

Лабораторная работа №2

Тема работы: Основные компоненты библиотеки Windows.Forms

Задание:

Часть 1.

Создать приложение, отображающее движущийся вдоль края окна текст. Текст должен двигаться плавно, не выходя за границы отображаемой области окна (за рамку), и не отрываясь от края окна при изменении его размеров. Кроме того, текст должен плавно менять цвет в соответствии с вариантом. Двигаться текст должен по часовой стрелке (От левого верхнего угла к правому верхнему, от правого верхнего к правому нижнему, от правого нижнего к левому нижнему, от левого нижнего к левому верхнему).

Часть 2

Добавить в приложение еще одну форму, поместить на форму компоненты и организовать их взаимодействие в соответствии с индивидуальным заданием.

Выполнение:

Часть 1.

Разместить на форме компонент Label (Метка), и поместить его в точку с координатами (0; 0) (левый верхний угол). Разместить компонент Timer (Таймер) — этот компонент визуальный, но неграфический (не имеет отображаемого на форме элемента). Этот компонент и является одним из самых часто используемых визуальных, но неграфических компонентов. Он представляет системный таймер, специальный механизм, который выполняет некоторое действие через каждый определенный интервал времени. Выберите этот компонент и изучите его свойства. Их не так много — всего 2, которые нам сейчас интересны, это свойство «Enabled», задающее функционирует этот таймер или нет, и «Interval», определяющее тот самый интервал времени, с которым и будут происходить события. Задайте свойство «Enabled» равным «true», а свойство «Interval» равным 40 (можете взять любое значение от 10 до 200). Это число равно длительности интервала в миллисекундах (40 соответствует 25 событий/сек). Операционная система на самом деле не позволяет выдерживать интервалы таймеров с миллисекундной точностью, да и сам механизм работы таймеров не дает возможности такой точностью воспользоваться (подробнее о механизмах работы таймеров сообщений (не путать с системным таймером) вы узнаете в курсе операционных систем). В событиях у таймера есть всего одно — и именно оно вызывается когда заданный интервал истек. Создайте обработчик этого события любым удобным для вас способом. Далее необходимо задать поведение компонентов в соответствии с заданием. Для этого нам совершенно необходимо знать текущее направление движения. Несмотря на то, что здесь это возможно и без «привлечения лишних сущностей», однако в других случаях это невозможно, да и здесь связано с написанием достаточно большого по объему и сложности для новичка кода. Поэтому мы воспользуемся дополнительным элементом, а именно введем специальную переменную, которая называется переменной состояния. Запишем ее в глобальной области видимости класса — это обеспечит сохранение ее значения между событиями таймера.

Сразу же определим и так называемый перечислимый тип данных. Определение таких типов необязательно, с точки зрения аппаратного обеспечения все типы данных — это числа, но использование перечислений вместо числовых типов со списками констант (а уж тем более без оного) повышает удобочитаемость кода и предохраняет от некоторых видов ошибок.

enum Direction {Right, Down, Left, Up};

Direction direction = Direction.Right;

Изначально все глобальные переменные инициализируются нулевыми значениями, но более корректной является явная инициализация, поэтому присвоим переменной значение Direction.Right, обозначающее движение направо.

Для движения необходимо изменить координаты метки, содержащей текст. Поскольку у нас с каждым новым событием метка смещается относительно старого положения, то операция движения будет функцией от старого положения. Например, для движения направо, нам нужно увеличивать координату X метки. В коде:

label1.Left = label1.Left+10;

или, короче

label1.Left += 10;

Рекомендуется не использовать значения менее 5 и более 20, т.к. полученная скорость будет неудачной для демонстрации результата.

