

## Programação III

Concorrência  
Jocélio Passos

joceliopassos@bol.com.br

### Concorrência - Conceitos

- Sebesta cap 12 e Deitel cap 15
- Conhecida também como Multithreading
- Multiprocessamento : Computador como mais de um processador
- Multitarefa : Processador executa **tarefas (programas) diferentes** sincronizando seu tempo de processamento. Exemplo : Formatando, Baixando arquivo, Compilando programa

2/6/2008

2

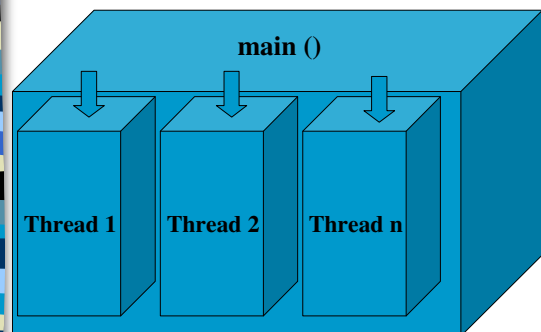
### Concorrência - Conceitos

- Thread : Execução simultânea (paralela) de **blocos de código** (subprogramas) do mesmo programa // Sebesta 495
- Multithreading : Processamento de mais de um thread
- Não é um conceito simples

2/6/2008

3

### Concorrência - Conceitos



2/6/2008

4

### Concorrência - Conceitos (Thread)

- “Imagine o **Sistema Operacional** como um **restaurante** e o **processador** da máquina como o **cozinheiro**. Para **cada programa** que executa (por exemplo, um aplicativo como o Excel), o Sistema Operacional **abre um processo** e coloca o código do programa em uma área de memória separada para ele. Seria **equivalente a uma mesa do restaurante**, quando um cliente a ocupa (uma área do restaurante que apenas ele usa enquanto faz a refeição).
- Um thread é um pedaço do programa (escrito em linguagem de máquina) que está sendo executado no processador. O thread seria o **ato** do cozinheiro de **executar** o pedido do cliente.”

2/6/2008

5

### Concorrência - Conceitos (Thread)

- “Todo processo obrigatoriamente tem um thread (ninguém vai a um restaurante para não pedir nada). Se você rodar um programa que suporte **múltiplos threads**, o aplicativo pode mandar vários threads para o Sistema Operacional simultaneamente. No nosso exemplo culinário, se o cozinheiro fosse **mono-threaded**, ele receberia o pedido da mesa e **executaria um prato por vez**, ou seja, quando terminasse de fazer um, começaria a fazer o outro. **Não seria muito mais eficiente se ele executasse todos os pratos ao mesmo tempo ?** Enquanto um está cozinhando, o outro já está sendo colocado no forno e assim por diante.”

2/6/2008

6

### Concorrência - Conceitos (Thread)

- “Isso é o que significa **multithreaded** : **rodar vários pedaços do programa simultaneamente dentro de um processo**. Cuidado para não confundir com multitarefa: esta é a capacidade de rodar vários processos ao mesmo tempo.
- Desta forma, um **gerenciador de banco de dados** pode **pesquisar vários pedidos de consultas no banco de dados ao mesmo tempo, sendo cada uma considerada um thread**. O Sistema Operacional sabe lidar com esses threads de forma a maximizar a performance.”

2/6/2008

7

### Concorrência - Conceitos (Thread)

- “Quem dispara a Thread ? O **programa em execução** precisa “**disparar**” esses threads para o Sistema Operacional. Essa é uma vantagem de se rodar programas escritos para um Sistema Operacionais multithreaded.
- Guarde bem estes conceitos: processo é um objeto criado quando um programa é executado ( por exemplo, um aplicativo como Excel ) e compreende um área de memória virtual unicamente para ele, e um ou mais threads de execução. O thread é uma porção do código do programa que está sendo executado e que pertence a um processo.”

2/6/2008

8

### Concorrência - Conceitos

- Vantagem : Métodos concorrentes **aumentam a flexibilidade** de programação.
- Aplicações
  - Problemas Particulares em Sistemas Operacionais
  - Simulação de sistemas físicos reais, múltiplos subsistemas concorrentes :
    - vôo de aeronaves em uma área controlada
    - Corpo Humano

2/6/2008

9

### Concorrência - Sincronização

- **Sincronização** : Mecanismo que controla a ordem de execução das tarefas
- **Cooperação** : quando a tarefa A precisa aguardar que a tarefa B conclua alguma atividade. Exemplo : produtor x Consumidor
- **Competição** : Tarefa A e B requerem o uso de algum tipo de recurso que não pode ser usado simultaneamente
- **Deadlock** : impasse que ocorre quando dois elementos de um processo estão aguardando que o outro responda. Nenhuma das tarefas renuncia ao recurso **Sebesta 474, Deitel 683 [771]**

2/6/2008

10

### Um programa Concorrente

- *// Show multiple threads printing at different intervals.*
- ```
public class ThreadTester {  
    public static void main( String args[] )  
    {  
        PrintThread thread1, thread2, thread3, thread4;  
        thread1 = new PrintThread( "thread1" );  
        thread2 = new PrintThread( "thread2" );  
        thread3 = new PrintThread( "thread3" );  
        thread4 = new PrintThread( "thread4" );  
        System.err.println( "\nStarting threads" );  
        thread1.start();  
        thread2.start();  
        thread3.start();  
        thread4.start();  
        System.err.println( "Threads started\n" );  
    }  
}
```

