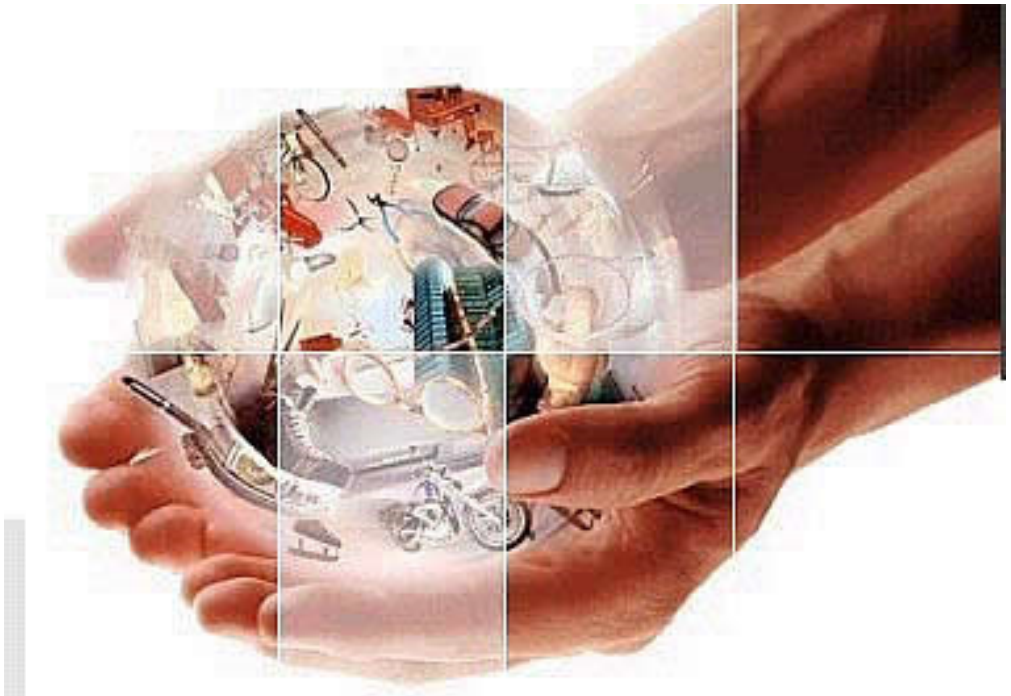
MBA em Desenvolvimento de Aplicações para Dispositivos Móveis

Desenvolvimento de Aplicações Android



Prof. MSc. Marcos Macedo
marcos@synapsystem.com.br

Versão 3.2 – Setembro/2013

ANDROID

MultiThreading

Atividades – Individual

Jogo de Ações - Multi-Threading

FIAP

- **Escreva um aplicativo Android para implementar o núcleo de um jogo de investimento que opera da seguinte maneira:**

1. Suponha que **R$ 100** é colocado em uma conta em um fundo seguro.

2. Dois consultores financeiros conhecidos como **'GoodNews'** e ‘**BadNews**' são atribuídos à conta. Ambos os investidores operam de forma independente um do outro.

3. **GoodNews** tem uma estratégia consiste em:
   a. pensar um pouco sobre o que fazer (leva um segundo de cada vez), e
   b. tomar uma decisão de investimento que sempre resulta em um ganho. O ganho é aleatória, isto é, valor menor ou igual a R$ 100. O ganho é adicionado à conta para aumentar a sua valor.

4. **BadNews** opera de forma semelhante, porém suas decisões sempre produzem, não sabemos o por quê, um perda. A perda é no máximo de **R$ 100** e deve ser subtraído da conta do equilíbrio.

