

Цикл презентаций «ООП на Delphi» посвящен объектно – ориентированному программированию с использованием одной из самых распространенных систем быстрой разработки приложений – Delphi

Используя данный учебный курс, можно самостоятельно овладеть основами объектно – ориентированного программирования на Delphi. Для расширения Ваших знаний к курсу приложен ряд учебных пособий и справочников по Delphi

---

## Цикл содержит 13 презентаций:

ООП на Delphi – 1: Знакомство с системой программирования Borland Delphi. Объекты (компоненты) и их свойства и методы

ООП на Delphi – 2: Первая программа на Delphi, сохранение и компиляция

ООП на Delphi – 3: Программное изменение свойств объектов

ООП на Delphi – 4: Условия в Delphi. Создание простого теста

ООП на Delphi – 5: Элементы ввода и вывода информации. Обработка исключений

ООП на Delphi – 6: Заставка программы и элемент таймер

**ООП на Delphi – 7: Программируем свою игрушку**

ООП на Delphi – 8: Меню программы, диалоги

ООП на Delphi – 9: Создаем свой текстовый редактор

ООП на Delphi – 10: Базы данных на Delphi

ООП на Delphi – 11: Калькулятор на Delphi. Обработка исключительных ситуаций

ООП на Delphi – 12: Создаем тестирующую систему

ООП на Delphi – 13: Графика на Delphi

---

Delphi использует язык программирования Объект Паскаль, поэтому лучше сначала изучить обычный Паскаль и поработать в ТурбоПаскале, а затем и переходить к Delphi – перейти будет очень просто, т.к синтаксис языка остается неизменным.

Изучение ООП на Delphi желательно проводить в старших профильных классах – количество часов, отводимое на информатику там вполне достаточно для освоения основ ООП на Delphi

Объектно –  
ориентированное  
программирование на

**Borland®**

**DELPHI - 7**

## DELPHI - 7

### На этом уроке:

Мы создадим свою игрушку для досуга, используя знания, полученные на предыдущих уроках

### Вопросы:

Здесь этот урок посвящен созданию  
игры «Раздави пауков»

# Создание игры «Раздави пауков»

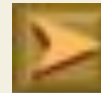
Вначале давайте сыграем в эту игру (выберите режим медленно), чтобы иметь представление о том что на этом уроке мы создадим. Играя, обратите внимание на используемые компоненты и функциональность программы

Хотя программирование игрушек не входит в наш курс, но это хороший способ попрактиковаться в использовании компонент и построении логики программы