2/6/2008

11

### Um programa Concorrente

- ```
class PrintThread extends Thread {  
    private int sleepTime;  
    // PrintThread constructor assigns name to thread  
    // by calling Thread constructor  
    public PrintThread( String name )  
    {  
        super( name );  
        // sleep between 0 and 5 seconds  
        sleepTime = (int) ( Math.random() * 5000 );  
        System.err.println( "Name: " + this.getName() +  
            "; sleep: " + this.sleepTime + " ou "  
            + this.sleepTime / (float) 1000 + " seconds" );  
    }  
}
```

2/6/2008

12

## Um programa Concorrente

```
■ // execute the thread
■ public void run()
■ {
■     // put thread to sleep for a random interval
■     try {
■         System.err.println( this.getName() + " going to sleep");
■         Thread.sleep( this.sleepTime );
■     }
■     catch ( InterruptedException exception ) {
■         System.err.println( exception.toString() );    }
■     // print thread name
■     System.err.println( this.getName() + " done sleeping");
■ }
```

2/6/2008

13

## Execução de um programa Concorrente

```
Name: thread1; sleep: 2287
Name: thread2; sleep: 4354
Name: thread3; sleep: 4907
Name: thread4; sleep: 3316
```

```
Starting threads
Threads started
```

```
thread1 going to sleep
thread2 going to sleep
thread3 going to sleep
thread4 going to sleep
thread1 done sleeping
thread4 done sleeping
thread2 done sleeping
thread3 done sleeping
Normal Termination
```

2/6/2008

14

## Métodos de Manipulação de Threads

- Deitel 683 [771], Sebesta 495
- **Thread** - Classe do pacote java.lang
- **void run ()** - método que contém os comando que definem o que a thread fará.
- **void start ()** - método que dispara a thread, chamando o método **run ()**
- **void sleep (int milliseconds)** - método que especifica quanto tempo (em milisegundo) a thread deve "dormir"
- **boolean interrupt ()** - método que interrompe a thread
- **boolean isInterrupted ()** - método que retorna se a thread está ou não interrompida
- **System.err.println( "mensagem" );** Deitel 754, 756 [820, 822]

2/6/2008

15

## Exercícios

- 1) **Observe a Thread exemplo :**
  - C:\...\DelphiX\Demos\Threads
- 2) **Observe o Applet exemplo :**
  - C:\[diretório java]\demo\applets\SortDemo
- 3) **Observe o Jogo exemplo :**
  - <http://www.jogolandia.com.br/jogos/la-camarera/index.php>
  - <http://clickjogos.uol.com.br/Jogos-online/Acao-e-Aventura/La-Camarera/>

2/6/2008

16

## Exercícios Propostos

- 1) Explique os tipos de concorrência.
  - Nível de :
    - Instrução
    - Comando
    - Unidade
    - Programa
  - Sebesta 468
- 2) O que é time-sliced ? Sebesta 468
- 3) Qual a diferença entre concorrência física e a concorrência lógica ? Sebesta 470
- 4) Como seria o processamento de um array de 500 elementos de forma concorrente ? Sebesta 470

2/6/2008

17

## Exercícios Propostos

- 5) Como computadores de múltiplos processadores podem ser subutilizados ? Sebesta 471
- 6) O que é uma tarefa disjunta ? Sebesta 471
- 7) O que é vivência (liveness) ? Sebesta 474
- 8) "A maioria das linguagens de programação não permite que os programadores especifiquem atividades concorrentes". Verdadeiro ou Falso.
- 9) "Multithreading fornece capacidades poderosas para o programador Java também disponíveis em C e C++, as linguagens em que Java é baseada". Verdadeiro ou Falso.
- 10) Cite exemplos da utilidade prática de programação concorrente ? Deitel 682 (1º parágrafo) [771]

2/6/2008

18

### Exercícios Propostos

- 11) Qual a relação entre o garbage collection e concorrência ? [Deitel 682 \[770\]](#)
- 12) O que são threads de baixa prioridade ? [Deitel 682 \[770\]](#)
- 13) Que exemplo prático mostra a complexidade do processamento concorrente? [Deitel 682 \[770\]](#)
- 14) Estude a citação do Deitel, página 58, 2º parágrafo.
- 15) O que é round-robin ? [Deitel 683 \[771\]](#)
- 16) O que é preempção ? [Deitel 683 \[771\]](#)
- 17) Implemente o programa produtor/consumidor sem sincronização ? [Deitel 689 \[778\]](#)

2/6/2008

19

### Exercícios Propostos

- 18) Implemente o programa produtor/consumidor com sincronização ? [Deitel 693 \[783\]](#)
- 19) O que são threads daemon ? [Deitel 702 \[795\]](#)
- 20) Quando ocorre a exceção [IllegalThreadStateException](#) ? [Deitel 683 \[771\]](#)
- 21) O que acontece com a thread enquanto ela está "dormindo" ? [Deitel 683 \[771\]](#)
- 22) Qual a função do scheduler ? [Sebesta 473, 496, Deitel 685 \[773\]](#)
- 23) "Quando um aplicativo Java inicia sua execução, um novo thread é criado (no qual o método run rodará) e main é chamado. Portanto, todos os programas Java rodam em threads". Verdadeiro ou Falso ?

2/6/2008

20