5. O objetivo do jogo é para ver qual estratégia será ***"bem sucedida"*** em primeiro lugar.
6. O jogo é limitado a um período de tempo não superior a **20 segundos**.
7. O jogo poderá terminar antes do tempo previsto, caso:
    a.) o saldo chega a **R$200** (o dobro da dotação inicial), ou
    b.) a conta está esgotada e não há mais dinheiro. Fornecer uma interface gráfica mostrando os seguintes elementos:
        i. c.) Tempo decorrido (20seg. max). Use um *ProgressBar* horizontal ou uma visualização semelhante para controle.
        ii. d.) O equilíbrio da conta. É um *ProgressBar* inicialmente no meio da escala. Ganhos movem o progresso para a direita (escala máxima), e as perdas reduzem o progresso enviando o marcador para a esquerda (0 ponto na escala relativa).
        iii. e.) O valor atual da conta.
        iv. f.) A sequência histórica escrita de ganhos e perdas.
        v. g.) Uma mensagem indicando o resultado do jogo.
        vi. h.) Botões: Começar / Encerrar.

Sinta-se livre para mudar qualquer item que melhora as especificações acima. Divirta-se!

Atividade – Desafio (estudos)

Multi-Threading - Cafeteria

FIAP

Nesta questão você deve implementar a simulação de um cafeteria como descrito a seguir. Em uma cafeteria, os consumidores competem pelos N banheiros disponíveis. Existe um quadro onde são colocadas as N chaves, uma para cada banheiro. Quando um consumidor chega na frente do quadro, ele pode pegar uma chave e usar o respectivo banheiro, e em seguida colocar a chave no quadro novamente. Se não houver nenhuma chave no quadro, o consumidor precisa esperar até um dos banheiros seja desocupado e a chave esteja disponível no quadro. Os consumidores executam baseados no pseudo-código a seguir:

```
print("Consumidor %d entrou na cafeteria.\n", id);
for (int i=0; i < K; i++)
{
        bebe café
        pega uma chave
        usa o banheiro
        coloca a chave de volta
}
print("Consumidor %d deixa a cafeteria.\n", id);
```

As operações "bebe café" e "usa o banheiro" apenas imprimem uma mensagem como "Consumidor X está tomando café" e em seguida faz a thread consumidor dormir entre 2 e 10 segundos. A simulação deve receber como entrada X (o número de consumidores), N (o número de banheiros/chaves disponíveis) e K (o mesmo do loop mostrado no pseudo--código). Você deve garantir que todos os consumidores consigam realizar suas operações e que nenhum dos consumidores use o mesmo banheiro ao mesmo tempo. Para cada consumidor deve existir uma thread no programa. Sua solução **não** deve usar blocos ou métodos ***synchronized*** nem travas explícitas e deve, ainda assim, garantir que não há condições de corrida, deadlocks, etc.

Multi-Threading – Agência de Contas

AgenteConta.java

```
1  package br.com.fiap ;
2
3  public class AgenteConta implements Runnable
4  {
5      private double saldo = 1000000.0 ;
6      private AgenteConta[ ] contas = new AgenteConta[ 3 ] ;
7
8      public AgenteConta( ) { }
9
10     public AgenteConta( AgenteConta c1, AgenteConta c2, AgenteConta c3 )
11     {
12         contas[ 0 ] = c1 ;
13         contas[ 1 ] = c2 ;
14         contas[ 2 ] = c3 ;
15     }
16
17     public void debitar( double valor )
18     {
19         if (valor <= saldo)
20         {
21             saldo -= valor ;
22         }
23         else
24         {
25             System.out.println( "Saldo Insuficiente" ) ;
26         }
27     }
28
29     public void creditar( double valor )
30     {
31         saldo += valor ;
32     }
33
34     public void transferir( AgenteConta c, double quantia )
35     {
36         c.saldo = c.saldo - quantia ;
37         this.saldo += quantia ;
38     }
```

