Технология Drag and Drop. Часть 1 Теоретические основы

Цели:

- Приобрести понятие о механизме *Drag&Drop*
- Получить понятие об этапах работы *Drag&Drop*
- Получить понятие о событиях, методах и свойствах, с помощью которых осуществляется контроль и управление работой механизма *Drag&Drop*
- Приобрести практические навыки использования механизма *Drag&Drop* для переноса визуальных компонент.

Технология *Drag&Drop* позволяет производить буксировку экранных объектов, а так же эффективно производить перенос информации между компонентами.

Буксировка осуществляется в четыре этапа:

- **Этап 1. Начало буксировки.**
  Возникает при перемещении мыши с нажатой левой кнопкой. Действие при начале буксировки задается свойством **DragMode** компонента. В режиме ***dmAutomatic*** компонент берет на себя обработку этого этапа. Если установить значение этого свойства в ***dmManual***, то программисту придется вручную обнаруживать начало буксировки (например, по событию **MouseDown**) и запускать ее механизм (метод **BeginDrag**). В момент начала буксировки компонент генерирует событие **OnStartDrag**.
  Методу **BeginDrag** в качестве параметра передается булевское значение которое показывает, будет ли курсор изменен на курсор буксировки мгновенно (***true***), либо только после перемещения кнопки мыши (***false***).

- **Этап 2. Буксировка объекта над компонентами.**

  Во время буксировки при перемещении курсора мыши над компонентами они генерируют событие **OnDragOver**. С помощью данного события можно задать реакцию компонента на буксировку чего-либо над ним. Параметры обработчика этого события следующие: ***Source*** - объект — источник буксировки (или буксируемый объект), ***X,Y*** - координаты курсора, ***Accept*** - булевская переменная, определяющая, принимает или нет данный компонент буксируемую информацию.

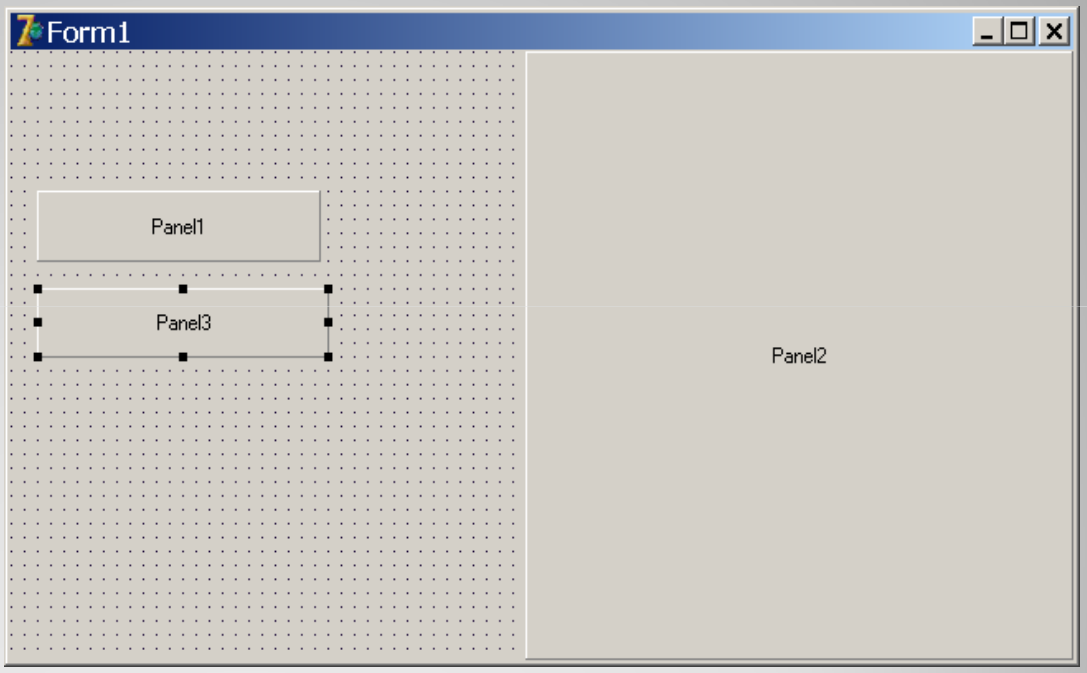
- **Этап 3. Оставление информации на целевом компоненте**.