Однако, нам нужно выполнять это действие ТОЛЬКО если текущее направление движения — вправо. Т.е. при выполнении определенных условий. Для проверки условий используется специальный условный оператор. В данном случае можно записать необходимое условие в виде условного оператора:

if (direction == Direction.Right)

label1.Left += 10;

Но если вспомнить все необходимые условия (их 4) и представить общий размер полученных языковых конструкций, рациональнее окажется использование оператора множественного ветвления:

switch (direction)

{

case Direction.Right:

label1.Left += 10;

break;

case Direction.Down:

label1.Top += 10;

break;

case Direction.Left:

label1.Left -= 10;

break;

case Direction.Up:

label1.Top -= 10;

break;

}

Теперь необходимо осуществить переключение направлений при достижении определенных условий, а именно — достижении краев. При этом метка не должна визуально выходить за границы формы. Необходимое условие для правого и нижнего края мы уже составили в лабораторной работе №1, если записать его в условный оператор, будет:

if (label1.Left > ClientRectangle.Width-label1.Width)

direction = Direction.Down;

Но оно не исключает «перелет». Поэтому нужно добавить корректировку координаты, чтобы она точно приходилась на правую границу формы. Также нужно помнить, что в задании существует требование на корректное поведение при масштабировании. Вообще, для корректного и удобного для программиста масштабирования элементов управления в VCL есть специальный механизм «якорей» (Anchors). В одноименном свойстве типа множество задаются привязки к соответствующим сторонам опорного элемента (в нашем случае это будет форма), расстояние до которых от границ элемента будет выдерживаться при масштабировании. Если заданы привязки к обеим границам одного направления (горизонтального или вертикального) элемент будет расширяться или сжиматься, чтобы сохранить расстояния до границ опорного элемента. По-умолчанию используется привязка к левому и верхнему краю. Мы будем динамически менять привязку в соответствии с направлением движения:

if (label1.Left > ClientRectangle.Width-label1.Width)

{

direction = Direction.Down;

label1.Left = ClientRectangle.Width-label1.Width;

}

Аналогично 3 остальных условия, важно только помнить, что верхняя и левая границы не имеют специальных свойств — по определению формы, они равны 0.

Следующая часть задания содержит работу с цветом текста метки. Цвет текста метки задается через свойство ForeColor, в коде:

label1.ForeColor = Color.FromArgb(0xFFFFFF);

Основные цвета приведены в таблице.

|  |  |  |  |
| --- | --- | --- | --- |
| Черный  0x000000 | Темно-синий  0x000080 | Темно-зеленый  0x008000 | Темно-голубой  0x008080 |
| Бордовый  0x800000 | Фиолетовый  0x800080 | Оливковый  0x808000 | Темно-серый  0x808080 |
| Светло-серый  0xC0C0C0 | Синий  0x0000FF | Зеленый  0x00FF00 | Голубой  0x00FFFF |
| Красный  0xFF0000 | Маджента  0xFF00FF | Желтый  0xFFFF00 | Белый  0xFFFFFF |

Также цвет можно «конструировать» из значений отдельных компонентов — красного, зеленого и синего:

label1.ForeColor = Color.FromArgb(0xFF, 0xFF, 0xFF); // белый

label1.ForeColor = Color.FromArgb(255, 255, 255); // белый

label1.ForeColor = Color.FromArgb(255, 0, 0); // красный

Кроме того, требуется плавное изменение цвета, т.е. функция, а не простое присвоение определенного значения (константы). Для построения соответствующей функции воспользуемся школьной математикой. Нам нужно отобразить весь возможный диапазон изменения координат на отрезок цвета от 0 до 255. Этого можно достичь простым линейным преобразованием. Разделим текущую координату на длину возможного изменения (приведем к интервалу 0..1) и умножим на 255 — длину интервала представления цвета.

label1.ForeColor = Color.FromArgb(label1.Left\*255/(ClientRectangle.Width-label1.Width), 0, 0);

