast is ready!");
}
```

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/

async-await: főzéses példa (2)

Ne blokkoljuk a futást, amíg a háttérmunka folyik...

```
static async Task Main(string[] args)
{
    Coffee cup = PourCoffee();
    Console.WriteLine("coffee is ready");
    Egg eggs = await FryEggs(2);
    Console.WriteLine("eggs are ready");
    Bacon bacon = await FryBacon(3);
    Console.WriteLine("bacon is ready");
    Toast toast = await ToastBread(2);
    ApplyButter(toast);
    ApplyJam(toast);
    Console.WriteLine("toast is ready");
    Juice oj = PourOJ();
    Console.WriteLine("oj is ready");

    Console.WriteLine("Breakfast is ready!");
}
```

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/

async-await: főzéses példa (3)

Task elmentése ahelyett, hogy rögtön await-tel megvárnánk a végét…

```
Coffee cup = PourCoffee();
Console.WriteLine("coffee is ready");
Task<Egg> eggTask = FryEggs(2);
Egg eggs = await eggTask;
Console.WriteLine("eggs are ready");
Task<Bacon> baconTask = FryBacon(3);
Bacon bacon = await baconTask;
Console.WriteLine("bacon is ready");
Task<Toast> toastTask = ToastBread(2);
Toast toast = await toastTask;
ApplyButter(toast);
ApplyJam(toast);
Console.WriteLine("toast is ready");
Juice oj = PourOJ();
Console.WriteLine("oj is ready");

Console.WriteLine("Breakfast is ready!");
```

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/

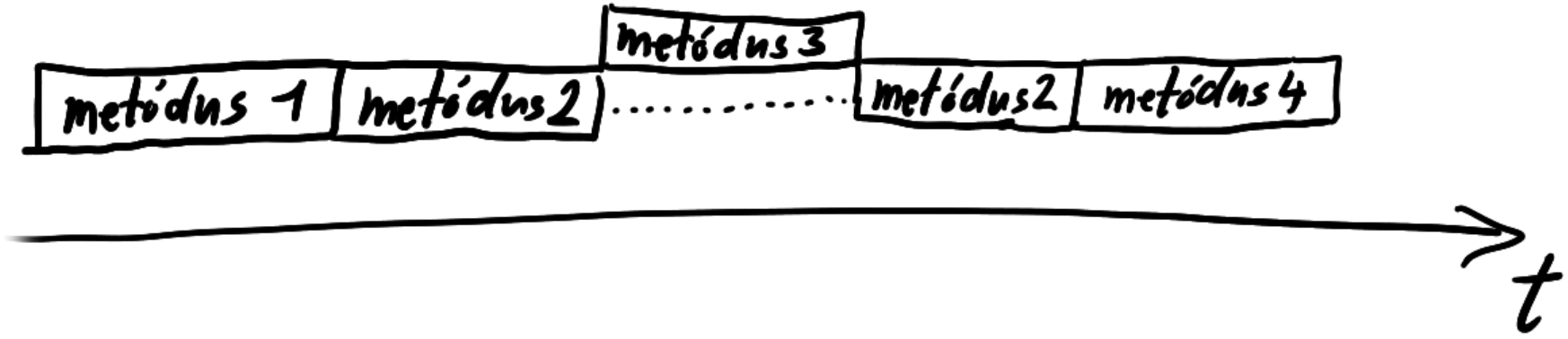
