**Игра «Поймай меня» на Delphi**

Принцип самой простой и хорошо известной большому количеству людей игры «Поймай меня» состоит в том, чтобы поймать кнопку, появляющуюся в разных частях окна программы. Каким образом это можно осуществить?

Сперва подготовим нашу форму к работе. Итак, в меню **File** создадим новое приложение: **File**>**New**>**Application.** Назовем игру «Игра — Поймай меня!». Заголовок формы находится в окне свойств **Object Inspector** на вкладке **Properties.** Меняем значение **Caption** на **Игра — Поймай меня!.** Выключим автопрокрутку (**AutoScroll** устанавливаем значение на **false**). Однако, при запуске программы можно обнаружить, что окошко программы появляется в том же месте, где и находилось перед компиляцией. Чтобы запустить программму в центре экрана, изменим значение **Position** на **poScreenCenter.** Заметим, что чем меньше размер окна, тем проще пройти игру. Возьмем, к примеру размер окна 400×400. Для этого поменяем свойства **Width** и **Height** на **400**. Кроме того, запретим программе разворачиваться на весь экран. Для этого изменим свойство **BorderStyle** на **bsDialog**и расположим наше главное окно поверх всех окон: меняем **FormStyle** на **fsStayOnTop.** Теперь еще раз сверимся с таблицей свойств главного окна:

|  |  |
| --- | --- |
| **Свойства Form1** | **Значение** |
| Caption | Игра — Поймай меня! |
| AutoScroll | false |
| Position | poScreenCenter |
| Width | 400 |
| Height | 400 |
| BorderStyle | bsDialog |
| FormStyle | fsStayOnTop |

Наше главное окно настроено, осталось добавить на него кнопку (**Button1**) и изменить свойства соответственно со следующей таблицей:

|  |  |
| --- | --- |
| **Свойства объекта Button1** | **Значение** |
| Caption | Начать игру |
| Left | 144 |
| Top | 152 |
| Width | 89 |

Далее, так как по условию нашей задачи кнопка должна постоянно менять свои координаты, добавим на форму еще один объект — **Timer** (на вкладке **System**). По-умолчанию таймер включен  и начнет выполняться сразу же после запуска программы. Но нам надо сделать, чтобы он запускался только после нажатия на кнопку «Начать игру». Поэтому поменяем значение **Enabled** на **False.** Кроме того, если наш цикл будет повторяться ежесекундно — кнопка будет менять свои координаты медленно, игра станет более легкой и не интересной. Поэтому давайте свойство **Interval** изменим на **500**.

|  |  |
| --- | --- |
| **Свойства объекта Timer1** | **Значение** |
| Enabled | False |
| Interval | 500 |

Итак, внешний вид программы создан, первоначальные настройки объектов заданы, перейдем к процессу программирования игры «Поймай меня». В будущем, возможно, мы добавим еще несколько объектов.

Прежде всего, необходимо прописать событие для кнопки, чтобы при нажатии на кнопку поменялось ее значение на «Поймай меня!» и запустился таймер. Для этого выделим нашу кнопку и в окне **Object Inspector** перейдем на вкладку **Events.** Затем, найдем процедуру **OnClick** и рядом пропишем ее название, например, **OnButtonClick.** Нажимаем на нее два раза кнопкой мыши (или, по желанию, Enter) и у нас появляется окно Unit1.pas со следующим кодом:

procedure TForm1.OnButtonClick(Sender: TObject);  
begin

end;

end.

Теперь между строками **begin** и **end** пропишем код, позволяющий поменять значение кнопки «Начать игру» на «Поймай меня!». Для этого обратимся к кнопке **Button1**, укажем ей свойство, которое необходимо поменять (**Button1.Caption**) и присвоим ей новое значение — «Поймай меня!»:

Button1.Caption:=’Поймай меня!’;

Дальше нам нужно запустить сам таймер. Для этого точно так же обращаемся к объекту **Timer1**, выбираем свойство **Enabled** и меняем его значение на **True**:

Tier1.Enmabled:=true;

Перейдем к объекту **Timer1**. Нажимаем на значок секундомера два раза и прописываем код, позволяющий кнопке появится в случайной координате нашего окна:

Button1.Left:=random(form1.ClientWidth-button1.Width);  
Button1.Top:=random(form1.ClientHeight-button1.Height);

Если Вам генератор случайных чисел (random) еще не знаком — рекомендую сначала научиться делать более простые программы с помощью Turbo Pascal и только после этого переходить на объектно-ориентированное программирование.

Но что же означает form1.ClientWidth и form1.ClientHeight? Конечно же, это только внутренняя рабочая область нашей формы без полей, автопрокруток и т.п. Если бы мы взяли просто form1.width, мы бы натолкнулись на некоторые неточности и наша кнопка стала бы выезжать за пределы окна. Кроме того, чтобы в нашем случае кнопка не выходила за пределы окна, надо вычесть ее ширину и высоту. Попробуем запустить программу. Уже становится более интересно. Но как узнать, нажали ли мы на эту кнопку? Добавим счетчик, позволяющий подсчитывать количество кликов, а результат высветим на экран.

