|  |
| --- |
| Министерство образования и науки Российской Федерации |
| федеральное государственное бюджетное образовательное учреждение высшего образования |
| "Российский экономический университет имени Г.В. Плеханова" |
| **МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ** |

УТВЕРЖДАЮ

Заместитель директора по учебной работе

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Д.А. Клопов

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 года

**Лабораторная работа № 8**

ОП 14 Объектно-ориентированное программирование

Тема: Отображения статистики графическим способом. Диаграммы. Графики.

**Специальность: 09.02.05 «Прикладная информатика (по отраслям)»**

Квалификация: техник-программист

Разработчик: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Ю.А. Русакова

Рассмотрено ЦМК

«Профессиональных модулей 09.02.05»

«31» августа 2018 года. Протокол № 1

Председатель ЦМК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Л.А.Соколова

**Цель работы:** Реализовать программу отображающую построение графика и диаграммы используя пространство имён System.Drawing, классы Graphics, Brushes, Pens и методы DrawString, DrawRectangle, FillRectangle, а также работу массивами и чтение данных из файла.

**Количество часов на выполнение работы:** Продолжительность выполнения данной практической работы составляет 2 академических часа.

**Программное обеспечение:** Windows 7 и выше, Microsoft Office Word 2010 и выше, Visual Studio 2015 и выше.

**Теоретические сведения к выполнению лабораторной работы:**

2D-графика делится, как вы знаете, на растровую и векторную. Растровое изображение — это набор цветных пикселей, заданных в прямоугольной области, хранящихся в файлах \*.bmp, \*.jpg, \*.png и т.п. Самый простой растровый редактор — программа Paint. Векторная графика намного экономнее (по объемам памяти) растровой. Так для рисования прямоугольника достаточно задать координаты двух точек (левого верхнего и правого нижнего углов) и цвет и толщину линии. В этом разделе в основном рассмотрим методы векторной графики.

Пространство имен System.Drawing (Рисование) обеспечивает доступ к функциональным возможностям графического интерфейса GDI+ , используя около 50 классов. Пространства имён расширяющие функциональные возможности библиотеки System.Drawing: System.Drawing.Drawing2D, System.Drawing.Imaging, System.Drawing.Printing, System.Drawing.Text.

Класс **Graphics** предоставляет методы рисования на устройстве отображения (другие термины — графический контекст, «холст»). Определимся сразу, на чем мы хотим рисовать. Далее в примерах он обозначается как объект g.

Такие классы, как Rectangle (прямоугольник) и Point (точка) инкапсулируют элементы GDI+. Отметим, что Point вообще то является структурой (struct) с полями x,y. Это уточнение несущественно, так как в C#структуры похожи на классы, a инициализация объекта-структуры point может выглядеть так же, как инициализация объекта-класса:  
Point point= new Point();

Класс **Pen** (перо) используется для рисования линий и кривых, а классы, производные от класса Brush (кисть) используются для закраски замкнутых контуров.

Класс **GraphicsPath** позволяет задавать последовательность соединенных линий и кривых, класс Region описывает внутреннюю часть графической формы, состоящей из многоугольников и контуров.

Класс **Image** – абстрактный базовый класс предоставляет функциональные возможности для производных классов Bitmap и Metafile. Bitmap используется для работы с пиксельными изображениями. Metafile определяет графический метафайл, содержащий записи, описывающие последовательность графических операций, которые могут быть записаны (созданы) и воспроизведены (отображается). Этот класс не наследуется.

**Класс Graphics**

Он инкапсулирует поверхность рисования GDI+. Этот класс не наследуется. Методов в этом классе огромное количество.

