



RAD STUDIO LIVE 2022
2 DIAS DE RAD STUDIO EM PORTUGUÊS
23 E 24 DE FEVEREIRO

 PPL - Parallel Programming Library
(Marcelo Varela - MVP)

embarcadero®

USANDO PARALELISMO EM SUAS APLICAÇÕES



MARCELO VARELA - MINI CURRÍCULO

- Embarcadero MVP (Most Valuable Professional)
- Principais Certificações
 - Delphi Developer Certification 7/2005/2006/2007/XE
 - Master Delphi Developer Certification
 - Delphi Instructor Certification
- Palestras e eventos
 - Delphi Conference Brasil de 2009 a 2021
 - BonCon 2005, 2006 e 2007
- Atividade profissional
 - 20 anos em experiência em desenvolvimento de software
 - Professor de Sistemas de Informação do IFRN
 - Mestre em Engenharia de Software na UFRN/IMD
 - Pós-Graduado em Desenvolvimento de Sistemas Web pela UnP
 - Bacharel e Sistemas de Informação
- Artigos
 - ActiveDelphi, Linha de Código e iMasters.

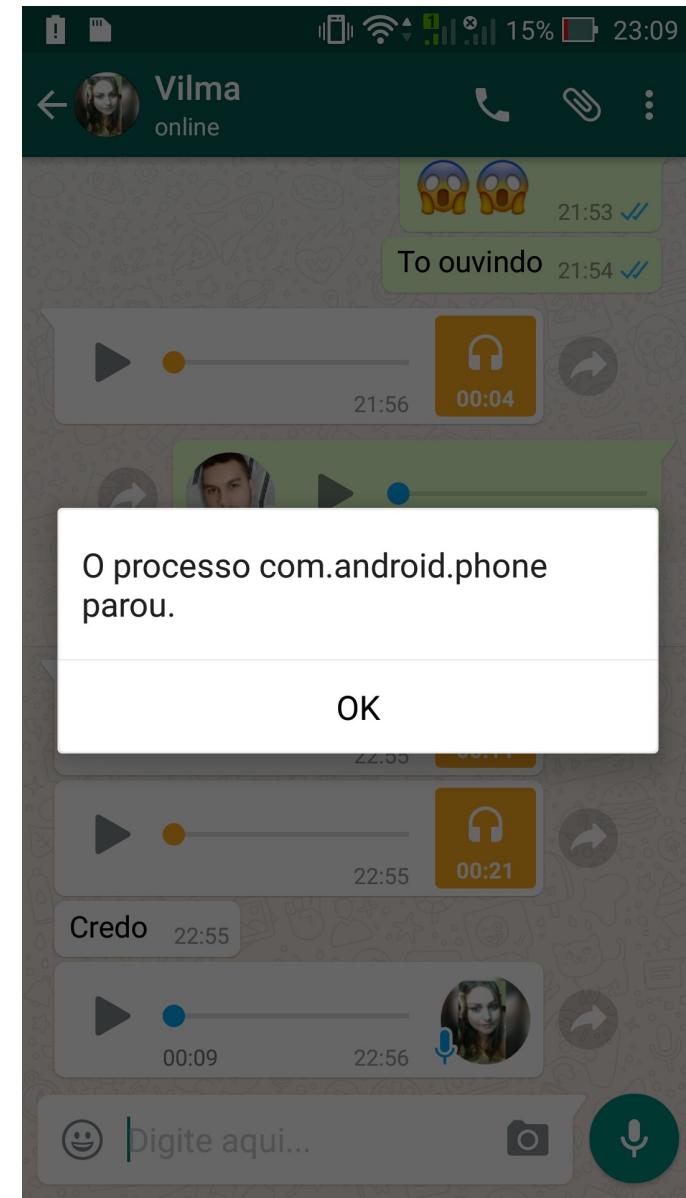
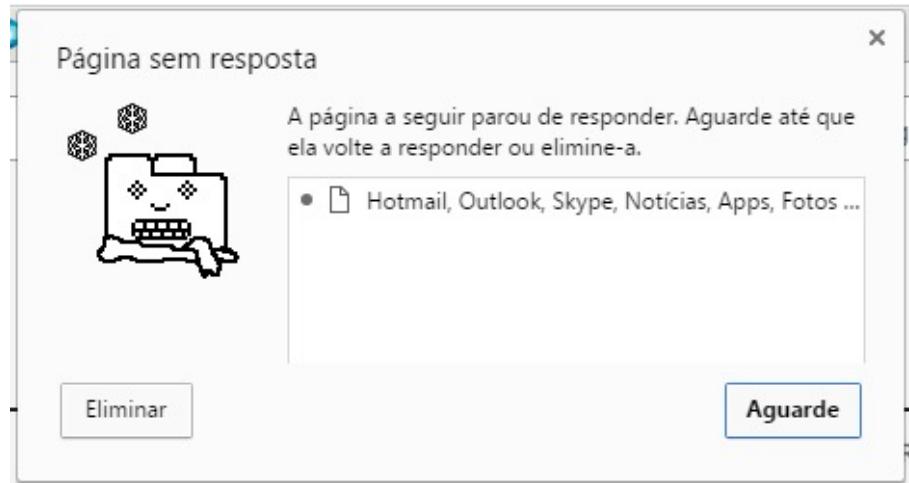


AGENDA

- Fundamentos
- Parallel Programming Library
 - Task
 - Future
 - For
 - REST Client ExecuteAsync
 - OmniThreadLibrary
- Dicas
- Conclusão



O QUE VOCÊ ACHA DISSO?



