Os controles são o principal meio de interação com o usuário. Digitando e clicando, e movendo-se pelos controles nos formulários do seu aplicativo, o usuário pode manipular dados e realizar as tarefas desejadas.

Este capítulo aborda os seguintes tópicos:

- Conhecendo controles e dados
- Selecionando o controle correto para a tarefa
- Facilitando o uso dos controles
- Estendendo formulários

Conhecendo controles e dados

Pode haver dois tipos de controles nos formulários: controles ligados aos dados e controles livres. Quando o usuário interage com controles de ligação, os valores digitados ou selecionados são armazenados na fonte de dados, que pode ser um campo de tabela, um campo de cursor ou uma variável. Para ligar um controle a dados, defina sua propriedade ControlSource, ou, no caso de grades, a propriedade RecordSource.

Se você não definir a propriedade ControlSource de um controle, o valor digitado ou selecionado pelo usuário no controle será apenas armazenado como uma definição de propriedade. O valor não será gravado no disco nem armazenado na memória após a vida útil do controle.

Efeito da definição da propriedade ControlSource sobre controles

Controle	Efeito
Caixa de verificação	Se a ControlSource for um campo de uma tabela, os valores NULL, valores lógicos verdadeiro (.T.) ou falso (.F.) ou valores numéricos 0, 1 ou 2 no campo ControlSource farão com que a caixa de verificação seja selecionada, limpa ou fique acinzentada quando o ponteiro do registro se movimentar pela tabela.
Coluna	Se a ControlSource for um campo de tabela, o usuário estará editando diretamente o campo quando editar valores na coluna. Para ligar uma grade inteira a dados, defina a propriedade RecordSource da grade.
Caixa de listagem ou caixa de combinação	Se a ControlSource for uma variável, o valor selecionado pelo usuário na lista será armazenado na variável. Se a ControlSource for um campo de uma tabela, o valor será armazenado no campo na posição do ponteiro do registro. Se um item da lista corresponder ao valor do campo na tabela, o item será selecionado na lista quando o ponteiro do registro se movimentar pela tabela.
Botão de opção	Se a ControlSource for um campo numérico, será gravado 0 ou 1 no campo, dependendo de o botão estar ou não selecionado. Se a ControlSource for lógica, .T. ou .F. será gravado no campo, dependendo de o botão estar ou não selecionado. Se o ponteiro do registro se movimentar pela tabela, o valor do botão de opção será atualizado para refletir o novo valor do campo. Se a ControlSource do controle OptionGroup do

	botão de opção (e não do botão propriamente dito) for um campo de caracteres, a legenda do botão de opção será armazenada no campo se o botão de opção for escolhido. Observe que a fonte do controle para um botão de opção (diferentemente de um controle OptionGroup) não pode ser um campo de caracteres, senão o Visual FoxPro reportará uma falta de correspondência entre tipos de dados quando o formulário for executado.
Controle de rotação	O controle de rotação reflete e grava valores numéricos na campo ou variável subjacente.
Caixa de texto ou caixa de edição	O valor no campo da tabela é exibido na caixa de texto. As alterações feitas pelo usuário nesse valor são regravadas na tabela. O movimento do ponteiro do registro afeta a propriedade Value da caixa de texto.

Algumas das tarefas que você deseja realizar com controles exigem que os dados estejam ligados ao controle. Outras, não.

Selecionando o controle correto para a tarefa

Os controles do Visual FoxPro são flexíveis e versáteis. Embora existam vários controles que você possa utilizar para realizar uma tarefa específica, é necessária uma abordagem consistente dos controles utilizados para que o usuário saiba o que esperar quando vir a interface fornecida por você. Por exemplo, uma etiqueta tem um evento Click, tal como um botão de comando, mas o usuário familiarizado com interfaces gráficas espera clicar sobre botões de comando para executar ações.

A maior parte das funcionalidades que você pode desejar conferir aos seus formulários irá se enquadrar em uma das categorias abaixo:

- Fornecer ao usuário um conjunto de opções pré-determinadas
- Aceitar entrada do usuário que não possa ser pré-determinada
- Aceitar entrada do usuário dentro de determinado intervalo
- Permitir que os usuários executem ações específicas
- Executar ações específicas em determinados intervalos
- Exibir informações

Fornecendo um conjunto de opções pré-determinadas

Uma das maneiras mais diretas de assegurar a validade dos dados em um banco de dados é fornecer ao usuário um conjunto pré-determinado de opções. Controlando as opções do usuário, você poderá assegurar que nenhum dado inválido seja armazenado no banco de dados. Os controles abaixo lhe permitem fornecer ao usuário um conjunto de opções pré-determinadas:

- Grupos de Botões de opção
- Caixas de listagem e Listas suspensas
- Caixas de verificação

Utilizando grupos de botões de opção

Os grupos de botões de opção são recipientes que contêm botões de opção. Normalmente, os botões de opção permitem que o usuário especifique uma dentre uma série de opções operacionais

em uma caixa de diálogo, em vez de fornecer uma entrada de dados. Por exemplo, os botões de opção podem ser utilizados para especificar a saída para um arquivo, uma impressora ou para visualização da impressão, conforme descrito no capítulo 12, Adicionando consultas e relatórios.

Definindo o número de botões de opção em um grupo de botões de opção

Quando você cria um grupo de botões de opção em um formulário, são incluídos, como padrão, dois botões de opção. Para determinar quantos botões de opção existem em um grupo, altere a propriedade ButtonCount.

Para definir o número de botões de opção em um grupo

- Defina a propriedade ButtonCount com o número desejado de botões de opção.

  Por exemplo, para ter um grupo de seis botões de opção, defina a propriedade ButtonCount do grupo de botões de opção como 6.

A propriedade Value do grupo indica o botão que foi selecionado. Por exemplo, se o usuário selecionar o quarto botão de opção em um grupo de seis, o valor do grupo de botões de opção será 4.

Se a propriedade ControlSource do grupo for um campo de caracteres, ou se a propriedade Value for definida como um valor de caractere antes que o formulário seja executado, a propriedade Value do grupo será a legenda do botão de opção selecionado.

Definindo propriedades de botões de opção

Para ajustar manualmente os elementos individuais de um grupo de botões de comando ou de botões de opção no **Criador de formulários**, selecione Editar no menu de atalho do grupo.

Você pode definir propriedades para botões individuais na janela **Propriedades**.. Você também pode definir essas propriedades em tempo de execução, especificando o nome do botão de opção e a definição de propriedade desejada. Por exemplo, a linha de código a seguir, incluída no código do método ou evento de alguns objetos no mesmo formulário que o grupo de botões de opção, define a legenda de `optCust` no grupo de botões de opção `opgChoices`:

```
THISFORM.opgChoices.optCust.Caption = "Classificar por Cliente"
```

Você também pode definir estas propriedades em tempo de execução, utilizando a propriedade Buttons e especificando o número de índice do botão de opção no grupo. Por exemplo, se `optCust` for o terceiro botão no grupo, a linha de código a seguir também definirá a legenda de `optCust`:

```
THISFORM.opgChoices.Buttons(3).Caption = "Classificar por Cliente"
```

Para definir propriedades para todos os botões de um grupo

- Utilize o método SetAll do grupo.

  Por exemplo, a linha de código a seguir desativa todos os botões de um grupo de botões de opção denominado `opgMyGroup`, em um formulário:

  ```
  THISFORM.opgMyGroup.SetAll("Enabled",.F., "OptionButton")
  ```

Ativando e desativando botões em um grupo

O exemplo anterior mostra como desativar através da linguagem de programação todos os botões de opção em um grupo. Quando os botões estão desativados, eles são exibidos nas cores especificadas nas propriedades DisabledForeColor e DisabledBackColor dos botões de opção. Também é possível definir a propriedade Enabled do grupo de botões de opção como (.F.) para desativar o grupo; no entanto, não haverá indicação visual para o usuário.

Determinando o botão de opção selecionado no momento

Você pode utilizar a propriedade Value do grupo de botões de opção para determinar qual botão de opção do grupo está selecionado. Se a origem do controle do botão for numérica, haverá cinco botões de opção em um grupo. Se o terceiro botão estiver selecionado, a propriedade Value do

grupo de botões de opção será 3; caso nenhum botão de opção esteja selecionado, a propriedade Value do grupo de botões de opção será 0.

Também é possível determinar a legenda do botão de opção selecionado utilizando as propriedades Value e Buttons do grupo. Por exemplo, a linha de código a seguir armazena a propriedade Caption do botão de opção selecionado numa variável `cSelected`.

```
oGroup = THISFORM.opg1
cSelected = oGroup.Buttons(oGroup.Value).Caption
```

Filtrando listas com botões de opção

Se você tiver um pequeno conjunto de filtros de tabela pré-determinados, poderá utilizar botões de opção para permitir que os usuários alternem entre os filtros.

O exemplo abaixo pressupõe um formulário com uma caixa de listagem (`lstCustomers`) e com um grupo de botões de opção que contém três botões de opção.

Definições de propriedades para a caixa de listagem

Objeto	Propriedade	Definição
lstCustomers	RowSourceType	2 - Alias
lstCustomers	RowSource	Cliente

Os filtros são definidos no código de evento Click dos botões de opção.

Código do evento para filtrar uma lista quando o usuário seleciona um botão de opção

Objeto	Evento	Código

optAll	Click	`SET FILTER TO`
		`GO TOP`
		`THISFORM.lstCustomers.Requery`

optCanada	Click	`SET FILTER TO customer.country = "Canadá"`
		`GO TOP`
		`THISFORM.lstCustomers.Requery`

optUK	Click	`SET FILTER TO customer.country = "Reino Unido"`
		`GO TOP`
		`THISFORM.lstCustomers.Requery`

Quando o usuário fechar o formulário, não se esqueça de redefinir o filtro incluindo SET FILTER TO no evento Click do botão de fechamento ou no evento Destroy.

Dica Para atualizar uma lista quando a fonte da lista tiver sido alterada, utilize o método Requery.

Utilizando botões de opção para armazenar escolhas do usuário em uma tabela

Embora não seja muito comum, você pode utilizar botões de opção para obter informações de um usuário a serem armazenadas em uma tabela, salvando a propriedade Caption. Caso você disponha de um aplicativo de teste padronizado, por exemplo, poderá utilizar botões de opção para permitir que o usuário escolha entre várias opções - A, B, C ou D. Você pode também utilizar botões de opção para indicar o sexo em uma tabela de funcionários.

Para armazenar a propriedade Caption de um botão de opção em uma tabela

1. Defina a propriedade Value do grupo de botões de opção como uma seqüência vazia.
2. Defina a propriedade ControlSource do grupo de botões de opção como um campo de caractere de uma tabela.

Por exemplo, se as legendas dos botões de opção no grupo forem “A”, “B”, “C” e “D” e a ControlSource do grupo de botões de opção for um campo de caractere, quando o usuário selecionar o botão com a legenda “B”, “B” será armazenado no campo.

Para ver um exemplo de um teste de múltipla escolha utilizando botões de opção, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e escolha “Apresenta várias opções a um

usuário”.

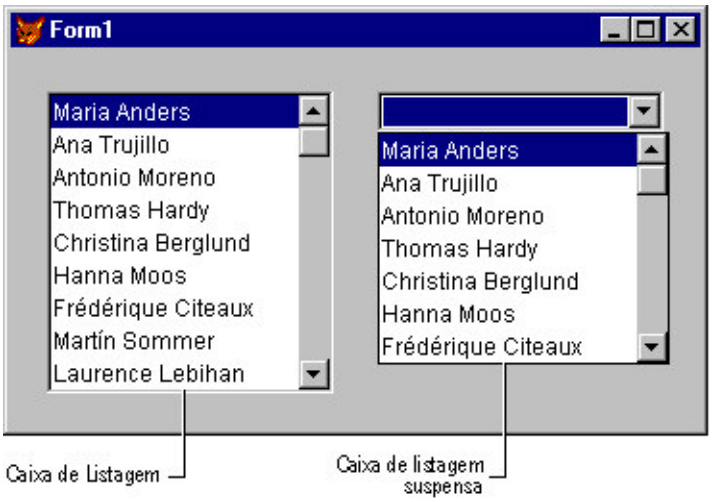
Utilizando caixas de listagem e caixas de listagem suspensas

As caixas de listagem e as caixas de listagem suspensas (controles de caixa de combinação com a propriedade Style definida como 2–Lista Suspensa) fornecem ao usuário listas roláveis que contêm uma série de opções ou itens de informação. Em uma caixa de listagem, vários itens podem ficar sempre visíveis. Em uma caixa de listagem suspensa somente um item fica visível, mas o usuário pode clicar sobre o botão seta-para-baixo para exibir uma lista rolável de todos os itens de uma caixa de listagem suspensa.

Execute SOLUTION.APP no diretório SAMPLES\SOLUTION para ver vários exemplos que demonstram a utilização de caixas de listagem e de caixas de listagem suspensas, incluindo o seguinte:

- Adicionar figuras a uma lista.
- Selecionar vários itens em uma lista.
- Preencher uma lista com valores de origens diferentes.
- Exibir várias colunas em uma lista.
- Classificar itens de listas.
- Mover itens entre listas.

Caixa de listagem e caixa de listagem suspensa com a mesma definição de propriedade RowSource



Dica Se houver espaço no formulário e você quiser enfatizar as opções disponíveis, utilize uma lista. Para economizar espaço e enfatizar o item selecionado no momento, utilize uma caixa de listagem suspensa.