В третьем столбце таблицы указывается число перегрузок метода, различающихся набором параметров (используйте интеллектуальную подсказку IntelliSense для выбора нужного Вам варианта метода).

|  |  |  |
| --- | --- | --- |
| **Имя метода** | **Описание** | **Число перегрузок** |
| Clear(Color) | Очищает всю поверхность рисования и выполняет заливку поверхности указанным цветом фона. | 1 |
| CopyFromScreen(Point, Point, Size) | Выполняет передачу данных о цвете, соответствующих прямоугольной области пикселей, блоками битов с экрана на поверхность рисования объекта Graphics. | 4 |
| Dispose() | Освобождает все ресурсы, используемые данным объектом Graphics. | 1 |
| DrawArc(Pen, Rectangle, Single, Single) | Рисует дугу, которая является частью эллипса, заданного структурой Rectangle. | 4 |
| DrawBezier(Pen, Point, Point, Point, Point) | Рисует кривую Безье, определяемую четырьмя структурами Point. | 3 |
| DrawBeziers(Pen, Point[]) | Рисует несколько (N) кривых Безье, определяемых массивом из (3N+1) структур Point. | 2 |
| DrawCloseCurve(Pen, Point[ ]) | Рисует замкнутый фундаментальный сплайн | 4 |
| DrawEllipse(Pen, Rectangle) | Рисует эллипс | 4 |
| DrawIcon(Icon, Rectangle) | Рисует значок | 2 |
| DrawImage(Image image, int x, int y) | Рисует заданное изображение image, используя его фактический размер в месте с координатами (x,y) | 30 |
| DrawLine(Pen, Point, Point) | Проводит линию, соединяющую две структуры Point. | 4 |
| DrawLines(Pen, Point[ ]) | Рисует набор сегментов линий, которые соединяют массив структур Point. | 2 |
| DrawPath(Pen, gp) | Рисует пером Pen объект GraphicsPath gp. | 1 |
| DrawPie(Pen, Rectangle, Single, Single) | Рисует сектор, который определяется эллипсом, заданным структурой Rectangle и двумя радиалtьными линиями. | 4 |
| DrawPolygon(Pen, Point[]) | Рисует многоугольник, определяемый массивом структур Point. | 2 |
| DrawRectangle(Pen, Rectangle) | Рисует прямоугольник, определяемый структурой Rectangle. | 3 |
| DrawRectangles(Pen, Rectangle[]) | Рисует набор прямоугольников, определяемых структурами Rectangle. | 2 |
| DrawString(String, Font, Brush, PointF) | Создает указываемую текстовую строку в заданном месте с помощью определяемых объектов Brush и Font. | 6 |
| Equals(Object) | Определяет, равен ли заданный объект текущему объекту. (Унаследовано от Object.) | 1 |
| ExcludeClip(Rectangle) | Обновляет вырезанную область данного объекта Graphics, чтобы исключить из нее часть, определяемую структурой Rectangle. | 1 |
| ExcludeClip(Region) | Обновляет вырезанную область данного объекта Graphics, чтобы исключить из нее часть, определяемую структурой Region. | 1 |
| FillClosedCurve(Brush, Point[]) | Заполняет внутреннюю часть замкнутой фундаментальной кривой, определяемой массивом структур Point. | 6 |
| FillEllipse(Brush, Rectangle) | Заполняет внутреннюю часть эллипса, определяемого ограничивающим прямоугольником, который задан структурой Rectangle. | 4 |
| FillPath(Brush, GraphicsPath) | Заполняет внутреннюю часть объекта GraphicsPath. | 1 |
| FillPie(Brush, Rectangle, Single, Single) | Заполняет внутреннюю часть сектора, определяемого эллипсом, который задан структурой RectangleF, и двумя радиальными линиями. | 3 |
| FillPolygon(Brush, Point[]) | Заполняет внутреннюю часть многоугольника, определяемого массивом точек, заданных структурами Point. | 4 |
| FillRectangle(Brush, Rectangle) | Заполняет внутреннюю часть прямоугольника, определяемого структурой Rectangle. | 4 |
| FillRegion(Brush, Region) | Заполняет внутреннюю часть объекта Region. | 1 |
| Flush() | Вызывает принудительное выполнение всех отложенных графических операций и немедленно возвращается, не дожидаясь их окончания. | 2 |
| IntersectClip(Region) | Обновляет вырезанную область данного объекта, включая в нее пересечение текущей вырезанной области и указанной структуры | 3 |
| ResetClip() | Сбрасывает выделенную область g, делая ее бесконечной | 1 |

