

Практическая работа №10, ЭКРАННАЯ ЗАСТАВКА

Постановка задачи



Создайте программу, являющуюся экранной заставкой, которая на черном фоне выводит текущее время. Выводимое сообщение перемещается по экрану случайным образом.

После запуска программы появляется изображение аналогичное рис.10.1.

Рис.10.1.

Новым в этой работе является:

- компонента **TTimer** (вкладка палитры компонентов **System**),
- создание программы с экранной заставкой и расширением *.scr.

Пояснение к задаче

Экранная заставка занимает весь экран, не имеет системной строки, главного меню и обрамления главного окна. Да и окно у экранной заставки, как правило, одно. Еще одно различие – файл с экранной заставкой имеет расширение *.scr (от англ.screen - экран) а не *.exe, как программа. Тем не менее, экранная заставка – та же программа.

Каким образом мы можно реализовать такую заставку в виде часов? Ответ напрашивается сам собой: установить в центре экрана обычную метку, на которой в свойстве **Caption** выводить текущее время и перемещать эту метку по экрану по случайной траектории.

План разработки программы

1. Откройте новый проект и сохраните код программы и проект под именами, например, **Main.pas** и **Scr_Clock.lpr**.
2. Разместите на рабочем поле две компоненты **Label** и **TTimer** и дадим им новые имена.

Выделенный объект	Вкладка окна Инспектор объектов	Имя свойства/Имя события	Значение/Действие
Form1	Properties /Свойства	Name	fMain
Label1	Properties /Свойства	Name	IClock

3. Все основные свойства объектов fMain (Form1) и lClock (Label1) установим в момент создания формы (начала работы программы) без «ручной работы».

Выделенный объект	Вкладка окна Инспектор объектов	Имя события	Значение/Действие
fMain	Events/ Событие	OnCreat	<pre>// Для Form изменение свойств fMain.BorderStyle:=bsNone; fMain.WindowState:=wsMaximized; fMain.Width:=Screen.Width; fMain.Height:=Screen.Height; fMain.Color:=clBlack; fMain.Cursor:=crNone; // Для Label изменение свойств lClock.ParentColor:=True; lClock.Caption:='00:00:00'; lClock.Font.Color:=clLime; lClock.Font.Size:=50; lClock.Font.Style:=[fsBold]; lClock.Top:=fMain.Height div 2; lClock.Left:=fMain.Width div 2;</pre>

Комментарий

I. Для компоненты **fMain** (Form1):

1. Свойство **BorderStyle** установим в **bsNone** – отсутствует системная строка.
2. Свойство **WindowState** установим в **wsMaximized** – окно должно занимать весь экран.
3. Свойства **Width** и **Height** установим в зависимости от размеров экрана.
4. Свойство **Color** установим в **clBlack**, чтобы форма стала черной.
5. Чтобы в работающей заставке не было видно указателя мыши свойство формы **Cursor** переведем в **crNone**.

II. Для компоненты **lClock** (Label1):

1. Свойство **ParentColor** установим **True** – заставит метку всегда сливаться с формой, а цвет шрифта будет контрастировать с общим фоном.
2. Свойство **Caption** установим 00:00:00.
3. Свойства **Font.Color**, **Font.Size**, **Font.Style** определяют цвет, размер и стиль.
4. Свойства **Top** и **Left** задают место начального положения объекта (отступы сверху и слева) – середина экрана.

4. Компонент **TTimer** является не визуальным, т.е. невидимый для пользователя, поэтому его можно установить в любое место на форме.

Interval - это свойство устанавливает интервал времени, когда таймер срабатывает. По умолчанию, он равен 1000, что соответствует 1 секунде.

OnTimer - событие срабатывает всякий раз, когда заканчивается установленный интервал.

5. Алгоритм движения надписи:

Сгенерируем случайное число от 0 до 3, получится 4 варианта – 4 направления движения: вверх, вниз, влево или вправо. Двигать будем, скажем, на 50 пикселей влево-вправо, или на 25 пикселей вверх-вниз.