Métodos e propriedades comuns de listagem

As propriedades de caixa de listagem a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
ColumnCount	O número de colunas na caixa de listagem.
ControlSource	Onde é armazenado o valor selecionado pelo usuário

	na lista.
MoverBars	Se são exibidas barras de movimentação à esquerda dos itens da lista para que o usuário possa reorganizar com facilidade a ordem dos itens.
Multiselect	Se o usuário pode selecionar mais de um item por vez na lista.
RowSource	A origem dos valores exibidos na lista.
RowSourceType	Se a RowSource é um valor, uma tabela, uma instrução SQL, uma consulta, uma matriz, uma lista de arquivos ou uma lista de campos.

Observação A propriedade Value de uma lista pode ser numérica ou caractere. O padrão é numérico. Defina a propriedade Value como uma seqüência vazia se a RowSource for um valor de caractere e você desejar que a propriedade Value reflita a seqüência de caracteres do item selecionado na lista. Você pode pressionar a BARRA DE ESPAÇOS e, em seguida, a tecla BACKSPACE para inserir uma seqüência de caracteres vazia para a propriedade na janela **Propriedades**.

Os métodos de caixa de listagem abaixo são freqüentemente utilizados.

Método	Descrição
AddItem	Adiciona um item a uma lista com uma RowSourceType 0.
RemoveItem	Remove um item de uma lista com uma RowSourceType 0.
Requery	Atualiza a lista se os valores na RowSource tiverem sido alterados.

Preenchendo uma caixa de listagem ou uma caixa de combinação

Você pode preencher uma caixa de listagem com itens de várias fontes, definindo as propriedades RowSourceType e RowSource.

Selecionando o tipo de dado para uma caixa de combinação ou de listagem

A propriedade RowSourceType determina o tipo de fonte que ocupa a caixa de listagem ou a caixa de combinação — por exemplo, uma matriz ou uma tabela. Depois de definir a propriedade RowSourceType, especifique a origem dos itens da lista, definindo a propriedade RowSource.

RowSourceType	Fonte dos itens da lista
0	Nenhuma. Adiciona itens à lista via programação.
1	Valor
2	Alias
3	Instrução SQL
4	Consulta (.QPR)
5	Matriz
6	Campos
7	Arquivos
8	Estrutura
9	Pop-up. Incluído para manter a compatibilidade com versões anteriores.

As seções a seguir descrevem as várias definições de RowSourceType.

Nenhuma Se você definir a propriedade RowSourceType como 0, que é o padrão, a lista não será preenchida automaticamente. Você pode adicionar itens à lista utilizando o método AddItem:

```
frmForm1.lstMyList.RowSourceType = 0
```

```
frmForm1.lstMyList.AddItem("Primeiro Item")
frmForm1.lstMyList.AddItem("Segundo Item")
frmForm1.lstMyList.AddItem("Terceiro Item")
```
O método RemoveItem permite remover itens da lista. Por exemplo, a linha de código a seguir remove "Segundo Item" da lista:

```
frmForm1.lstMyList.RemoveItem(2)
```
Valor Se você definir a propriedade RowSourceType como 1, poderá especificar vários valores na propriedade RowSource para que sejam exibidos na lista. Se definir a propriedade RowSource através da janela **Propriedades**, inclua uma lista de itens separados por vírgulas. Se definir RowSource utilizando a linguagem de programação, coloque essa lista entre aspas:

```
Form1.lstMyList.RowSourceType = 1
Form1.lstMyList.RowSource = "um,dois,três,quatro"
```
Alias Se você definir a propriedade RowSourceType como 2, poderá incluir valores de um ou mais campos em uma tabela aberta.

Se a propriedade ColumnCount for 0 ou 1, a lista exibirá valores no primeiro campo da tabela. Se você definir a propriedade ColumnCount como 3, a lista exibirá valores nos três primeiros campos da tabela. Para exibir campos em uma ordem diferente daquela em que eles estão armazenados na tabela, defina a propriedade RowSourceType como 3 (Instrução SQL) ou 6 (Campos).

Observação Se. RowSourceType é 2 (Alias) ou 6 (Campos), quando um usuário escolher um novo valor na lista, o ponteiro de registro da tabela irá para o registro com o valor desse item.

Instrução SQL Se você definir a propriedade RowSourceType como 3 (Instrução SQL), inclua uma instrução SELECT - SQL na propriedade RowSource. Por exemplo, a instrução a seguir seleciona todos os campos e todos os registros da tabela Cliente para um cursor:

```
SELECT * FROM Customer INTO CURSOR mylist
```
Se você definir RowSource utilizando a linguagem de programação, lembre-se de colocar a instrução SELECT entre aspas.

Observação Como padrão, as instruções SELECT do Visual FoxPro sem cláusulas INTO exibem imediatamente o cursor resultante em uma janela **Pesquisar**. Visto que raramente se deseja esse comportamento em uma instrução RowSource SQL, inclua uma cláusula INTO CURSOR na instrução SELECT.

Consulta Se você definir a propriedade RowSourceType como 4, poderá ocupar a caixa de listagem com os resultados de uma consulta designada por você no **Criador de consultas**. Quando a propriedade RowSourceType estiver definida como 4, defina RowSource como o arquivo .QPR. Por exemplo, a linha de código a seguir define a propriedade RowSource de uma lista como uma consulta.

```
THISFORM.List1.RowSource = "region.qpr"
```
Se você não especificar uma extensão de arquivo, o Visual FoxPro irá adotar a extensão .QPR.

Matriz Se você definir a propriedade RowSourceType como 5, a lista será ocupada com os itens de uma matriz. Você pode criar uma propriedade matriz do formulário ou do conjunto de formulários para a RowSource ou utilizar uma matriz criada em outra parte do aplicativo.

Para obter informações sobre como criar propriedades de matriz, consulte o capítulo 9, Criando formulários.

Solucionando problemas A definição da propriedade RowSource de uma lista é avaliada pelo Visual FoxPro de acordo com a necessidade do aplicativo e não apenas do método em que você define essa propriedade. É preciso ter em mente essa abrangência. Se você criar uma matriz local em um método, o escopo dessa matriz será esse método, e ela não estará disponível em todas as situações em que o Visual FoxPro precisa avaliar a definição da propriedade. Se você definir a RowSource de uma lista como uma propriedade de matriz do formulário ou do conjunto de formulários, deverá fazer referência à propriedade com relação à lista e não com relação ao método no qual definiu a propriedade. Por exemplo, se houver uma propriedade de matriz de formulário denominada `arrayprop`, as linhas de código a seguir no Init do formulário irão produzir resultados

diferentes:

```
THIS.lst1.RowSource = "THIS.arrayprop"      && Erro
THIS.lst1.RowSource = "THISFORM.arrayprop" && Nenhum erro.
```

Para ocupar uma lista com os elementos de uma matriz multidimensional

1. Defina a propriedade RowSourceType como 5.
2. Defina a propriedade RowSource como a matriz multidimensional.
3. Defina a propriedade ColumnCount como o número de colunas a serem exibidas.
4. Defina a propriedade ColumnWidths com as larguras desejadas para cada coluna.

Campos Se você definir a propriedade RowSourceType como 6, poderá especificar um campo ou uma lista de campos delimitada por vírgulas para ocupar a lista, como:

```
contato,empresa,país
```

Você pode incluir os seguintes tipos de informação na propriedade RowSource de uma lista com a propriedade RowSourceType definida como 6–Campos:

- campo
- alias.campo
- alias.campo, campo, campo, ...

Se você quiser que a lista contenha campos de várias tabelas, defina a propriedade RowSourceType como 3–Instrução SQL.

Ao contrário de RowSourceType como 2–Alias, a RowSourceType como 6–Campos lhe permite exibir campos independentemente de suas posições reais na tabela.

Arquivos Se você definir a propriedade RowSourceType como 7, a lista será ocupada com arquivos do diretório atual. Além disso, as opções na lista lhe permitem selecionar unidade de disco e diretório diferentes para os nomes de arquivo a serem exibidos na lista.

Lista preenchida com arquivos de um diretório

Defina RowSource como o esqueleto do tipo de arquivo que você deseja exibir na lista. Por exemplo, para exibir tabelas do Visual FoxPro na lista, defina a propriedade RowSource como ***.dbf**.

Estrutura Se você definir a propriedade RowSourceType como 8, a lista será ocupada com os campos da tabela especificada quando você definiu a propriedade RowSource. Esta definição de RowSourceType será útil se você quiser apresentar ao usuário uma lista de campos em que ele deve procurar valores ou uma lista de campos pelos quais ordenar uma tabela.

Pop-up Se você definir a propriedade RowSourceType como 9, poderá preencher a lista a partir de um pop-up anteriormente definido. Esta opção foi incluída para manter a compatibilidade com versões anteriores.

Criando uma caixa de listagem de várias colunas

Embora o número padrão de colunas em uma caixa de listagem seja uma, uma caixa de listagem no Visual FoxPro pode conter quantas colunas você desejar. Uma caixa de listagem de várias colunas difere de uma grade, sendo que em uma caixa de listagem de várias colunas, você seleciona uma linha por vez, enquanto em uma grade você pode selecionar células individuais; e os dados de uma lista não podem ser editados diretamente.

Para exibir várias colunas em uma caixa de listagem

1. Defina a propriedade ColumnCount como o número de colunas desejadas.
2. Defina a propriedade ColumnWidths. Por exemplo, se a caixa de listagem tiver três colunas, o comando a seguir definirá as larguras das colunas como 10, 15 e 30, respectivamente:

   ```
   THISFORM.listbox.ColumnWidths = "10, 15, 30"
   ```
3. Defina a propriedade RowSourceType como **6 - Campos**.
4. Defina a propriedade RowSource como os campos a serem exibidos nas colunas. Por exemplo, o comando abaixo define a origem das três colunas em uma caixa de listagem de 3 colunas como os campos contato, cidade e país da tabela de clientes:

   ```
   form.listbox.RowSource = "contato,cidade,país"
   ```

Observação Para que as colunas sejam alinhadas corretamente, você deve definir a propriedade ColumnWidths ou alterar a propriedade FontName para uma fonte com espaçamento simples.

Quando a propriedade RowSourceType da lista for definida como 0 - Nenhum, você pode utilizar o método AddListItem para adicionar itens a uma caixa de listagem de várias colunas. Por exemplo, o código a seguir adiciona texto a colunas específicas em uma caixa de listagem:

```
THISFORM.lst1.ColumnCount = 3
THISFORM.lst1.Columnwidths = "100,100,100"
THISFORM.lst1.AddListItem("linha1 col1", 1,1)
THISFORM.lst1.AddListItem("linha1 col2", 1,2)
THISFORM.lst1.AddListItem("linha1 col3", 1,3)
THISFORM.lst1.AddListItem("linha2 col2", 2,2)
```

Permitindo que o usuário selecione vários itens em uma Caixa de listagem

O comportamento padrão de uma lista permite que seja selecionado um item por vez. No entanto, você pode permitir que o usuário selecione vários itens em uma lista.

Para permitir a seleção de vários itens em uma lista

- Defina a propriedade MultiSelect da lista como verdadeira (.T.).

Para processar os itens selecionados, isto é, para copiá-los para uma matriz ou incorporá-los a outra parte do aplicativo, efetue um loop pelos itens da lista e processe aqueles para os quais a propriedade Selected é verdadeira (.T.). O código a seguir pode ser incluído no evento InteractiveChange de uma caixa de listagem para exibir os itens selecionados em uma caixa de combinação, `cboSelected`, e o número de itens selecionados em uma caixa de texto, `txtNoSelected`:

```
nNumberSelected = 0  && uma variável para pegar o número
THISFORM.cboSelected.Clear   && limpa a caixa de combinação
FOR nCnt = 1 TO THIS.ListCount
IF THIS.Selected(nCnt)
nNumberSelected = nNumberSelected + 1
THISFORM.cboSelected.Additem (THIS.List(nCnt))
ENDIF
ENDFOR
THISFORM.txtNoSelected.Value = nNumberSelected
```

Permitindo que o usuário adicione itens a uma caixa de listagem

Além de permitir que o usuário selecione itens em uma caixa de listagem, você pode permitir que ele adicione itens a uma lista interativamente.

Para adicionar itens a uma lista interativamente

- Utilize o método AddItem.

No exemplo a seguir, o código no evento KeyPress de uma caixa de texto adiciona o texto da caixa de texto à caixa de listagem e limpa o texto da caixa de texto quando o usuário pressiona ENTER:

```
LPARAMETERS nKeyCode, nShiftAltCtrl
IF nKeyCode = 13     && Inserir chave
THISFORM.lstAdd.AddItem(This.Value)
THIS.Value = ""
ENDIF
```

Permitindo que o usuário digite dados de uma lista em uma tabela

Se a propriedade ControlSource estiver definida como um campo, tudo que o usuário selecionar na lista será gravado na tabela. Esta é uma maneira fácil de ajudar a garantir a integridade dos dados da tabela. Embora o usuário possa inserir o dado errado, não poderá digitar um valor inválido.

Por exemplo, se você oferecer ao usuário uma lista de estados ou países para seleção, ele não poderá digitar uma sigla de país ou estado inválida.

Permitindo que o usuário vá para um registro selecionando um valor em uma lista

Em várias ocasiões você desejará permitir que o usuário selecione o registro que ele deseja visualizar ou editar. Por exemplo, você pode fornecer ao usuário uma lista de nomes de clientes. Quando ele selecionar um cliente na lista, você selecionará o registro do cliente na tabela e exibirá as informações de cliente em caixas de texto do formulário. Você pode fazer isso de várias maneiras, dependendo da fonte de dados de seu formulário.