**Класс Pen**

Класс Pen определяет объект, используемый для рисования прямых линий и кривых. Этот класс не наследуется. Конструкторы класса (тоже пример перегрузки методов):

1. Pen(Color) инициализирует новый экземпляр класса Pen с указанным цветом.
2. Pen(Color, Single) инициализирует новый экземпляр класса Pen с указанными свойствами Color и Width. (Width — устанавливает ширину пера Pen).

Например:

Pen redPen = new Pen(Color.Red); // толщина пера по умолчанию 1 пиксельили используя промежуточную переменную green

Color green = Color.Green;

Pen greenPen = new Pen(green, 4.5f);

Можно вводить новый объект без указания явного имени пера (пример создания динамического объекта — пера):

g.DrawRectangle(new Pen(Color.Red, 3), r);

Например, для рисования прямоугольника r красным пером толщиной 3 пикселя, используя графический контекст («холст») g.

**Класс Brush**

Класс Brush определяет объекты, которые используются для заливки внутри графических фигур, таких как прямоугольники, эллипсы, круги, многоугольники и дорожки. Это абстрактный базовый класс, который не может быть реализован. Для создания объекта «кисть» используются классы, производные от Brush, такие как SolidBrush, TextureBrush и LinearGradientBrush, например:

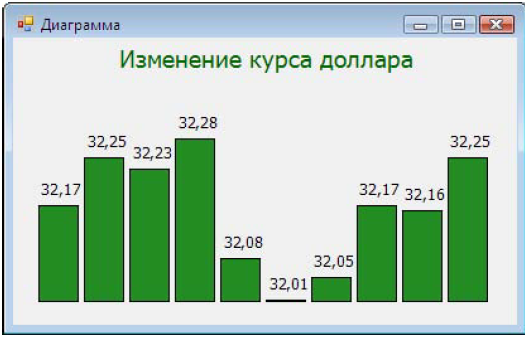
SolidBrush br = new SolidBrush(Color.Aqua);

g.FillRectangle(br, r); // закраска прямоугольника r на «холсте» g

Или без явного объявления объекта «кисть» br:  
g.FillRectangle(new SolidBrush(Color.Red), r);

**Этапы выполнения:**

1. Необходимо создать три формы. На них после запуска будут отображены следующие графики и диаграммы:







1. Реализация столбчатой диаграммы:

string header;

double[] d;

int N;

Graphics g;

public Form1()

{

InitializeComponent();

try

{

StreamReader sr = new StreamReader(Application.StartupPath + "\\g.txt",

Encoding.Default);

header = sr.ReadLine();

N = File.ReadAllLines(Application.StartupPath + "\\g.txt").Length - 1;

d = new double[N];

int i = 0;

string st = sr.ReadLine();

while ((st != null) && (i < N))

{

d[i++] = Convert.ToDouble(st);

st = sr.ReadLine();

}

sr.Close();

Paint += new PaintEventHandler(Diagram);

}

catch (FileNotFoundException ex)

{

MessageBox.Show(ex.Message + "\n" + "(" + ex.GetType().ToString() + ")",

"График", MessageBoxButtons.OK,

MessageBoxIcon.Error);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "График", MessageBoxButtons.OK,

MessageBoxIcon.Error);

}

}

private void Diagram(object sender, PaintEventArgs e)

{

g = e.Graphics;

//Рисуем заголовок

Font hFont = new Font("Tahoma", 12, FontStyle.Bold);

int x = (ClientSize.Width - (int)g.MeasureString(header, hFont).Width) / 2;

g.DrawString(header, hFont, Brushes.Blue, x, 5);