Left - это расстояние в пикселях от левого края формы до компонента. Допустим, у метки **Left** равен 100. То есть, от края формы до метки 100 пикселей. Если мы прибавим еще 50, то тем самым, сдвинем метку вправо, а если наоборот, отнимем, то сдвинем влево.

Для движения вверх-вниз используем свойство **Top** – расстояние от верхней части формы до компонента. Движения будут аналогичные, но на 25 пикселей.

6. Для события **OnTimer** компонента **TTimer**:

Выделенный объект	Вкладка окна Инспектор объектов	Имя события	Значение/Действие
TTime	Events/ Событие	OnTimer	<pre> procedure TfMain.Timer1Timer (Sender: TObject); Var X: Byte; begin lClock.Caption:= TimeToStr(Now); X:= Random(4); case X of 0: lClock.Left:= lClock.Left + 50; 1: lClock.Left:= lClock.Left - 50; 2: lClock.Top:= lClock.Top + 25; 3: lClock.Top:= lClock.Top - 25; end; //если вышла - возвращаем её обратно //если ушла влево: if lClock.Left<0 then lClock.Left:=0; //если ушла вверх: if lClock.Top<0 then lClock.Top:=0; //если ушла вправо: if (lClock.Left+lClock.Width>fMain.Width) then lClock.Left:=fMain.Width-lClock.Width; //если ушла вниз: if (lClock.Top + lClock.Height)> fMain.Height then lClock.Top:=fMain.Height-lClock.Height; end; </pre>

Комментарий

1. `lClock.Caption:= TimeToStr(Now);`
Присваиваем свойству **Caption** метки **lClock** текущее время.
2. Функция **Random(4)** возвращает случайное число от 0 до 3.

Если получим 0, то двигать будем вправо: прибавлять к свойству **Left** метки 50 пикселей. Если получим 1, то двигаем влево, отнимая 50 пикселей от свойства **Left**. Если получим 2, то двигаем вниз: прибавляем 25 пикселей к свойству **Top** метки. Ну и если получим 3, то двигаем вверх, прибавляя 25 пикселей к свойству **Top**.

Причем может случиться и так, что надпись окажется за пределами окна. В этом случае нам нужно будет вернуть её обратно в эти пределы. Сделаем это следующим образом: если метка ушла влево, то свойству **Left** просто присвоим значение 0 - метка вернется в пределы формы и окажется прижатой к левому краю.

Если метка ушла за правый край, то присвоим свойству **Left** ширину формы минус ширину метки, таким образом, метка окажется прижатой к правой части формы и будет в её пределах.

Точно таким же образом будем возвращать метку, если она уйдет вверх или вниз за пределы формы.

7. Проверите работу заставки, только не забудьте предварительно сохранить изменения. Чтобы закрыть окно, у которого нет для этого кнопок и меню, воспользуйтесь стандартными кнопками Windows **<Alt + F4>**.

8. Если вас устраивает, что для закрытия заставки приходится нажимать **<Alt + F4>**, то ничего больше делать не нужно. Но мне бы хотелось, чтобы эту заставку можно было закрыть кнопкой **<Esc>** (код #27 в таблице символов ANSI), как любую другую нормальную заставку, то для события **fMain** компонента **OnKeyPress**:

