

Nesta seção você encontra artigos intermediários sobre Delphi Win32 e Delphi .NET



Desenvolvendo uma Aplicação Completa com PocketStudio

Aprenda a criar aplicações para o sistema operacional PalmOS – Parte 1

A partir deste artigo, criaremos uma aplicação para PalmOS completa utilizando a ferramenta PocketStudio. Algumas considerações são importantes em relação à programação para Palm Top. Atualmente existem *Palms* com sistema operacional Windows Mobile chamados Pocket PC's e os dispositivos com PalmOS. Existem muitas diferenças entre eles, mas a principal a ser entendida, é justamente a questão do Sistema Operacional.

Nesse mini-curso entenderemos as diferenças entre os sistemas operacionais PalmOS e Windows Mobile, conceitos de desenvolvimento móvel e exemplos práticos.

Criaremos uma aplicação voltada ao mercado de vendas, onde os representantes poderão cadastrar novos Clientes, Produtos e Pedidos. Em seguida poderão enviar os dados a matriz através da aplicação gerada e instalada no dispositivo móvel.

Conceitos, história e diferenças

PalmOS não é Windows, muitos imaginam que o PalmOS é uma versão do Windows para a plataforma móvel. Terrível engano, pois a estrutura interna é totalmente diferente e as chamadas às APIs também. Palms são aparelhos com menos memória que os Pocket PC's, o que não significa queda de performance pois a API do Palm compensa a diferença. O sistema operacional PalmOS ocupa em torno de 4 MB da memória, contra os mais de 200MB do Windows Mobile.

O custo dos portáteis equipados com PalmOS é bem menor que os com Windows Mobile. Certamente isso pesa na balança na hora em que empresas optam por informatizar sua equipe de vendas. Com o PocketStudio desenvolvemos aplicações para PalmOS, isso precisa estar bem claro. Um sistema desenvolvido com o PocketStudio não roda na API do Pocket PC.

PocketStudio não é Delphi, apesar de



Ricardo C. Boaro

(rboaro@aquasoft.com.br)

trabalha com desenvolvimento de sistemas em Delphi há mais de 10 anos e PocketStudio há 3 anos. Atualmente é gerente de informática na Di Hellen Indústria de Cosméticos, e atua como instrutor certificado Borland na Aquasoft Tecnologia da Informação parceira da Borland, em Porto Alegre – RS. Borland Instrutor, Delphi 7, 2007 e Certified.

algumas semelhanças com o IDE do Delphi 7 e anteriores, a forma de programar é diferente. Enquanto em sistemas desenvolvidos em Delphi trabalhamos com a linguagem Object Pascal, no PocketStudio usamos o Pascal puro. Após alguns esclarecimentos importantes vamos aos passos para instalação do PocketStudio e do PalmOS Emulador.

Instalando o emulador

Para testarmos nossas aplicações podemos instalar o PalmOS Emulador da empresa Access. O Emulador é provido de um arquivo executável, imagens de dispositivos e skins para tornar ainda mais atrativos os testes. Vamos efetuar o download do emulador pelo site Palm Brasil, que contém uma grande quantidade de programas e utilitários para Palm. O portal é parceiro da empresa Access mencionada anteriormente e mantém em sua área de downloads o PalmOS Emulador bem como imagens ("ROM's") dos aparelhos e skins.

Acesse o link www.palmbrasil.com.br e clique no link *Programas* na seção *downloads*. Logo em seguida abra o link *Palm Desktop e Simulador* na seção *Utilitários*. Na página 2 você encontrará o item *PalmOS Emulador 3.5* para download e mais abaixo as ROM's para download. Se preferir acesse diretamente o link www.palmbrasil.com.br/programas/palmos2.html.

Para efetuar o download é necessário ter cadastro e estar devidamente logado no site da Access. Para se cadastrar acesse o link <https://www.developerpavilion.com/palmos> e logo em seguida em *Register*. Preencha seus dados e faça o login. Logo após retorne ao site da Palm Brasil e clique nos links para download.

Você fará o download do arquivo *emulador-win.zip*. Também é interessante baixar o arquivo de *skins*, *PalmOS Emulador*

Skins 1.9 ("emulator-skins-19.zip"), presente nessa mesma página. Os *skins* são usados para visualizar o aparelho no emulador e no ambiente de desenvolvimento do PocketStudio como veremos mais adiante. Após ter feito o download de todos os arquivos necessários, descompacte-os na pasta criada e execute o arquivo *Emulador.exe*.