Кроме того эта игрушка – хорошее средство приручить мышку к своей руке

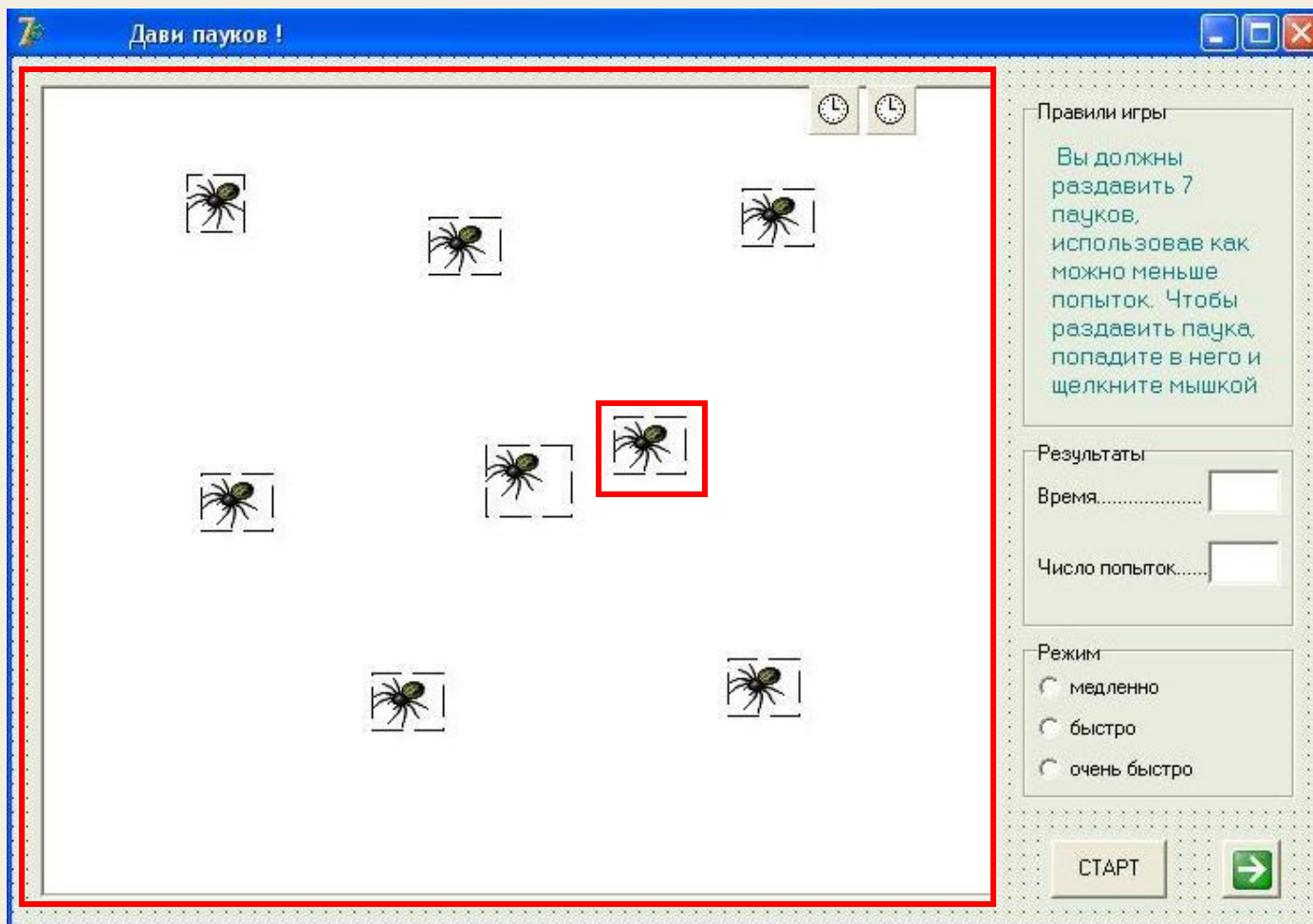
---

**Сыграть ->**



## ШАГ 1

Как обычно, запускаем Delphi и конструируем форму

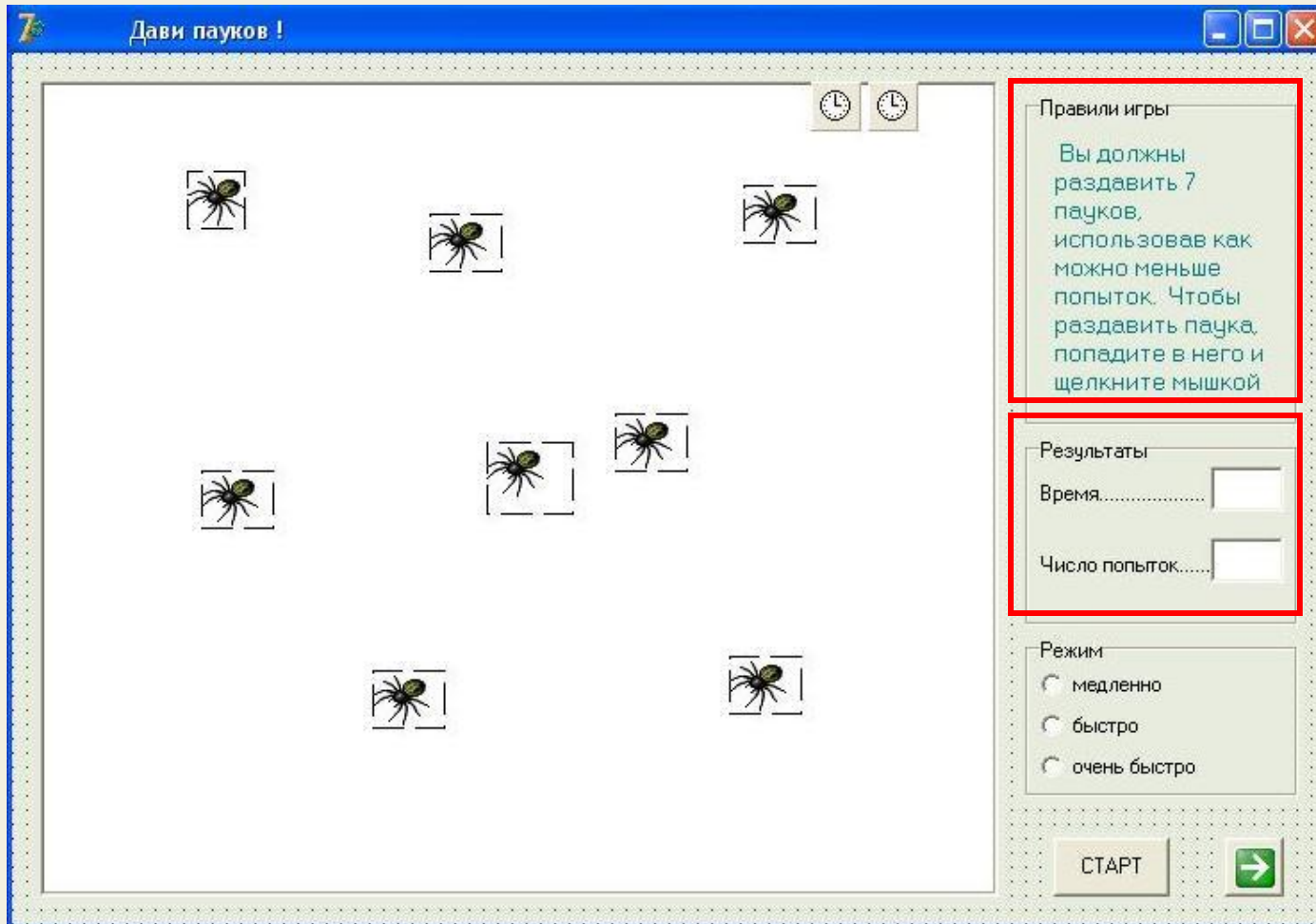


Положим на форму панель (Panel1) – это будет наше поле боя.  
Подберем цвет панели