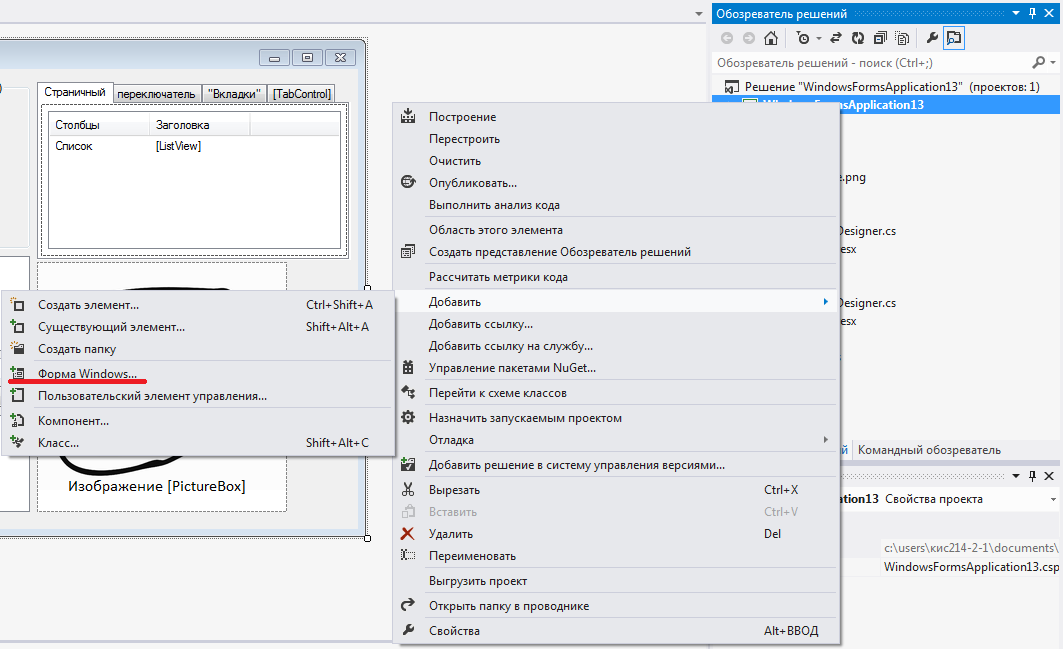
Обратите внимание, что умножение на 255 (0xFF) идет ПЕРЕД делением. Поскольку мы используем целочисленное, а не вещественное деление, то данный шаг необходим, иначе выражение будет принимать только 2 значения – 0 и 255 (посчитайте в уме, и убедитесь, что это так).

Аналогично для всех остальных изменений цвета. Поскольку изменение цвета осуществляется в зависимости от направления движения, это действие также стоит поместить в оператор множественного ветвления.

Часть 2.

Для выполнения этой части работы следует создать еще одну форму и сделать ее видимой.

Для этого можно воспользоваться главным меню (Проект -> Добавить форму Windows), можно контекстным меню обозревателя решений:



Поскольку вторя форма автоматически не создается, ее создание необходимо прописать вручную.

В обработчике события Shown первой формы допишем создание и вызов второй формы:

Form2 newForm = new Form2();

newForm.Show();

Для связи второй формы с первой (как это требуется для выполнения задания) нам необходима ссылка из второй формы на первую. Добавим такую ссылку в описание класса второй формы Form2:

public Form1 form1;

и, соответственно, добавим инициализацию этого поля в процедуру создания формы:

newForm.form1 = this;

Для работы с первой формы из второй теперь достаточно обратиться к полю form1.

На Рисунке 1 вы можете ознакомиться с основными базовыми элементами управления современных графических интерфейсов. Здесь будут описаны их основные свойства и приведены примеры кода, не относящиеся к выданному заданию.

Все видимые компоненты.

Свойство Name — не является в полном понимании этого термина свойством. Обозначает идентификатор, по которому доступен внутри формы данный компонент (должно быть корректным идентификатором C#).

Свойство Left — X координата компонента, тип integer (число).

Свойство Top — Y координата компонента, тип integer (число).

Свойство Width — ширина компонента, тип integer (число).

Свойство Height — высота компонента, тип integer (число).

Свойство Visible — видимость компонента. Невидимый компонент не отображается на форме в время выполнения программы (но, тем не менее, создается и существует). Тип Boolean (логический).

Свойство Enabled — доступность компонента. Недоступный компонент отображается как правило плоским и серым цветом. Большинство событий не работают, если компонент недоступен. Тип Boolean (логический).

Компонент Button.

Основное свойство — Text. Отвечает за надпись на кнопке, тип string (строка).

Основное событие — Click. Вызывается, когда пользователь нажал на кнопку.

Компонент TextBox

Основное свойство — Text. Текст, введенный пользователем, тип string (строка).

