# Урок 3. Создание анимации

*Анимация* – это имитация движения, производимая быстрым отображением на экране серии связанных между собой изображений. Настоящая анимация требует программного перемещения объектов, и часто использует также изменение размера или формы изображения.

Для создания анимации (иллюзии движения на экране какого-либо объекта) применяется принцип смены кадров (изображений), как это делается в мультипликации. Программа, имитирующая движение, должна реализовывать следующие этапы:

* создание изображения на экране;
* реализация временной паузы для того, чтобы глаз зафиксировал изображение;
* коррекция изображения.

Анимация часто используется для показа движения объектов.

## Элемент управления Timer (таймер)

Чтобы объекты двигались, можно использовать циклы, но удобнее воспользоваться элементом управления *Timer* (находится на Панели элементов в разделе *Компоненты*).

Элемент управления *Timer* – это невидимый секундомер, который позволяет использовать системные часы. Он может быть использован как таймер для обратного отсчета от заданного интервала времени, что используется при создании пауз в программе, или повторения действий через заданные интервалы времени.

*Некоторые свойства элемента управления Timer:*

|  |  |  |
| --- | --- | --- |
| *Свойство* | *Значение* | *Комментарии* |
| *Interval* | Int32, значение не может быть меньше единицы | Возвращает или задает время в миллисекундах (одна тысячная доля секунды) до вызова события *Tick* относительно момента, когда событие *Tick* произошло последний раз.  Если значение равно 1000, событие запускается раз в секунду. |
| *Enabled* | *true/false* | Возвращает или задает, запущен ли таймер.  Значение *true*, если данный таймер включен; в противном случае — значение *false* (значение по умолчанию). |

Изменить значения свойств можно как при разработке, так и во время выполнения программы.

Если во время разработки программы установить значение свойства *Enabled* равным *true*, таймер начнет отсчет времени сразу после того, как форма будет загружена в память.

*Некоторые методы элемента управления Timer:*

|  |  |  |  |
| --- | --- | --- | --- |
| *Методы* | *Описание* | *Пример использования* | *Аналогичное изменение свойства* |
| *Start* | Запускает таймер. | Timer1.Start(); | Timer1.Enabled = true; |
| *Stop* | Останавливает таймер. | Timer1.Stop(); | Timer1.Enabled = false; |

Основное событие таймера – *Tick*. Происходит по истечении заданного интервала таймера при условии, что таймер включен. Оно инициализируется системой, и частота появления события зависит от выставленного пользователем интервала: чем меньше установлено значение свойства *Interval*, тем чаще будет вызываться событие. Когда компонент включен, событие *Tick* вызывается через каждый интервал. В этом месте следует добавить исполняемый код.

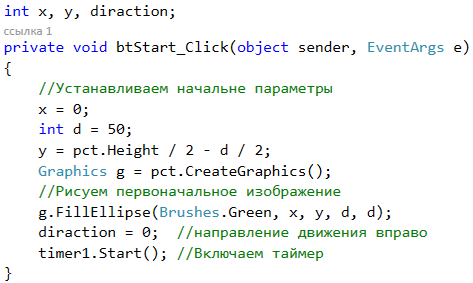
Когда таймер активизирован, он постоянно работает, выполняя через заданные промежутки времени свою процедуру события до тех пор, пока пользователь не остановит программу или не отключит таймер.

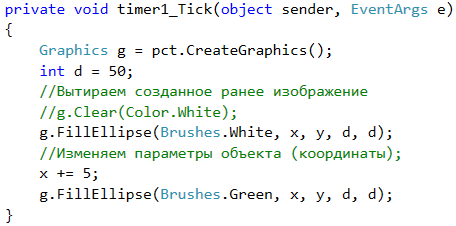
## Процесс создания анимации

В процесс создания анимации можно выделить следующие события:

1. При нажатии на кнопку, которая запускает процесс анимации, выполняем следующие действия:
   * устанавливаем начальные значения изображения (необходимо помнить, что переменные, которые будут изменяться во время анимации, должны быть объявлены общими для всех событий);
   * рисуем начальное изображение;
   * включаем таймер.
2. В событии *timer2\_Tick* выполняем следующие действия:
   * вытираем созданное ранее изображение;
   * изменяем параметры объекта (координаты, размеры или др.);
   * рисуем новое изображение.