No caso dos *skins*, uma pasta *Skins_v1.9* será criada ao descompactar. É necessário renomear a pasta *Skins*. Nas subpastas desse diretório existem alguns arquivos de imagem em formato JPEG. Copie-os para a pasta *Skins*. Eles serão usados para mostrar o dispositivo no ambiente de desenvolvimento. Feito isso execute o arquivo *Emulador.exe*, ele lhe perguntará se deseja criar um atalho no menu *Programas*. Confirme e indique em qual pasta deseja que o emulador crie o atalho.

Vamos falar um pouco sobre o emulador que servirá como ambiente de testes em nosso *Desktop*. Vamos iniciá-lo e dar uma olhada nas configurações necessárias para o funcionamento correto de nossa aplicação. A primeira vez que o iniciarmos, uma tela de configuração é exibida (**Figura 1**).

O título da janela *New Session* indica que a partir dela criaremos uma nova seção para trabalharmos em paralelo a nossa aplicação. Seção nesse caso serão todas as configurações que fizermos na tela de abertura e tudo que gravarmos no emulador, como por exemplo a nossa aplicação. Nos exercícios que desenvolvermos nos próximos artigos ficará bem entendido o que é e como utilizar uma seção no emulador. Na **Figura 1** podemos ver quatro opções de configuração:

- **ROM file:** ROM é o modelo do dispositivo que iremos emular. Para cada modelo

de Palm existe uma ROM para utilizarmos no emulador. A ROM que utilizaremos em nosso exercício esta gravada dentro do diretório de instalação do Emulador em uma pasta *ROM's*. Para a nossa aplicação selecione a ROM mostrada na figura. *Platform41_m505_enUS.rom*;

- **Device:** em *Device* irão aparecer todos os equipamentos Palm compatíveis com a ROM selecionada. Para o nosso exemplo escolha *Palm m505*;

- **Skin:** são as aparências disponíveis para o modelo que selecionamos em *Device*. Selecione a opção *Standard-English*;

- **RAM size:** você pode escolher a quantidade de memória que deseja utilizar na seção. Também será variável de acordo com o modelo do Palm escolhido, para o nosso exercício selecione *16,384K*;

Após todas as opções selecionadas clicamos em *OK*. E o emulador é carregado (**Figura 2**) exatamente igual ao modelo do Palm selecionado. Ele possui todas as funções do Palm verdadeiro, até a escrita *graffite* funciona como se estivéssemos escrevendo com a caneta.



