|  |
| --- |
| **Описание** |

ChartProvider – это класс, представляющий собой набор типовых функций для упрощения работы с массивами точек (замерами) и отображением их на графиках (объектах chart .NET).

**Основные возможности**

* Отображение массивов точек (замеров) на графике;
* Вывод значений массивов точек (замеров) в виде текста;
* Добавление на график горизонтальных линий, перемещаемых мышью;
* Удаление одного или всех замеров;
* Установка доступности (видимости) замеров;
* Сохранение массивов точек (замеров) в файл как текст, изображение или Excel-файл;
* Загрузка массивов точек (замеров) из Excel-файла;
* Настройка значений осей и внешнего вида графика;
* Увеличение масштаба графика мышью.

**Состав**

ChartProvider состоит из следующих компонентов:

* Класс типовых функций ChartProvider, файл

**ChartProvider.cs**;

* Окно настроек графиков, файлы:

**ChartSetsWindow.cs**,

**ChartSetsWindow.Designer.cs,**

**ChartSetsWindow.resx;**

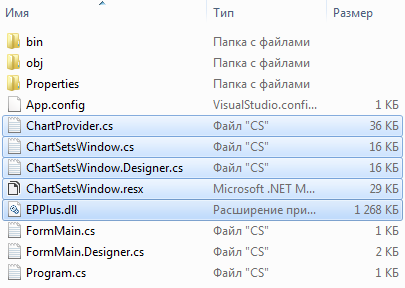
* Для работы с xml-файлами требуется открытая библиотека EPPlus (автор Jan Källman, <https://github.com/JanKallman/EPPlus>), файл

**EPPlus.dll**.

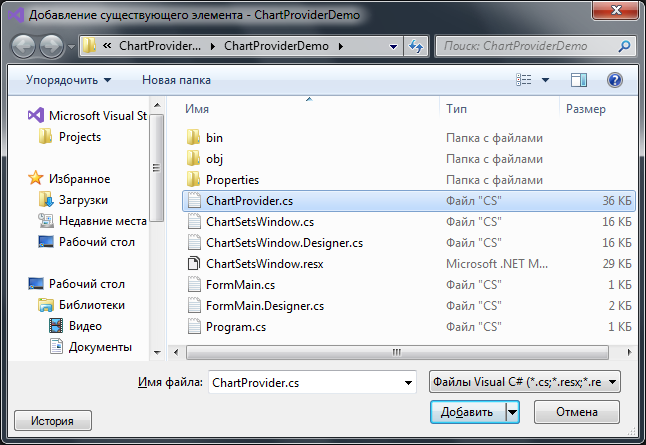
|  |
| --- |
| **Применение** |

**Добавление в проект**

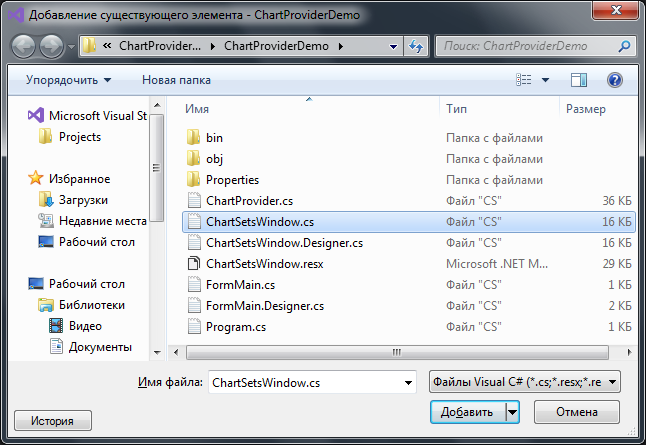
Во-первых, необходимо скопировать все 5 файлов из папки «ChartProvider» в папку Вашего проекта.



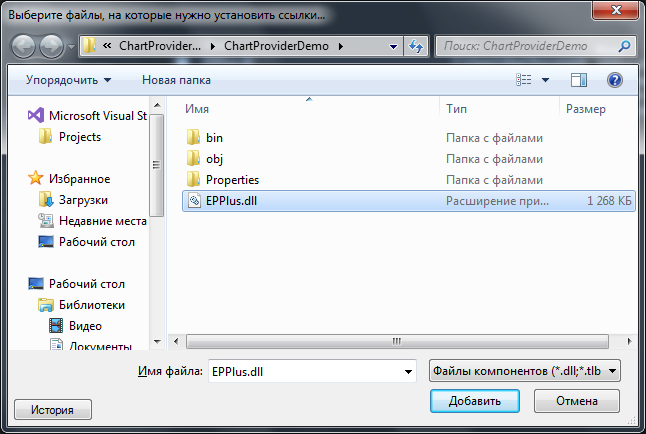
Далее, открыв проект в Visual Studio, необходимо кликнуть правой кнопкой мыши на название проекта, далее – «Добавить», «Существующий элемент…». В открывшемся окне выбрать файл **ChartProvider.cs** и нажать кнопку «Добавить».