async-await: főzéses példa (4)

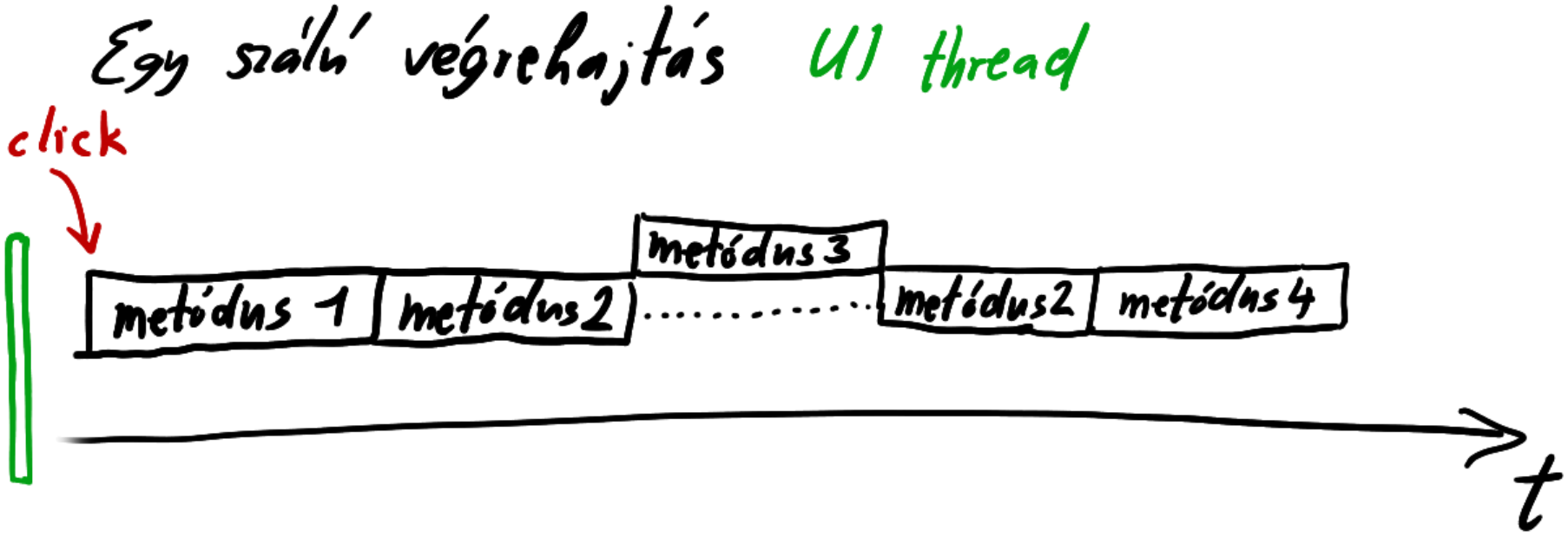
Csak akkor várjuk meg a végét egy műveletnek, amikor már tényleg kell az eredmény!

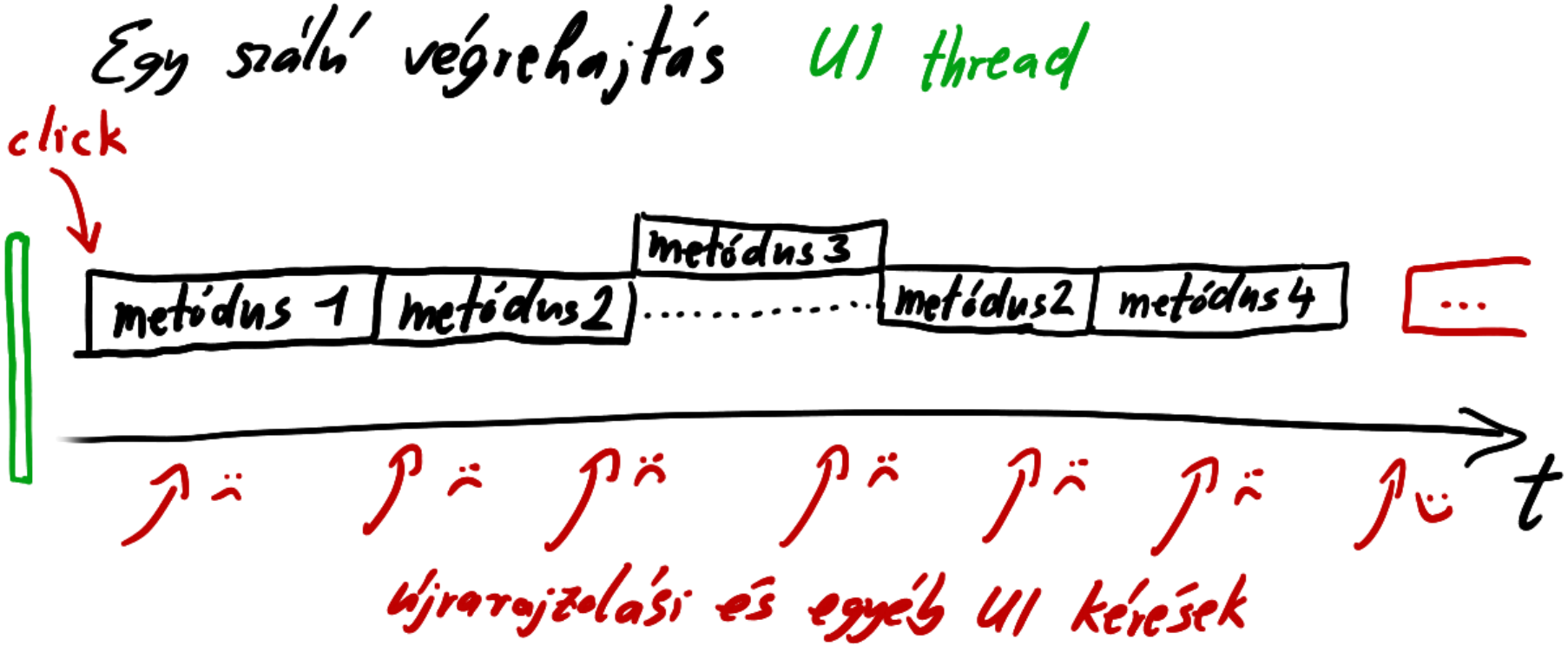
```
Coffee cup = PourCoffee();
Console.WriteLine("coffee is ready");
Task<Egg> eggTask = FryEggs(2);
Task<Bacon> baconTask = FryBacon(3);
Task<Toast> toastTask = ToastBread(2);
Toast toast = await toastTask;
ApplyButter(toast);
ApplyJam(toast);
Console.WriteLine("toast is ready");
