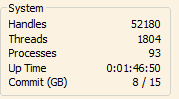
Kaj je to večnitnost(in preprost primer dveh niti v C#)?

Pri večnitnem programiranju uporabimo 2 ali več niti določenega procesa. Ker posamezni niti procesor(CPU) določi čas izvajanja, lahko na večjedernih procesorjih teče več niti hkrati. Takšnemu načinu izvajanja kode pravimo vzporedno procesiranje ali paralelizacija.

Pomembna razlika med večnitnim izvajanjem in vzporednim izvajanjem kode je, da se lahko večnitni proces izvaja tudi na enojedernem računalniku. V tem primeru ne gre za vzporedne procese, še zmeraj pa lahko dosežemo navidezno vzporednost, ker procesorska enota zelo dobro optimizira alokacijo izvajanja posameznih niti. V praksi na posameznem jedru tako teče več tisoč niti, ki si neprestano izmenjujejo procesorski čas. Posledica tega je, da lahko na enem računalniku uporabljamo več programov/procesov.



# Razlika med sinhronim in asinhronim programiranjem

Konec

C

A

B

Začetek

Konec

C

A

Začetek

B

Pri sinhronem programiranju se posamezni deli programa zaženejo v zaporedju eden za drugim

Pri asinhronem se ne izvajajo zaporedno, ampak se določeni deli izvedejo hkrati, kar v praksi pomeni da lahko naenkrat uporabljamo več funkcionalnosti programa. Načeloma se takšen program tudi hitreje izvede.

# Prednosti asinhronega(vzporednega) izvajanja programa

* Hitrejša koda
* Pri aplikacijah z uporabniškimi vmesniki lahko izvajamo druge procese medtem ko čakamo, da uporabnik vnese podatke
* Hitrost narašča s številom jeder/procesorjev

# Slabosti

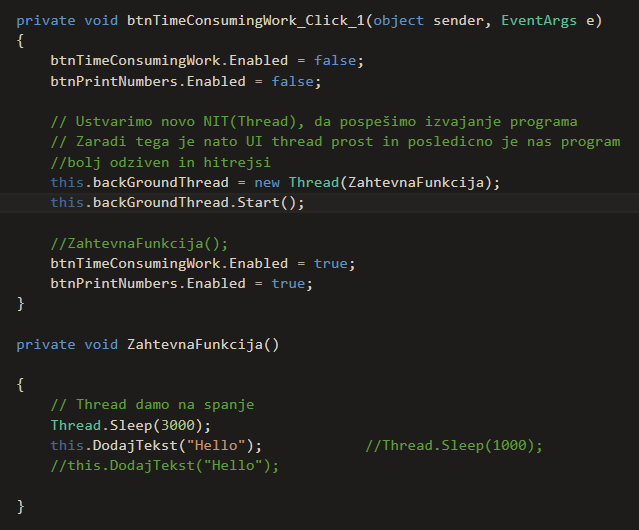
* Paziti je treba na sinhronizacijo niti, vzpostaviti mehanizem, ki skrbi da se določena niti začne izvajati šele, ko se je predhodna zaključila.
* Sočasnost deljenih virov, če dve niti bereta/spreminjata skupno datoteko, je treba paziti, da je ne dostopata hkrati.
* Na nižjem nivoju je težko predvideti kdaj se bo določena niti izvedla, ker si ne sledijo v logičnem vrstenm redu.

V praksi je v .net ogrodju poskrbljeno za nemoteno izvajanje večnitnih aplikacij z uporabo različnih pristopov.

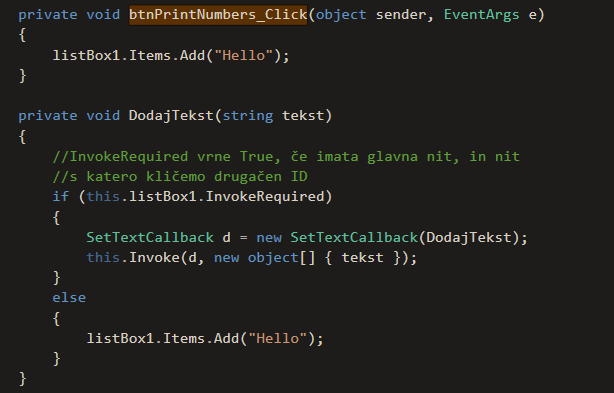
# Različne metode za pisanje večjedernih aplikacij

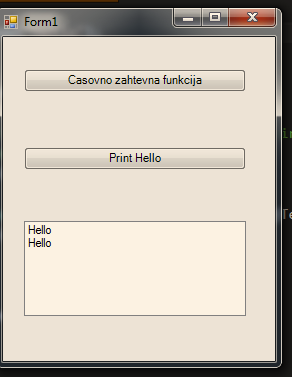
* PLINQ
* Background worker(Izvaja kodo na ločeni niti. Preko dogodkov([events](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/events/)) komunicira z glavno nitjo programa)
* TPL(Task parallel library)
* Tasks
* Threads(nižjenivojsko večnitno programiranje)

Poglejmo si primer Windows Forms aplikacije, ki prikaže delovanje 2 različnih niti, ki ju zažene uporabnik preko GUI vmesnika.



Uporabimo Thread objekt Threading imenskega prostora.





Ob kliku na zgornji gumb se zažene niti, ki po 3 sekundah izpiše “Hello” v okvir. Medtem pa lahko poljubno pritiskamo na spodnji gumb, ki se v Main niti izvaja nemoteno in izpisuje “Hello” v okvir.