Через компонент Image помещаем на панель несколько жучков (8), которые будут прыгать по полю

**ШАГ 1**

Как обычно, запускаем Delphi и конструируем форму

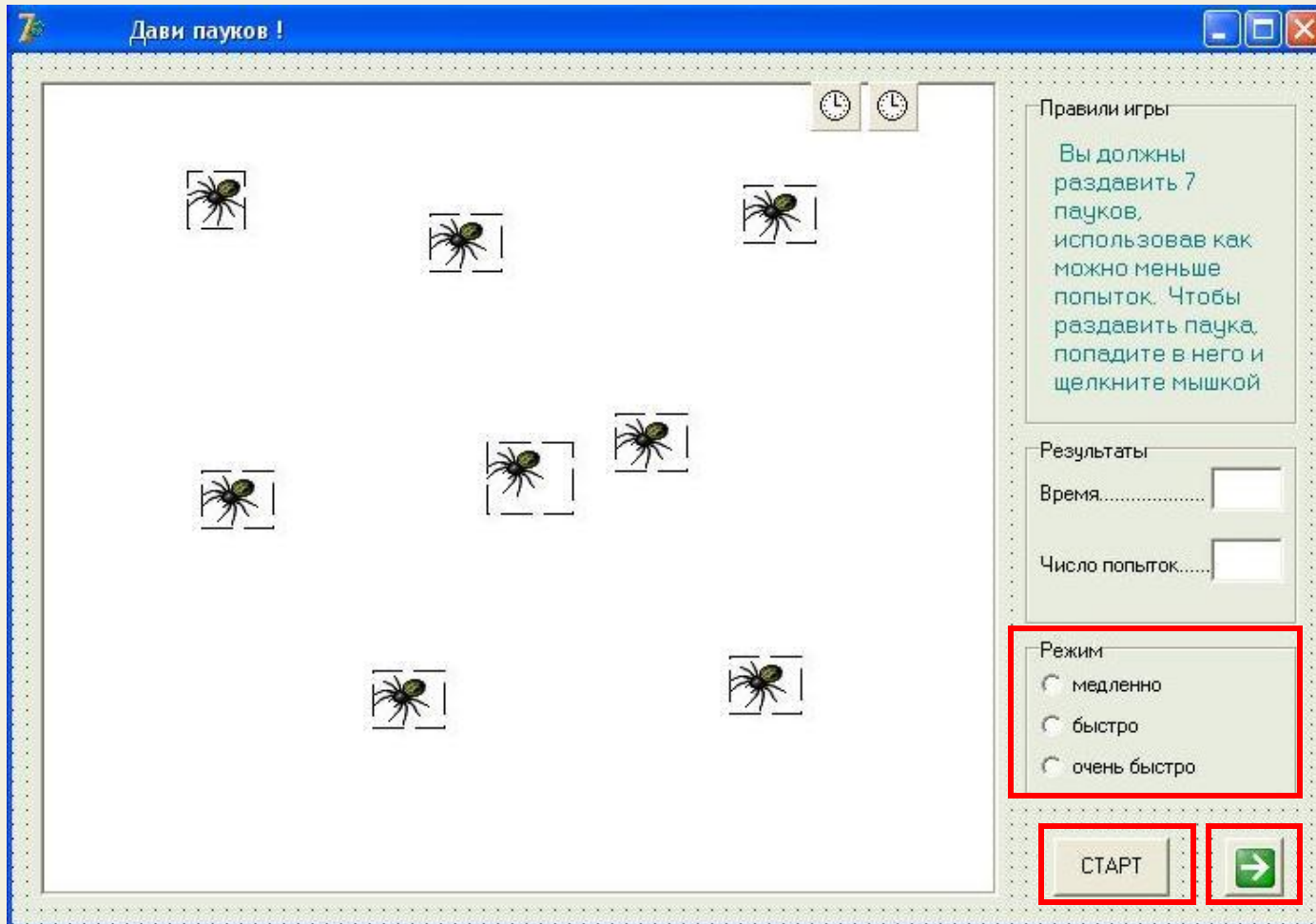


Вставляем **Group Box**, в котором размещаем **Label** с правилами игры

Вставляем **Group Box**, в котором размещаем 2 **Label**-а и 2 **Мето** для вывода результатов: времени, затраченного на игру и числа попыток