Figura 1. Primeira tela do Emulador

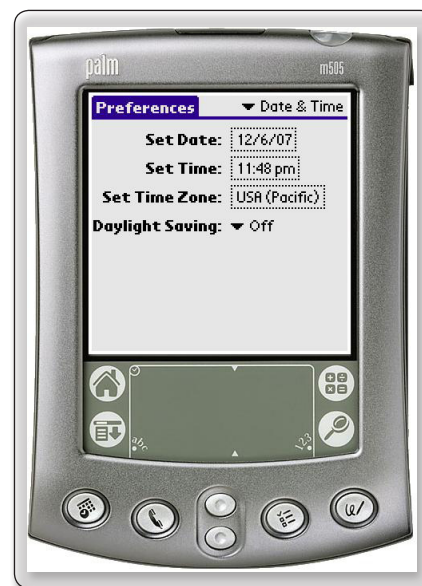


Figura 2. Emulador do Palm funcionando



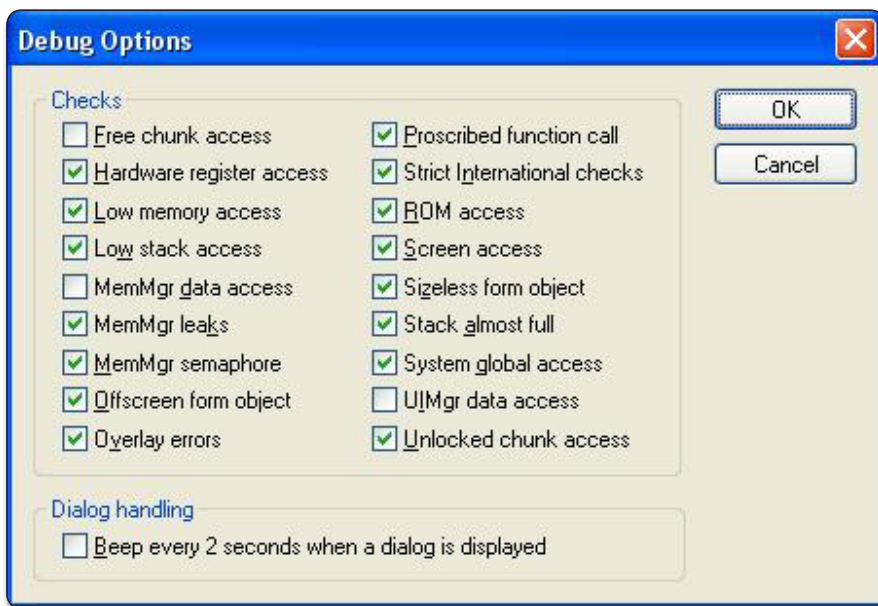


Figura 3. Configurações de Debug do Emulador



Figura 4. Menu principal e a paleta de componentes

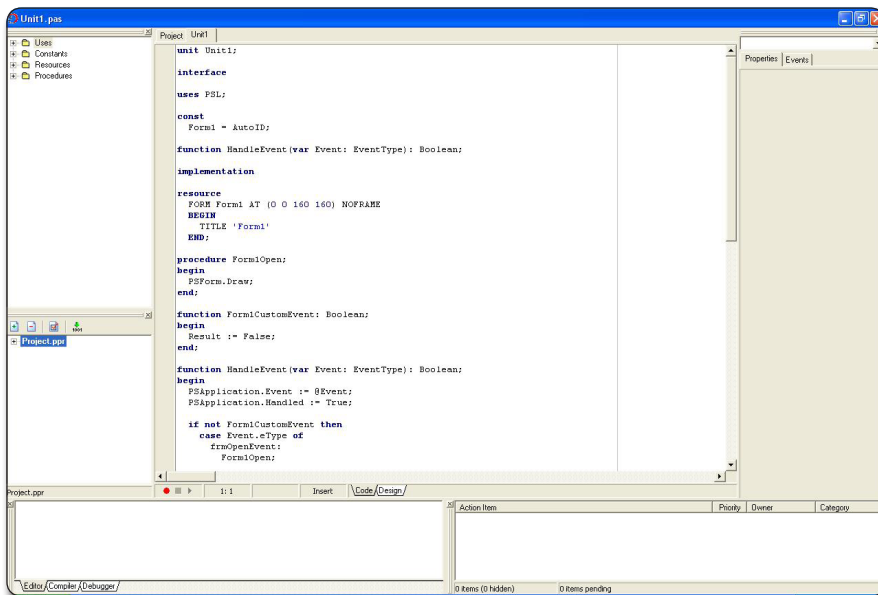


Figura 5. Janela principal do PocketStudio



Nota: A escrita Graffiti é um sistema de reconhecimento de escrita utilizado em todos os PDA's da plataforma PalmOS até a versão 4.1, baseado no alfabeto simples, onde em realidade não chega a ser um "reconhecimento da escrita" de verdade, pois o usuário precisa aprender as combinações de cada letra do alfabeto, símbolos, pontuações etc.

Antes de iniciarmos nossa aplicação é necessário que façamos mais algumas configurações em nosso emulador para evitarmos problemas. Clicando com o botão direito do mouse sobre ele navegue até a opção *Settings* e escolha o item *Debugging*. Aberta a caixa de diálogo desmarque os itens *Free chunk access*, *MemMgr data access* e *UIMgr data access* como mostrado na Figura 3.

Essas opções que desmarcamos, existem no emulador para futuras versões do sistema operacional do PalmOS, sendo assim, na versão atual precisam ser desmarcadas para evitar erros de *runtime* quando rodarmos o projeto no Pocket Studio. Muitos desenvolvedores perdem horas tentando encontrar o problema de sua aplicação, quando na verdade é apenas a configuração do emulador.