Juice oj = PourOJ();
Console.WriteLine("oj is ready");

Egg eggs = await eggTask;
Console.WriteLine("eggs are ready");
Bacon bacon = await baconTask;
Console.WriteLine("bacon is ready");

Console.WriteLine("Breakfast is ready!");
```

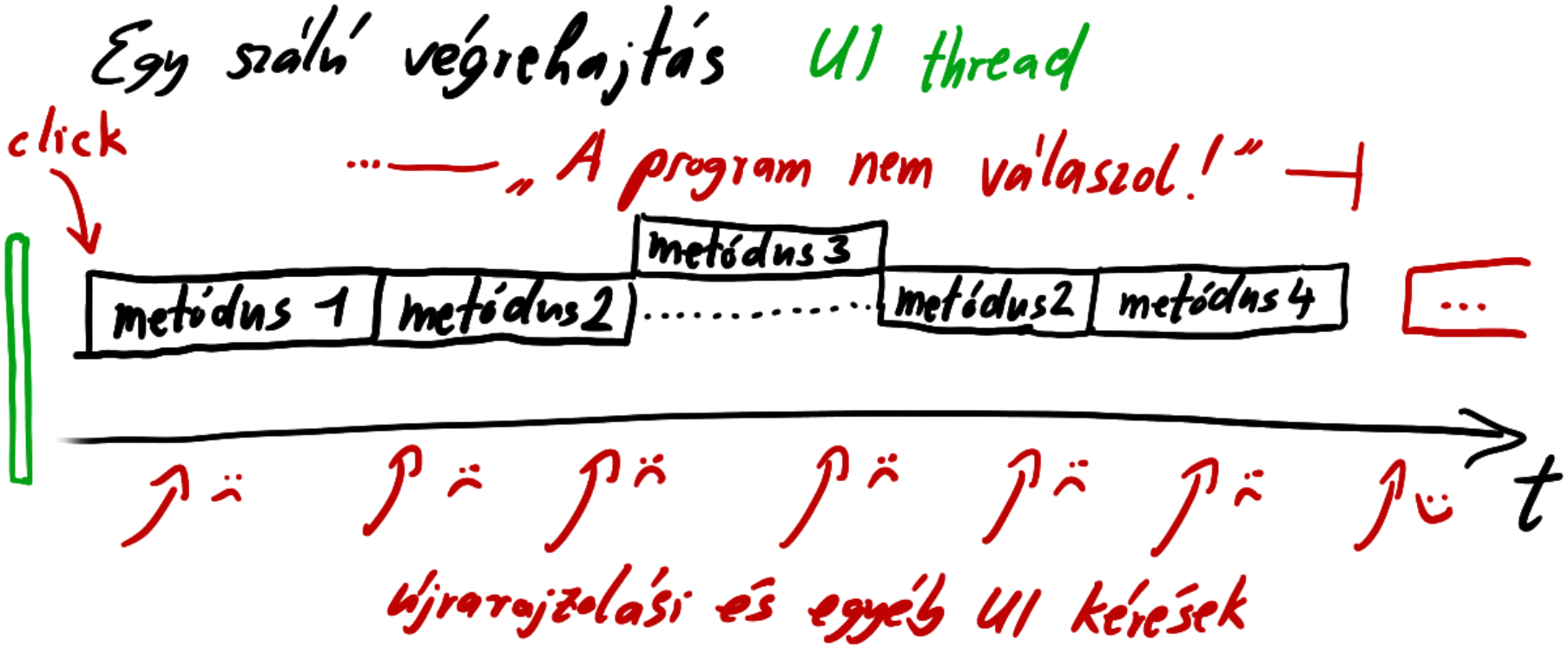
https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/

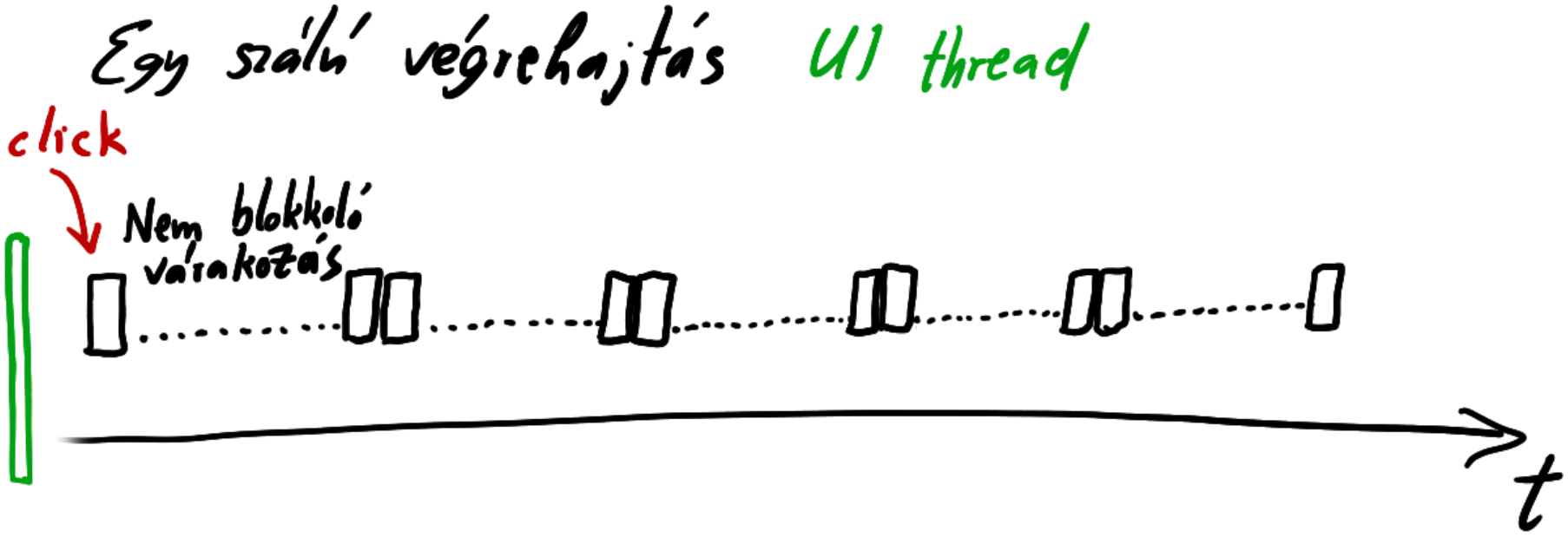
Egy szálú végrehajtás



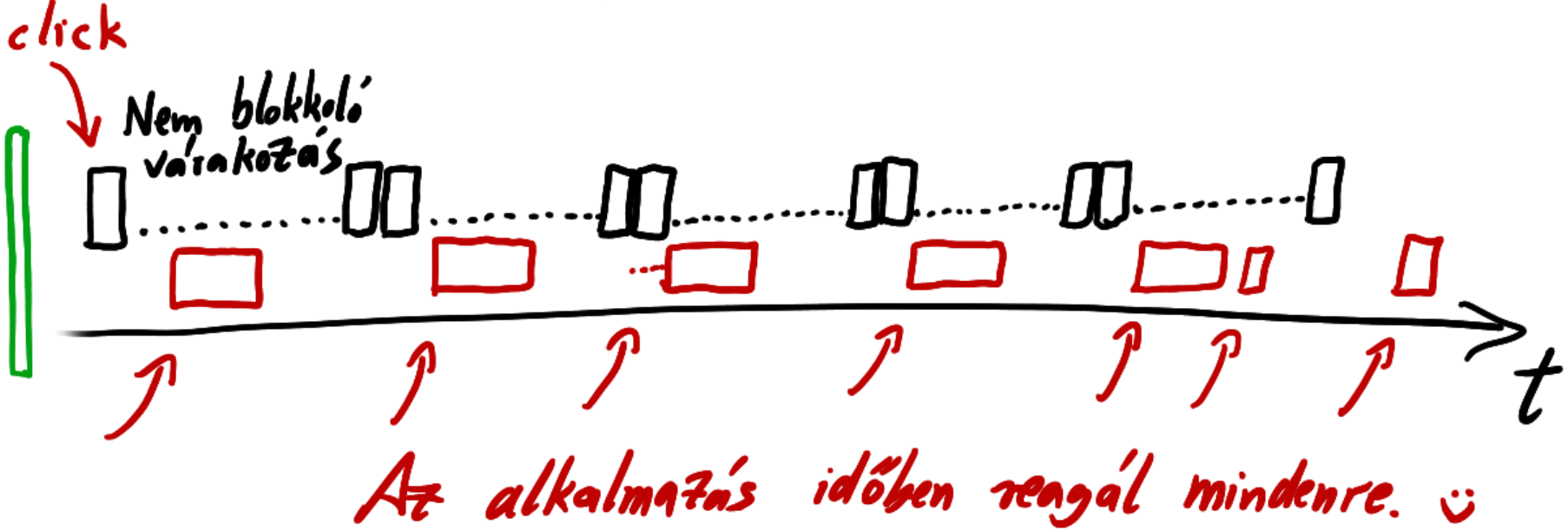

Egy szálú végrehajtás