RowSourceType	Selecionando o registro apropriado
2 - Alias 6 - Campos	Quando o usuário seleciona um valor na lista, o ponteiro de registro é definido automaticamente como o registro desejado. Emita THISFORM.Refresh no evento InteractiveChange da lista para exibir os novos valores de outros controles do formulário.

0 - Nenhuma
1 - Valor
3 - Instrução SQL
4 - QPR
5 - Matriz
	No evento InteractiveChange, selecione a tabela que possui o registro com os valores desejados e, em seguida, procure o valor desejado. Por exemplo, se a propriedade RowSource contiver números de identificação de cliente da tabela de clientes, utilize este código:
	`SELECT customer`
	`LOCATE FOR THIS.Value = cust_id`
	`THISFORM.Refresh`

Atualizando uma exibição um para n baseada em um valor da lista

Quando o usuário decide ir para um registro escolhendo um valor em uma lista, você poderá ter um relacionamento de um para n que deve refletir o ponteiro do registro alterado na tabela pai. Você pode implementar essa funcionalidade com tabelas locais e visualizações locais ou remotas.

Tabelas locais

Se a RowSourceType da lista for 2(Alias) ou 6(Campos) e a RowSource for uma tabela local com um relacionamento definido no ambiente de dados do formulário, emita `THISFORM.Refresh` no evento InteractiveChange quando o usuário selecionar um novo valor. O lado n do relacionamento um para n exibe automaticamente somente os registros correspondentes à expressão da tabela pai envolvida na relação.

Visualizações

Atualizar uma exibição um para n será um pouco diferente se a RowSource da caixa de listagem for uma visualização local ou remota. O exemplo a seguir descreve como criar um formulário com uma caixa de listagem e uma grade. A caixa de listagem exibe os valores do campo `cust_id` na tabela `TESTDATA!Customer`. A grade exibe os pedidos associados com o campo `cust_id` selecionado na caixa de listagem.

Primeiro, no **Criador de visualizações** crie uma visualização parametrizada para os pedidos. Ao criar a visualização no **Criador de visualizações**, defina o critério de seleção para a chave estrangeira como uma variável. No exemplo a seguir, a variável se chama `m.cCust_id`.

Visualização parametrizada utilizando uma variável

A seguir, ao criar o formulário, siga as etapas no procedimento a seguir. Observe que a visualização requer um valor para o parâmetro que não está disponível quando o formulário é carregado. Definindo a propriedade NoDataOnLoad do objeto Cursor da visualização como verdadeira (.T.), você evitará que a visualização seja executada até que a função REQUERY( ) seja chamada, quando então o usuário já terá selecionado um valor para a variável usada na visualização parametrizada.

▸ **Para criar uma lista um para n baseada em visualizações locais ou remotas**

1. Adicione a tabela e a visualização parametrizada ao ambiente de dados.
2. Na janela **Propriedades** do objeto cursor da visualização em **Ambiente de dados**, defina a propriedade NoDataOnLoad como verdadeira (.T.).
3. Defina a propriedade RowSourceType da caixa de listagem como **6 - Campos**, e defina sua propriedade RowSource como o campo ao qual se faz referência como chave estrangeira no parâmetro da visualização.

   No exemplo, você definiria a propriedade RowSource como `customer.cust_id`.
4. Defina a propriedade RecordSource da grade como o nome da visualização criada anteriormente.
5. No código do evento InteractiveChange da caixa de listagem, armazene o valor da caixa de listagem na variável e consulte a visualização novamente, como neste exemplo:

```
m.cCust_id = THIS.Value
*considerando que o nome da visualização é orders_view
=REQUERY("orders_view")
```

Para obter maiores informações sobre visualizações remotas e locais, consulte o capítulo 8, Criando visualizações.

Exibindo registros filho em uma lista

Você pode exibir registros de um relacionamento um para n em uma lista, para que a lista exiba os registros filho do relacionamento à medida que o ponteiro de registro se mover pela tabela pai.

▸ **Para exibir registros filho em uma lista**

1 Adicione uma lista ao formulário.

2 Defina a propriedade ColumnCount da lista como o número de colunas que você deseja exibir.

Por exemplo, se quiser exibir os campos `Order_id`, `Order_net` e `Shipped_on` na lista, defina a propriedade ColumnCount como 3.

3 Defina a propriedade ColumnWidths com as larguras apropriadas para exibir os campos selecionados.

4 Defina a propriedade RowSourceType da lista como **3 – Instrução SQL**.

5 Defina a propriedade RowSource como a instrução SELECT. Por exemplo, a instrução a seguir seleciona três campos da tabela de pedidos para o registro atual na tabela de clientes:

```
SELECT order_id, order_net, shipped_on from orders ;
WHERE order.cust_id = customer.cust_id ;
INTO CURSOR temp
```

6 No evento Init do formulário e no código que move o ponteiro do registro pela tabela, consulte novamente a lista:

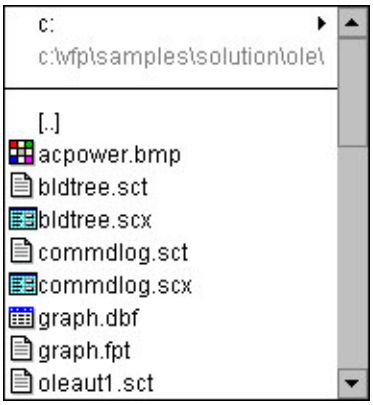
```
THISFORM.lstChild.Requery
```

Adicionando figuras a itens de uma lista

Você pode definir a propriedade Picture da lista como o arquivo .BMP que deseja exibir ao lado dos itens da lista.

Por exemplo, é possível preencher uma caixa de listagem com arquivos. Você pode querer colocar um bitmap diferente ao lado do arquivo quando ele for uma tabela, um programa ou algum outro tipo de arquivo.

Caixa de listagem com figuras



O código a seguir está associado ao evento Click da caixa de listagem:

```
FOR iItem = 5 TO THIS.ListCount          && arquivos que começam no quinto item
cExtension = UPPER(RIGHT(THIS.List(iItem),3))
DO CASE
CASE cExtension = "DBF"
THIS.Picture(iItem) = "tables.bmp"
CASE cExtension = "BMP"
THIS.Picture(iItem) = "other.bmp"
CASE cExtension = "PRG"
THIS.Picture(iItem) = "programs.bmp"
CASE cExtension = "SCX"
THIS.Picture(iItem) = "form.bmp"
OTHERWISE
THIS.Picture(iItem) = IIF("]" $ cExtension, ;
"", "textfile.bmp")
ENDCASE
ENDFOR
```

Utilizando caixas de verificação

Você pode utilizar caixas de verificação para permitir que o usuário especifique um estado booleano: Verdadeiro ou Falso, Ativado ou Desativado, Aberto ou Fechado. No entanto, às vezes Verdadeiro ou Falso não representam uma avaliação precisa, como no caso de perguntas sem resposta em um questionário Verdadeiro/Falso.

Para ver exemplos de como utilizar caixas de verificação, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e, em seguida, escolha Checkbox na seção Control.

Existem quatro estados possíveis para uma caixa de verificação, conforme determinados pela propriedade Value.

Exibição	Propriedade Value
Check1	0 ou .F.
Check2	1 ou .T.
Check3	2
Check4	.NULL.

A propriedade Value da caixa de verificação reflete o tipo de dado da última atribuição. Se você definir a propriedade como verdadeira (.T.) ou falsa (.F.), o tipo será Lógico até você definir a propriedade como um valor numérico.

Dica Um usuário pode exibir um valor nulo em uma caixa de verificação, pressionando CTRL+0.

Armazenando ou exibindo campos lógicos

Se você definir a propriedade ControlSource da caixa de verificação como um campo lógico de uma tabela, a caixa de verificação será exibida como marcada quando o valor do registro atual for verdadeiro (.T.), como não marcada, quando
o valor no registro atual for falso (.F.), e como acinzentada quando um valor nulo (.NULL.) estiver no registro atual.

Aceitando entrada que não pode ser pré-determinada

Nem sempre é possível prever todos os valores possíveis que o usuário pode precisar digitar em um controle. Os controles a seguir permitem aceitar entrada do usuário que não possa ser pré-determinada:

- Caixas de Texto
- Caixas de Edição
- Caixas de Combinação

Utilizando caixas de texto

A caixa de texto é o controle básico que permite que o usuário adicione ou edite dados armazenados em um campo não-memo de uma tabela.

Para ver exemplos de como utilizar caixas de texto, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e, em seguida, escolha Textbox na seção Controls.

Para referir ou alterar o texto exibido na caixa de texto utilizando a linguagem de programação

- Defina ou faça referência à propriedade Value.

Se você definir uma propriedade ControlSource para a caixa de texto, o valor exibido na caixa de texto será armazenado na propriedade Value da caixa de texto e na variável ou campo especificada(o) na propriedade ControlSource.

Validando os dados em uma caixa de texto

Para verificar ou confirmar o valor na caixa de texto, inclua código no método associado ao evento Valid. Se o valor for inválido, será retornado falso (.F.) ou 0. Se Valid retornar falso (.F.), será exibida uma mensagem "Entrada inválida". Se você quiser exibir sua própria mensagem, inclua o comando WAIT WINDOW ou a função MESSAGEBOX( ) no código Valid e retorne 0.

Por exemplo, se uma caixa de texto aceita que o usuário digite a data de um compromisso, você poderá verificá-la para certificar-se de que a data ainda não transcorreu, incluindo o código abaixo no evento Valid da caixa de texto:

```
IF CTOD(THIS.Value) < DATE( )
= MESSAGEBOX("Você deve informar uma data futura",1)
RETURN 0
ENDIF
```

Selecionando texto quando a caixa de texto é destacada

Para selecionar todo o texto quando o usuário insere a caixa de texto com o teclado, defina a propriedade SelectOnEntry como verdadeira (.T.).

Formatando o texto em uma caixa de texto

Você pode utilizar a propriedade InputMask para determinar os valores que podem ser digitados na caixa de texto e a propriedade Format para determinar o modo como os valores são exibidos na caixa de texto.

Utilizando a propriedade InputMask

A propriedade InputMask determina as características de cada caractere digitado na caixa de texto. Por exemplo, você pode definir a propriedade InputMask como 999.999,99 para limitar a entrada a valores numéricos inferiores a 1.000.000, com duas casas decimais. A vírgula e o ponto serão exibidos na caixa de texto antes que o usuário digite qualquer valor. Se o usuário pressionar uma tecla de caractere, o caractere não será exibido na caixa de texto.

Se você tiver um campo lógico e quiser que o usuário possa digitar "S" ou "N", mas não "T" ou "F", defina a propriedade InputMask com "S".

Aceitando senhas do usuário em uma caixa de texto

Com freqüência um aplicativo desejará obter informações confidenciais do usuário, como uma senha. Você pode utilizar uma caixa de texto para obter essas informações sem torná-las visíveis na tela.

Para aceitar entrada do usuário sem exibir o valor real

- Defina a propriedade PasswordChar da caixa de texto como * ou outro caractere genérico.

Se você definir a propriedade PasswordChar como qualquer valor que não uma seqüência vazia, as propriedades Value e Text da caixa de texto conterão o valor real digitado pelo usuário na caixa de texto, mas a caixa de texto exibirá um caractere genérico para cada tecla pressionada pelo usuário.

Digitando datas em uma caixa de texto

As caixas de texto possuem várias propriedades que você pode definir para facilitar a seus usuários a inserção de valores de data.

Propriedade	Descrição

Century	Define se os dois primeiros dígitos do ano são exibidos ou não.
DateFormat	Formata a data na caixa de texto como um dos quinze formatos pré-determinados, como americano, alemão, japonês.
StrictDateEntry	Definir StrictDateEntry como 0 - Livre permitirá que um usuário digite datas em formatos mais flexíveis do que o padrão de 99/99/99.

Propriedades comuns de caixa de texto

As propriedades de caixa de texto abaixo são normalmente definidas na hora da criação.

Propriedade	Descrição
Alignment	Se o conteúdo da caixa de texto é alinhado à esquerda ou à direita, centralizado ou automático. O alinhamento automático depende do tipo de dado. Os números, por exemplo, são alinhados à direita e os caracteres são alinhados à esquerda.
ControlSource	O campo ou variável de tabela cujo valor é exibido na caixa de texto.
InputMask	Especifica a regra para entrada de dados que cada caractere digitado deve seguir. Para obter informações específicas sobre InputMask, consulte a Ajuda.
SelectOnEntry	Se o conteúdo da caixa de texto é selecionado automaticamente quanto a caixa de texto recebe o foco.
TabStop	Se o usuário pode utilizar a tabulação para acessar o controle. Se TabStop estiver definida como .F., o usuário ainda poderá selecionar a caixa de texto, clicando sobre ela.

Utilizando caixas de edição

Você pode permitir que o usuário edite o texto de campos caractere extensos ou de campos memo em caixas de edição. As caixas de edição permitem a quebra de linha automática e oferecem ao usuário a possibilidade de movimentar-se pelo texto utilizando as teclas de direção, as teclas PAGE UP e PAGE DOWN e as barras de rolagem.

Para ver exemplos de como utilizar caixas de edição, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e, em seguida, escolha Editbox na seção Controls.