**ШАГ 1**

Как обычно, запускаем Delphi и конструируем форму



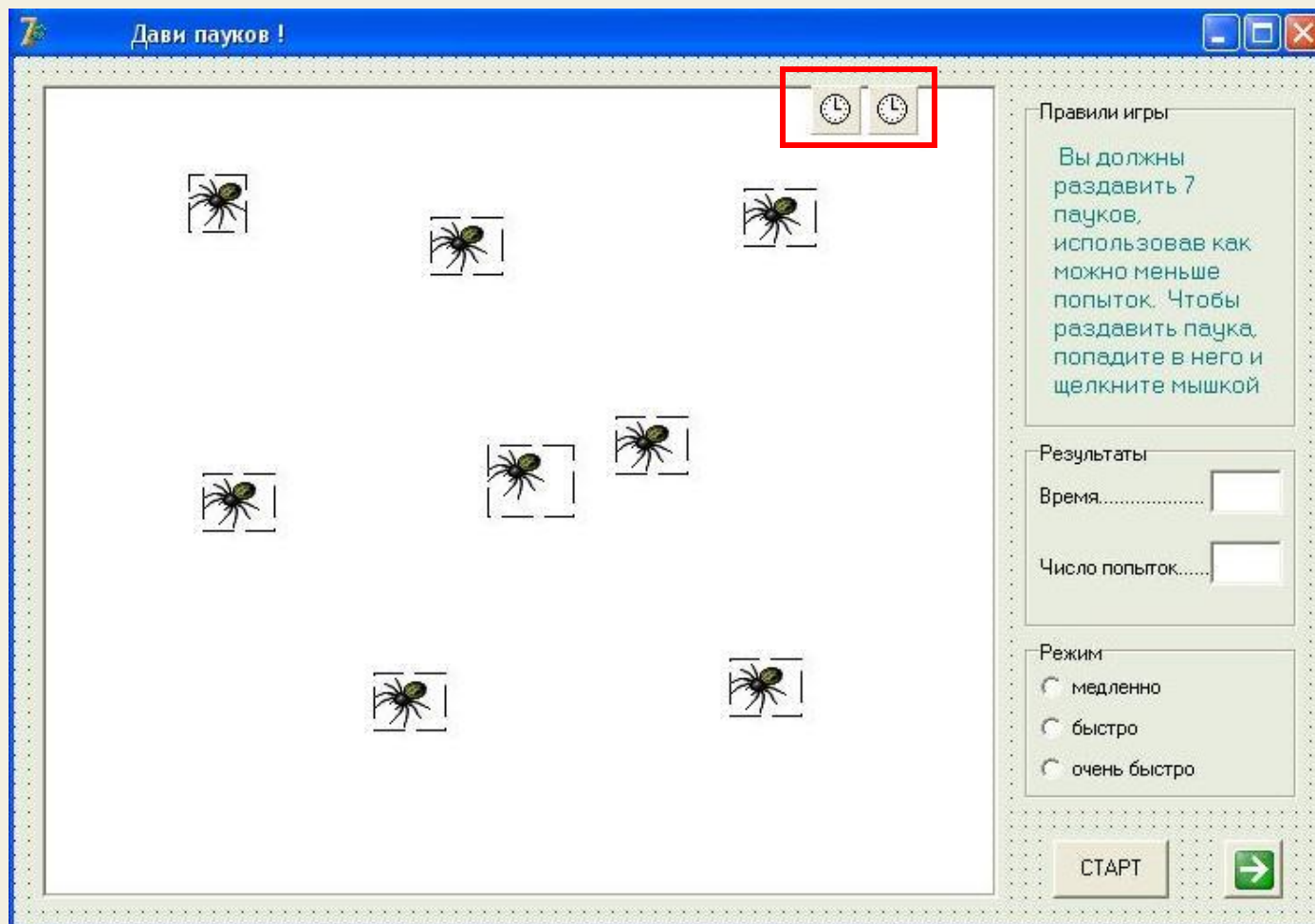
В следующем **Group Box**-е вставляем **Radio Group** на три переключателя для выбора режима игры

Размещаем кнопки **СТАРТ** и **ВЫХОД**



**ШАГ 1**

Как обычно, запускаем Delphi и конструируем форму



Поместим на форму **два таймера**:

**Первый** будет отсчитывать время, которое мы затратили на игру

**Второй** будет заставлять прыгать паучков через определенные промежутки времени

**ШАГ 2**

Сейчас приступим к написанию кода игры и начнем с **события создания формы (On Create)**, возникающего каждый раз самым первым при запуске приложения