UI thread
click
metódus 1
metódus2
metódus 3
metódus2
metódus 4
t


Egy szálú végrehajtás
UI thread
click
metódus 1
metódus2
metódus 3
metódus2
metódus 4
...
t
újrarajzolási és egyéb UI kérések


Egy szálú végrehajtás
UI thread
click
„A program nem válaszol!"
metódus 1
metódus 2
metódus 3
metódus 2
metódus 4
...
t
újrarajzolási és egyéb UI kérések

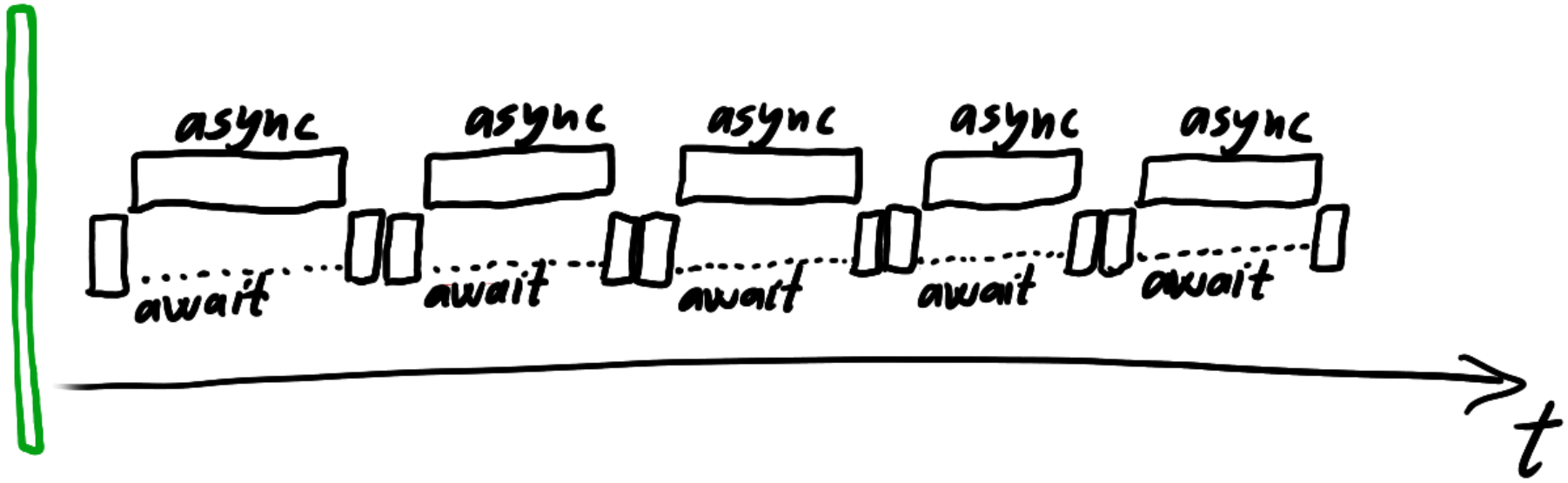

Egy szálú végrehajtás
UI thread
click
Nem blokkoló
várakozás
t

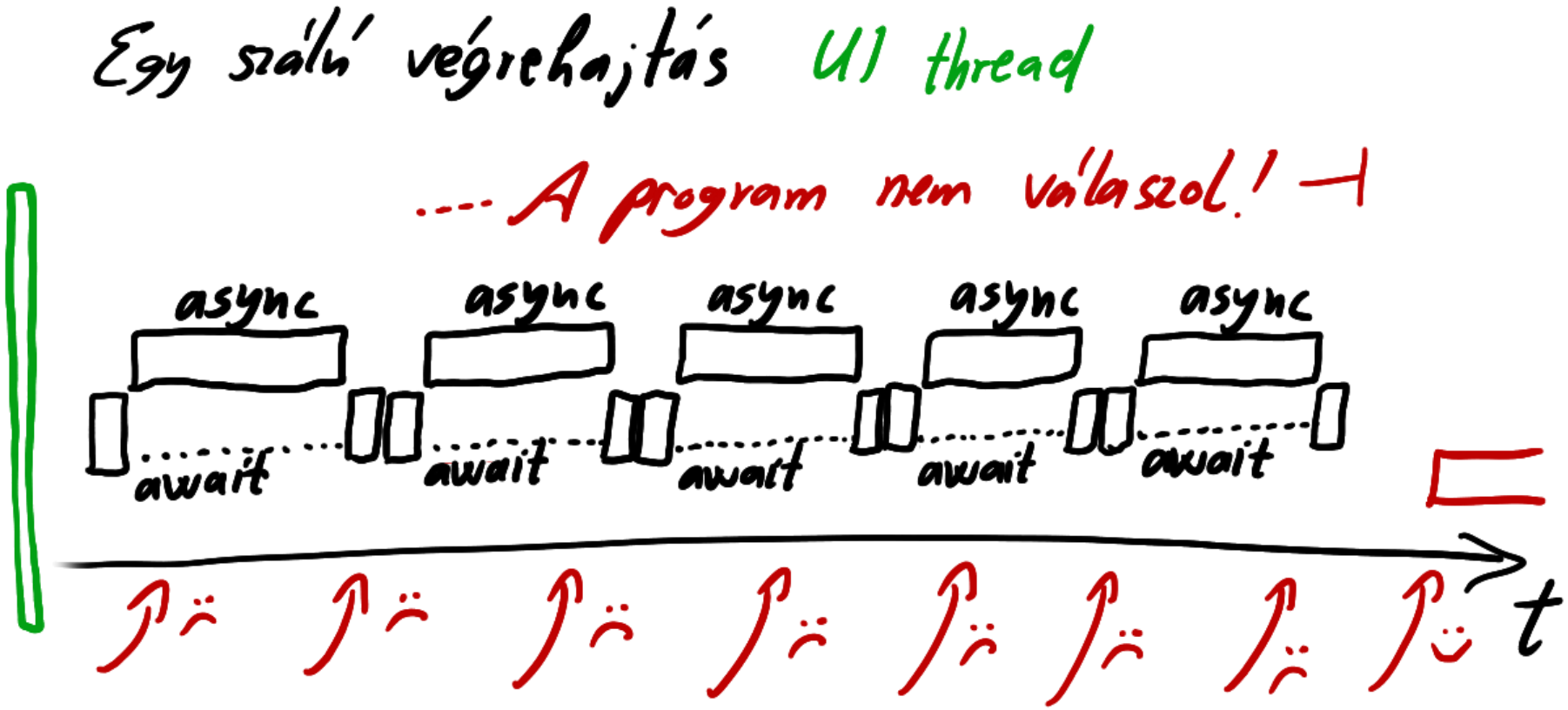