Instalando e conhecendo o PocketStudio

No site da WinSoft podemos baixar uma versão de avaliação do PocketStudio. Acesse o link www.winsoft.sk/pstudio.htm, encontre a seção *Products* e em seguida PocketStudio. No final da página encontre o link *PocketStudio 2.1 Standard 45-day trial version*. Será solicitado que entre com Nome, Empresa e E-mail. Os dados e instruções para *download* virão em seu endereço eletrônico, por isso informe um endereço válido.

A instalação é simples, basta executar o *Setup.exe*. Serão apresentadas algumas telas, confirmando o local da instalação, se será criado um atalho etc. Basta confirmar as mesmas, no final clicamos em *Finish* e o PocketStudio está instalado e pronto para ser utilizado. Recentemente instalei-o no Windows Vista e funcionou perfeitamente.

Vamos conhecer o IDE do PocketStudio

para depois iniciarmos o desenvolvimento de nossa aplicação. Nas **Figuras 4 e 5** podemos ver o menu principal juntamente com a paleta de componentes e ainda a janela principal da ferramenta, onde encontram-se as janelas *Code Editor*, *Project Manager*, *Object Inspector* entre outras.

As janelas que aparecem no ambiente são:

- No canto superior esquerdo está a janela do *Code Explorer*, onde podemos acessar as *procedures* da *Unit*;

- Logo abaixo está localizada a janela *Project Manager*;

Essa janela contém todas as *Units* do projeto;

- No canto inferior esquerdo está a janela *Messages* onde as mensagens e erros de complicação são mostradas;

- No canto inferior direito está a janela *To-Do-List* (“Lista de tarefas”) que é um modo de controlarmos as implementações a serem feitas na aplicação;

- No centro temos o editor de código (“Code Editor”) do PocketStudio, onde cada *Unit* é representado por uma aba. Pressionando *F12* alternamos entre o *Code Editor* e o formulário;

- Finalmente no canto superior direito está o *Object Inspector*, acessível pela tecla *F11* como no Delphi. Nessa janela configuramos as propriedades e eventos dos objetos visuais de nossa aplicação;

Todas essas janelas aparecem ancoradas por padrão, mas podem ser fechadas ou configuradas como você preferir.

No PocketStudio, uma configuração que considero útil é podermos mostrar o *skin* do Palm diretamente na IDE como mostrado na **Figura 6**. *Skin* é a imagem do dispositivo propriamente dita. Para isso clicamos em *Tools|Environment Options>Designer*. Marque a opção *Display skin* e em seguida escolha o *skin* que desejar. No campo *Path to skins* configure o caminho onde as imagens foram salvas. É importante fazer o download dessas imagens, caso contrário não será possível visualizá-las.

Ao voltarmos ao PS temos o formulário já com a *skin* posicionada, isso facilita muito para montarmos o layout da tela da nossa aplicação. Voltando ao *Environment Options*, vamos dar uma olhada na aba *Debugger* que possui configurações importantes relacionadas ao emulador.

Na opção *Emulator* informamos o local onde está o executável do emulador para que o PocketStudio possa rodar nossas aplicações. Se você instalou o PocketStudio e o PalmOS Emulator nos diretórios indicados na instalação, isso é configurado por *Default*. Caso tenha mudado o diretório, principalmente do Emulador é importante configurar esse item, caso contrário o PocketStudio irá encontrar o emulador.

Construindo a aplicação

Todas as configurações prontas, vamos iniciar o desenvolvimento de nossa aplicação. Nosso objetivo é criar uma aplicação que servirá de apoio à vendas, ou seja, os representantes de uma empresa irão utilizá-la para digitar e enviar os pedidos à empresa. Imaginemos uma empresa que possui vários representantes e os mesmos fazem pedidos diariamente. Estes são enviados para a empresa por fax. Além de ter um custo elevado, a empresa ainda tem o trabalho de digitar todos os pedidos em seu ERP.

Nossa aplicação, além de diminuir consideravelmente o custo de envio dos pedidos, irá agilizar o processo, pois não será mais necessário digitar os pedidos, os mesmo serão importados do nosso dispositivo diretamente para o ERP.

Entendido o objetivo de nosso exercício iremos começar pela tela principal, onde criaremos o menu e alguns botões para acessarmos os cadastros e telas de consulta da nossa aplicação.

