

# Operacijski sistemi



Niti

# Vsebina

- Niti
- Zakaj niti?
- Izvedba niti
- Nitni izzivi

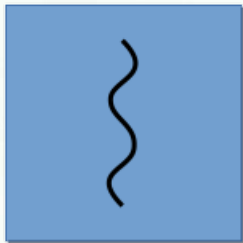
# Niti

- Glavni nalogi **procesov**
  - **lastništvo** oz. zaščita virov
  - **izvajanje** kode
  - neodvisnost obeh nalog
- Zakaj obeh nalog ne bi ločili?
  - procesi skrbijo za lastništvo virov
  - niti pa skrbijo za izvajanje kode
    - seveda brez virov ni mogoče izvajati kode

# Niti

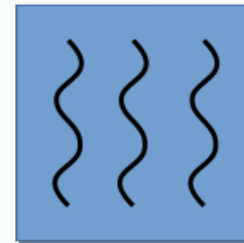
- Enonitnost in večnitnost
  - (single- vs. multi- threading)

en proces: ena nit na proces



npr. Microsoft DOS

en proces: več niti na proces



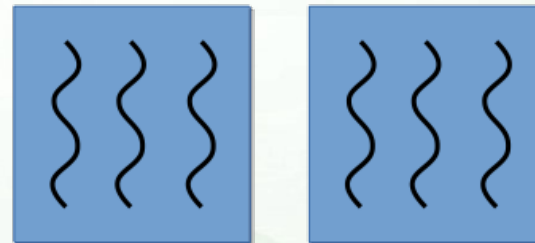
npr. Java VM

več procesov: ena nit na proces



npr. nekateri Unixi

več procesov: več niti na proces




npr. Microsoft Windows, Solaris,  
moderni Unixi



# Niti

- Kaj je nit?
  - skrbi za **izvajanje kode**
  - ima svoj **izvajalni kontekst**
    - stanje, registre in sklad
  - pripada nekemu procesu
    - vse niti procesa imajo **isti naslovni prostor**
    - in tudi ostale vire, npr. odprte datoteke



lahkokategorni  
proces

# Zakaj niti?

- Prednosti večnitnosti
  - ustvarjanje in končanje niti je hitrejše
    - brez inicializacije naslovnega prostora
  - učinkovit preklop niti
    - preklop med nitmi istega procesa je bolj učinkovit kot preklop med procesa
    - naslovnega prostora ni potrebno preklopiti, ni potrebno prazniti TLB
  - učinkovita medprocesna komunikacija
    - niti si delijo pomnilnik

# Zakaj niti?

- Prednosti večnitnosti
  - boljši izkoristek več-procesorskega sistema
  - boljša izkoriščenost virov
    - skupni naslovni prostor oz. deljenje pomnilnika
    - manjša poraba pomnilnika
    - cenejša sinhronizacija
  - specializacija in modularnost
    - vsaka nit niti skrbi za svoje opravilo
    - asinhrono procesiranje

# Zakaj niti?

- Primeri uporabe
  - spletni brskalnik
    - nit za upodabljanje strani, niti za nalaganje podatkov
  - urejevalnik besedila
    - upodabljanje, odziv na uporabniške akcije, preverjanje slovnice
  - spletni strežnik
    - glavna sprejemna nit, nit za vsako zahtevo
  - večnitnost v jedru OS
    - niti za rokovanje prekinitev, nit za menjavo strani itd.



# Izvedba niti

- Izvedba niti na **jedrnem nivoju**
  - OS razume in skrbi za niti 😊
  - podpora nitim preko sistemskih klicev
  - razvrščevalnik razvršča niti (tako kot procese)
  - niti lahko izkoriščajo več procesorjev
  - OS poskrbi za morebitno sinhronizacijo
  - sistemski klici, ki blokirajo, niso problematični
    - blokira le ustrezna nit