Итак, добавим 4 объекта надписи (TLabel). Переименуем эти объекты в соотвествии со следующими таблицами:

|  |  |
| --- | --- |
| **Свойства объекта Label1** | **Значение** |
| Caption | Поймано: |
| Left | 8 |
| Top | 8 |
| Height | 13 |
| Width | 49 |

|  |  |
| --- | --- |
| **Свойства объекта Label2** | **Значение** |
| Name | SumClick |
| Caption | 0 |
| Left | 64 |
| Top | 8 |
| Height | 13 |
| Width | 6 |

|  |  |
| --- | --- |
| **Свойства объекта Label3** | **Значение** |
| Caption | Осталось: |
| Left | 296 |
| Top | 8 |
| Height | 13 |
| Width | 52 |

|  |  |
| --- | --- |
| **Свойства объекта Label4** | **Значение** |
| Name | Time |
| Caption | 0 |
| Left | 352 |
| Top | 8 |
| Height | 13 |
| Width | 12 |

Пойдем дальше, так как у нас будет использоваться всего одна кнопка, соответственно нужно определить, при каком условии нужно запускать программу, а при каком — подсчитывать количество нажатий. Можно оттолкнуться от таймера — если он работает — значит нужно ловить количество нажатий, если же нет — запускаем программу с самого начала.

И тут появляется самая распространенная ошибка. Программа Delphi, так же, как и Pascal, очень строгая программа, особенно к ошибкам. И если к значению **SumClick** прибавить просто единицу (SumClick.Caption:=SumClick.Caption+1;), то это будет то же самое, что если мы сложим какой-то текст и 1. Это просто невозможно! Поэтому сначала надо перевести значение строки SumClick в число. Специально для этого есть функция StrToInt(). Тогда, чтобы перевести количество кликов в число, необходимо прописать следующее: StrToInt(SumClick.Caption);

Но это еще не все. Теперь, чтобы нам присвоить полученное значение, нужно обратно перевести значение из числа в строку. Для этого есть функция IntToStr(). Таким образом наш счетчик прописываем следующим образом:

SumClick.Caption:=IntToStr(StrToInt(SumClick.Caption)+1);

Или построчно, используя переменную a, как integer:

a:=StrToInt(SumClick.Caption);     // переводим строку в число  
a:=a+1;     // прибавляем единицу  
SumClick.Caption:=IntToStr(a);     // переводим обратно число в строку

Таким образом наш код примет следующий вид:

if Timer1.Enabled=false then  
begin  
Button1.Caption:=’Поймай меня!’;  
Timer1.Enabled:=true;  
end  
else  
SumClick.Caption:=IntToStr(StrToInt(SumClick.Caption)+1);

Однако любая программа должна каким-то образом придти к своему логическому завершению. Для этого добавим еще один объект **Timer2**. Он будет отсчитывать количество оставшегося времени (в сек). Допустим поставим секундомер на 1 минуту. Соответственно в 1-ой минуте содержится 60 секунд. В программировании у объекта Timer (свойство Interval) значение 1000 равняется одной секунде (скорость отсчета). В нашей программе 60 секунд мы устанавливаем для объекта Time.

Time.Caption:=’15’;

|  |  |
| --- | --- |
| **Свойства объекта Timer2** | **Значение** |
| Enabled | False |
| Interval | 1000 |

Этот таймер будет запускаться вместе со вторым таймером во время нажатия на кнопку «Начать игру».

if Timer1.Enabled=false then  
begin  
Button1.Caption:=’Поймай меня!’;  
Timer1.Enabled:=true;  
Timer2.Enabled:=true;  
end  
else  
SumClick.Caption:=IntToStr(StrToInt(SumClick.Caption)+1);

Перейдем ко второму таймеру. Здесь действует такая же функция, как и у той, что мы прописывали для кнопки подсчета количества кликов. Единственное отличие — нам нужно не складывать предыдущее значение на 1, а вычитать.

Time.Caption:=IntToStr(StrToInt(Time.Caption)-1);

Теперь добавим условие, чтобы узнать, осталось ли у нас еще время или нет. Если же осталось — переходим к счетчику времени и вычитаем единицу. Если не осталось — выключаем все таймеры, ставим кнопку в первоначальное положение и проверяем, сколько раз мы ее поймали. Если ни разу — высвечиваем сообщение о том, что мы проиграли, иначе — мы выиграли.

if StrToInt(Time.Caption)>0 then  
Time.Caption:=IntToStr(StrToInt(Time.Caption)-1)  
else  
begin  
Timer1.Enabled:=false;  
Timer2.Enabled:=false;  
Button1.Left:=144;  
Button1.Top:=152;  
Button1.Caption:=’Повторить’;  
if SumClick.Caption=’0′ then  
ShowMessage(‘Вы проиграли!’)  
else  
ShowMessage(‘Поздравляем! Вы выиграли и поймали кнопку ‘+SumClick.Caption+’ раз’);  
end;