Abra o PocketStudio e crie uma nova aplicação em *File|New>Application*. O PocketStudio abrirá uma nova aplicação e será apresentado um formulário em branco. Antes de começarmos a montar o layout vamos salvar nossa aplicação. Acesse o menu *File|Save as* e salve o formulário como “uPrincipal.pas” e o projeto como “Vendas.ppr” em uma pasta de sua preferência. Feito isso vamos alterar algumas propriedades do formulário principal.

No *Object Inspector* modifique a propriedade *Title* para “Vendas” e seu *Name* para “FrmPrincipal”. Após isso pressione *Ctrl + F9* para compilar a nossa aplicação e note que erros foram apresentados, ao todo dois. Esse erros

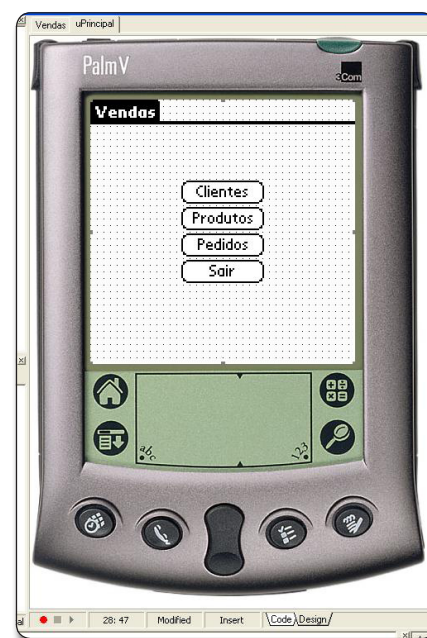


Figura 6. Imagem do dispositivo no IDE

foram causados pela alteração do nome de nosso formulário. Apenas mude de *Form1* para “FrmPrincipal” nos dois locais onde o compilador apontar.

Antes de continuarmos com o desenvolvimento de nossa aplicação vamos dar uma olhada no diretório onde foi salvo nosso projeto para entendermos os tipos de arquivos criados pelo PocketStudio.

Note que mesmo após compilarmos nossa aplicação não temos um arquivo com a extensão .exe comum em aplicações desktop. Isso por que no PalmOS os executáveis têm extensão .prc.

Voltando ao PocketStudio, o primeiro passo da nossa aplicação é criarmos os botões de acesso aos cadastros e telas de consulta e também o botão para fecharmos a aplicação. Nossa aplicação terá os formulários de *Clientes*, *Produtos*, *Pedidos* e *Itens de Pedidos*.

Inclua quatro *Buttons* da paleta *Form*. Os nomes para cada botão são: *btnClientes*, *btnProdutos*, *btnPedidos* e *btnSair*, respectivamente. Aumente a largura do primeiro botão digitando 50 na propriedade *Width*. Agora vamos aumentar a largura dos demais botões de uma só vez. Para isso selecione todos os botões pressionando a tecla *Shift* e clicando em cada botão. Feito isso dê uma clique de direita na mouse em cima de um dos cinco botões

e escolha a opção *Size*. Marque o item *Grow to largest* no grupo de opções *Width* e confirme a caixa de diálogo. Com isso deixamos todos os botões com o tamanho do maior componente da seleção.

Com o grupo de botões ainda selecionados, clique novamente com o direito e escolha *Align*. Centralize-os no formulário marcando o último item de cada grupo de opções, ou seja, *Center in window*. Veja o layout sugerido na **Figura 7**.

Analisando a linguagem

Vamos começar a entender um pouco mais a maneira como o compilador do PocketStudio trabalha. Clicando na aba *Code* ou pressionando *F12*, vamos ao fonte de nossa *Unit* principal.

No início da *Unit* temos o nome de nosso formulário principal. Logo abaixo, na seção *Interface*, ficam definidos os métodos visíveis para as outras *Units* bem como a cláusula *Uses*, onde encontramos

declarada a *Unit PSL*, que é a *PSLibrary*. Ela é um conjunto de bibliotecas em forma de arquivos *.pas* que simplificam o trabalho do desenvolvedor, pois tem várias funções para manipularmos objetos visuais e banco de dados no Palm. Podemos compará-la a VCL do Delphi. Mais adiante estudaremos essa biblioteca mais detalhadamente.