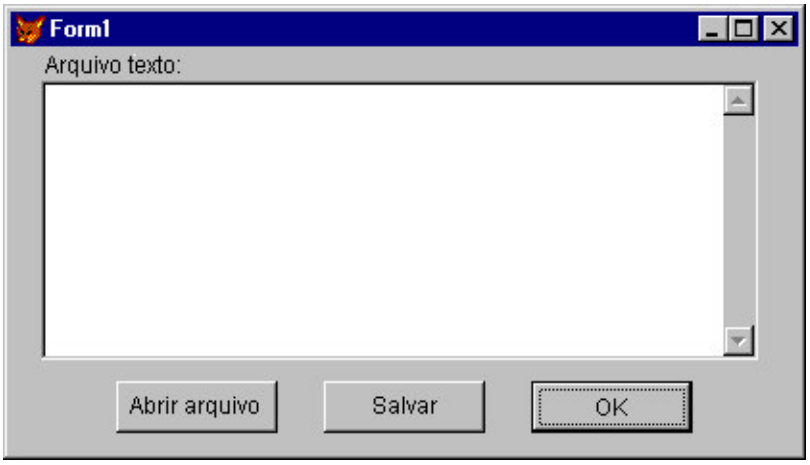
Permitindo que o usuário edite um campo memo em uma caixa de edição

Tudo o que você precisa fazer para permitir que o usuário edite um campo memo em uma caixa de edição é definir a propriedade ControlSource da caixa de edição como o campo memo. Por exemplo, se existir um campo memo denominado `comments` em um tabela chamada `log`, você poderá definir a propriedade ControlSource de uma caixa de edição como `log.comments`, para permitir que o usuário edite o campo memo na caixa de edição.

Permitindo que o usuário edite um arquivo de texto em uma caixa de edição

Você pode permitir também que o usuário edite um arquivo de texto em uma caixa de edição. O formulário abaixo demonstra isso.

Formulário de exemplo para editar um arquivo de texto em uma caixa de edição



Um botão OK no formulário o fecha com o seguinte comando no código do evento Click:

```
RELEASE THISFORM
```

Os outros dois botões neste exemplo, `cmdOpenFile` e `cmdSave`,permitem que o usuário abra um arquivo de texto e salve o arquivo após as edições.

Código associado ao evento Click de cmdOpenFile

Código	Comentários
`CREATE CURSOR textfile ;` `  (filename c(35), mem m)`	Criar um Cursor com um campo de caractere para conter o nome do arquivo de texto e um campo memo para conter o conteúdo do arquivo de texto.
`APPEND BLANK`	Adiciona um registro em branco ao Cursor.
`REPLACE textfile.FileName WITH ;` `  GETFILE("TXT")`	Utilizar a função GETFILE( ) para retornar o nome do arquivo a ser aberto. Armazenar o nome no campo FileName do Cursor.

`IF EMPTY(textfile.FileName)`
`  RETURN`
`ENDIF`
	Se o usuário escolher **Cancelar** na caixa de diálogo **Get File**, o campo **FileName** ficará vazio e não haverá arquivo a ser aberto.

`APPEND MEMO mem FROM ;` `  (textfile.FileName) OVERWRITE`	Preencher o campo memo com o texto do arquivo.
`THISFORM.edtText.ControlSource = ;` `  "textfile.mem"`	Definir a ControlSource da caixa de edição no formulário.
`THISFORM.Refresh`	
`THISFORM.cmdSave.Enabled = .T.`	Ativar o botão Salvar.

Depois que o arquivo foi aberto e editado, o botão Salvar permite que o usuário grave as alterações de volta no arquivo.

Código associado ao Evento Click de cmdSave

Código	Comentários
`COPY MEMO textfile.mem TO ;` `  (textfile.filename)`	Sobrescreve o valor antigo no arquivo com o texto no campo memo.

Manipulando texto selecionado em uma caixa de edição

As caixas de edição e as caixas de texto possuem três propriedades que lhe permitem trabalhar com texto selecionado: SelLength, SelStart e SelText.

Para selecionar texto por meio do programa, utilize as propriedades SelStart e SelLength. Por exemplo, as linhas de código abaixo selecionam a primeira palavra em uma caixa de edição.

```
Form1.edtText.SelStart = 0
Form1.edtText.SelLength = AT(" ", Form1.edtText.Text) - 1
```

Dica Quando você altera a propriedade SelStart, a caixa de edição rola para exibir a nova SelStart. Se você alterar SelStart em um loop, como por exemplo ao pesquisar texto, o seu código será executado mais rapidamente se você incluir `THISFORM.LockScreen = .T.` antes do processamento e `THISFORM.LockScreen = .F.` depois do processamento.

Você pode acessar texto selecionado em uma caixa de edição ou caixa de texto, com a propriedade SelText. Por exemplo, a linha de código a seguir irá colocar todo o texto selecionado em letras maiúsculas:

```
Form1.edtText.SelText = UPPER(Form1.edtText.SelText)
```

Propriedades comuns de caixa de edição

As propriedades de caixa de edição a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
AllowTabs	Se o usuário pode inserir tabulações na caixa de edição em vez de ir para o controle seguinte. Se você permitir tabulações, certifique-se de indicar que o usuário pode ir para o controle seguinte pressionando CTRL+TAB.
HideSelection	Se o texto selecionado na caixa de edição fica selecionado visivelmente quando a caixa de edição não está destacada..
ReadOnly	Se o usuário pode alterar o texto na caixa de edição.
ScrollBars	Se existem barras de rolagem verticais.

Utilizando caixas de combinação

O controle de caixa de combinação tem a funcionalidade de uma caixa de listagem e de uma caixa de texto. Existem dois estilos de caixa de combinação: Caixa de combinação suspensa e Caixa de listagem suspensa. Especifique qual deles deseja, alterando a propriedade Style do controle. As listas suspensas são descritas em Utilizando caixas de listagem e caixas de listagem suspensas anteriormente neste capítulo.

Caixa de combinação suspensa

O usuário pode clicar no botão de uma caixa de combinação suspensa para ver uma lista de opções, ou digitar um novo item diretamente na caixa ao lado do botão. A propriedade Style padrão de uma caixa de combinação é 0(Caixa de combinação suspensa).

Adicionando itens do usuário a listas de combinação suspensas

Para adicionar o novo valor do usuário à caixa de combinação suspensa, você pode utilizar a linha de código a seguir no método associado ao evento Valid da caixa de combinação:

```
THIS.AddItem(THIS.Text)
```

No entanto, antes de adicionar um item, seria recomendável verificar se o valor já não consta da caixa de combinação suspensa:

```
lItemExists = .F. && considerar que o valor não está na lista.
FOR i = 1 to THIS.ListCount
IF THIS.List(i) = THIS.Text
```

```
lItemExists = .T.
EXIT
ENDIF
ENDFOR

IF !lItemExists
   THIS.AddItem(THIS.Text)
ENDIF
```

Propriedades comuns de caixa de combinação

As propriedades de caixa de combinação a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
ControlSource	Especifica o campo da tabela onde é armazenado o valor que o usuário seleciona ou digita.
InputMask	Para caixas de combinação suspensas, especifica o tipo de valor que pode ser digitado.
IncrementalSearch	Especifica se o controle tenta comparar um item da lista à medida que o usuário digita cada letra.
RowSource	Especifica a origem dos itens da caixa de combinação.
RowSourceType	Especifica o tipo da fonte da caixa de combinação. Os valores RowSourceType de uma caixa de combinação são os mesmos dos de uma lista. Para obter uma explicação sobre cada um, consulte a Ajuda ou a explicação sobre caixas de listagem anteriormente neste capítulo.
Style	Especifica se a caixa de combinação é uma caixa de combinação suspensa ou uma listagem suspensa.

Aceitando entrada numérica em um intervalo específico

Embora seja possível definir a propriedade InputMask e incluir código no evento Valid para garantir que os valores numéricos digitados em caixas de texto fiquem dentro de um intervalo específico, a maneira mais fácil de verificar o intervalo de valores é utilizar um controle de rotação.

Utilizando controles de rotação

Você pode utilizar controles de rotação para permitir que o usuário faça seleções “girando” pelos valores ou digitando valores diretamente na caixa de controle rotação.

Definindo o intervalo de valores que o usuário pode selecionar

Defina as propriedades KeyboardHighValue e SpinnerHighValue como o maior valor que os usuários poderão digitar no controle de rotação.

Defina as propriedades KeyboardLowValue e SpinnerLowValue como o menor valor que os usuários poderão digitar no controle de rotação.

Decrementando um controle de rotação quando o usuário clica sobre o botão Acima

Às vezes, se o controle de rotação refletir um valor como “prioridade”, você desejará que o usuário possa aumentar a prioridade de 2 para 1, clicando sobre o botão Acima. Para fazer com que o número do controle de rotação seja decrementado quando o usuário clicar sobre o botão Acima, defina a propriedade Increment como -1.

Girando por valores não-numéricos

Embora o valor de um controle de rotação seja numérico, você pode utilizar o controle de rotação e uma caixa de texto para permitir que o usuário gire por vários tipos de dados. Por exemplo, se você quiser que o usuário gire por um intervalo de datas, poderá dimensionar o controle de rotação de modo que apenas os botões fiquem visíveis, e posicionar a caixa de texto ao lado dos botões do controle de rotação. Defina a propriedade Value da caixa de texto como uma data e, nos eventos UpClick e DownClickdo controle de rotação, incremente ou decremente a data.

Dica Você pode utilizar a função API GetSystemMetrics do Windows para definir a largura do seu controle de rotação, de modo que apenas os botões sejam visíveis e tenham a melhor largura para a exibição do bitmap das setas acima e abaixo.

1. Defina a propriedade BorderStyle do controle de rotação como 0.
2. Inclua o código abaixo na Init do controle de rotação:

```
DECLARE INTEGER GetSystemMetrics IN Win32api INTEGER
THIS.Width = GetSystemMetrics(2) && SM_CXVSCROLL
```

Propriedades comuns de controle de rotação

As propriedades de controle de rotação a seguir são normalmente definidas em tempo de criação.

Propriedade	Descrição
Interval	De quanto incrementar ou decrementar o valor sempre que o usuário clicar sobre os botões Acima ou Abaixo.
KeyboardHighValue	O maior valor que pode ser digitado na caixa de texto do controle de rotação.
KeyboardLowValue	O menor valor que pode ser digitado na caixa de texto do controle de rotação.
SpinnerHighValue	O maior valor que o controle de rotação exibirá quando o usuário clicar sobre o botão Acima.
SpinnerLowValue	O menor valor que o controle de rotação irá exibir quando o usuário clicar sobre o botão Abaixo.

Permitindo ações específicas

Em várias ocasiões, você quer permitir que o usuário adote ações específicas sem qualquer relação com a manipulação de valores. Por exemplo, você pode permitir que o usuário feche um formulário, abra outro formulário, mova-se pela tabela, salve ou cancele edições, execute um relatório ou consulta ou várias outras ações.

Utilizando botões de comando e grupos de botões de comando

Um dos locais mais comuns para se colocar código de ações específicas é o evento Click de um botão de comando.

Tornando o botão de comando a opção padrão

Defina a propriedade Default como verdadeira (.T.), para tornar o botão de comando a opção padrão. A opção padrão tem uma borda mais espessa que a dos outros botões de comando. Quando um botão de comando é a opção padrão, seu evento Click será executado quando o usuário pressionar ENTER.

Observação Se o objeto selecionado em um formulário for uma caixa de edição ou uma grade, o

código associado ao evento Click da opção padrão não será executado quando o usuário pressionar ENTER. Quando se pressiona ENTER em uma caixa de edição, um retorno de carro e uma alimentação de linha são adicionados ao valor dela. Quando se pressiona ENTER em uma grade, um campo adjacente é selecionado. Para executar o evento Click do botão padrão, pressione CTRL+ENTER.

Propriedades comuns de botão de comando

As propriedades de botão de comando a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
Cancel	Especifica que o código associado ao evento Click do botão de comando será executado quando o usuário pressionar ESC.
Caption	Texto exibido no botão.
DisabledPicture	Arquivo .BMP exibido quando o botão está desativado.
DownPicture	Arquivo .BMP exibido quando o botão é pressionado.
Enabled	Se o botão pode ou não ser selecionado.
Picture	Arquivo .BMP exibido no botão.

Também é possível incluir botões de comando em um grupo para que você possa manipulá-los individualmente ou como um grupo.

Gerenciando opções de botão de comando em nível do grupo

Se você quiser trabalhar com um único procedimento de método para todos os códigos de eventos Click dos botões de comando de um grupo, poderá anexar o código ao evento Click do grupo de botões de comando. A propriedade Value do grupo de botões de comando indica o botão que foi clicado, conforme demonstrado no exemplo de código a seguir:

```
DO CASE
   CASE THIS.Value = 1
      WAIT WINDOW "Você clicou em " + THIS.cmdCommand1.Caption NOWAIT
      * executa uma ação
   CASE THIS.Value = 2
      WAIT WINDOW "Você clicou em " + THIS.cmdCommand2.Caption NOWAIT
      * executa uma outra ação
   CASE THIS.Value = 3
      WAIT WINDOW "Você clicou em " + THIS.cmdCommand3.Caption NOWAIT
      * executa uma terceira ação
ENDCASE
```

Observação Se o usuário clicar sobre o grupo de botões de comando, mas não sobre um botão específico, a propriedade Value continuará mostrando o último botão de comando selecionado.

Se você escreveu um código para o evento Click de um botão específico do grupo, esse código será executado, em vez do código de evento Click do grupo, quando o usuário selecionar aquele botão.

