



THE DEVELOPER'S CONFERENCE

Trilha – Delphi

Anderson Dapper Rocha

Desenvolvedor Delphi na Server Softwares Para Varejo

Quem é esse cara?



- Técnico em Informática
 - Comecei a programar aos 14 anos de idade em Clipper
 - Trabalho a 17 anos com Delphi
- Técnico em Contabilidade
- Trabalho com ERP e Automação comercial desde quando conheci o Delphi
- Atualmente sou desenvolvedor na Server Softwares para Varejo, uma das maiores no ramo calçadista.



THE
DEVELOPER'S
CONFERENCE

Criando Serviços Linux com Delphi



Apoio

SERVER
SOFTWARES PARA VAREJO

O que é um serviço Linux?



- Uma aplicação, que roda em background (daemon), e possui um script para inicialização durante o boot ou manualmente.

“Vim para ouvir como se cria um script para iniciar uma aplicação!?!?!?”

Requisitos de um DAEMON



- Desassociação do tty (terminal) de controle
- Tornar-se líder de uma sessão
- Tornar-se líder de um grupo de processo
- Executar como uma tarefa de plano de fundo por bifurcação (fork) e saída (uma ou duas vezes).
- Definir o diretório raiz (/) como o diretório de trabalho atual. (Evitar que o processo mantenha qualquer diretório em uso)
- Alterar a umask para 0 para possibilitar que open(), creat() e outras chamadas do sistema operacional forneçam suas próprias máscaras de permissão e não dependam da umask do chamador



- Fechar todos os arquivos herdados no momento da execução que são deixados abertos pelo processo pai, incluindo descritores de arquivo 0, 1 e 2 para os fluxos padrões (stdin, stdout e stderr). Arquivos requeridos serão abertos posteriormente.
- Usar um logfile, o console ou /dev/null como stdin, stdout e stderr

E agora?



- Posix.Daemons.pas
- Posix.Syslog.pas
 - Paolo Rossi (<https://github.com/paolo-rossi>)
- Linux.Utils.pas
 - Anderson Dapper (<https://github.com/toppermitz>)



```
uses
  System.SysUtils,
  Posix.Daemon,
  Posix.Syslog,
  Linux.Utils;

begin
  try
    TPosixDaemon.Setup(procedure(ASignal: TPosixSignal)
    begin
      //Tratar as sinalizações que um processo pode receber
      case ASignal of
        TPosixSignal.Termination:
          begin
            //Implementar o código para ser executado ao receber
            // o sinal de finalização (SIGKILL)
            TPosixDaemon.Stop;
          end;
        TPosixSignal.Reload:
          begin
            //Implementar o código que deve ser executado ao processo
            // ser sinalizado para reload (SIGHUP)
          end;
        end;
      end;
    end,'meupid');

    //Criar uma thread nesse ponto ou implementar o segundo
    // parametro do TPosixDaemon.Run

    TPosixDaemon.Run(1000,procedure
      var
        LHostName : String;
      begin
        //Codigo a ser rodado no loop infinito
        LHostName := TLinuxUtils.RunCommandLine('hostname').Text.trim;
      end);

  except
    on E:Exception do
      begin
        TPosixDaemon.LogOpen;
        TPosixDaemon.LogError(E.ClassName + ' - ' + E.Message);
        TPosixDaemon.LogClose;
      end;
    end;
  end;
end.
```

As benditas units

Setup de sinalizações

O core da nossa aplicação

Tratamento mínimo de
exceção



Fontes



➤ <https://github.com/toppermitz/Linux-TDC2019/>

Dúvidas??



AVALIAÇÃO DE PALESTRAS



THE
DEVELOPER'S
CONFERENCE



<https://thedeveloperconf.com/tdc/2019/avaliacao>

Apoio

SERVER
SOFTWARES PARA VAREJO