Работа в мышью в Delphi

Последнее изменение: 03.10.2016

В Delphi есть два основных подхода получения информации о курсоре мыши: через набор функций и через интерфейс Controls.TMouse (возможно более удобный). Но всё же я собираюсь рассказать про оба. Тем более, что TMouse работает через эти самые функции.

Положение курсора

Для начала стоит разобраться с системой координат. Она у нас глобальная, с началом в левом верхнем углу экрана, ось X направлена вправо, ось Y вниз. Максимальное значение координат X и Y будет, соответственно, ширина и высота вашего экрана в пикселях. Их проще всего получить используя Forms.Screen:

```
var
  w, h: integer;
begin
  w := Screen.Width; // ширина
  h := Screen.Height; // высота
end;
```

Заметьте, что координаты правого нижнего угла зависят от разрешения.

Получение координат

Перейдём к получению самих координат. Для этого есть функция Windows.GetCursorPos

```
function GetCursorPos(var lpPoint: TPoint): BOOL; stdcall;
```

Пожалуй, использовать её не очень удобно - переменная, куда записывается нужная информация, передаётся в функцию в виде параметра. Однако, я согласен с тем, что хранить координаты вместе удобнее, используя тип Windows.TPoint. Для тех, кто не знает как работать в TPoint (мало ли?), получение координат осуществляется примерно так:

```
var
  p: TPoint;
  x, y: integer;
begin
  GetCursorPos(p);
  x := p.X;
  y := p.Y;
end;
```

Нет, всё-таки неудобно. Давайте лучше воспользуется вторым методом: для этого в модуле "Controls" объявлен объект Controls.Mouse:

```
var
  Mouse: TMouse;
```

С ней работать уже приятнее, ведь не обязательно заводить лишние переменные, т.к. Mouse.CursorPos уже и есть та переменная типа TPoint. Итого:

```
var
  x, y: integer;
begin
  x := Mouse.CursorPos.X;
  y := Mouse.CursorPos.Y;
end;
```

Перемещение курсора

На этот раз, пожалуй, у первого варианта преимущество. Достаточно взять и просто вызвать Windows.SetCursorPos:

```
function SetCursorPos(X, Y: Integer): BOOL; stdcall;
```

Со вторым вариантом, впрочем, тоже особых проблем не возникает. Достаточно преобразовать пару чисел (x, y) в TPoint и сохранить его:

```
Mouse.CursorPos := Point(x, y);
```

Стоит заметить, что не получится просто изменить только одну из координат:

```
Mouse.CursorPos.X := 100; //[Error]: Left side cannot be assigned to
```

Вот, что очень удобно - перемещение курсора происходит моментально, без прохождения промежуточных точек.

Нажатие кнопок мыши

Перейдём к более активным действиям мышью. Рассмотрим функцию Windows.mouse_event:

```
procedure mouse_event(dwFlags, dx, dy, dwData, dwExtraInfo: DWORD); stdcall;
```

Спецификация на MSDN

Вообще, это достаточно сложная функция, ей можно и курсор двигать, но я ограничусь нажатиями. Куда более полное описание есть здесь.

Нажатие всегда происходит под курсором, поэтому передавать координаты сюда бесполезно, надо лишь переместиться в нужную точку до того, как нажимать. Вот список констант, определяющих, что именно мы хотим нажать:

```
const
  MOUSEEVENTF_LEFTDOWN = $0002; // нажать левую кнопку
  MOUSEEVENTF_LEFTUP = $0004; // отпустить левую кнопку
  MOUSEEVENTF_RIGHTDOWN = $0008; // нажать правую кнопку
  MOUSEEVENTF_RIGHTUP = $0010; // отпустить правую кнопку
  MOUSEEVENTF_MIDDLEDOWN = $0020; // нажать среднюю кнопку
  MOUSEEVENTF_MIDDLEUP = $0040; // отпустить среднюю кнопку
  MOUSEEVENTF_XDOWN = $0080; // нажать X-кнопку
  MOUSEEVENTF_XUP = $0100; // отпустить X-кнопку
```

Часть из них точно уже есть в Вашей версии Delphi в модуле "Windows", здесь они приведены на случай, если чего-то не хватает.

Внимание! Не забывайте отпускать нажатые кнопки!

Итого, что мы имеем: в dwFlags определяем действие; dx и dy это 0; в dwData код X-кнопки, которую нажимаем, иначе 0; dwExtraInfo тоже 0. Возможные значения кода X-кнопки:

```
const
  XBUTTON1 = $0001; // первая X-кнопка
  XBUTTON2 = $0002; // вторая X-кнопка
```

Прокрутка колёсика

Пользуясь всё той же функцией mouse_event можно прокручивать колёсико. Для этого имеется две константы:

```
// Код события
const
  MOUSEEVENTF_WHEEL = $0800; // прокрутить колёсико

// Единицы прокрутки
const
  WHEEL_DELTA = 120; // Один шаг колёсика
```

Применение такое: dwFlags - код события прокрутки; dx и dy - 0; dwData - величина, на которую прокручиваем; dwExtraInfo - 0. Значение dwData может быть как кратно WHEEL_DELTA:

```
mouse_event(MOUSEEVENTF_WHEEL, 0, 0, WHEEL_DELTA * 3, 0);
```

Что совершит прокрутку в три шага колёсика; так и может быть куда меньше, например:

```
mouse_event(MOUSEEVENTF_WHEEL, 0, 0, WHEEL_DELTA div 5, 0);
```

Что совершит прокрутку в 1/5 шага. Знак dwData определяет направление прокрутки: положительное значит движение вверх, вращение от пользователя; отрицательное - вниз, вращение на пользователя. Однако, из-за того, что у этого параметра указан беззнаковый тип DWORD (диапазон 0..4294967295), то, если у Вас не получилось использовать отрицательные значения, - воспользуйтесь приведением типов к DWORD:

```
mouse_event(MOUSEEVENTF_WHEEL, 0, 0, DWORD(-2 * WHEEL_DELTA), 0);
```

Скрытие курсора

Специальная функция Windows.ShowCursor, в качестве параметра принимает состояние, которое должен принять курсор (True - видимый, False - невидимый):

```
function ShowCursor(bShow: BOOL): Integer; stdcall;
```

Примеры

Пример 1: Переместить курсор 120 пикселей вправо и на 40 вверх.

```
// Комбинируя 1 и 2 способы
// Вспоминаем как направлены оси X и Y - вправо и вниз
SetCursorPos(Mouse.CursorPos.X + 120, Mouse.CursorPos.Y - 40);
```

Пример 2: Нажать левую кнопку, если курсор расположен строго в левом нижнем углу.

```
// Возможный диапазон X от 0 до Screen.Width - 1
// Возможный диапазон Y от 0 до Screen.Height - 1
if (Mouse.CursorPos.X = 0) and (Mouse.CursorPos.Y = Screen.Height - 1) then
begin
  mouse_event(MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0);
  mouse_event(MOUSEEVENTF_LEFTUP, 0, 0, 0, 0); // не забываем отпускать
end;
```

Кстати, код можно немного укоротить, передав разом нажатие и отпускание в качестве одного действия:

```
  mouse_event(MOUSEEVENTF_LEFTDOWN or MOUSEEVENTF_LEFTUP, 0, 0, 0, 0);
```

Не знаю, предусмотрено ли это, но у меня прекрасно работает 😁

Смотреть последнюю версию статьи онлайн