Propriedades comuns de grupo de botões de comando

As propriedades de grupo de botões de comando a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
ButtonCount	Número de botões de comando no grupo.
BackStyle	Se o grupo de botões de comando tem fundo transparente ou opaco. Um fundo transparente parece ter a mesma cor do objeto subjacente, geralmente um formulário ou página.

Executando ações específicas a intervalos determinados

O controle Timer lhe permite executar ações ou verificar valores a intervalos específicos.

Utilizando o controle Timer

Os controles Timer respondem à passagem do tempo independentemente da interação do usuário. Portanto, você pode programá-los para executar ações a intervalos regulares. É comum utilizá-los na verificação do relógio do sistema para ver se está na hora de executar uma determinada tarefa. Os cronômetros também são úteis em outros tipos de processamento em segundo plano.

Para ver exemplos de como utilizar cronômetros, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e escolha Timer na seção Controls.

Cada cronômetro possui uma propriedade Interval, que especifica o número de milissegundos decorridos entre um evento do cronômetro e o seguinte. A menos que esteja desativado, o cronômetro continuará a receber um evento (apropriadamente denominado evento Timer) a intervalos de tempo mais ou menos iguais. A propriedade Interval tem algumas limitações que devem ser consideradas quando você estiver programando um cronômetro:

- O intervalo pode ir de 0 a 2.147.483.647, inclusive, o que significa que o intervalo mais longo será de aproximadamente 596,5 horas (mais de 24 dias).
- Não existe garantia de que o intervalo transcorrerá em período exato de tempo. Para assegurar a precisão, o cronômetro deve verificar o relógio do sistema quando precisar, ao invés de tentar manter controle interno do tempo acumulado.
- O sistema gera 18 tiques de relógio por segundo, portanto, embora a propriedade Interval seja medida em milissegundos, a verdadeira precisão de um intervalo é, no máximo, um dezoito avos de segundo.
- Se o seu aplicativo, ou outro, estiver fazendo grandes exigências ao sistema—como loops longos, cálculos extensos ou acesso a porta, rede ou disco—o aplicativo pode não obter eventos do cronômetro com a freqüência especificada pela propriedade Interval.

Colocando um controle Timer em um formulário

Colocar um controle Timer em um formulário é como desenhar qualquer outro controle: selecione a ferramenta cronômetro na **Barra de ferramentas controles**, clique e arraste-a para o formulário.

Um controle Timer

O cronômetro aparece no formulário na hora da criação para que você possa selecioná-lo, visualizar suas propriedades e escrever um procedimento de evento para ele. Em tempo de execução ele fica invisível, e a sua posição e o seu tamanho são irrelevantes.

Inicializando um controle Timer

Um controle Timer possui duas propriedades básicas:

Propriedade	Definição
Enabled	Se você quiser que o cronômetro comece a funcionar assim que o formulário for carregado, defina como

	verdadeira (.T.). Caso contrário, deixe esta propriedade definida como falsa (.F.). Você pode optar por fazer com que um evento externo (como um clique em um botão de comando) inicie a operação do cronômetro.
Interval	Número de milissegundos entre eventos do cronômetro.

Observe que a propriedade Enabled do cronômetro é diferente da de outros objetos. Para a maioria dos objetos, a propriedade Enabled determina se o objeto pode responder a um evento gerado pelo usuário. Com o controle Timer, se Enabled for definida como falsa (.F.), a operação do cronômetro será suspensa.

Lembre-se de que o evento Timer é periódico. A propriedade Interval não determina “por quanto tempo”, mas sim “com que freqüência”. A duração do intervalo depende da precisão desejada. Visto que existe um certo potencial interno de erro, faça intervalos de metade da precisão desejada.

Observação Quanto maior a freqüência com que um evento de cronômetro é gerado, mais tempo do processador é consumido para responder ao evento. Isto pode reduzir o desempenho geral. Não defina um intervalo muito pequeno, a menos que seja necessário.

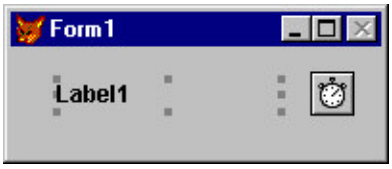
Respondendo ao evento Timer

Decorrido o intervalo do controle Timer, o Visual FoxPro gera o evento Timer. Normalmente, você responde a este evento verificando alguma condição geral, como o relógio do sistema.

O relógio digital é um aplicativo bastante simples, porém extremamente útil, que envolve um controle Timer. Depois de compreender o funcionamento do aplicativo, você poderá aprimorá-lo para que funcione como um alarme, um cronômetro ou algum outro dispositivo de temporização.

O aplicativo do relógio digital inclui um cronômetro e um rótulo com uma borda. Na hora da criação, o aplicativo tem a seguinte aparência:

Aplicativo do relógio digital



Em tempo de execução, o cronômetro fica invisível.

Controle	Propriedade	Definição
lblTime	Caption	
Timer1	Interval	500 (meio segundo)
Timer1	Enabled	Verdadeiro

O único procedimento no aplicativo é o procedimento de evento Timer:

```
IF THISFORM.lblTime.Caption != Time()
THISFORM.lblTime.Caption = Time()
ENDIF
```

A propriedade Interval do cronômetro está definida como 500, seguindo a regra de definir o intervalo como a metade do menor período que você deseja distinguir (um segundo, neste caso). Isto pode fazer com que o código do cronômetro atualize o rótulo com a mesma hora duas vezes em um segundo. Isto pode gerar oscilações perceptíveis na tela, e por isso o código realiza testes para verificar se a hora está diferente do que está exibido no rótulo, antes de alterar a legenda.

Exibindo informações

Um dos princípios de uma boa criação é tornar visíveis informações relevantes. Os controles a seguir podem ser utilizados para exibir informações para os usuários:

- Imagens
- Rótulos
- Caixas de texto
- Caixas de edição
- Formas

Utilizando imagens

O controle Image lhe permite adicionar figuras (arquivos .BMP) a um formulário. Um controle Image possui o conjunto completo das propriedades, eventos e métodos de outros controles, assim, ele pode ser alterado dinamicamente em tempo de execução. O usuário pode interagir com imagens clicando, clicando duas vezes, etc.

A tabela abaixo lista algumas das propriedades principais de um controle de imagem.

Propriedade	Descrição
Picture	A figura (arquivo .BMP) a ser exibida.
BorderStyle	Se existe ou não uma borda visível para a imagem.
Stretch	Se Stretch estiver definida como 0–Cortar, partes da figura que ultrapassam as dimensões do controle Image não serão exibidas. Se Stretch estiver definida como 1–Isométrico, o controle Image preservará as dimensões originais da figura .BMP e exibirá o máximo permitido pelas dimensões do controle Image. Se Stretch estiver definida como 2–Estender, a figura será ajustada para corresponder exatamente à altura e à largura do controle Image.

Utilizando rótulos

Os rótulos diferem das caixas de texto, pois:

- Não podem ter uma fonte de dados.
- Não podem ser editados diretamente.
- Não é possível tabular até eles.

É possível alterar, através da linguagem de programação, as propriedades Caption e Visible de um rótulo para adaptar a exibição do rótulo à presente situação.

Propriedades comuns de rótulos

As propriedades de rótulo abaixo são normalmente definidas na hora da criação.

Propriedade	Descrição
Caption	O texto exibido pelo rótulo.
AutoSize	Se o tamanho do rótulo é ajustado ao comprimento da legenda.
BackStyle	Se o rótulo é Opaco ou Transparente.
WordWrap	Se o texto exibido no rótulo pode ser quebrado em

linhas adicionais.

Utilizando caixas de texto e de edição para exibir informações

Defina a propriedade ReadOnly de caixas de texto e de edição para exibir informações que o usuário pode visualizar, mas não pode editar. Se você desativar somente uma caixa de edição, o usuário não poderá percorrer o texto.

Utilizando formas e linhas

Formas e linhas ajudam a agrupar visualmente elementos do formulário. As pesquisas demonstraram que a associação de itens relacionados ajuda o usuário a aprender e entender uma interface, o que facilita o uso do aplicativo.

As propriedades de Forma a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
Curvature	Um valor entre 0 (ângulos de 90 graus) e 99 (círculo ou oval).
FillStyle	Se a forma é transparente ou tem o padrão de preenchimento do segundo plano especificado.
SpecialEffect	Se a forma é simples ou 3D. Isso só tem efeito quando a propriedade Curvature está definida como 0.

As propriedades de Linha a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
BorderWidth	Quantos pixels de largura tem uma linha.
LineSlant	Quando a linha não é horizontal ou vertical, a direção da inclinação. Os valores válidos para esta propriedade são barra (/) e barra invertida (\ ).

Utilizando gráficos de formulário para exibir informações

Você pode exibir graficamente informações em um formulário utilizando os métodos de formulário a seguir.

Método	Descrição
Circle	Desenha uma figura circular ou um arco em um formulário.
Cls	Limpa gráficos e texto de um formulário.
Line	Desenha uma linha em um formulário.
Pset	Define um ponto em um formulário como uma cor específica.
Print	Imprime uma seqüência de caracteres em um formulário.

Para ver exemplos que demonstram elementos gráficos de formulário, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e, em seguida, escolha Form Graphics da seção Control.

Aprimorando a exibição de controles

Botões de comando, caixas de verificação e botões de opção podem exibir figuras além de uma

legenda. Todos estes controles possuem propriedades que permitem especificar figuras a serem exibidas nos controles:

Propriedade	Descrição
DisabledPicture	Figura exibida no botão quando o botão está desativado.
DownPicture	Figura exibida no botão quando o botão está pressionado.
Picture	Figura exibida no botão quando o botão está ativado e não está pressionado.

Se você não especificar um valor DisabledPicture, o Visual FoxPro exibirá a figura acinzentada quando o controle estiver desativado. Se você não especificar um valor DownPicture, o Visual FoxPro exibirá a figura com as cores do segundo plano alteradas, para que o botão pareça estar pressionado quando o for.

Se você não quiser exibir uma legenda além da figura, defina a propriedade Caption como uma seqüência vazia, excluindo a legenda padrão na caixa Edição de Propriedade da janela **Propriedades**.

Utilizando máscaras de figura

Uma figura .BMP freqüentemente contém espaço em branco que você não deseja que apareça nos controles. Uma borda branca ao redor de uma imagem com forma irregular pode dar aparência indesejada ao seu controle. Para evitar esse problema, o Visual FoxPro cria uma máscara padrão temporária para a figura. As áreas brancas recebem um atributo transparente para que a cor subjacente do botão ou do segundo plano apareçam. Para manter brancas algumas áreas brancas do .BMP, crie uma máscara para ele que substituirá o padrão.

Para criar uma máscara para um .BMP

1. Abra o arquivo .BMP no Microsoft Paint ou em outro utilitário de bitmap.
2. Enegreça todas as áreas da figura que você deseja exibir exatamente como estão no arquivo .BMP. Deixe brancas as áreas que você deseja que fiquem transparentes.
3. Salve o arquivo no mesmo diretório e com o mesmo nome do arquivo .BMP, mas com uma extensão .MSK.

Quando o Visual FoxPro carrega um arquivo .BMP especificado pela propriedade Picture para um botão de comando, botão de opção ou caixa de verificação, ele procura um arquivo .MSK correspondente no mesmo diretório. Caso exista um arquivo .MSK com o mesmo nome do .BMP no diretório, o Visual FoxPro o utilizará como máscara da figura. Todas as áreas brancas na figura .MSK ficarão transparentes no .BMP. Todas as áreas pretas na figura .MSK serão exibidas exatamente como estavam no .BMP.

Observação A figura .BMP e a figura .MSK devem ter as mesmas dimensões para que a máscara possa representar a área . do BMP.

Manipulando várias linhas de dados

O Visual FoxPro fornece uma ferramenta bastante poderosa—o objeto grade—para exibir e manipular várias linhas de dados.

Utilizando grades

A grade é um objeto recipiente. Da mesma forma que um conjunto de formulários pode conter formulários, uma grade pode conter colunas. Além disso, as colunas contêm cabeçalhos e controles, cada um com os seus próprios conjuntos de propriedades, eventos e métodos, proporcionando um

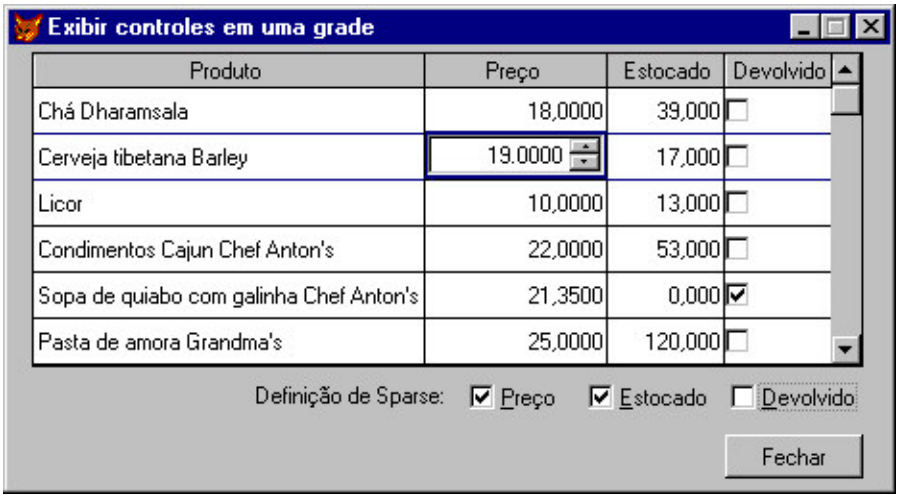
grande controle sobre os elementos da grade.