Multi-Threading – Agência de Contas

FIAP

```
39
40      public void run( )
41      {
42          for (long i = 0; i < 1000000; i++)
43          {
44              contas[ 0 ].debitar( 1 ) ;
45              contas[ 1 ].debitar( 1 ) ;
46              contas[ 2 ].debitar( 1 ) ;
47              this.creditar( 3 ) ;
48              this.transferir( contas[ 0 ], 1 ) ;
49              this.transferir( contas[ 1 ], 1 ) ;
50              this.transferir( contas[ 2 ], 1 ) ;
51          }
52      }
53
54      public static void main( String[ ] args )
55      {
56          long t = System.currentTimeMillis( ) ;
57          AgenteConta c1 = new AgenteConta( ) ;
58          AgenteConta c2 = new AgenteConta( ) ;
59          AgenteConta c3 = new AgenteConta( ) ;
60          AgenteConta c4 = new AgenteConta( c1, c2, c3 ) ;
61          c1.contas = new AgenteConta[ ]  { c2, c3, c4 } ;
62          c2.contas = new AgenteConta[ ]  { c1, c3, c4 } ;
63          c3.contas = new AgenteConta[ ]  { c1, c2, c4 } ;
64          Thread t1 = new Thread( c1 ) ;
65          Thread t2 = new Thread( c2 ) ;
66          Thread t3 = new Thread( c3 ) ;
67          Thread t4 = new Thread( c4 ) ;
68          t1.start( ) ;
69          t2.start( ) ;
70          t3.start( ) ;
71          t4.start( ) ;
72          try
73          {
74              t1.join( ) ;
75              t2.join( ) ;
76              t3.join( ) ;
77              t4.join( ) ;
78          }
79          catch (InterruptedException e)
80          {
81              System.out.println( "Erro" + e.getMessage( ) ) ;
82          }
83          System.out.println( " C1: " + c1.saldo + " C2: " + c2.saldo + " C3: " + c3.saldo + " C4: " + c4.saldo ) ;
84          System.out.println( " Tempo: " + ( System.currentTimeMillis( ) - t ) ) ;
85      }
86  }
87
```

Multi-Threading – Agência de Contas

FIAP

a-) Modifique a implementação da classe AgenteConta utilizando **somente travas explícitas** de modo que, ao final da execução, suas quatro instâncias tenham o mesmo saldo(1.000.000,00). É proibido modificar o método *main()* e é proibido também usar trava ou monitor global único para todas as *threads*. Também é proibido mudar a lógica da aplicação (i.e., cada instância de AgenteConta deve continuar debitando das três outras contas, creditando para si e, em seguida, transferindo uma unidade para cada uma das outras contas) e a sequência de operações realizadas no método *run()*.

Obrigado


धन्यवाद
Hindi
多謝
Traditional Chinese
Grazie
Italian
ขอบคุณ
Thai
Gracias
Spanish
Merci
French
Спасибо
Russian
شكرا
Arabic
Obrigado
Brazilian Portuguese
Danke
German
多谢
Simplified Chinese
நன்றி
Tamil
ありがとうございました
Japanese
감사합니다
Korean

Referências Bibliográficas

FIAP

- Android Developer's Website

- Google Maps API external library
  http://code.google.com/android/add-ons/google-apis/maps-overview.html
- MightyPocket
  http://www.mightypocket.com/2010/08/android-screenshots-screen-capture-screen-cast/
- Numerous Forums & other developer sites, including:
  http://www.javacodegeeks.com/2011/02/android-google-maps-tutorial.html
  http://efreedom.com/Question/1-6070968/Google-Maps-Api-Directions
  http://www.mail-archive.com/android-developers@googlegroups.com/msg28487.html
  http://android.bigresource.com/ threads
  http://groups.google.com/group/android-developers threads
  Many http://stackoverflow.com threads
  http://www.anddev.org/google_driving_directions_-_mapview_overlayed-t826.html
- Zainan Victor Zhou – for advice and his own tutorial

Referências Bibliográficas

Google Android – 3ª edição
Ricardo R. Lecheta

Android para Desenvolvedores
Lúcio Camilo Oliva Pereira

Desenvolvimento de Aplicações Android
Rick Rogers
John Lombardo
Zigurd Mednieks
Blake Meike

FIAP

A MELHOR FACULDADE DE TECNOLOGIA

www.fiap.com.br - Central de Atendimento: (11) 3385-8000

Campi: Aclimação I Aclimação II Paulista Alphaville

Copyright © 2013 Prof. MSc. Marcos Macedo

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proíbido sem o consentimento formal, por escrito, do Professor (autor).