Em seguida nos deparamos com uma constante *FrmPrincipal* do tipo *AutoID*. É o identificador para nosso formulário; Essa seção de constantes contém todos os identificadores para todos os objetos existentes em nossa *Unit*, visuais ou não. Note que o primeiro objeto a ser identificado é nosso formulário *FrmPrincipal*.

Em seguida temos uma função chamada de *HandleEvent*, que recebe como parâmetro um evento do tipo *EventType* e retorna um valor booleano. Todo formulário tem o seu próprio *EventHandle*. Essa função é responsável por controlar todos

os eventos que chegam ao formulário. Após codificarmos nossos botões, voltaremos a ela para entendermos melhor o seu funcionamento (**Listagem 1**).

Mais adiante existe a seção *Implementation*, também encontrada em Delphi, onde são definidos todos os métodos e os objetos acessíveis apenas para essa *Unit*. Note que para cada botão que colocamos o PocketStudio gerou uma constante com o valor *AutoID* atribuído, ou seja, um identificador.

```
const
  btnClientes = AutoID;
  btnProdutos = AutoID;
  btnPedidos = AutoID;
  btnSair = AutoID;
```

Em *resource* temos os valores das propriedades configuradas no *Object Inspector*. Após codificarmos os eventos comuns para todos os formulários, falaremos mais sobre eles na sequência de nosso exemplo.

Codificando o formulário

O primeiro botão que vamos codificar será o *btnSair*, para podermos finalizar nossa aplicação sem precisarmos clicar no ícone da *Home* representado por uma “casinha” no dispositivo. Clique duas vezes no *btnSair* e note que somos levados ao editor de código do Palm no evento *BtnSairSelect*. Codifique de acordo com o código da **Listagem 2**.

No código da **Listagem 2** declaramos uma variável do tipo *EventType*. Em seguida atribuímos o valor *appStopEvent* a sua propriedade *eType*. Por fim invocamos o método *EvtAddEventToQueue* passando

Listagem 1. Trecho da *Unit uPrincipal*

```
unit uPrincipal;
interface
uses PSL;
const
  FrmPrincipal = AutoID;
function HandleEvent(var Event: EventType): Boolean;
implementation
const
  btnClientes = AutoID;
  btnProdutos = AutoID;
  btnPedidos = AutoID;
  btnSair = AutoID;
resource
  FORM FrmPrincipal AT (0 0 160 160) NOFRAME
  BEGIN
    TITLE 'Vendas'
    BUTTON btnClientes 'Clientes' AT (56 50 48 12)
    BUTTON btnProdutos 'Produtos' AT (56 66 48 12)
    BUTTON btnPedidos 'Pedidos' AT (56 82 48 12)
    BUTTON btnSair 'Sair' AT (56 98 48 12)
  END;
```

Listagem 2. Código do botão *Sair*

```
procedure BtnSairSelect;
var
  Evt: EventType;
begin
  Evt.eType := appStopEvent;
  EvtAddEventToQueue(@Evt);
end;
```

Listagem 3. Loop principal do projeto gerado pelo PocketStudio

```
procedure EventLoop;
var
  Event: EventType;
  Error: UInt16;
begin
  repeat
    EvtGetEvent(Event, evtWaitForever);
    if not SysHandleEvent(Event) then
      if not MenuHandleEvent(nil, Event, Error) then
        if not ApplicationHandleEvent(Event) then
          FrmDispatchEvent(Event); // sairá do loop quando receber um appStopEvent.
  until Event.eType = appStopEvent;
end;
```



Figura 7. Layout sugerido para a tela principal

para ele nossa variável. Isso terá efeito dentro da *Unit* principal do projeto onde encontramos o código da **Listagem 3**.

A variável *Evt*, declarada com o tipo *EventType* irá receber a atribuição *appStopEvent* que indica ao sistema operacional do PalmOS que deve encerrar o *loop* que mantém a aplicação aberta. Em outras palavras, a aplicação será finalizada.

A **Listagem 3** possui o código completo do método *EventLoop* localizado no fonte principal do projeto, no nosso caso chamado de *Vendas*.

Nota: A *Unit Vendas* se assemelha-se ao *Source* principal de um sistema desenvolvido em Delphi acessado pela opção *Project>View Source*.

