

# Operacijski sistemi



Niti – API

# Vsebina

- Java
- Pthreads
- Windows

# Java

- Izvedba niti

- uporaba vmesnika `Runnable`
- prekrijemo metode `run()`

```
public interface Runnable {  
    public void run();  
}
```

- Ustvarjanje niti

- uporaba razreda `Thread`
  - ime niti: `getName()` / `setName()`
  - prioriteta niti: `getPriority()` / `setPriority()`

# Java

- **Neposredno** ustvarjanje niti
  - **razširimo** razred `Thread` in povozimo metodo `run()`

```
class MojaNit extends Thread {  
    public void run() {  
        // opravilo  
        . . .  
    }  
}
```

- ustvarimo nit: ustvarimo objekt

```
MojaNit nit1 = new MojaNit();  
MojaNit nit2 = new MojaNit("Nitka");
```



# Java

- **Posredno ustvarjanje niti**

- ustvarimo razred, ki **izvede** `Runnable` in povozimo `run()`

```
class MojaNit implements Runnable {  
    public void run() {  
        // opravilo  
        . . .  
    }  
}
```

- ustvarimo nit: razredu `Thread` podtaknemo razred, ki izvede nit

```
Thread nit1 = new Thread(new MojaNit());  
Thread nit2 = new Thread(new MojaNit(), "Nitkica");
```

# Java

- Zakaj oboje (neposredno in posredno) deluje?
  - pogledamo v `Thread.run`

```
public class Thread implements Runnable {  
    private Runnable target;  
  
    Thread (Runnable target) {  
        this.target = target;  
        . . .  
    }  
  
    public void run() {  
        if (target != null)  
            target.run();  
    }  
  
    . . .  
}
```

# Java

- Zagon niti
  - klic `nit.start()`
- Konec niti
  - konec metode `run()`
  - poskrbimo za prizanesljivo končanje

```
volatile private boolean running;  
  
public void run() {  
    running = true;  
    while (running) {  
        // ponavljajoče se opravilo  
    }  
}
```

# Java

- Čakanje niti, da se konča
  - klic `nit.join()`
  - izjema `InterruptedException`

```
try {  
    // kličemo join od niti na  
    // kateremo želimo počakati  
    nit.join();  
} catch (InterruptedException e) {  
    e.printStackTrace()  
}
```



# Java

- Spanje niti

- `Thread:static void sleep(long milisecs)`

- Prepustitev CPE

- `Thread:static void yield()`

# Pthreads

- Knjižnica `pthread.h`
  - zaglavje: `include <pthread.h>`
  - prevajanje: `gcc -lpthread koda.c`
- Konec niti
  - ko se konča funkcija, ki izvede nit ali
  - klic `pthread_exit()`
- Identiteta niti
  - funkcija `pthread_t pthread_self()`

# Pthreads

- Nit implementiramo s funkcijo oblike

- `void* nit(void* arg)`

```
void *nit1(void *arg) {  
    // id niti  
    printf("Sem %li in sem ena uboga nit .\n", pthread_self());  
}  
  
void *nit2(void *arg) {  
    // prenos niza  
    printf("Sem %li in imam %s\n", pthread_self(), (char*)arg);  
}  
  
void *nit3(void *arg) {  
    // prenos int-a  
    printf("Sem %li in imam %li.\n", pthread_self(), (long)arg);  
    pthread_exit((void*)42);  
}
```

identifikator  
niti

končanje niti z  
izhodnim statusom

# Pthreads

```
int main() {  
    // priprava atributov  
    pthread_attr_t attr;  
    pthread_attr_init(&attr);    // privzeti atributi  
    pthread_attr_setdetachstate(&attr PTHREAD_CREATE_JOINABLE);  
  
    // zagon niti  
    pthread_t t1, t2, t3;  
    pthread_create(&t1, &attr, nit1, NULL);  
    pthread_create(&t2, &attr, nit2, "Juhuhu poletje je tu.");  
    pthread_create(&t3, &attr, nit3, (void*)42);  
    pthread_attr_destroy(&attr);  
  
    // čakanje na niti, da se končajo  
    pthread_join(t1, NULL);  
    pthread_join(t2, NULL);  
    pthread_join(t3, NULL);  
}
```



# Pthreads

- Nit implementiramo s funkcijo oblike

- `void* nit(void* arg)`

```
void *nit1(void *arg) {  
    // id niti  
    printf("Sem %li in sem ena uboga nit .\n", pthread_self());  
}  
  
void *nit2(void *arg) {  
    // prenos niza  
    printf("Sem %li in imam %s\n", pthread_self(), (char*)arg);  
}  
  
void *nit3(void *arg) {  
    // prenos int-a  
    printf("Sem %li in imam %li.\n", pthread_self(), (long)arg);  
    pthread_exit((void*)42);  
}
```

identifikator  
niti

končanje niti z  
izhodnim statusom

# Windows

- Knjižnica `windows.h`
  - zaglavje: `include <windows.h>`
- Konec niti
  - ko se konča funkcija, ki izvede nit ali
  - klic `pthread_exit()`
- Identiteta niti
  - funkcija `pthread_t pthread_self()`

# Windows

```
DWORD WINAPI nit(LPVOID arg) {  
    ...  
}  
  
int main() {  
    DWORD ThreadId;  
    HANDLE ThreadHandle;  
  
    // stvaritev niti  
    ThreadHandle = CreateThread(  
        NULL,    // privzeta varnost  
        0,       // privzeta velikost sklada  
        nit,     // funkcija niti  
        42,      // argument funkcije niti  
        &ThreadId);    // vrne identifikator niti  
  
    // čakanje na nit  
    WaitForSingleObject(ThreadHandle, INFINITE);  
    CloseHandle(ThreadHandle);  
}
```