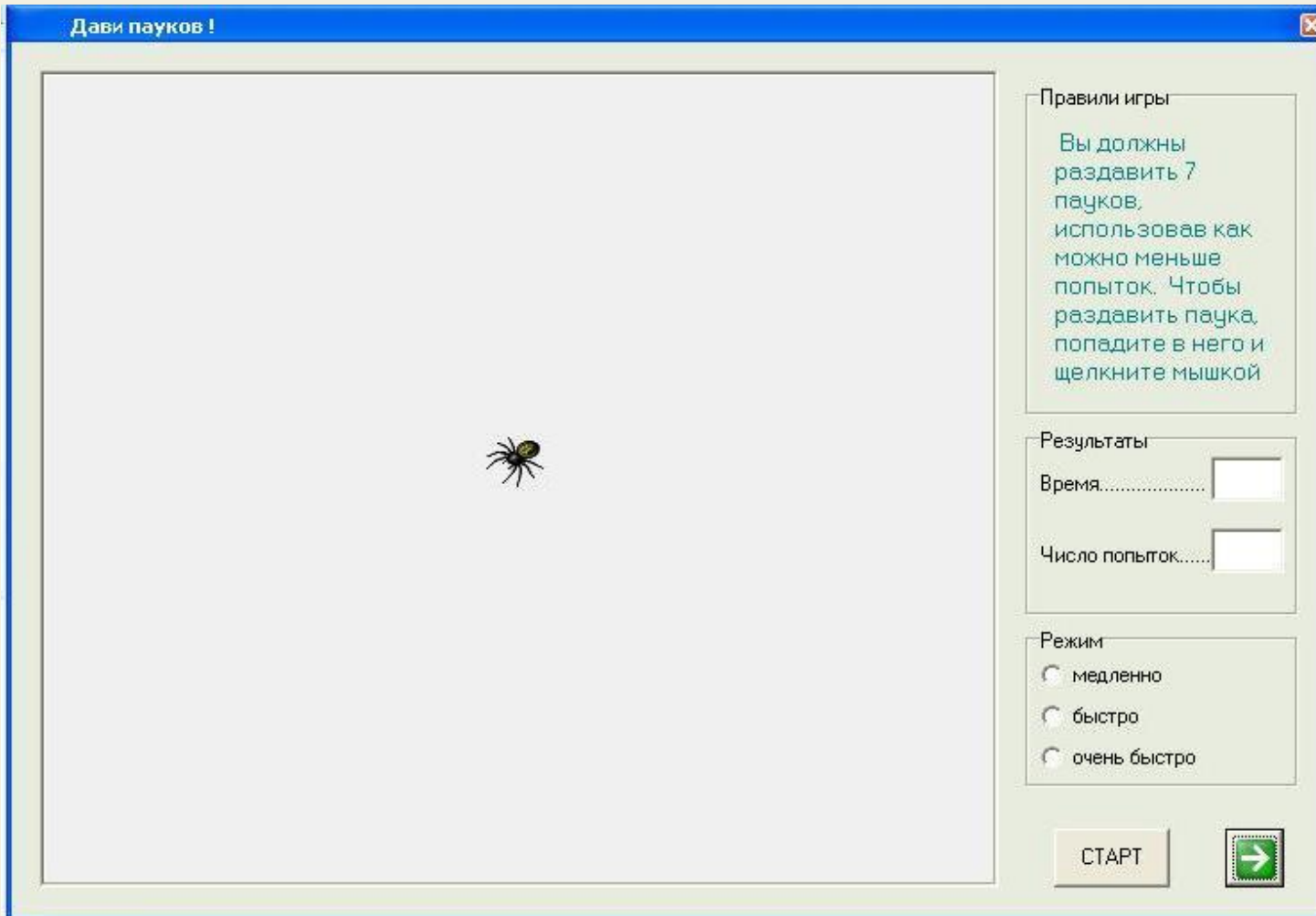
```
procedure TForm1.FormCreate(Sender: TObject);  
begin  
panel1.Color:=rgb(240,240,240);  
panel1.Enabled:=false;  
timer1.Enabled:=false;  
  timer2.Enabled:=false;  
Image1.Visible:=false;  
Image2.Visible:=false;  
Image3.Visible:=false;  
Image4.Visible:=false;  
Image5.Visible:=false;  
Image6.Visible:=false;  
Image7.Visible:=false;  
image8.Visible:=true;
```

Давайте сделаем, чтобы при запуске игры панель1 (поле боя) было неактивным, серым и на нем в качестве логотипа в середине один паучок. Время игры идти не должно

- Красим панель серой краской
- Делаем ее недоступной
- Оба таймера делаем СТОП
- Паучков, которые будут прыгать, делаем невидимыми
- Паучка – логотип делаем видимым

**ШАГ 2**

Сейчас приступим к написанию кода игры и начнем с **события создания формы (On Create)**, возникающего каждый раз самым первым при запуске приложения



В результате при запуске программы мы увидим такую форму

Чтобы во время игры нельзя было изменить размеры формы (и поля боя), сделайте свойство формы **BorderStyle** равным **Tool Window**