Com ele fica fácil entendermos que ele estará em um *loop*, até que o atributo passado para ele seja do tipo *appStopEvent*, como mostrado na última linha do código. Compilamos nosso projeto com *Ctrl + F9*, se tudo estiver certo, salvamos o projeto clicando no menu *File>Save*. Para rodarmos a aplicação basta clicarmos em

F9, e o PocketStudio automaticamente abrirá o emulador. Veja nossa aplicação funcionando na **Figura 8**.

Clicando em sair nossa aplicação é encerrada e o PocketStudio fica piscando na barra de menu indicando que recebeu uma chamada, no caso do nosso Emulador no momento em que finalizamos a aplicação.

Criando menus no PocketStudio

Continuando nosso exemplo, vamos criar um menu em nosso formulário principal. Os menus estão localizados na paleta *Menu* na barra de componentes do PS. O menu deve ser adicionado diretamente no *Code Editor* acima da seção *resource*. Então clique acima da palavra reservada *resource*, selecione o componente *Menu* na paleta e clique uma vez no local informado anteriormente. Você notará a inclusão de uma nova constante como mostrado a seguir:

```
const
  Menu1 = AutoID;
resource
  MENU Menu1
  BEGIN
  END;
```

Como podemos ver o PocketStudio declarou uma constante *Menu1 = AutoID* para o componente e uma área de *resource* com o nome do componente ("Menu1") e *begin..end*. Continuando a construir nosso menu abrimos uma linha entre o *Begin* e o *End* e adicionamos o segundo componente da paleta *Menu* que é o *PullDown*. Feito isso abrimos mais uma linha entre o *Begin* e *End* do *PullDown* e colocamos o próximo componente, um *Item*. Na **Listagem 4** veja código completo para após a criação do menu.

O próximo componente a ser adicionado chamado *Separator*, que é apenas um separador entre os menus como o próprio nome diz. Como teremos apenas um item de menu não será necessário adicioná-lo ao código. Feito isso estamos com o esqueleto de nosso menu criado. Após inserirmos os menus no código, vamos renomeá-los para entendermos bem o funcionamento de cada componente adicionado.

O primeiro adicionado foi o menu principal, e por *default* nomeado como *Menu1*, vamos renomeá-lo para *MnuAplicacao*.

Impressão Rápida em Matricias...

RDprint 4.0

O mais completo componente para impressão em MATRICIAIS !
LIDERANÇA absoluta na sua categoria !
 Ideal para Notas Fiscais, Duplicatas, Boletos Bancários, etiquetas e relatórios em geral.

- Opção para impressão colorida
- Ajustes de margens para impressão gráfica
- Opção para ocultar a barra de progresso
- Variáveis PAGINAS, DATA, HORA e TÍTULO

Novo form de SETUP com :

- Mapeamento das impressoras e Modelos
- Seleção de páginas igual ao word (1-5,7,8)
- Opção para Inverter e Agrupar cópias na impressão

Novo form de PREVIEW com:

- Função para Procura de TEXTO no relatório
- ROLAGEM com salto automático de página
- ARRASTO da imagem do preview
- StatusBar com informações da impressão
- Novos ícones personalizados

* Disponível para Delphi 5, 6, 7, 2005 e 2006 (VCL)
 * Compatível com todas as versões do Windows
 * Imprime em portas LPT / COM e USB (modo gráfico)

RDprint Setup

Configuração da Impressão

Impressora: HP DeskJet 870Cxi Propriedades...

Modelo: Gráfico - Compatível com Windows Visualizar

Intervalo de Páginas:
☐ Todas
☐ Página Atual
☒ Páginas: 1-5,7

Cópias:
 Número de Cópias: 3
☒ Agrupar
☐ Ordem inversa

Imprimir: Todas as páginas do intervalo

Digite a páginas e/ou intervalos separados por vírgula. Por exemplo: 1,3,5-12

Ok Cancelar

Deltress Informática

Fone/Fax (14) 3454-7880
www.deltress.com.br

Página: 2 de 19 87% Impressora: HP DeskJet 870Cxi Gráfico * O RDprint 4.0 não imprime gráficos !

Muita atenção na hora de renomear os componentes pois no PocketStudio precisamos renomear o *resource* do componente e na declaração da *const* do mesmo.

O outro componente adicionado foi o *PullDown*, vamos mudar seu nome de *PullDown* para *Arquivo*. Esse componente não tem *resourceID* pois é apenas o título do nosso menu e não terá nenhum efeito associado a ele.