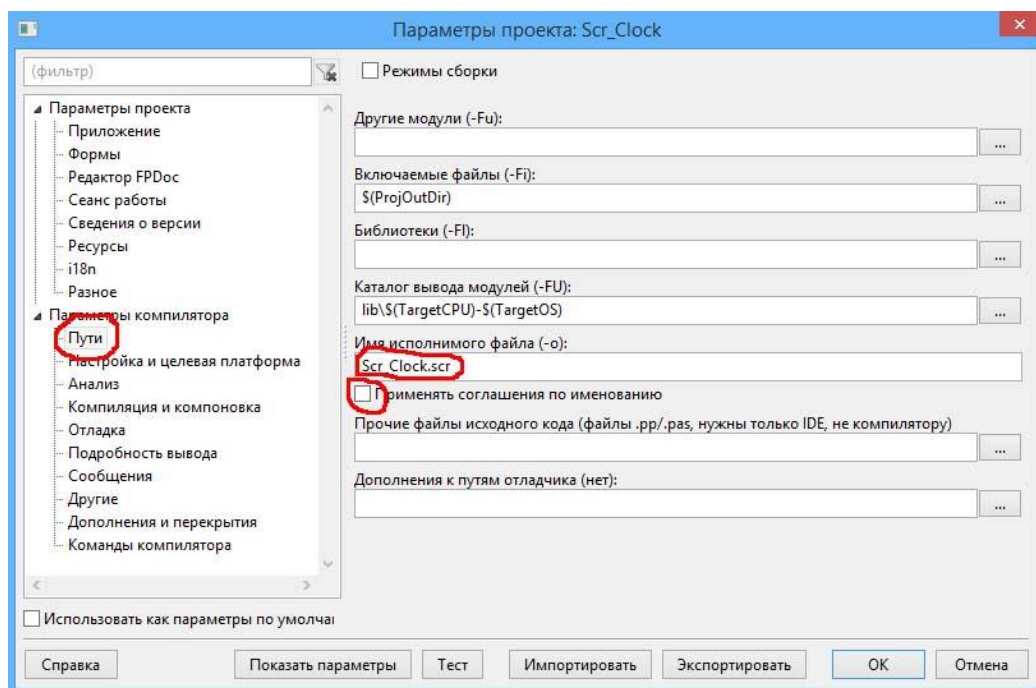
Выделенный объект	Вкладка окна Инспектор объектов	Имя события	Значение/Действие
fMain	Events/ Событие	OnKeyPress	<code>if Key = #27 then Close;</code>

Проверьте работу программы.

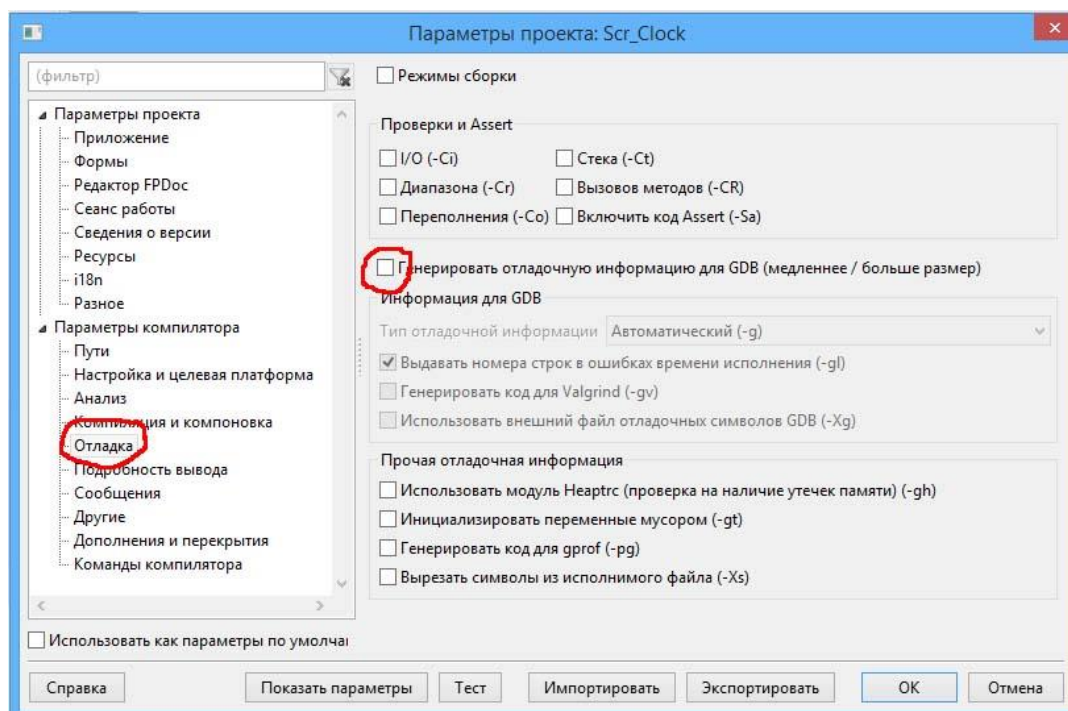
9. Заставка на этом еще не закончена. Прежде всего, из-за включенной отладочной информации программа имеет большой размер, почти 15 мегабайт. Кроме того, мы получили исполняемый exe-файл, а нам нужна заставка с расширением *.scr.