Recipiente	Pode conter
Grade	Colunas
Coluna	Cabeçalhos, controles

O objeto Grade lhe permite apresentar e manipular linhas e colunas de dados em um formulário ou página. Uma aplicação particularmente útil do controle Grid é criar formulários um para n, como um formulário de fatura.

Para ver exemplos de como utilizar grades, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e, em seguida, escolha Grid na seção Controls.

Um formulário com uma grade preenchida



Para adicionar um controle Grid a um formulário

- Na **Barra de ferramentas controles de formulário**, selecione o botão **Grade** e arraste para dimensionar na janela **Formulário**.

Se você não especificar um valor de RecordSource para a grade e uma tabela estiver aberta na área de trabalho atual, a grade exibirá todos os campos dessa tabela.

Definindo o número de colunas de uma grade

Uma das primeiras propriedades que convém definir para o controle Grid é o número de colunas.

Para definir o número de colunas em uma grade

1. Selecione a propriedade ColumnCount na lista **Propriedade e Métodos**.
2. Na caixa **Propriedade**, digite o número de colunas desejado.

Se a propriedade ColumnCount estiver definida como –1 (o padrão), a grade terá, em tempo de execução, um número de colunas equivalente ao número de campos na tabela associada à grade.

Ajustando manualmente a exibição da grade na hora da criação

Após adicionar colunas à grade, você poderá alterar a largura das colunas e a altura das linhas. Você pode definir manualmente as propriedades de altura e largura dos objetos linha e coluna na janela **Propriedades** ou definir visualmente estas propriedades no modo criação de grade.

Para alternar para o modo criação de grade

- Selecione **Editar** no menu de atalho da grade.

  – Ou –

- Na caixa **Objeto** da janela **Propriedades**, selecione uma coluna da grade.

Quando você está no modo criação de grade, uma borda espessa é exibida ao redor da grade. Para sair do modo criação de grade, selecione o formulário ou outro controle.

Para ajustar a largura das colunas em uma grade

1. No modo criação de grade, posicione o ponteiro do mouse entre os cabeçalhos das colunas da grade de modo que o ponteiro se transforme em uma barra com setas apontando para a direita e para a esquerda.
2. Selecione a coluna e arraste até que ela atinja a largura desejada.

   – Ou –

   Defina a propriedade Width da coluna na janela **Propriedades**.

Para ajustar a altura das linhas em uma grade

1. No modo criação de grade, posicione o ponteiro do mouse entre o primeiro e o segundo botão no lado esquerdo do controle **Grid** para que o ponteiro se transforme em uma barra com setas apontando para cima e para baixo.
2. Selecione a linha e arraste até que ela atinja a altura desejada.

   – Ou –

   Defina a propriedade Height da coluna na janela **Propriedades**.

Dica Você pode evitar que um usuário altere a altura das linhas de grade em tempo de execução, definindo AllowRowSizing como falsa (.F.).

Definindo a fonte dos dados exibidos na grade

É possível definir a fonte de dados da grade e de cada coluna individualmente.

Para definir a fonte de dados de uma grade

1. Selecione a grade e, em seguida, clique sobre a propriedade RecordSourceType na janela **Propriedades**.
2. Defina a propriedade RecordSourceType como **0 - Tabela**, se desejar que o Visual FoxPro abra a tabela, ou como **1 - Alias** se desejar que a grade seja ocupada com os campos de uma tabela que já esteja aberta.
3. Clique sobre a propriedade RecordSource na janela **Propriedades**.
4. Digite o nome do alias ou tabela que deve atuar como fonte de dados da grade.

Se quiser especificar campos específicos a serem exibidos em colunas específicas, você poderá definir também a fonte de dados de uma coluna.

Para definir a fonte de dados de uma coluna

1. Selecione a coluna e, em seguida, clique sobre a propriedade ControlSource na janela **Propriedades**.
2. Digite o nome do alias ou tabela, e o campo que deve atuar como fonte dos valores exibidos na coluna. Por exemplo, você pode digitar:

   ```
   Orders.order_id
   ```

Adicionando registros a uma grade

Você pode permitir que os usuários adicionem novos registros a uma tabela exibida em uma grade, definindo a propriedade AllowAddNew da grade como verdadeira (.T.). Quando a propriedade AllowAddNew for definida como verdadeira, novos registros serão adicionados à tabela quando o último registro estiver selecionado e o usuário pressionar a tecla SETA ABAIXO.

Se desejar ter mais controle sobre o momento em que um usuário adiciona um novo registro a uma tabela, poderá definir a propriedade AllowAddNew como falsa (.F.), o padrão, e utilizar os comandos APPEND BLANK ou INSERT para adicionar novos registros.

Configurando um formulário um para n utilizando o controle de grade

Um dos usos mais comuns de uma grade é o de exibir os registros filho de uma tabela, enquanto as caixas de texto exibem os dados dos registros pai. Quando o usuário percorre os registros da tabela pai, a grade exibe os registros filho apropriados.

Se há um ambiente de dados para seu relatório que inclui um relacionamento de um para n entre duas tabelas, é muito fácil exibir o relacionamento um para n no formulário.

Para configurar um formulário um para n com um ambiente de dados

1. Arraste os campos desejados da tabela pai do **Criador de ambientes de dados** para o formulário.
2. Arraste a tabela relacionada do **Criador de ambientes de dados** para o formulário.

Quase sempre, você desejará criar um ambiente de dados para seu formulário ou conjunto de formulários. No entanto, não é tão complicado criar um formulário um para n sem utilizar o **Criador de ambientes de dados**.

Para configurar um formulário um para n sem criar um ambiente de dados

1. Adicione caixas de texto ao formulário para exibir os campos desejados da tabela primária.
2. Defina a propriedade ControlSource das caixas de texto como a tabela primária.
3. Adicione uma grade ao formulário.
4. Defina a propriedade RecordSource da grade como o nome da tabela relacionada.
5. Defina a propriedade LinkMaster da grade como o nome da tabela primária.
6. Defina a propriedade ChildOrder da grade como o nome da marca de índice da tabela relacionada correspondente à expressão relacional da tabela primária.
7. Defina a propriedade RelationalExpr da grade como a expressão que associa a tabela relacionada à tabela primária. Por exemplo, se a marca ChildOrder estiver indexada em `"lastname + firstname"`, defina RelationalExpr como a mesma expressão.

Qualquer que seja o modo utilizado para configurar o formulário um para n, você pode adicionar controles de navegação para percorrer a tabela pai e atualizar os objetos do formulário. Por exemplo, o código abaixo poderia ser incluído no evento Click de um botão de comando:

```
SELECT orders && se orders for a tabela pai
SKIP
IF EOF( )
GO BOTTOM
ENDIF
THISFORM.Refresh
```

Exibindo controles em colunas da grade

Além de exibir dados de campo em uma grade, você pode ter controles nas colunas de uma grade de forma a poder apresentar ao usuário caixas de texto, caixas de verificação, caixas de listagem suspensas, controles de rotação e outros controles. Por exemplo, se existir um campo lógico em uma tabela, quando você executar o formulário o usuário saberá quais valores de registro são verdadeiros (.T.) e quais são falsos (.F.), examinando se a caixa de verificação está definida. Alterar o valor é tão fácil quanto definir ou limpar a caixa de verificação.

Você pode adicionar controles a colunas de grade interativamente no **Criador de formulários** ou escrever um código para adicionar os controles às colunas em tempo de execução.

Para adicionar controles interativamente a uma coluna da grade

1. Adicione uma grade a um formulário.
2. Na janela **Propriedades**, defina a propriedade ColumnCount da grade como o número de colunas desejadas.

   Por exemplo, digite **2** para uma grade com duas colunas.

3 Na janela **Propriedades** selecione a coluna pai para o controle na caixa Objeto.

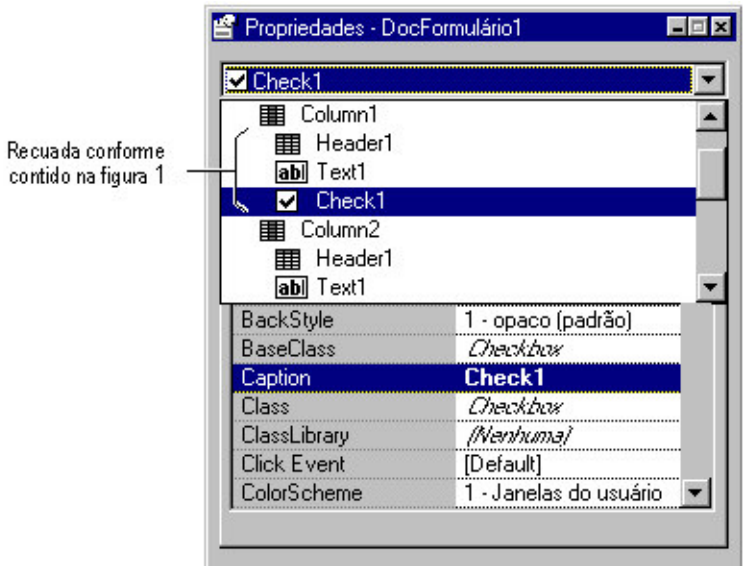
Por exemplo, selecione Coluna1 para adicionar um controle à Coluna1. A borda da grade muda para indicar que você está editando um objeto contido nela quando seleciona a coluna.

4 Selecione o controle desejado na **Barra de ferramentas controles de formulário** e clique sobre a coluna pai.

O novo controle não será exibido na coluna da grade no **Criador de formulários**, mas estará visível em tempo de execução.

5 Na janela **Propriedades**, certifique-se de que o controle seja exibido recuado sob a coluna pai na caixa Objeto.

Uma caixa de verificação adicionada a uma coluna de grade



Se o novo controle for uma caixa de verificação, defina a propriedade Caption da caixa de verificação como " " e a propriedade Sparse da coluna como falsa (.F.).

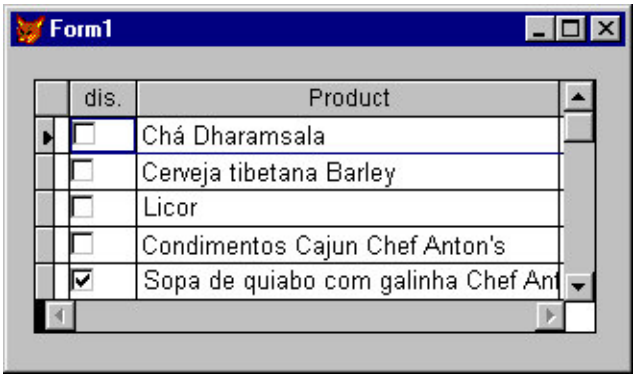
6 Defina a propriedade ControlSource da coluna pai como o campo desejado da tabela.

Por exemplo, a ControlSource da coluna na ilustração a seguir é `products.discontinu` do TESTDATA.DBC em VFP\SAMPLES\DATA.

7 Defina a propriedade CurrentControl da coluna pai como o novo controle.

Quando você executar o formulário, o controle será exibido na coluna da grade.

A caixa de verificação é exibida na coluna em tempo de execução.



Dica Para poder centralizar uma caixa de verificação em uma coluna de grade, crie uma classe recipiente, adicione-lhe uma caixa de verificação e ajuste a posição da caixa na classe recipiente.

Adicione a classe recipiente à coluna de grade e defina a ControlSource da caixa de verificação como o campo desejado.

Para remover controles de colunas de grade no Criador de formulários

1 Na caixa **Objeto** da janela **Propriedades**, selecione o controle.

2 Ative o **Criador de formulários**.

Se a janela **Propriedades** estiver visível, o nome do controle será exibido na caixa **Objeto**.

3 Pressione a tecla DELETE.

Você também pode adicionar controles a uma coluna de grade utilizando o método AddObject no código.

Para adicionar controles a uma coluna de grade por programa

- No evento Init da grade, utilize o método AddObject para adicionar o controle à coluna da grade e defina a propriedade CurrentControl da coluna.

Por exemplo, as linhas de código a seguir no evento Init de uma grade adicionam dois controles a uma coluna da grade e especificam um deles como o controle atual:

```
THIS.grcColumn1.AddObject("spnQuantity", "SPINNER")
THIS.grcColumn1.AddObject("cboQuantity", "COMBOBOX")
THIS.grcColumn1.CurrentControl = "spnQuantity"
* As linhas de código a seguir asseguram que o controle seja visível
* e seja exibido em cada linha da grade
THIS.grcColumn1.spnQuantity.Visible = .T.
THIS.grcColumn1.Sparse = .F.
```

Neste exemplo, Column1 tem três valores de controle atuais possíveis:

- `spnQuantity`
- `cboQuantity`
- `Text1` (o controle padrão)

Observação As propriedades definidas no nível de grade não são passadas para as colunas ou cabeçalhos. Da mesma maneira, você deve definir diretamente as propriedades dos cabeçalhos e os controles nele contidos; eles não herdam suas propriedades das definições feitas no nível da coluna.

Dica Para obter a melhor exibição das caixas de combinação em colunas de grade, defina as seguintes propriedades de caixa de combinação:

```
BackStyle = 0      && Transparente
Margin = 0
SpecialEffect = 1 && Simples
BorderStyle = 0    && Nenhuma
```

Utilizando formatação condicional em grades

A formatação especial de uma grade poderá facilitar para o usuário a varredura dos registros da grade e a localização de determinadas informações. Para fornecer uma formatação condicional, utilize as propriedades dinâmicas de cor e fonte de uma coluna.