//рисуем диаграмму

Font dFont = new Font("Tahoma", 9);

double max = d[0], min = d[0];

for (int i = 1; i < N; i++)

{

if (d[i] > max) max = d[i];

if (d[i] < min) min = d[i];

}

int x1, y1, w, h;

w =(ClientSize.Width - 40-5 \*(N - 1))/N;

x1 = 20;

for (int i = 0; i < N; i++)

{

y1 = ClientSize.Height - 20 -

(int)((ClientSize.Height - 100) \* (d[i] - min) / (max - min));

g.DrawString(d[i].ToString(), dFont, Brushes.Black, x1, y1 - 20);

h = ClientSize.Height - y1 - 20 + 1;

g.FillRectangle(Brushes.HotPink, x1, y1, w, h);

g.DrawRectangle(Pens.Khaki,x1,y1,w,h);

x1 += w + 5;

}

}

private void Form1\_Resize(object sender, EventArgs e)

{

Invalidate();

}

1. Реализация круговой диаграммы:

string header;

int N = 0;

double[] dat;

double[] p;

string[] title;

public Form1()

{

InitializeComponent();

try

{

StreamReader sr = new StreamReader(Application.StartupPath + "\\date.txt",

Encoding.Default);

header = sr.ReadLine();

N = (File.ReadAllLines(Application.StartupPath + "\\date.txt").Length) / 2;

dat = new double[N];

p = new double[N];

title = new string[N];

int i = 0;

string st = sr.ReadLine();

while ((st != null) && (i < N))

{

title[i] = st;

st = sr.ReadLine();

dat[i++] = Convert.ToDouble(st);

st = sr.ReadLine();

}

sr.Close();

Paint += new PaintEventHandler(CDiagram);

double sum = 0;

for (int j = 0; j < N; j++) sum += dat[j];

for (int j = 0; j < N; j++) p[j] = (double)(dat[j] / sum);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Диаграмма",

MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void CDiagram(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

//Рисуем заголовок

Font hFont = new Font("Tahoma", 12, FontStyle.Bold);

int x = (ClientSize.Width - (int)g.MeasureString(header, hFont).Width) / 2;

g.DrawString(header,hFont,Brushes.Red, x, 10);

//Рисуем легенду

Font lFont = new Font("Tahoma",9);

int d = ClientSize.Height - 70,// диаметр диаграммы

x0 = 30,// координаты верхнего левого угла

y0 = (ClientSize.Height - d) / 2 + 10,

lx = 60 + d,//область легенды

ly = y0 + (d - N \* 20 + 10) / 2,

swe,//длина дуги сектора

sta = -90;// начальная точка дуги сектора

Brush fBrush = Brushes.White;

for (int i = 0; i < N; i++)

{

swe = (int)(360 \* p[i]);

switch (i)

{

case 0: fBrush = Brushes.YellowGreen; break;

case 1: fBrush = Brushes.Gold; break;

case 2: fBrush = Brushes.Pink; break;

case 3: fBrush = Brushes.Violet; break;

case 4: fBrush = Brushes.OrangeRed; break;

case 5: fBrush = Brushes.RoyalBlue; break;

case 6: fBrush = Brushes.SteelBlue; break;

case 7: fBrush = Brushes.Chocolate; break;

case 8: fBrush = Brushes.LightCyan; break;

case 9: fBrush = Brushes.Maroon; break;

}

if (i == N - 1) swe = 270 - sta;

g.FillPie(fBrush,x0,y0,d,d,sta,swe);

g.DrawPie(Pens.Black, x0, y0, d, d, sta, swe);

g.FillRectangle(fBrush, lx, ly + i \* 20, 20, 10);

g.DrawRectangle(Pens.Black, lx, ly + i \* 20, 20, 10);

g.DrawString(title[i]+" - "+p[i].ToString("P"),lFont,Brushes.Black,

lx+24,ly+i\*20-3);

sta += swe;

}

}

private void Form1\_Resize(object sender, EventArgs e)