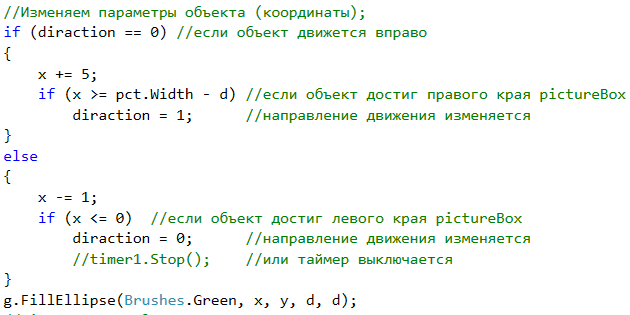
*Задача. Создать следующую анимацию: круг движется на экране слева направо вдоль горизонтальной прямой, проходящей через центр объекта для вывода изображений pictureBox.*





*Продолжение задачи. …достигнув края pictureBox, круг меняет направление на противоположное.*

В таком случае процесс изменения координат может быть следующим:



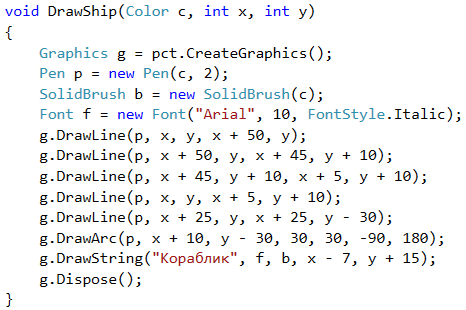
## Анимация изображения, созданного с помощью графических методов

Используя событие таймера можно двигать изображения, созданные с помощью графических методов, как простого объекта типа круг, квадрат или линия, так и комбинированного. В последнем случае логично написать процедуру, которая рисует указанный объект, опираясь на координату одной из его точек. А затем вызывать эту процедуру в событии таймера, изменяя эту координату.

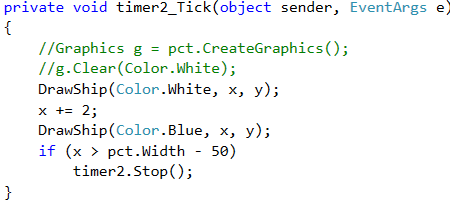
При использовании графически методов каждый раз рисуется новое изображение, а значит, старое нужно очищать. Если в объекте области вывода изображений нет других рисунков, то можно полностью очищать область рисования. Иначе нужно перерисовывать старое изображение цветом фона.

*Задача. Создать следующую анимацию: кораблик движется в объекте области вывода изображений pictureBox слева направо. Кораблик останавливается, когда достигнет края области.*

Процедура рисование кораблика цветом *c*:



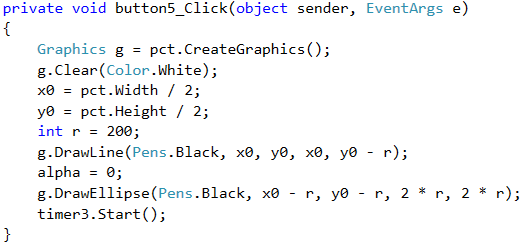
В событии таймера сначала рисуем кораблик белым цветом (или стираем все, что было нарисовано ранее), затем изменяем координату *x* и рисуем кораблик синим цветом, пока он не достигнет края *PictureBox*.

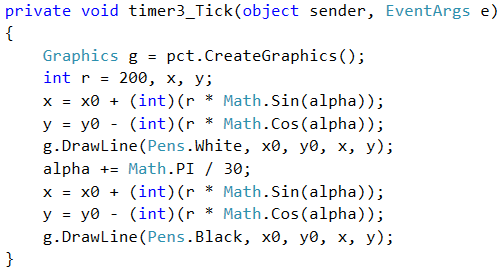


## Анимация изображения с использование эффектов вращения

*Задача. Создать следующую анимацию: один конец отрезка зафиксирован в центре объекта области вывода изображений pictureBox, а второй движется по окружности, проходящей через центр pictureBox, как секундная стрелка в часах.*







В этой задаче:

*alpha* – это угол, на который поворачивается секундная стрелка,

*x0, y0* – координаты конца отрезка, который находится в центре, вокруг которого происходит вращение,

*x, y* – координаты второго конца отрезка, который движется по окружности.