## ШАГ 3

## Опишем процедуру нажатия на кнопку СТАРТ

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  k:=0;
  s:=0;
  p:=0;
  panel1.Enabled:=true;
  panel1.Color:=rgb(255,255,255);
  image1.Visible:=true;
  image2.Visible:=true;
  image3.Visible:=true;
  image4.Visible:=true;
  image5.Visible:=true;
  image6.Visible:=true;
  image7.Visible:=true;
  timer1.Enabled:=true;
  timer2.Enabled:=true;
  image8.Visible:=false;;
  panel1.Caption:='';
  memo1.Text:='';
  memo2.Text:='';
  if radiogroup1.ItemIndex=0 then timer1.Interval:=800;
  if radiogroup1.ItemIndex=1 then timer1.Interval:=500;
  if radiogroup1.ItemIndex=2 then timer1.Interval:=300;
end;
```

Здесь мы видим три переменных: **k,s,p**.

Для чего они введены?

- 1) В переменной **k** будет храниться количество попаданий по паучкам во время игры
- 2) В переменной **s** будет храниться время, затраченное на игру
- 3) В переменной **p** будет храниться число попыток раздавить паучков (считая число попаданий и число промахов)

При каждом старте эти переменные должны обнуляться

**ШАГ 3**Опишем процедуру нажатия на кнопку **СТАРТ**

```
procedure TForm1.Button1Click(Sender: TObject);
begin
k:=0;
s:=0;
p:=0;
panel1.Enabled:=true;
panel1.Color:=rgb(255,255,255);
image1.Visible:=true;
image2.Visible:=true;
image3.Visible:=true;
image4.Visible:=true;
image5.Visible:=true;
image6.Visible:=true;
image7.Visible:=true;
timer1.Enabled:=true;
timer2.Enabled:=true;
image8.Visible:=false;
panel1.Caption:='';
memo1.Text:='';
memo2.Text:='';
if radiogroup1.ItemIndex=0 then timer1.Interval:=800;
if radiogroup1.ItemIndex=1 then timer1.Interval:=500;
if radiogroup1.ItemIndex=2 then timer1.Interval:=300;
end;
```

Делаем **Panel1** (поле боя) доступной и красим ее белым цветом

Все 7 паучков, которые будут прыгать, делаем **ВИДИМЫМИ**

Восьмого паучка (логотип) делаем **НЕВИДИМЫМ**

## ШАГ 3

## Опишем процедуру нажатия на кнопку СТАРТ

```
procedure TForm1.Button1Click(Sender: TObject);
begin
k:=0;
s:=0;
p:=0;
panel1.Enabled:=true;
panel1.Color:=rgb(255,255,255);
image1.Visible:=true;
image2.Visible:=true;
image3.Visible:=true;
image4.Visible:=true;
image5.Visible:=true;
image6.Visible:=true;
image7.Visible:=true;
timer1.Enabled:=true;
timer2.Enabled:=true;
image8.Visible:=false;
panel1.Caption:='';
memo1.Text:='';
memo2.Text:='';
if radiogroup1.ItemIndex=0 then timer1.Interval:=800;
if radiogroup1.ItemIndex=1 then timer1.Interval:=500;
if radiogroup1.ItemIndex=2 then timer1.Interval:=300;
end;
```

Запускаем оба таймера:  
пошел отсчет времени и  
паучки запрыгали

Очищаем все надписи на  
панели и в Мето для  
вывода результата (Эти  
надписи у нас появятся в  
конце игры)



**ШАГ 3**Опишем процедуру нажатия на кнопку **СТАРТ**

```
procedure TForm1.Button1Click(Sender: TObject);
begin
k:=0;
s:=0;
p:=0;
panel1.Enabled:=true;
panel1.Color:=rgb(255,255,255);
image1.Visible:=true;
image2.Visible:=true;
image3.Visible:=true;
image4.Visible:=true;
image5.Visible:=true;
image6.Visible:=true;
image7.Visible:=true;
timer1.Enabled:=true;
  timer2.Enabled:=true;
image8.Visible:=false;;
panel1.Caption:='';
memo1.Text:='';
memo2|.Text:='';
if radiogroup1.ItemIndex=0 then timer1.Interval:=800;
if radiogroup1.ItemIndex=1 then timer1.Interval:=500;
if radiogroup1.ItemIndex=2 then timer1.Interval:=300;
end;
```

А здесь идет **проверка условия:**

Если выбран первый переключатель(с индексом 0), то интервал прыгания паучков устанавливается 0,8 сек – **медленно**

Если второй – 0,5 сек (**быстро**)

Если третий – 0,3 сек (**очень быстро**)

Таким образом каждый может выбрать свой темп игры

Режим

медленно

быстро

очень быстро

**ШАГ 4**

Опишем событие щелканья мышкой по паучку. Здесь может быть два случая, для каждого из которых надо написать процедуру обработки:

1. Мы попали по паучку (возникает событие `ImageX.Click`)
2. Мы не попали по паучку, но тогда мы попали по панели (возникает событие `Panel1.Click`)