# Izvedba niti

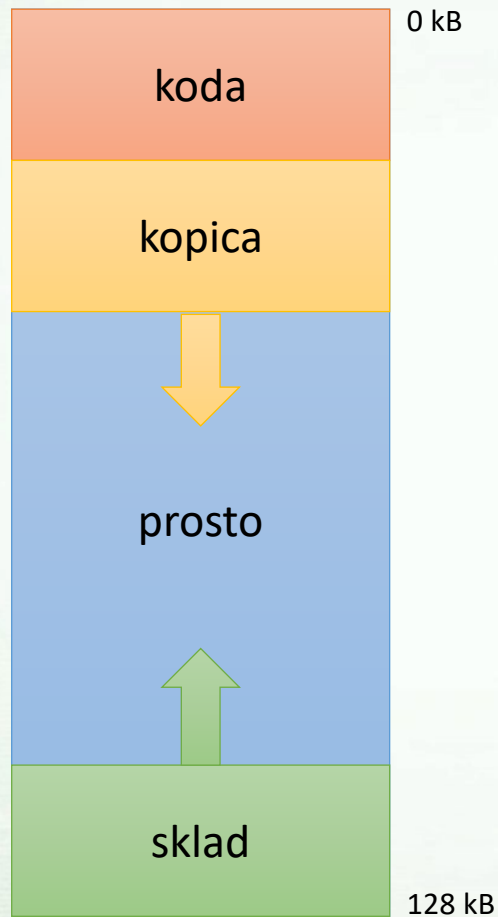
- Izvedba niti na **uporabniškem nivoju**
  - upravljanje niti je izvedeno v uporabniškem programu
    - uporabe preko uporabniške knjižnice za nitenje
    - razvrščanje niti je lahko prilagojeno aplikaciji
    - lahko tečejo v kateremkoli OS
    - hitrejši preklop med nitmi (vse se zgodi na uporabniškem nivoju)
  - OS se teh niti ne zaveda
    - če ena nit blokira, potem blokira vse niti (celoten proces)
    - vse niti so izvedene v sklopu enega samega procesa
    - nezmožnost izkoriščanja več procesorjev in razvrščanja
    - nekatere slabosti je možno zaobiti s posebnimi tehnikami
      - jacketing: pretvorba blokirnega klica v ne-blokirnega

# Izvedba niti

- Izvedba niti na **uporabniškem nivoju**
  - preslikava uporabniških niti v jedrne
    - **več-v-eno**
      - več uporabniških niti se preslika na eno jedrno nit
      - popolnoma uporabniške izvedbe
    - **več-v-več**
      - več uporabniških niti se preslika v manj ali enako število jedrnih niti
      - npr. IRIX, HP-UX, Tru64 Unix, Solaris <9
    - **ena-v-eno**
      - ena uporabniška nit upravljana s strani ene jedrne niti
      - Windows, Linux, Solaris 9+
    - **ena-v-več**
      - raziskovalni OS, niti se lahko selijo med procesi

# Izvedba niti

- Večnitni naslovni prostor





# Izvedba niti

- Mehanizem za izvedbo niti
  - **deskriptor niti** (kontrolni blok niti)
    - podatkovna struktura (v jedru), ki hrani podatke v zvezi z nitjo
    - registri, stanje niti, sklad, PC
    - računovodske info. v zvezi z nitjo
  - souporaba pomnilnika
    - navidezni pomnilnik: enaka preslikava iz navideznega prostora v fizični pomnilnik

# Izvedba niti


- Preklop niti
  - podobno preklopu procesa
  - ni zamenjave naslovnega prostora
  - ni praznenja preslikovalnega pomnilnika (TLB)
- Stvaritev niti

| Operacija   | Uporabniške niti | Jedrne niti | Procesi |
|-------------|------------------|-------------|---------|
| null fork   | 34               | 948         | 11300   |
| signal wait | 37               | 441         | 1840    |

null fork      stvaritev, razvrščanje, izvedba, končanje klica prazne funkcije  
signal wait    čakanje na izvedbo ustvarjene niti/procesa

# Nitni izzivi

- Kaj naredi `fork()` v večnitnem procesu?
  - kloniranje vseh niti procesa
  - kloniranje samo klicoče niti
- Kaj naredi `exec()` v večnitnem procesu?
  - nadomesti celoten proces, torej vse niti



Kaj pa zagon novega programa v otroku:  
`fork()` & `exec()`

# Nitni izzivi

- Pokončanje niti
  - končanje niti preden zaključi opravilo
  - **asinhrono končanje**
    - niti se konča takoj
    - Java: `Thread.stop()` ... deprecated
  - **odloženo končanje**
    - niti se kočna, ko zaključi trenutno opravilo
    - navadno vsebuje preverjanje ali se mora končati
- sproščanje zasedenih virov
  - Kdo sprosti vire, če se niti asinhrono konča?



# Nitni izzivi

- Rokovanje signalov
  - Katera nit rokuje signal?
    - tista, ki je vzrok, za pošiljanje signala
    - vse niti procesa
    - samo izbrane niti procesa
    - izbrana niti rokuje vse signale

# Nitni izzivi

- Število niti in bazeni niti (thread pool)
  - strežnik: ena nit na zahtevo
  - veliko zahtev → veliko niti
  - lahko povzroči preobremenitev sistema
  - rešitev: bazen niti (thread pool)
    - končno število (delavskih) niti (worker thread)
    - po končanju opravila se ponovno uporabijo
    - niti so vnaprej pripravljene
    - ni ustvarjanja in sproščanja niti