Свойство Lines. Строки, введенные пользователем в многострочное поле ввода. Тип — массив строк string[];

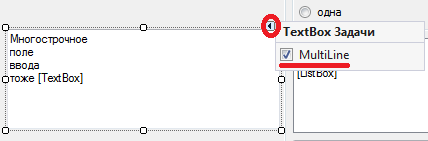
Свойство MaxLength. Максимальная длина вводимого текста, тип int (число).

Свойство Multiline. Задает режим работы — однострочный/многострочный.

Основное событие — TextChanged. Вызывается, когда текст в поле ввода изменяется пользователем.

Метод AppendText. Добавляет текст в конец. Используется для добавления строк.

Компонент переключается в другой режим изменением свойства MultiLine или с помощью стрелочки в конструкторе форм:



Компонент CheckBox

Основное свойство — Checked, тип Boolean (логический). Состояние кнопки — отмечена (true) или нет (false).

Свойство Text — текст подписи, тип string (строка).

Событие — Click. Вызывается, когда пользователь нажал на кнопку. Как правило, при этом изменяется состояние кнопки.

Событие — CheckChanged. Вызывается при изменении состояния флаговой кнопки.

Компонент ListBox

Основное свойство — Items, коллекция. Содержит элементы списка.

Свойство SelectedIndex — выделенный элемент, тип int (число). Первый элемент имеет индекс 0. Если ни один элемент не выделен, это свойство имеет значение -1.

Свойство SelectedItem — выделенный элемент, тип object (абстрактный объект). Если ни один элемент не выделен, это свойство имеет значение null.

Событие — Click. Вызывается, когда пользователь нажал на список. Как правило, при этом изменяется выделенный элемент (но не обязвтельно).

Компонент ComboBox

Основное свойство — Items, коллекция. Содержит элементы списка.

Свойство SelectedIndex — выделенный элемент, тип int (число). Первый элемент имеет индекс 0. Если ни один элемент не выделен, это свойство имеет значение -1.

Свойство SelectedItem — выделенный элемент, тип object (абстрактный объект). Если ни один элемент не выделен, это свойство имеет значение null.

Свойство DropDownStyle — внешний вид выпадающего списка. Тип перечисление, специально объявленный. ComboBoxStyle.DropDown — список с полем ввода, ComboBoxStyle.DropDownList — список выбора.

Основное событие — TextChanged. Вызывается, когда текст в поле ввода изменяется пользователем.

Событие — SelectedIndexChanged. Вызывается при выборе варианта из выпадающего списка.

Варианты заданий:

Часть 1

|  |  |
| --- | --- |
| Вариант 1 | Вариант 2 |
| Вариант 3 | Вариант 4 |

Часть 2

Вариант 1

Кнопка, с подписью «Выключить», при нажатии, отключающая движение метки из Части 1. После первого нажатия меняет подпись на «Включить». При повторном нажатии включает движение метки из части 1 и меняет подпись обратно на «Выключить».

Группировочная панель, на ней:

Поле ввода, без введенного текста. Справа от нее кнопка с подписью «Добавить». Еще правее – кнопка «Удалить». Ниже список выбора. Ниже списка – метка. При нажатии кнопки «Добавить» очистить поле ввода и добавить введенный текст в конец вариантов в списке выбора. При выборе элементов из этого списка, в метке отобразить индекс выбранного элемента, начиная с 1, с подписью, например: «Выбранный элемент: 1, Текст, введенный пользователем», или «Выбранный элемент: Нет». При отсутствии выбранного элемента, отобразить это состояние в метке. При нажатии кнопки «Удалить», удалить выбранный элемент из списка выбора, если таковой элемент есть. Иначе — выдать соответствующее сообщение.

Вариант 2

Флаговая кнопка, с подписью «Движение включено» и положением по-умолчанию «отмечено». Движение метки из части 1 должно соответствовать состоянию флаговой кнопки.

Группировочная панель, на ней:

Многострочное поле ввода, без текста. Кнопка с подписью «Добавить строку». Кнопка с подписью «Удалить 1-ю строку». Метка, отображающая количество строк текста в поле ввода, с поясняющей надписью, например: «Количество строк: 4». При нажатии на кнопку из поля ввода удаляется первая строка. Если строк нет — вывести соответствующее сообщение. При нажатии на кнопку «Добавить строку» добавить в конец строку с произвольным текстом (не пустую).