Para finalizarmos nosso primeiro exemplo de menu falta renomearmos o componente *MenuItem1* para *MnuSair* e mudar seu texto associado de *MenuItem* para *Sair*. A letra “T” que está comentada entre chaves no código acima, funciona como uma tecla de atalho no Palm. Descomente-a para usarmos como tecla de atalho.

Para cada *Item* adicionado é necessário nomear e inserir uma referência a ele na área de constantes do menu. Por isso modifique de *MenuItem1* para *MnuSair* e crie uma constante como podemos ver a seguir na **Listagem 5**.

Como mencionado, declare uma constante como *AutoID* para o componente *MnuSair*. Isso é necessário pois o PS não declara automaticamente os itens de menu.

Feito isso, precisamos associar o menu ao formulário. Para isso pressione *F12* para que possamos ver o formulário. Em seguida *F11* ou vá direto ao *Object Inspector*. Na propriedade *Menu* selecione o *FrmPrincipal*.

Por fim falta dizer para o PS o que ele deve fazer com esse menu, ou melhor,

qual será o efeito ao clicarmos com a caneta no menu *Sair*. Logicamente finalizaremos a aplicação, ou seja, faremos a codificação do evento responsável por executar as ações do *Menu*. Para isso clique duas vezes no evento *OnMenu* do formulário e digite o código da **Listagem 6**.

Observe que fizemos um *case..of* no *Item*, parâmetro recebido no evento, e verificamos qual menu foi clicado. Caso seja *MnuSair*, único item criado, executamos o procedimento de encerramento da aplicação.

Conclusão

Neste artigo tivemos o primeiro contato com o PocketStudio e o PalmOS Emulator, iniciamos nossa aplicação de vendas, e criamos a tela principal da aplicação. E também trabalhamos com botões e menus.

No próximo artigo iremos incrementar a tela principal, adicionando algumas consistências. Também veremos a tela de consulta de clientes, o que irá nos levar ao primeiro contato com *Banco de Dados* no PocketStudio, conceitos e formas de manipulação. Até o próximo artigo, e bons códigos a todos! ●

Listagem 4. Criação do menu

```
const
  Menu1 = AutoID;
resource
  MENU Menu1 >>> Componente Menu
  BEGIN
    PULLDOWN 'PullDown' >>> Componente PullDown
    BEGIN
      MENUITEM MenuItem1 'MenuItem' {'T'} >>> Componente MenuItem
    END
  END;
```

Listagem 5. Menu criado e alterado

```
const
  MnuPrincipal = AutoID;
  { MnuSair será associado ao método menu do Formulário }
  MnuSair = AutoID;
resource
  { Componente Menu renomeado para MnuPrincipal }
  MENU MnuPrincipal
  BEGIN
    { Componente PullDown renomeado para Arquivo }
    PULLDOWN 'Arquivo'
    BEGIN
      { Componente MenuItem renomeado para Sair }
      MENUITEM MnuSair 'Sair' 'S'
    END
  END;
```

Listagem 6. Código do evento OnMenu

```
procedure FrmPrincipalMenu(Item: UInt16);
var
  Evt: EventType;
begin
  case Item of
    MnuSair:
      begin
        Evt.eType := appStopEvent;
        Evt.AddEventToQueue(@Evt);
      end;
  end;
end;
```

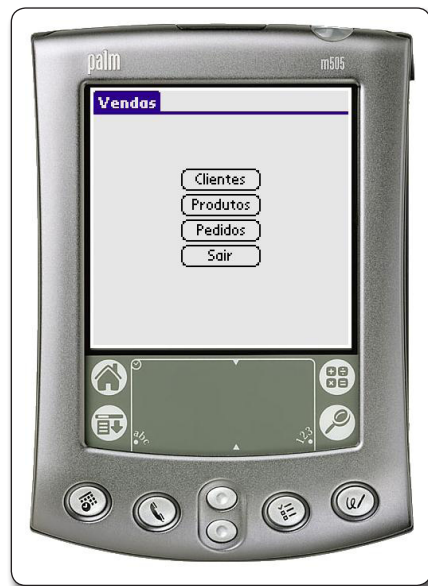


Figura 8. Sistema vendas funcionando no emulador