Por exemplo, você pode adicionar uma grade a um formulário e definir a propriedade ColumnCount como 2. Defina a propriedade ControlSource da primeira coluna como `orders.to_name` e a propriedade ControlSource da segunda coluna como `orders.order_net`. Para exibir os totais de pedidos inferiores a 500,00 com a cor de primeiro plano preta e os totais de pedidos superiores ou iguais a 500,00 com a cor de primeiro plano vermelha, inclua a linha abaixo no código de evento Init da grade:

```
THIS.Column2.DynamicForeColor = ;
"IIF(orders.order_net >= 500, RGB(255,0,0), RGB(0,0,0))"
```

Propriedades comuns de grade

As propriedades de grade a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
ChildOrder	A chave estrangeira da tabela filho que está associada à chave primária da tabela pai.
ColumnCount	Número de colunas. Se ColumnCount for definida como –1, o número de colunas da grade será equivalente ao número de campos na RecordSource da grade.
LinkMaster	A tabela pai dos registros filho exibidos na grade.
RecordSource	Os dados a serem exibidos na grade.
RecordSourceType	De onde vêm os dados exibidos na grade: de uma tabela, um alias, uma consulta, ou uma tabela selecionada pelo usuário em resposta a uma solicitação.

Propriedades comuns de coluna

As propriedades de coluna abaixo são normalmente definidas na hora da criação.

Propriedade	Descrição
ControlSource	Os dados a serem exibidos na coluna. Este é, freqüentemente, um campo de uma tabela.
Sparse	Se Sparse estiver definido como verdadeira (.T.), os controles de uma grade só serão exibidos como controles quando a célula da coluna for selecionada. As outras células da coluna exibem o valor dos dados subjacentes em uma caixa de texto. A definição de Sparse como verdadeira (.T.) permite redesenhar mais rapidamente se o usuário estiver percorrendo uma grade com várias linhas exibidas.
CurrentControl	Qual controle está ativo na grade. O padrão é Text1, mas se você adicionar um controle à coluna, poderá especificá-lo como CurrentControl.

Observação A propriedade ReadOnly de um controle da coluna é substituída pela propriedade ReadOnly da coluna. Se você definir a propriedade ReadOnly do controle de uma coluna no código associado ao evento AfterRowColChange, a nova definição permanecerá válida enquanto você estiver nessa célula.

Facilitando a utilização dos controles

Você quer tornar o mais fácil possível para o usuário o entendimento e a utilização dos controles. As teclas de acesso, a ordem de tabulação, o texto das Descrições de ferramentas e a desativação seletiva contribuem, todos, para uma criação mais prática.

Definindo teclas de acesso

Uma tecla de acesso permite que um usuário selecione um controle de qualquer parte do formulário pressionando ALT e a tecla.

Para especificar uma tecla de acesso para um controle

- Coloque uma barra invertida e um sinal de menor que (\<) antes da letra desejada na propriedade Caption.

Por exemplo, a definição de propriedade a seguir para a legenda de um botão de comando torna O a tecla de acesso.

```
\<Open
```
O usuário pode selecionar o botão de comando em qualquer parte do formulário pressionando ALT+O.

Para especificar uma tecla de acesso para uma caixa de texto ou caixa de edição

1. Crie um rótulo com uma barra invertida e um sinal de menor que (\<) na frente da letra desejada, como em `C\<ustomer`.
2. Certifique-se de que o rótulo seja o controle na ordem de tabulação imediatamente anterior à caixa de texto ou de edição que deverá ser destacada.

Definindo a ordem de tabulação de controles

A ordem de tabulação padrão dos controles no formulário é a ordem em que eles foram adicionados ao formulário.

Dica Defina a ordem de tabulação dos controles para que o usuário possa percorrer facilmente os controles em uma ordem lógica.

Para alterar a ordem de tabulação dos controles

1. Na **Barra de ferramentas criador de formulários** selecione o botão **Definir ordem de tabulação**.
2. Clique duas vezes sobre a caixa ao lado do controle que você deseja que seja destacado inicialmente quando o formulário for aberto.
3. Clique sobre a caixa ao lado dos outros controles na ordem em que você deseja que sejam tabulados.
4. Clique em qualquer lugar fora das caixas de ordem de tabulação para finalizar.

Também é possível definir a ordem de tabulação dos objetos no formulário por listagem, dependendo da definição na guia **Criação de formulários** da caixa de diálogo **Opções**.

Você pode definir a ordem de seleção para os botões de opção e comando de um grupo de controle. Para mover um grupo de controle com o teclado, o usuário deverá tabular até o primeiro botão no grupo de controle e utilizar as teclas de direção para selecionar os outros botões no grupo.

Para alterar a ordem de seleção de botões em um grupo de controle

1. Na janela **Propriedades**, selecione o grupo na lista **Objeto**. Uma borda espessa indica que o grupo está em modo de edição.
2. Selecione a janela **Criador de formulários**.
3. No menu **Exibir**, escolha **Ordem de tabulação**.
4. Defina a ordem de seleção da mesma forma que você definiria a ordem de tabulação para os controles.

Definindo a propriedade ToolTipText

Cada controle possui uma propriedade ToolTipText que permite que você especifique o texto exibido quando o usuário posicionar o ponteiro do mouse sobre o controle. As descrições são particularmente úteis para botões com ícones em vez de texto.

Para especificar o texto de uma descrição de ferramenta

- Na janela **Propriedades**, selecione a propriedade **ToolTipText** e digite o texto desejado.

A propriedade ShowTips do formulário determina se o texto da descrição de ferramenta será exibido ou não.

Alterando a exibição do ponteiro do mouse

Você pode alterar a exibição do ponteiro do mouse para fornecer dicas visuais aos seus usuários sobre os diferentes estados em que seu aplicativo pode estar.

Por exemplo, na classe tsBaseForm do aplicativo de exemplo Tasmanian Traders, um método WaitMode altera o ponteiro do mouse para um cursor padrão do estado de espera. Antes de executar qualquer código que possa levar algum tempo para ser processado, o aplicativo Tasmanian Traders passa um valor de verdadeiro (.T.) para o método WaitMode, que altera o ponteiro para fazer o usuário saber que o processamento está em andamento. Quando o processamento for completado, uma chamada para WaitMode com falso (.F.) restaurará o ponteiro padrão do mouse.

```
* Método WaitMode da classe tsBaseForm
LPARAMETERS tlWaitMode

lnMousePointer = IIF(tlWaitMode, MOUSE_HOURGLASS, MOUSE_DEFAULT)
thisform.MousePointer = lnMousePointer
thisform.SetAll('MousePointer', lnMousePointer)
```

Se quiser alterar o ponteiro do mouse para uma forma diferente da dos ponteiros padrão, defina a propriedade MousePointer como 99 (Personalizar) e defina a propriedade MouseIcon para o seu próprio arquivo de Cursor (.CUR) ou ícone (.ICO).

Ativando e desativando controles

Defina a propriedade Enabled dos controles como falsa (.F.) se a funcionalidade do controle não estiver disponível em uma situação específica.

Ativando e desativando botões em um grupo

Para ativar ou desativar botões de opção ou botões de comando individuais em um grupo, defina a propriedade Enabled de cada botão como verdadeira (.T.) ou falsa (.F.). Também é possível desativar ou ativar todos os botões em um grupo definindo a propriedade Enabled do grupo, como na linha de código abaixo:

```
frmForm1.cmgCommandGroup1.Enabled = .T.
```

Quando você define a propriedade Enabled de um grupo de botões de opção ou de um grupo de botões de comando como falsa (.F.), todos os botões do grupo são desativados, mas não serão exibidos com as cores de primeiro e de segundo planos desativadas. A definição da propriedade Enabled do grupo não altera a propriedade Enabled de cada botão no grupo. Isso permite desativar um grupo de botões que tenha alguns dos botões já desativados. Quando você ativar o grupo, os botões que estavam originalmente desativados permanecem desativados.

Se você quiser desativar todos os botões de um grupo para que apareçam desativados, e não quiser conservar informações sobre que botões estavam originalmente desativados ou ativados, utilize o método SetAll do grupo assim:

```
frmForm1.opgOptionGroup1.SetAll("Enabled", .F.)
```

Permitindo que o usuário arraste e solte

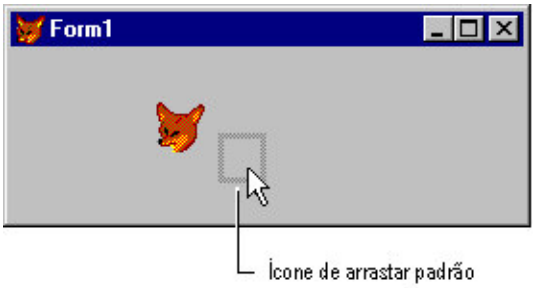
Quando cria aplicativos do Visual FoxPro, você pode arrastar objetos do **Gerenciador de projetos**, do **Criador de bancos de dados** e do **Criador de ambientes de dados** para localizações desejadas em formulários e relatórios. Os recursos de arrastar-e-soltar do Visual FoxPro lhe permitem estender este recurso ao usuário em tempo de execução.

Essa capacidade de arrastar-e-soltar estende-se a operações com vários formulários. O usuário pode arrastar o controle para qualquer lugar na tela, inclusive para outros formulários. Você pode responder a qualquer operação de arrastar desde que o destino esteja dentro do seu aplicativo.

Para ver exemplos de arrastar-e-soltar, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e, em seguida, escolha General na seção Controls.

Quando o usuário arrasta um controle, o Visual FoxPro apresenta um contorno cinza com o mesmo tamanho do objeto e que se move com o ponteiro do mouse. Você pode substituir este comportamento padrão especificando um arquivo de cursor (.CUR) para a propriedade DragIcon de um controle.

Arrastando um controle Image em tempo de execução



Observação Arrastar um controle em tempo de execução não altera automaticamente a sua localização. Você pode fazer isso, mas deve programar você mesmo o reposicionamento, conforme descrito na seção Gerando o Movimento do Controle, posteriormente neste capítulo. A operação de arrastar é freqüentemente utilizada apenas para indicar que alguma ação deve ser executada; o controle volta à sua posição original depois que o usuário solta o botão do mouse.

Utilizando as propriedades, os eventos e o método arrastar-e-soltar a seguir, você poderá especificar o significado de uma operação de arraste e como ela pode ser iniciada (se for o caso) para qualquer controle determinado.

Para	Utilize este recurso
Ativar o arraste automático ou manual de um controle.	Propriedade DragMode
Especificar que ícone será exibido quando o controle for arrastado.	Propriedade DragIcon
Reconhecer quando um controle for soltado no objeto.	Evento DragDrop
Reconhecer quando um controle for arrastado sobre o objeto.	Evento DragOver
Iniciar ou parar o arraste manual.	Método Drag

Todos os controles visuais podem ser arrastados em tempo de execução e todos os controles compartilham as propriedades listadas na tabela anterior. Os formulários reconhecem os eventos DragDrop e DragOver, mas não possuem propriedades DragMode e DragIcon.

Ativando o modo arrastar automático

Para permitir que o usuário arraste um controle sempre que clicar sobre ele, defina a propriedade DragMode do controle como 1. Isso ativa o arraste automático do controle. Quando você define o modo arrastar como Automático, esse modo permanece sempre ativado.

Observação Enquanto uma operação de arraste automático está ocorrendo, o controle que está sendo arrastado não reconhece outros eventos do mouse.

Respondendo quando o usuário solta o objeto

Quando o usuário libera o botão do mouse depois de arrastar um controle, o Visual FoxPro gera um evento DragDrop. Você pode responder a esse evento de várias maneiras. Pode reposicionar o controle na nova localização (indicada pela última posição do contorno cinza). Lembre-se de que o controle não se move automaticamente para a nova localização.

Dois termos são importantes quando as operações de arrastar-e-soltar são abordadas — *origem* e *destino*.

Termo	Significado
Origem	O controle que está sendo arrastado.

Destino	O objeto no qual o usuário solta o controle. Esse objeto, que pode ser um formulário ou um controle, reconhece o evento DragDrop.

Um controle se tornará o destino se a posição do mouse estiver dentro de suas bordas quando o botão for liberado. Um formulário será o destino se o ponteiro estiver em uma parte em branco do formulário.

O evento DragDrop recebe três parâmetros: *oSource*, *nXCoord* e *nYCoord*. O parâmetro *oSource* é uma referência ao controle que foi soltado no destino. Os parâmetros *nXCoord* e *nYCoord* contêm as coordenadas horizontal e vertical, respectivamente, do ponteiro do mouse no destino.

Visto que *oSource* é um objeto, ele é utilizado exatamente como um controle—você pode referir-se às suas propriedades ou chamar um de seus métodos. Por exemplo, as instruções a seguir do código associado ao evento DragDrop verificam se o usuário soltou um controle sobre ele mesmo:

```
LPARAMETERS oSource, nXCoord, nYCoord
IF oSource.Name != THIS.Name
   * Execute uma ação.
ELSE
   * O controle foi solto sobre ele mesmo.
   * Execute uma outra ação.
ENDIF
```

Todos os tipos de controle possíveis para *oSource* possuem uma propriedade Visible. Portanto, você pode tornar um controle invisível quando ele for solto em uma parte específica de um formulário ou em um outro controle. A linha a seguir, no código associado ao evento DragDrop de um controle de imagem, faz com que um controle arrastado desapareça quando for solto sobre a imagem:

```
LPARAMETERS oSource, nXCoord, nYCoord
oSource.Visible = .F.
```

Indicando zonas válidas para soltar

Quando você ativa o recurso arrastar-e-soltar, pode ajudar os usuários incluindo dicas visuais sobre onde o usuário pode e não pode soltar um controle. A melhor maneira de fazer isso é alterar a propriedade DragIcon da origem no código associado ao evento DragOver.