Исправляем эти недостатки:

1. Выбираем команду меню **<Проект -> Параметры проекта>**.
2. В разделе **<Параметры компилятора>** выбираем подраздел **<Пути>**.
3. В поле **<Имя исполнимого файла (-o)>** вместо **Scr_Clock** указываем **Scr_Clock.scr** (то есть, добавляем нужное расширение).
4. Отключаем флажок **<Применять соглашения по именованию>**, чтобы не было конфликта расширений.



5. В разделе <Параметры компилятора> переходим на подраздел <Отладка>.
6. Отключаем флажок <Генерировать отладочную информацию для GDB>, чтобы уменьшить размер полученного файла.



7. Сохраняем проект, и заново его запускаем. В результате получаем требуемый файл **Scr_Clock.scr** размером чуть больше 1,5 мегабайта.

Установка заставки в Windows

Большинство из вас наверняка знает, как полученную заставку установить в своей Windows. Для тех, кто этого все-таки не знает, расскажу подробно:

1. Найдите файл **Scr_Clock.scr**, где вы храните проекты.

2. Скопируйте этот файл в папку **C:\Windows**
3. Щелкните правой кнопкой по свободному от окон месту **Рабочего стола** и выберите команду "**Свойства**".
4. В открывшемся окне "**Свойства: Экран**" перейдите на вкладку "**Заставка**".
5. Там выберите нашу заставку и требуемый интервал её появления, после чего нажмите **<ОК>**:

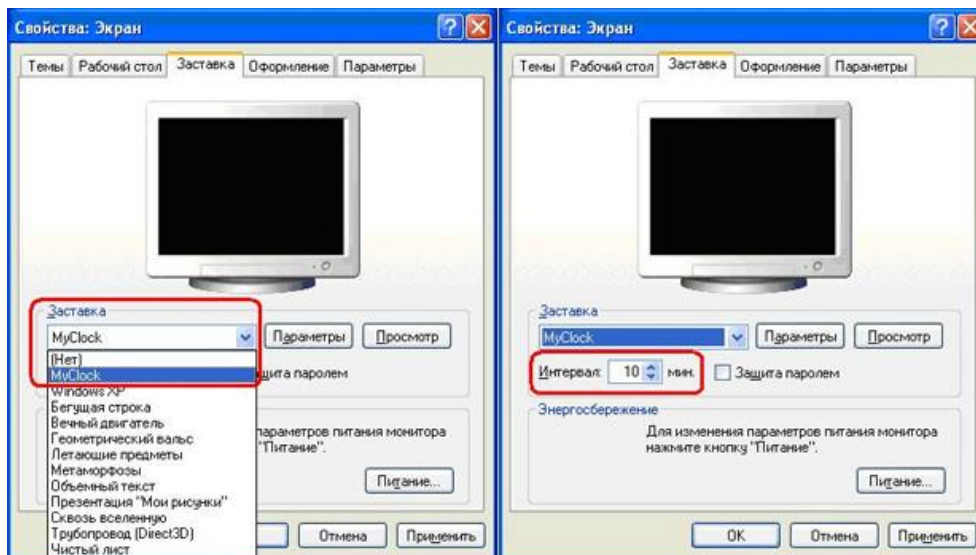


Рис. 10.2. Выбор заставки и интервала

Если пользователь в течение указанного интервала не будет двигать мышью и не нажмет никакой клавиши, то появится наша заставка.