Есть способ довавить **ProgressBar** (или другой компонент Delphi) в управление **ListView** .

Большинство менеджеров загрузки использует вид списка, чтобы визуально показать количество остающейся загрузки. Компонент Delphi **TListView** отображает список элементов различными способами.

TProgressBar отображает простую полосу прогресса и обеспечивает индикатор продвижения в пределах приложения.

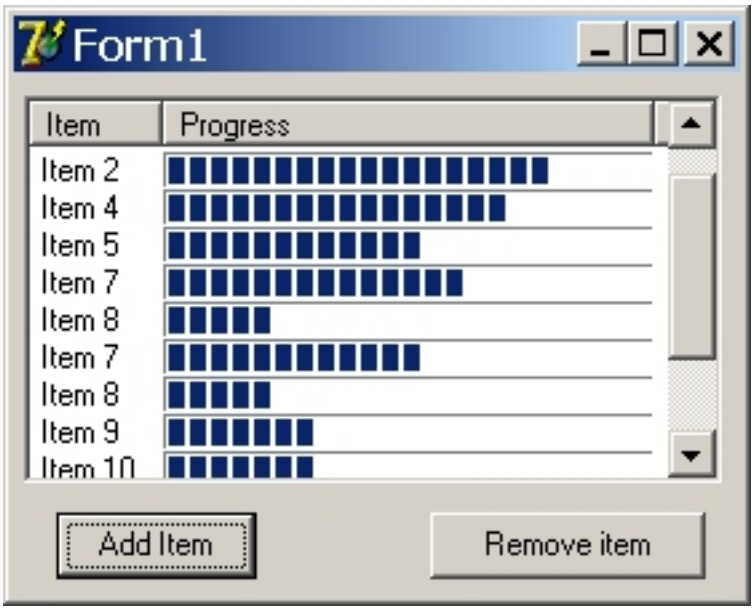
Но можно разместить **ProgressBar** в столбец окна **ListView**...

ProgressBar в ListView

Когда свойство *ViewStyle* управления **ListView** установлено в *vsReport*, каждый пункт появляется на отдельной строке с информацией, размещенной в столбцах. Крайний левый столбец содержит маленькую иконку и метку, а последующие столбцы стдержат подпункты, как определено приложением.

Вы, вероятно, каждый день это видите... когда используете Проводник Windows с установленным видом *"Таблица"*.

Предположим, что Вы имеете на форме **ListView** со свойством **ViewStyle**, установленным в
vsReport
. Два столбца определены, используя свойство
Columns
: первый столбец содержит имя пункта, а второй столбец должен содержать индикатор прогресса.



Когда кнопка **AddItem** нажата, новый пункт должен быть добавлен в список с **Progress Bar** :

```
procedure TForm1.AddItemButtonClick( Sender: TObject); const pbColumnIndex = 1; pbMax = 100; var li : TListItem; lv : TListView; pb : TProgressBar; pbRect : TRect; begin lv := ListViewEx1; //создать новый ListItem (для lv) li := lv.Items.Add; li.Caption := 'Item ' + IntToStr(lv.Items.Count); //создать ProgressBar, поместить его во второй столбец pb := TProgressBar.Create(nil); pb.Parent := lv; li.Data := pb; pbRect := li.DisplayRect(drBounds); pbRect.Left := pbRect.Left + lv.Columns[-1 + pbColumnIndex].Width; pbRect.Right := pbRect.Left + lv.Columns[pbColumnIndex].Width; pb.BoundsRect := pbRect; end; //AddItemButtonClick
```

Когда будет нажата кнопка **AddItemButton**, новый пункт списка (**TListItem**) будет добавлен в управление
ListView
(по имени
ListViewEx1
). Индикатор прогресса создан, ссылка на него добавлен в свойство
Data
списка, и наконец, индикатор помещен в столбец, определенный как
pbColumnIndex
. Используются некоторые вычисления, чтобы заставить индикатор прогресса появиться в правильном месте.

Когда Вы хотите удалить элемент из списка, Вы должны удостовериться, что "прикрепленная" память освобождена, и все индикаторы прогресса ниже выбранного перемещены на одну позицию вверх (т. к. это произойдет со всеми остальными пунктами

ListView
):

```
procedure TForm1.RemoveItemButtonClick( Sender: TObject); var lv : TListView; li : TListItem; i, idx : integer; pb : TProgressBar; begin lv := ListViewEx1; li := lv.Selected; if li nil then begin idx := li.Index; TProgressBar(li.Data).Free; lv.Items.Delete(idx); //переместить индикаторы выше for i := idx to -1 + lv.Items.Count do begin li := lv.Items.Item[i]; pb := TProgressBar(li.Data); pb.Top := pb.Top - (pb.BoundsRect.Bottom - pb.BoundsRect.Top); end; end; end; //RemoveItemButtonClick
```

Только для испытания, мы добавим некоторый фиктивный код внутри события *OnTime* управления
TTimer
, чтобы иметь некоторое продвижение индикаторов (для этого поместите компонент
TTimer
на форму и используйте следующий код для события
OnTime
).

```
procedure TForm1.Timer1Timer( Sender: TObject); var idx : integer; pb: TProgressbar; lv : TListView; begin lv := ListViewEx1; if lv.Items.Count = 0 then Exit; //случайно выберем пункт и //увеличим значение его индикатора idx := Random(lv.Items.Count); pb := TProgressBar(lv.Items[idx].Data); if pb.Position Запустите приложение, нажмите кнопку
```

AddItem
несколько раз и наблюдайте за продвижением прогресса.

Теперь попытайтесь изменить размеры любого из столбцов... :(УУУпссс!

Когда Вы изменяете размеры столбцов, Вы должны повторно установить индикаторы, но в **TListView** нет события, чтобы обработать изменение размера столбцов!

TListViewEx - ListView с событием изменения размера столбцов

Есть компонент **TListViewEx** - потомок **TListView** с *published* событиями *BeginColumnResize*, *ColumnResize* и *EndColumnResize*.

The code was provided by Peter Below (a TeamB member).

Теперь, со всей мощью **TListViewEx**, Вы можете легко обрабатывать ситуацию изменения пользователем ширины столбца:

```
procedure TForm1.ListViewEx1EndColumnResize(  sender: TCustomListView;
columnIndex,  columnWidth: Integer);  var  lv : TListViewEx;  idx : integer;  pb :
TProgressBar;  begin  lv := ListViewEx1;  //первый столбец  if columnIndex = 0 then
begin  for idx := 0 to -1 + lv.Items.Count do  begin  pb :=
TProgressBar(lv.Items[idx].Data);  pb.Left := columnWidth;  end;  end;  //столбец
индикатора прогресса  if columnIndex = 1 then  begin  for idx := 0 to -1 + lv.Items.Count
do  begin  pb := TProgressBar(lv.Items[idx].Data);  pb.Width := columnWidth;  end;
 end;  end;
```

Вы можете загрузить исходник TListViewEx . Это один файл *.pas. Вы только будете должны добавить компонент в существующий пакет.