{

Invalidate();

}

1. Реализация графика:

string header;

double[] d;

int N;

Graphics g;

public Form1()

{

InitializeComponent();

try

{

StreamReader sr = new StreamReader(Application.StartupPath + "\\g.txt",

Encoding.Default);

header = sr.ReadLine();

N = File.ReadAllLines(Application.StartupPath + "\\g.txt").Length-1;

d = new double[N];

int i = 0;

string st = sr.ReadLine();

while ((st != null) && (i < N))

{

d[i++] = Convert.ToDouble(st);

st = sr.ReadLine();

}

sr.Close();

Paint += new PaintEventHandler(Graf);

}

catch (FileNotFoundException ex)

{

MessageBox.Show(ex.Message + "\n" + "(" + ex.GetType().ToString() + ")",

"График", MessageBoxButtons.OK,

MessageBoxIcon.Error);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "График", MessageBoxButtons.OK,

MessageBoxIcon.Error);

}

}

private void Graf(object sender, PaintEventArgs e)

{

g = e.Graphics;

//Рисуем заголовок

Font hFont = new Font("Tahoma", 12, FontStyle.Bold);

int x = (ClientSize.Width - (int)g.MeasureString(header, hFont).Width) / 2;

g.DrawString(header, hFont, Brushes.Blue, x, 5);

//рисуем диаграмму

Font dFont = new Font("Tahoma", 9);

int sw = (int)((ClientSize.Width - 40) / (N-1));

double max = d[0], min = d[0];

for (int i = 1; i < N; i++)

{

if (d[i] > max) max = d[i];

if (d[i] < min) min = d[i];

}

int x1, y1, x2, y2;

x1 = 20;

y1 = ClientSize.Height - 20 -

(int)((ClientSize.Height - 100) \* (d[0] - min) / (max - min));

g.DrawRectangle(Pens.Black, x1 - 2, y1 - 2, 4, 4);

g.DrawString(d[0].ToString(), dFont, Brushes.Black, x1 - 10, y1 - 20);

for (int i = 1; i < N; i++)

{

x2 = 8 + i \* sw;

y2 = ClientSize.Height - 20 -

(int)((ClientSize.Height - 100) \* (d[i] - min) / (max - min));

g.DrawRectangle(Pens.Black, x2 - 2, y2 - 2, 4, 4);

g.DrawLine(Pens.Black, x1, y1, x2, y2);

g.DrawString(d[i].ToString(), dFont, Brushes.Black, x2 - 10, y2 - 20);

x1 = x2;

y1 = y2;

}

}

private void Form1\_Resize(object sender, EventArgs e)

{

Invalidate();

}

1. Задание:
   1. +В круговой диаграмме цвета областей задавать динамически в соответствие с числом секторов;
   2. + Сделать возможность пользователю выбирать внешний вид маркера на графике;
   3. +Маркер максимальной и минимальной точек выделять цветом;
   4. +Сделать возможность выбора документа, по которому будут строится диаграммы, учитывая проверку на возможность вывода информации и формат файла;
   5. Используя компоненты Panel и RadioButton организовать вывод диаграмм и графика на одной форме, при этом учитывать отличия документов, требуемых для вывода информации;
   6. + Отображать систему координат на графике;
   7. + Считывать значения курса указанной валюты из интернета и по нему строить график (***задание на дополнительную оценку***).

**Вывод:** При выполнении лабораторной работы, в которой …описываете что делали… были использованы следующие компоненты: …, из них дополнительно изучения требовали: …; для реализации функционала программы необходимо было применить классы: …; дополнительно в лабораторной работе было разобрано (применено, изучено) …. Основные проблемы при выполнении возникли с ….

**Контрольные вопросы:**

1. Каким классом по отношению к Pen является класс Brush?
2. Дайте определение классу Brush.
3. Дайте определение классу Image.
4. Какой класс используется для работы с пикселями?
5. Дайте определение классу Metafile.