```

procedure TForm1.Image1Click(Sender: TObject);
begin
  Image1.Visible:=false;
  k:=k+1;
  p:=p+1;
  memo2.Text:=inttostr(p);
  if k=7 then
  begin
    panel1.Font.Size:=20;
    panel1.Font.Color:=rgb(255,0,0);
    panel1.Caption:='Игра закончена';
    timer1.Enabled:=false;
    timer2.Enabled:=false;
  end;
end;

```

1. Процедура `Image1.Click` (опишем для первого паучка, а для остальных паучков будет то же самое)

Паука, в которого попали, делаем **невидимым** и он выходит из игры

Считаем **число попыток** и **число попаданий** (в случае попадания число попыток увеличиваем и число попаданий тоже)



**ШАГ 4**

Опишем событие щелканья мышкой по паучку. Здесь может быть два случая, для каждого из которых надо написать процедуру обработки:

1. Мы попали по паучку (возникает событие `ImageX.Click`)
2. Мы не попали по паучку, но тогда мы попали по панели (возникает событие `Panel1.Click`)

```

procedure TForm1.Image1Click(Sender: TObject);
begin
  Image1.Visible:=false;
  k:=k+1;
  p:=p+1;
  memo2.Text:=inttostr(p);
  if k=7 then
  begin
    panel1.Font.Size:=20;
    panel1.Font.Color:=rgb(255,0,0);
    panel1.Caption:='Игра закончена';
    timer1.Enabled:=false;
    timer2.Enabled:=false;
  end;
end;

```

1. Процедура `Image1.Click` (опишем для первого паучка, а для остальных паучков будет то же самое)

В `Memo2` выводим число попыток

Если это последний убитый паук ( $k=7$ ), то на панели красным крупным шрифтом выводим окончание игры и останавливаем таймеры

**ШАГ 4**

Опишем событие щелканья мышкой по паучку. Здесь может быть два случая, для каждого из которых надо написать процедуру обработки:

1. Мы попали по паучку (возникает событие `ImageX.Click`)
2. Мы не попали по паучку, но тогда мы попали по панели (возникает событие `Panel1.Click`)

## 2. Процедура `Panel1.Click`

```
procedure TForm1.Panel1Click(Sender: TObject);  
begin  
  p:=p+1;  
  Memo2.Text:=inttostr(p);  
end;
```

Увеличиваем счетчик  
попыток

Выводим число  
использованных попыток в  
`Memo2`

**ШАГ 5**

Опишем процедуры тиканья таймеров (**OnTimer**)

Напомню, что **Timer1** управляет частотой перемещения пауков, а **Timer2** – считает время игры (в переменной **S** и отображает в **Memo1**)

**Timer1**

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  randomize;
  Image1.Left:= 50+random(300);
  Image1.Top:= 50+random(300);
  Image2.Left:= 500-random(400);
  Image2.Top:= 400-random(300);
  Image3.Left:= random(300);
  Image3.Top:= random(300);
  Image4.Left:= 200+random(300);
  Image4.Top:= 50+random(300);
  Image5.Left:= 350-random(150);
  Image5.Top:= 350-random(150);
  Image6.Left:= random(500);
  Image6.Top:= random(400);
  Image7.Left:= 100+random(300);
  Image7.Top:= 50+random(300);
end;
```

Инициализируем генерацию случайных чисел

Генерируем для каждого паука его положение от левого края и верха поля боя случайным образом

(Здесь необходимо учесть, чтобы в результате **random** паук не выпрыгивал за поле боя – для этого посмотрите размеры панели в инспекторе объектов и подберите соответствующие **random**-ы

**ШАГ 5**

Опишем процедуры тиканья таймеров (**OnTimer**)

Напомню, что **Timer1** управляет частотой перемещения пауков, а **Timer2** – считает время игры (в переменной **S** и отображает в **Memo1**)

**Timer2**

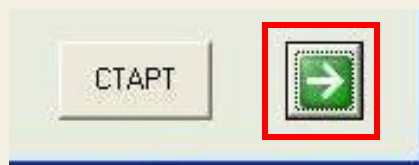
```
procedure TForm1.Timer2Timer(Sender: TObject);  
begin  
s:=s+0.1;  
Memo1.Text:=floattostr(s);  
end;
```

Увеличиваем значение **S**  
на 0,1 секунды

Выводим в **Memo1** показания  
переменной **s**, в результате в  
**Memo1** идет динамический  
подсчет времени игры с  
точностью до десятых долей  
секунды

**ШАГ 6**

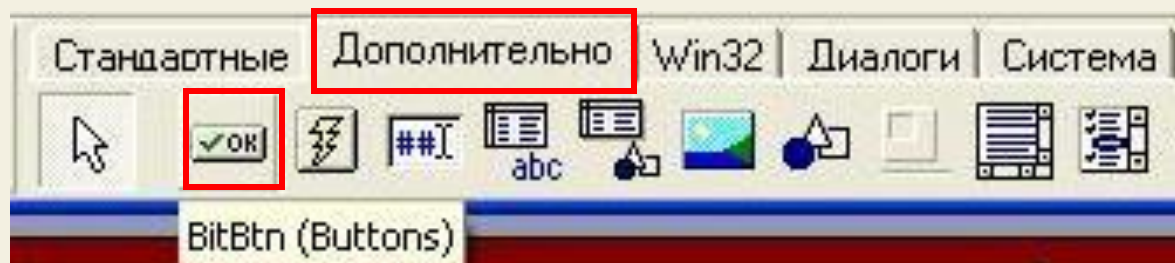
Кнопка **ВЫХОД** комментариев на требует ( метод **Close**)