Вариант 3

Группа переключателей, с подписью «Движение» и 2-мя вариантами: «Включено» и «Выключено». По-умолчанию отмечен 1-й вариант. При переключении состояние движения метки из части 1 должно соответствовать выбранному варианту.

Группировочная панель, на ней:

Выпадающий список с полем ввода, без текста и элементов. Правее кнопка с подписью «Добавить». Еще правее — кнопка «Удалить». Ниже метка. При нажатии кнопки «Добавить», добавить введенный пользователем текст в варианты выбора выпадающего списка. При нажатии кнопки «Удалить» — удалить выбранный вариант из списка. Если введенный текст не является выбранным вариантом — выдать соответствующее сообщение. В метке отображать номер выбранного варианта начиная с 1, с его содержимым и поясняющей подписью.

Вариант 4

Выпадающий список выбора, с 2-мя элементами: «Движение включено» и «Движение выключено». По-умолчанию выбран первый вариант. При переключении состояние движения метки из части 1 должно соответствовать выбранному варианту.

Группировочная панель, на ней:

Поле ввода, без введенного текста. Справа от нее кнопка с подписью «Добавить». Еще правее – кнопка «Удалить». Ниже список выбора. Ниже списка – метка. При нажатии кнопки «Добавить» очистить поле ввода и добавить введенный текст в конец вариантов в списке выбора. При выборе элементов из этого списка, в метке отобразить индекс выбранного элемента, начиная с 1, с подписью, например: «Выбранный элемент: 1, Текст, введенный пользователем», или «Выбранный элемент: Нет». При отсутствии выбранного элемента, отобразить это состояние в метке. При нажатии кнопки «Удалить», удалить выбранный элемент из списка выбора, если таковой элемент есть. Иначе — выдать соответствующее сообщение.

Вариант 5

Кнопка, с подписью «Выключить», при нажатии, отключающая движение метки из Части 1. После первого нажатия меняет подпись на «Включить». При повторном нажатии включает движение метки из части 1 и меняет подпись обратно на «Выключить».

Группировочная панель, на ней:

Многострочное поле ввода, без текста. Кнопка с подписью «Добавить строку». Кнопка с подписью «Удалить 1-ю строку». Метка, отображающая количество строк текста в поле ввода, с поясняющей надписью, например: «Количество строк: 4». При нажатии на кнопку из поля ввода удаляется первая строка. Если строк нет — вывести соответствующее сообщение. При нажатии на кнопку «Добавить строку» добавить в конец строку с произвольным текстом (не пустую).

Вариант 6

Флаговая кнопка, с подписью «Движение включено» и положением по-умолчанию «отмечено». Движение метки из части 1 должно соответствовать состоянию флаговой кнопки.

Группировочная панель, на ней:

Выпадающий список с полем ввода, без текста и элементов. Правее кнопка с подписью «Добавить». Еще правее — кнопка «Удалить». Ниже метка. При нажатии кнопки «Добавить», добавить введенный пользователем текст в варианты выбора выпадающего списка. При нажатии кнопки «Удалить» — удалить выбранный вариант из списка. Если введенный текст не является выбранным вариантом — выдать соответствующее сообщение. В метке отображать номер выбранного варианта начиная с 1, с его содержимым и поясняющей подписью.

Вариант 7

Группа переключателей, с подписью «Движение» и 2-мя вариантами: «Включено» и «Выключено». По-умолчанию отмечен 1-й вариант. При переключении состояние движения метки из части 1 должно соответствовать выбранному варианту.

Группировочная панель, на ней:

Поле ввода, без введенного текста. Справа от нее кнопка с подписью «Добавить». Еще правее – кнопка «Удалить». Ниже список выбора. Ниже списка – метка. При нажатии кнопки «Добавить» очистить поле ввода и добавить введенный текст в конец вариантов в списке выбора. При выборе элементов из этого списка, в метке отобразить индекс выбранного элемента, начиная с 1, с подписью, например: «Выбранный элемент: 1, Текст, введенный пользователем», или «Выбранный элемент: Нет». При отсутствии выбранного элемента, отобразить это состояние в метке. При нажатии кнопки «Удалить», удалить выбранный элемент из списка выбора, если таковой элемент есть. Иначе — выдать соответствующее сообщение.