POR QUE ESCREVER CÓDIGO ASSÍNCRONO?

- Operações assíncronas no nosso código são importantes quando desejamos executar tarefas que podem ser demoradas ou de segundo plano, sem que afetem a experiência de interação do usuário com a interface do nosso software.
- Operações assíncronas nos possibilitam ter uma interface gráfica responsiva, que não fica bloqueada esperando alguma operação terminar, ou impossibilitando que o usuário continue seu trabalho.
- Interfaces que travam, podem aparentar lentidão no processo, cancelamento desnecessário ou mesmo retrabalho do usuário, além de passar uma impressão ruim sobre seu software.



EXEMPLO DE QUANDO PARALELIZAR

- Um dos exemplos mais comuns é a execução de tarefas longas na *thread* principal. No Delphi, a *thread* principal é responsável por gerenciar a interface gráfica do usuário.
- Se estiver executando alguma tarefa longa e não processando eventos da interface do usuário (troca de mensagens do Windows, por exemplo), a interface do usuário ficará bloqueada.
- Podemos resolver esse problema movendo a tarefa longa para um novo segmento de código em segundo plano, que permitirá que o *thread* principal gerencie a interface do usuário.
- Um programa que não trava faz o usuário feliz.



PARALLEL PROGRAMMING LIBRARY

- É uma biblioteca, disponibilizada a partir da versão XE7, que torna o trabalho do desenvolvedor mais simples no que se refere a programação paralela
- Ela promove uma facilitação na criação e gerenciamento de novos segmentos dentro do seu processo principal. É como trabalhar com Thread, só que mais fácil e poderoso
- Multiplataforma
- System.Threading
- Task, Future e For (loop)



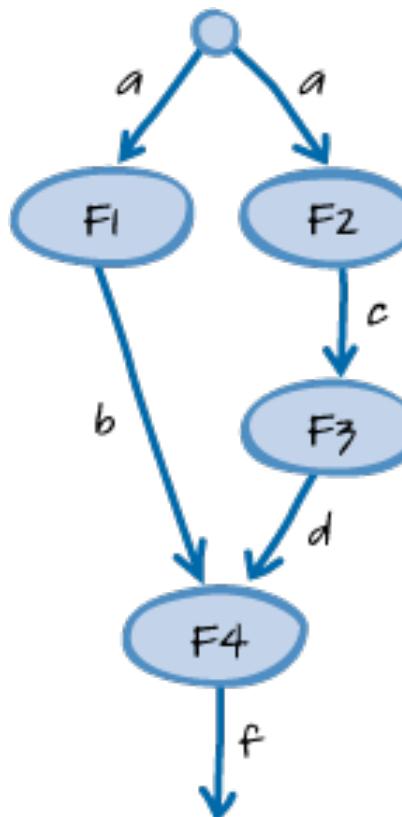
TASK

- Uma Task (tarefa) funciona como *procedure* que roda em paralelo, ou seja, uma rotina sem retorno.
- Uma Task não inicia imediatamente após a chamada do método Start, elas são enfileiradas e iniciadas quando há disponibilidade na CPU.



FUTURE

- Uma Future (tarefa futura) funciona como *function* que roda em paralelo, ou seja, uma ronina com retorno.
- Veja um exemplo:
 - `a := 10;`
 - `b := F1(a);`
 - `c := F2(a);`
 - `d := F3(c);`
 - `f := F4(b, d);`
 - `Result := f;`



FOR

- Um For, usando a PPL, divide um laço FOR em pedaços que rodam em paralelo.
- A chamada de um Parallel.For não é assíncrona, por isso, se desejar evitar o efeito *freeze* de tela, use em conjunto com uma Task.



DEMONSTRAÇÃO



DICAS

- Cuidado com a concorrência de recursos
- Preocupe-se em deixar sua aplicação responsive e agradável para o seu usuário
- Use sempre o Synchronize/Queue quando for acessar/modificar componentes



CONCLUSÃO

- A PPL é uma biblioteca pronta para você usar os recursos de paralelismo e multicore.
- Você não tem mais desculpas para criar aquelas Apps com a tela travando!
- Referências:
 - <https://docwiki.embarcadero.com/>
 - <http://www.omnithreadlibrary.com/>
 - Livro: Delphi High Performance, de Primož Gabrijelčič





marcelologica



linkedin.com/in/marcelovarelasouza



@marcelologica



marcelo.varela.9