O código a seguir, no evento DragOver de um controle, indica ao usuário que o controle não é um destino válido onde soltar. Neste exemplo, `cOldIcon` é uma propriedade do formulário definida pelo usuário.

```
LPARAMETERS oSource, nXCoord, nYCoord, nState
DO CASE
CASE nState = 0 && Entrar
THISFORM.cOldIcon = oSource.DragIcon
oSource.DragIcon = "NODROP01.CUR"
CASE nState = 1 && Sair
oSource.DragIcon = THISFORM.cOldIcon
ENDCASE
```

Controlando quando a operação de arrastar começa ou é interrompida

O Visual FoxPro possui a definição Manual para a propriedade DragMode, que dá a você mais controle do que a definição Automático. A definição Manual lhe permite especificar quando um controle pode e não pode ser arrastado. (Quando DragMode está definida como Automático, o controle pode ser sempre arrastado, desde que a definição não seja alterada.)

Por exemplo, você pode querer ativar a operação de arrastar em resposta a eventos MouseDown e MouseUp ou em resposta a um comando de menu ou do teclado. A definição Manual também lhe permite reconhecer um evento MouseDown antes que a operação de arrastar tenha início, para que você possa registrar a posição do mouse.

Para ativar a operação de arrastar por meio do código, deixe DragMode com sua definição padrão (0

–Manual). Em seguida, utilize o método Drag sempre que quiser começar a arrastar um objeto ou parar de arrastá-lo:

recipiente.controle.Drag(*nAction*)
Se *nAction* for 1, o método Drag começará a arrastar o controle. Se *nAction* for 2, o controle será soltado, gerando um evento DragDrop. O valor 0 para *nAction* cancela a operação de arrastar. O efeito é semelhante ao de fornecer um valor 2, com a exceção de que não ocorre nenhum evento DragDrop.

Observação Para ativar a operação de arrastar-e-soltar a partir de uma caixa de listagem, o melhor lugar para chamar o método Drag é no código associado ao evento MouseMove da caixa de listagem de origem, após determinar se o botão do mouse está pressionado. Para obter um exemplo, consulte o LMOVER.SCX no diretório VFP\SAMPLES\CONTROLS\LISTS.

Gerando movimento do controle em uma operação de arrastar-e-soltar

Você pode querer que o controle de origem mude de posição depois que o usuário liberar o botão do mouse. Para fazer com que o controle se mova para a nova localização do mouse, utilize o método Move. Por exemplo, o código a seguir no evento DragDrop de um formulário move o controle que é arrastado para o local da soltura:

```
LPARAMETERS oSource, nXCoord, nYCoord
oSource.Move(nXCoord, nYCoord)
```

Esse código pode não produzir exatamente os efeitos desejados, pois o canto superior esquerdo do controle fica posicionado na localização do mouse. O código a seguir posiciona o centro do controle na localização do mouse:

```
LPARAMETERS oSource, nXCoord, nYCoord
oSource.Move ((nXCoord – oSource.Width / 2), ;
(nYCoord – oSource.Height / 2))
```

O código funciona melhor quando a propriedade DragIcon está definida como um valor diferente do padrão (o retângulo cinza). Quando o retângulo cinza está sendo utilizado, o usuário normalmente deseja que o controle se mova precisamente para a última posição do retângulo cinza. Para isso, registre a posição inicial do mouse no controle de origem. Utilize então essa posição como deslocamento quando o controle for movido. Para obter um exemplo, consulte DDROP.SCX no diretório SAMPLES\SOLUTION\FORMS.

Para registrar a posição inicial do mouse

1. Especifique o modo arrastar manual do controle.
2. Declare duas variáveis em nível de formulário, n`DragX` e n`DragY`.
3. Ative o arraste quando ocorrer um evento MouseDown. Além disso, armazene o valor de *nXCoord* e *nYCoord* nas variáveis em nível de formulário deste evento..
4. Desative o arraste quando ocorrer o evento MouseUp.

Estendendo formulários

As molduras de página permitem que você estenda a superfície dos formulários e os controles ActiveX permitem que você amplie a funcionalidade deles.

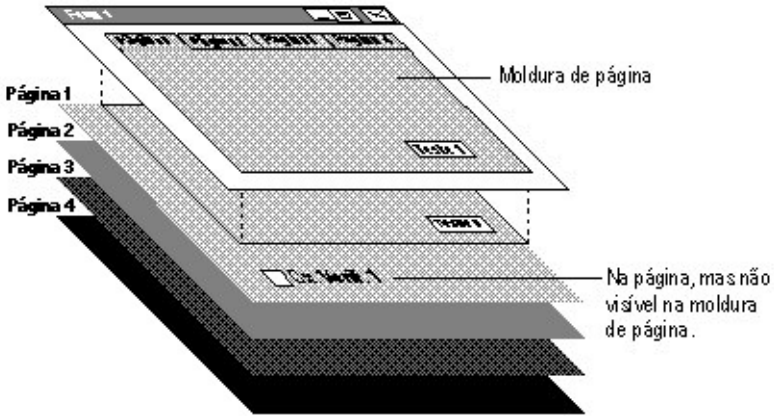
Utilizando molduras de página

Uma moldura de página é um objeto recipiente que contém páginas. As páginas, por sua vez, contêm controles. As propriedades podem ser definidas no nível do controle, da moldura de página ou da página.

Para ver exemplos de como utilizar molduras de página, execute SOLUTION.APP no diretório SAMPLES\SOLUTION e, em seguida, escolha Pageframe na seção Controls.

Imagine que a moldura da página é um recipiente tridimensional que apresenta camadas de páginas. Somente os controles na página superior (no alto da moldura de página) podem ficar visíveis e ativos.

Várias páginas em uma moldura de página de um formulário



A moldura de página define a localização das páginas e quanto da página é visível. O canto superior esquerdo de uma página está ancorado no canto superior esquerdo da moldura de página. Podem ser colocados controles em páginas que ultrapassem as dimensões da moldura de página. Esses controles ficam ativos mas não visíveis, a menos que você altere, por programação, as propriedades Height e Width da moldura de página para torná-los visíveis.

Utilizando páginas em um aplicativo

Com as molduras de página e as páginas, você pode criar formulários ou caixas de diálogo com guias utilizando o mesmo tipo de recursos de interface existentes no **Gerenciador de projetos**.

Além disso, as molduras de página lhe permitem definir uma região do formulário onde poderá inserir e remover controles facilmente. Por exemplo, nos Assistentes, a maior parte do formulário permanece constante, mas uma área dele é alterada a cada etapa. Em vez de criar cinco formulários para as etapas dos assistentes, você pode criar um formulário com uma moldura de página e cinco páginas.

SOLUTION.APP, (no diretório SAMPLES\SOLUTION) contém dois exemplos de molduras de página que demonstram como utilizar molduras com e sem guias.

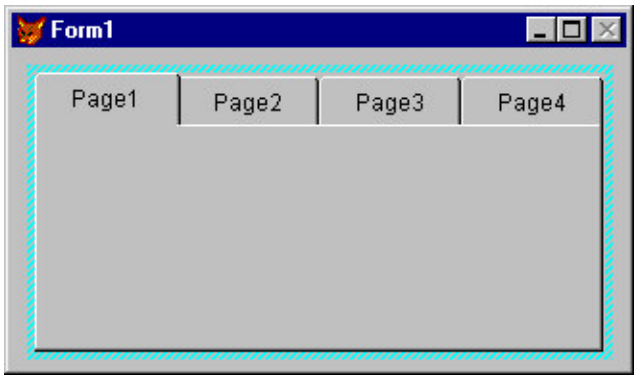
Adicionando molduras de página a um formulário

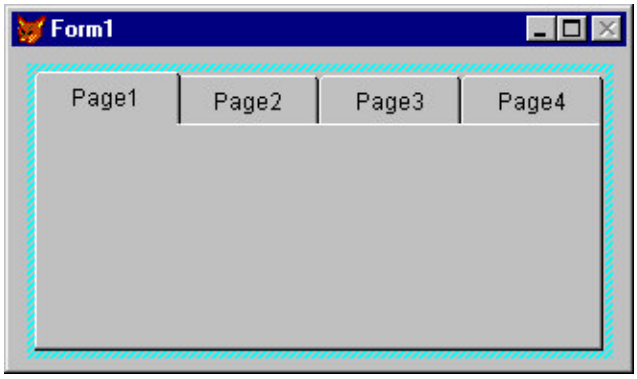
Você pode incluir uma ou mais molduras de página em qualquer formulário.

Para adicionar uma moldura de página a um formulário

1. Na **Barra de ferramentas controles de formulário**, selecione o botão **Moldura de página** e arraste até a dimensão desejada na janela **Formulário**.
2. Defina a propriedade PageCount para indicar o número de páginas a serem incluídas na moldura.

Moldura de página com quatro páginas





3 No menu de atalho da moldura, selecione **Editar** para ativar a moldura como um recipiente. A borda da moldura de página fica mais larga para indicar que ela está ativa.

4 Adicione controles da mesma maneira que os adicionaria a um formulário.

Observação Da mesma forma que com outros controles de recipiente, você deve selecionar a moldura de página e selecionar **Editar** no menu acionado pelo botão direito do mouse, ou selecionar o recipiente na lista suspensa Objeto na janela **Propriedades**,, para que o recipiente seja selecionado (fique com uma borda mais larga) antes de você adicionar controles à página que está criando. Se você não ativar a página como um recipiente antes de adicionar controles, estes serão adicionados ao formulário e não à página, embora possa parecer que eles estão na página.

▸ **Para selecionar outra página na moldura de página**

1 Ative a moldura de página como um recipiente clicando com o botão direito do mouse sobre ela e escolhendo **Editar**.

2 Selecione a guia da página que deseja utilizar.

– Ou –

- Selecione a página na caixa **Objeto** na janela **Propriedades**.

– Ou –

- Selecione a página na caixa **Página** na parte inferior do **Criador de formulários**.

Adicionando controles a uma página

Quando você adiciona controles a uma página, eles só ficarão visíveis e ativos quando a página relativa a eles estiver ativa.

▸ **Para adicionar controles a uma página**

1 Na caixa **Objeto** da janela **Propriedades**, selecione a página. Será exibida uma borda ao redor da moldura de página, indicando que você pode manipular os objetos contidos.

2 Na **Barra de ferramentas controles de formulário**,selecione o botão de controle que deseja e arraste para dimensioná-lo na página.

Gerenciando legendas extensas em guias de página

Se as legendas das guias forem maiores do que o tamanho que pode ser exibido na guia devido à largura da moldura da página e ao número de páginas, você tem duas opções:

- Definir a propriedade TabStretch como **1 - Linha simples**, para exibir somente os caracteres das legendas que se ajustem nas guias. Linha simples é o padrão.
- Definir a propriedade TabStretch como **0 - Linhas múltiplas**, para empilhar as guias de modo que a legenda inteira de todas as guias fique visível.

Alterando páginas por meio de programação

Seja a moldura de página exibida com guias ou não, você pode tornar uma página ativa pelo programa, utilizando a propriedade ActivePage. Por exemplo, o código a seguir, no procedimento do evento Click de um botão de comando em um formulário altera a página ativa de uma página de moldura do formulário para a terceira delas:

```
THISFORM.pgfOptions.ActivePage = 3
```

Propriedades comuns de moldura de página

As propriedades de moldura de página a seguir são normalmente definidas na hora da criação.

Propriedade	Descrição
Tabs	Se as guias estão visíveis para as páginas.
TabStyle	Se as guias têm ou não do mesmo tamanho, e se juntas têm a mesma largura da moldura da página.
PageCount	O número de páginas na moldura de página.

Controle de recipiente OLE

Para adicionar um objeto OLE a um formulário, clique sobre essa ferramenta e arraste até dimensioná-la na janela **Formulário**. Essa ferramenta pode representar um objeto servidor, como o Microsoft Excel ou Microsoft Word, ou um controle de ActiveX, caso o diretório SYSTEM do Windows contenha controles ActiveX (arquivos com a extensão .OCX). Para obter informações gerais sobre controles de ActiveX, consulte o capítulo 16, Adicionando OLE.

Controle de ligação de OLE

Para criar um objeto OLE acoplado em um formulário, clique sobre essa ferramenta e arraste para dimensioná-la na janela **Formulário**. Depois de criar o objeto, conecte-o a um campo Geral em uma tabela. Utilize então o objeto para exibir o conteúdo do campo. Por exemplo, se você armazenar documentos do Word em um campo Geral, poderá exibir o conteúdo desses documentos utilizando um objeto OLE acoplado em um formulário.

Para criar um objeto OLE acoplado

1. Crie ou abra um formulário.
2. Na **Barra de ferramentas controles de formulário**,selecione o botão **Controle de ligação de OLE** e arraste-o para sua dimensão no formulário.
3. Ligue o objeto OLE a um campo Geral definindo a propriedade ControlSource do objeto.

Para obter um exemplo de como utilizar o controle de ligação de OLE, consulte o capítulo 16, Adicionando OLE.