Вариант 8

Выпадающий список выбора, с 2-мя элементами: «Движение включено» и «Движение выключено». По-умолчанию выбран первый вариант. При переключении состояние движения метки из части 1 должно соответствовать выбранному варианту.

Группировочная панель, на ней:

Многострочное поле ввода, без текста. Кнопка с подписью «Добавить строку». Кнопка с подписью «Удалить 1-ю строку». Метка, отображающая количество строк текста в поле ввода, с поясняющей надписью, например: «Количество строк: 4». При нажатии на кнопку из поля ввода удаляется первая строка. Если строк нет — вывести соответствующее сообщение. При нажатии на кнопку «Добавить строку» добавить в конец строку с произвольным текстом (не пустую).

Вариант 9

Кнопка, с подписью «Выключить», при нажатии, отключающая движение метки из Части 1. После первого нажатия меняет подпись на «Включить». При повторном нажатии включает движение метки из части 1 и меняет подпись обратно на «Выключить».

Выпадающий список с полем ввода, без текста и элементов. Правее кнопка с подписью «Добавить». Еще правее — кнопка «Удалить». Ниже метка. При нажатии кнопки «Добавить», добавить введенный пользователем текст в варианты выбора выпадающего списка. При нажатии кнопки «Удалить» — удалить выбранный вариант из списка. Если введенный текст не является выбранным вариантом — выдать соответствующее сообщение. В метке отображать номер выбранного варианта начиная с 1, с его содержимым и поясняющей подписью.

Вариант 10

Флаговая кнопка, с подписью «Движение включено» и положением по-умолчанию «отмечено». Движение метки из части 1 должно соответствовать состоянию флаговой кнопки.

Группировочная панель, на ней:

Поле ввода, без введенного текста. Справа от нее кнопка с подписью «Добавить». Еще правее – кнопка «Удалить». Ниже список выбора. Ниже списка – метка. При нажатии кнопки «Добавить» очистить поле ввода и добавить введенный текст в конец вариантов в списке выбора. При выборе элементов из этого списка, в метке отобразить индекс выбранного элемента, начиная с 1, с подписью, например: «Выбранный элемент: 1, Текст, введенный пользователем», или «Выбранный элемент: Нет». При отсутствии выбранного элемента, отобразить это состояние в метке. При нажатии кнопки «Удалить», удалить выбранный элемент из списка выбора, если таковой элемент есть. Иначе — выдать соответствующее сообщение.

Вариант 11

Группа переключателей, с подписью «Движение» и 2-мя вариантами: «Включено» и «Выключено». По-умолчанию отмечен 1-й вариант. При переключении состояние движения метки из части 1 должно соответствовать выбранному варианту.

Группировочная панель, на ней:

Многострочное поле ввода, без текста. Кнопка с подписью «Добавить строку». Кнопка с подписью «Удалить 1-ю строку». Метка, отображающая количество строк текста в поле ввода, с поясняющей надписью, например: «Количество строк: 4». При нажатии на кнопку из поля ввода удаляется первая строка. Если строк нет — вывести соответствующее сообщение. При нажатии на кнопку «Добавить строку» добавить в конец строку с произвольным текстом (не пустую).

Вариант 12

Выпадающий список выбора, с 2-мя элементами: «Движение включено» и «Движение выключено». По-умолчанию выбран первый вариант. При переключении состояние движения метки из части 1 должно соответствовать выбранному варианту.

Выпадающий список с полем ввода, без текста и элементов. Правее кнопка с подписью «Добавить». Еще правее — кнопка «Удалить». Ниже метка. При нажатии кнопки «Добавить», добавить введенный пользователем текст в варианты выбора выпадающего списка. При нажатии кнопки «Удалить» — удалить выбранный вариант из списка. Если введенный текст не является выбранным вариантом — выдать соответствующее сообщение. В метке отображать номер выбранного варианта начиная с 1, с его содержимым и поясняющей подписью.