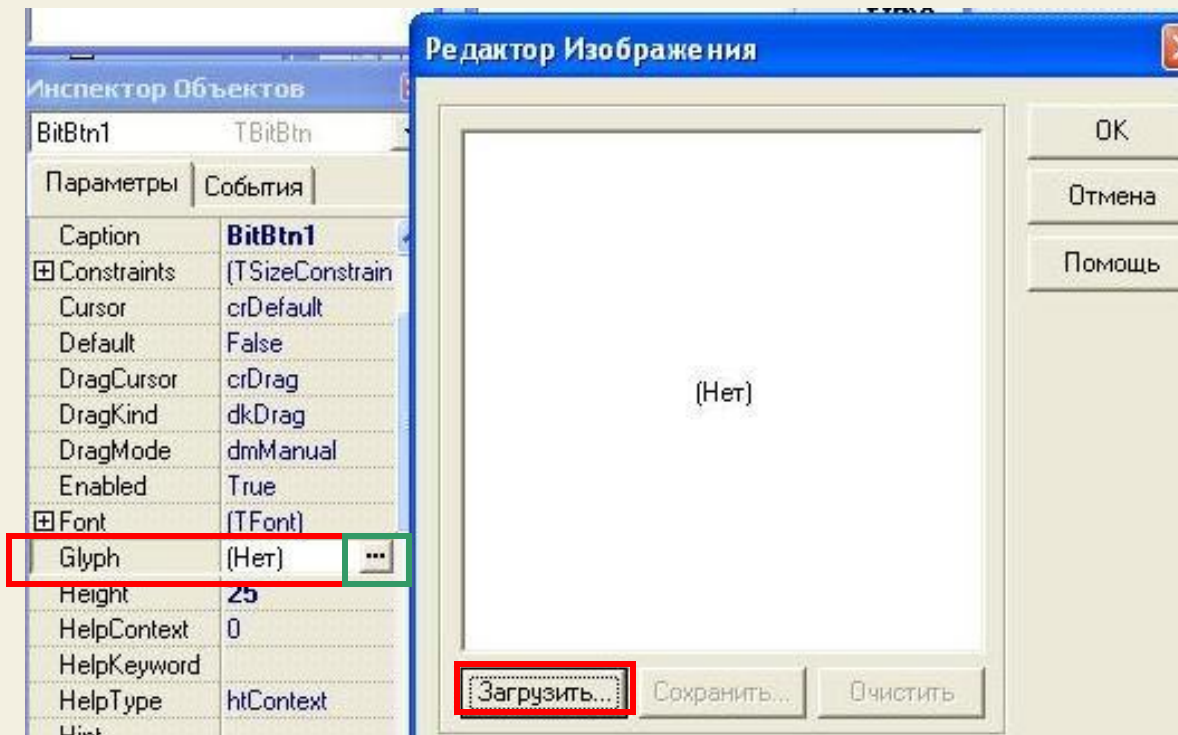
Но вместо обычной командной кнопки с надписью **ВЫХОД** мы видим в приложении **кнопку с пиктограммой**

Давайте разберемся, как ее сделать.

Это тоже командная кнопка, но с картинкой ( **Bit Button**), и находится она на вкладке **Дополнительно**



Помещаем кнопку на форму и в инспекторе объектов раскрываем свойство кнопки **Glyph**, дальше – **загрузить** и находим маленькую картинку (\*.bmp) или рисуем сами



## ШАГ 7

И последний шаг: сохраняем проект и компилируем его.  
Сейчас можно и поиграть

---

Поиграть ->



**Итак, на этом уроке мы создали простую игровую программу (обозначим ее версией 1.0). Вместо паучков мы могли взять, например, уток, зайчиков ... – и тогда у нас получилась бы другая игра (про охотников). Этих уток можно заставить появляться по очереди на 1-2 секунды, чтобы за это время успеть выстрелить, причем для каждой картинке можно использовать свой таймер**

**Понятно, что здесь огромное поле творчества. Поэтому попробуйте сделать свою версию игры, измените ее функциональность, внесите и реализуйте свои идеи и мысли**

## **ИТОГИ УРОКА:**

**На этом уроке мы создали свою игрушку для досуга, используя знания, полученные на предыдущих уроках**

---

## **НА СЛЕДУЮЩЕМ УРОКЕ:**

**ООП на Delphi – 8:**

**Мы познакомимся с созданием меню программы, статусной панели, а также различными диалогами и сообщениями**



Домнин Константин Михайлович

E – mail: [kdomnin@list.ru](mailto:kdomnin@list.ru)

2006 год.