Egy szálú végrehajtás UI thread



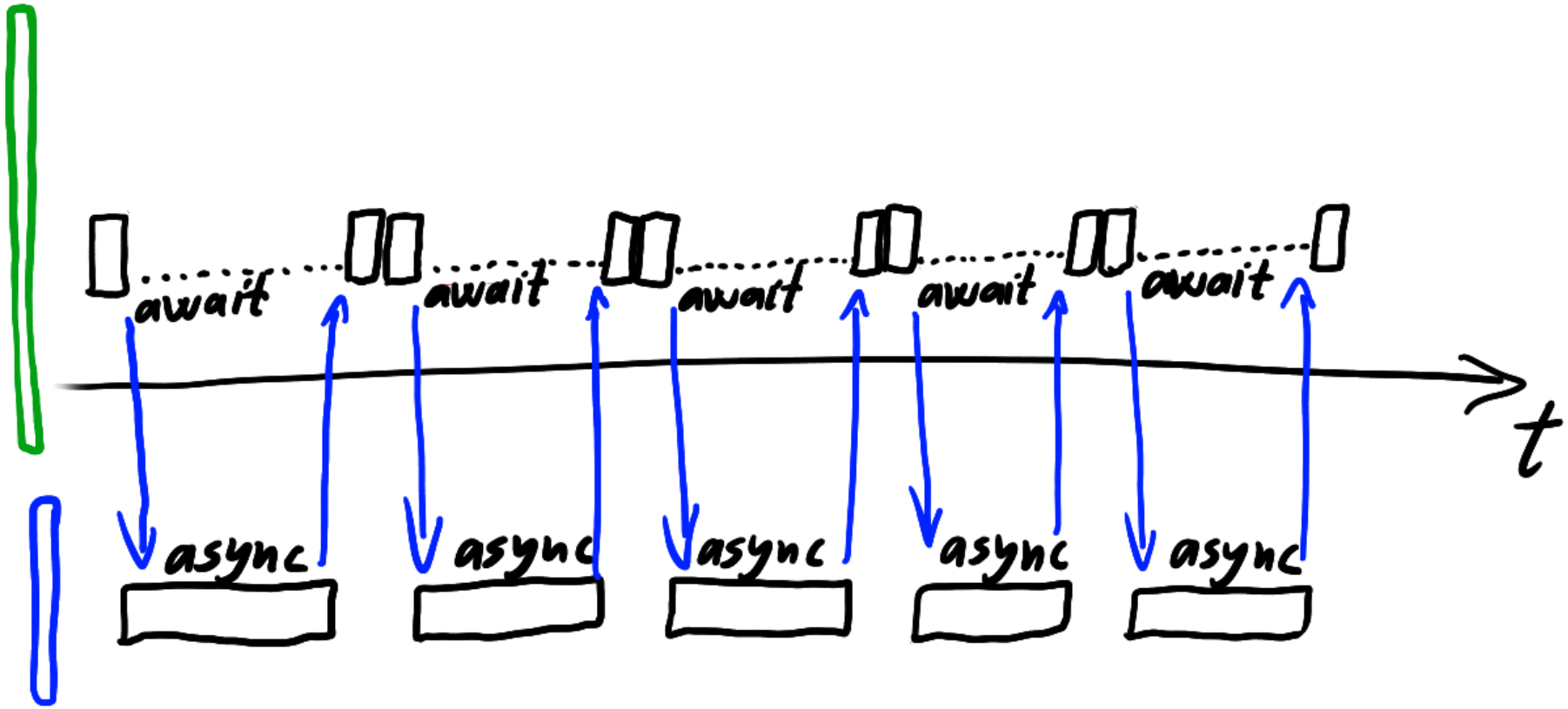
Az alkalmazás időben reagál mindenre. ☺

Egy szálú végrehajtás UI thread




Egy szálú végrehajtás
UI thread
.... A program nem válaszol! -|
async
async
async
async
async
await
await
await
await
await
t

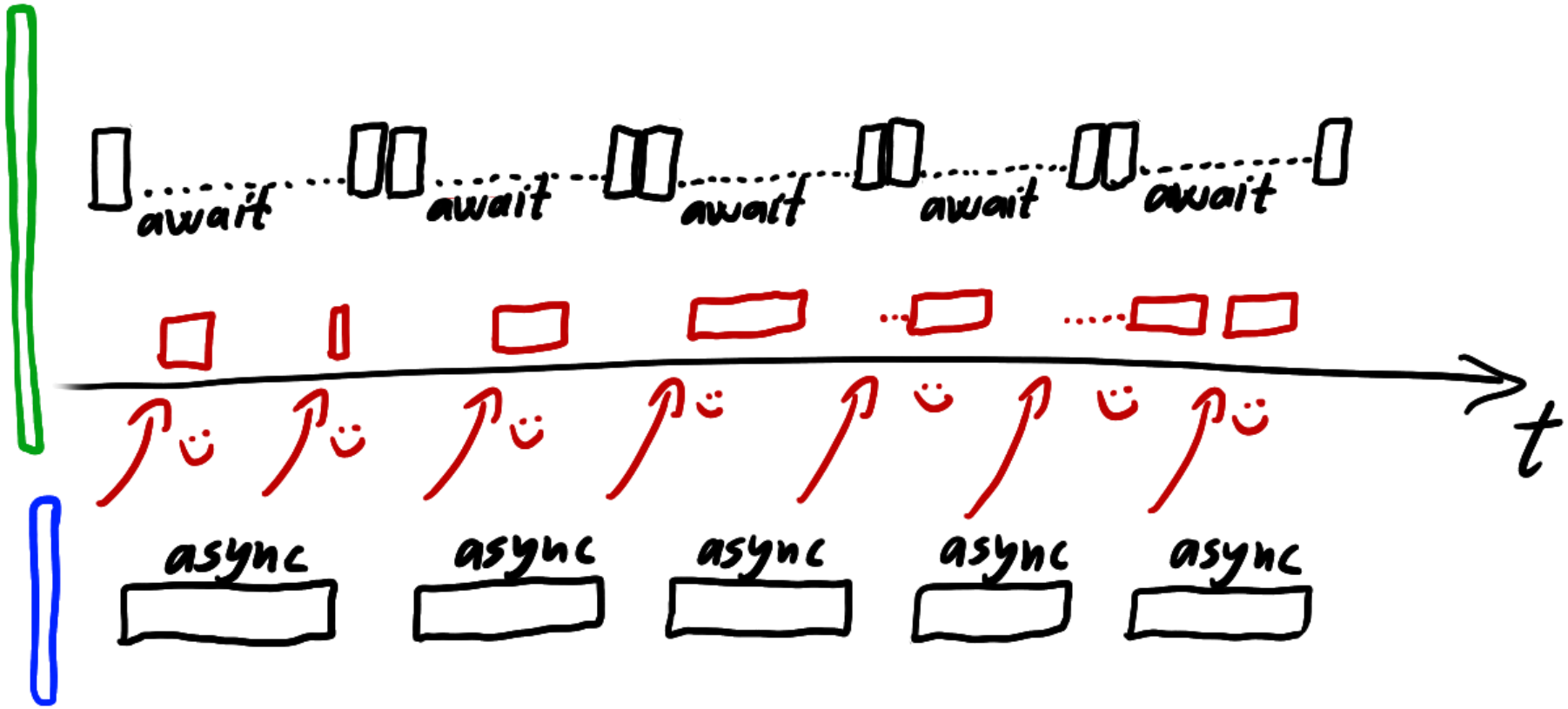
Többszálú végrehajtás UI thread Háttérszál



Többszálú végrehajtás

UI thread

Háttérszál



ThreadingLab mint példa...

UI thread, feladatai, background thread

- Webes kérések, nagy fájlok, számításigényes dolgok

Megéri 50 szálat indítani?

Kitekintés: WinRT (UWP)

- WinRT alatt minden async, ennek jelentősége és következményei.

“Thread safe”

- Példa többszálúsági gondokra
- Kritikus versenyhelyzetek
- Tipikus megoldás: lock
- https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/lock-statement

Kitekintés: funkcionális elvek

- Funkcionális programozás és többszálúság
- Linq AsParallel (Mandelbrot példa)