  Во время буксировки при отпускании кнопки мыши над целевым компонентом он генерируют событие **OnDragDrop**. В обработчике именно этого события должно быть произведено то действие, которому соответствует буксировка. Следует отметить, что если событие **OnDragOver** не принимает данный компонент (***Accept***=false), то событие **OnDragDrop** сгенерировано не будет.

- **Этап 4. Завершение буксировки**.
  Компонент - источник буксировки в момент отпускания генерирует событие **OnEndDrag**, параметр ***Target*** которого соответствует другому, целевому компоненту. В обработчике этого события можно задать действия, выполняющиеся по окончании буксировки.

Задание
Перенос панелей

Создадим первое приложение, осуществляющее перенос объекта по форме. Для этого создадим новое приложение. Разместим на форме три компонента TPanel. Полученный макет формы должен выглядеть примерно следующим образом.



Зададим свойство ***DragMode*** компонентов ***Panel1*** и ***Panel3*** в ***dmAutomatic***.

Зададим события таким образом, чтобы компонент ***Panel1*** можно было перетаскивать по всей форме и по второй панели (***Panel2***), а компонент ***Panel3*** только по форме и нельзя было перенести его на панель (***Panel2***).

Для реализации этого зададим следующие события:

- ***OnDragOver*** компонента ***Panel2.*** Задает в качестве принимаемого объекта только ***Panel1***.

Для остальных объектов ***Accept*** устанавливается в ***False***:

```
procedure TForm1.Panel2DragOver(Sender, Source: TObject; X, Y:
    Integer;
  State: TDragState; var Accept: Boolean);
begin
  if Source=Panel1 then Accept:=true else Accept:=false;
end;
```

- ***OnDragOver*** компонента ***From1***. Задает в качестве принимаемого объекта любой объект класса ***TPanel***. Для остальных объектов ***Accept*** устанавливается в ***False***:

```
procedure TForm1.FormDragOver(Sender, Source: TObject;
   X, Y: Integer;
  State: TDragState; var Accept: Boolean);
begin
  if Source is TPanel then Accept:=true else Accept:=false;
end;
```

Проверка **is** в условном операторе позволяет проверить принадлежность компонента из параметра **Source** к указанному классу – в данном случае **TPanel**.

- ***OnDragDrop*** компонента ***Form1.*** Устанавливает новое местоположение брошенного над формой компонента. Свойство ***Parent*** задается для установки нового родителя для визуального компонента (например, при переносе компонента с ***Panel2*** на ***Form1***):

```
procedure TForm1.FormDragDrop(Sender, Source: TObject; X, Y:
    Integer);
begin
  TPanel(Source).Parent:=Form1;
  TPanel(Source).Left:=X;
  TPanel(Source).Top:=Y;
end;
```

- ***OnDragDrop*** компонента ***Panel2.*** Устанавливает новое местоположение брошенного над панелью (***Panel2***) компонента. Свойство ***Parent*** задается для установки нового родителя для визуального компонента (например, при переносе компонента с ***Form1*** на ***Panel2***).

```
procedure TForm1.Panel2DragDrop(Sender, Source: TObject;
   X, Y: Integer);
begin
  TPanel(Source).Parent:=Panel2;
  TPanel(Source).Left:=X;
  TPanel(Source).Top:=Y;
end;
```

Модифицируйте указанный код таким образом, чтобы компоненты ***Panel1*** и ***Panel3*** могли переносится друг на друга.
Добавьте на форму еще одну панель ***Panel4***. Реализуйте ее перемещение аналогичным ***Panel1*** образом, **не устанавливая ее свойство *DragMode*** в ***dmAutomatic*** (воспользуйтесь методом ***BeginDrag*** в событии ***OnMouseDown***).