Аналогично для файла **ChartSetsWindow.cs**. Обратите внимание, данные файлы необходимо добавлять именно по одному, а не оба сразу.



После этого необходимо кликнуть правой кнопкой мыши на название проекта, далее – «Добавить», «Ссылка…». Откроется Менеджер ссылок, в нем необходимо нажать кнопку «Обзор». В открывшемся окне выбрать файл **EPPlus.dll** и нажать кнопку «Добавить».



**Создание объекта**

В начале файла главной формы необходимо прописать директиву

|  |
| --- |
| using ChartProviders; |

В классе главной формы необходимо объявить объект типа ChartProvider с именем, к примеру, ChartMain:

|  |
| --- |
| public partial class FormMain : Form  {  ChartProvider ChartMain; // Объект управления главным графиком |

На форме необходимо создать график (объект типа «chart») с именем, к примеру, chart1. В конструкторе класса главной формы необходимо инициализировать объект ChartMain, указав имя данного графика:

|  |
| --- |
| // Конструктор главной формы  public FormMain()  {  ChartMain = new ChartProvider(chart1); // Инициализация объекта  } |

По умолчанию объекты типа ChartProvider работают с 10-ю замерами, если же планируется работа с иным числом, то после имени графика указывается количество необходимых замеров. Например, для 3-х замеров:

|  |
| --- |
| ChartMain = new ChartProvider(chart1, 3); // Инициализация объекта |

Таким образом, теперь объект ChartMain связан с графиком chart1 и управляет им.

Должен получиться следующий код:

|  |
| --- |
| using ChartProviders;  namespace ChartProviderDemo  {  public partial class FormMain : Form  {  ChartProvider ChartMain; // Объект управления главным графиком  // Конструктор главной формы  public FormMain()  {  InitializeComponent();  ChartMain = new ChartProvider(chart1); // Инициализация объекта  }  }  } |

**Добавление вывода значений замеров в виде текста**

На форме необходимо создать текстовое поле (объект типа «textBox») с именем, к примеру, textBox1. В конструкторе класса главной формы необходимо вызвать функцию AddTextDisplaying объекта ChartMain и передать в нее

1-м аргументом – имя связанного текстового поля,

2-м аргументом – имя измеряемой величины по оси X,

3-м аргументом – имя измеряемой величины по оси Y:

|  |
| --- |
| ChartMain.AddTextDisplaying(textBox1, "Время", "Ток"); |

2-й и 3-й аргументы необязательны.

С одним объектом типа ChartProvider можно связать только одно текстовое поле.

**Добавление горизонтальных линий**

Предположим, необходимо отображать на графике 2 горизонтальные линии. Для этого на форме необходимо создать 2 поля цифрового ввода (объекты типа «numericUpDown») с именами, к примеру, numericMain1 и numericMain2. В конструкторе класса главной формы 2 раза необходимо вызвать функцию AddHorizontalLine объекта ChartMain и передать в нее

1-м аргументом – имя связанного поля цифрового ввода,

2-м аргументом – цвет линии (выбирается из класса Color или SystemColors):

|  |
| --- |
| ChartMain.AddHorizontalLine(numericMain1, Color.Red);  ChartMain.AddHorizontalLine(numericMain2, Color.Blue); |

Должен получиться следующий код:

|  |
| --- |
| // Конструктор главной формы  public FormMain()  {  InitializeComponent();  ChartMain = new ChartProvider(chart1); // Инициализация объекта  ChartMain.AddTextDisplaying(textBox1, "Время", "Ток");  ChartMain.AddHorizontalLine(numUpDownMain1, Color.Red);  ChartMain.AddHorizontalLine(numUpDownMain2, Color.Blue);  } |

Объект типа ChartProvider позволяет создавать на графике несколько горизонтальных линий.

**Изменение доступности увеличения графика мышью**

По умолчанию, график, после привязки к нему объекта ChartProvider, получает возможность увеличения мышью. Если эта возможность не требуется, необходимо в конструкторе класса главной формы вызвать функцию SetScalingByMouseAvailability объекта ChartMain и передать в нее false:

|  |
| --- |
| ChartMain.SetScalingByMouseAvailability(false); |

**Отображение массивов точек (замеров)**

На форме необходимо создать кнопку вывода замеров (объект типа «button») с именем, к примеру, btnMainGetData. Двойным кликом по ней создастся функция события Click. Внутри нее необходимо вызвать функцию Display объекта ChartMain и передать в нее

1-м аргументом – массив X-координат точек,

2-м аргументом – массив Y-координат точек:

|  |
| --- |
| // Кнопка "Сгенерировать замер"  private void btnMainGetData\_Click(object sender, EventArgs e)  {  double[] x = new double[50]; // Массив X-координат точек  double[] y = new double[50]; // Массив Y-координат точек  /\* Некий код, получающий значения этих массивов \*/  ChartMain.Display(x, y);  } |

**Добавление окна настроек графика**

На форме необходимо создать кнопку открытия окна настроек графика (объект типа «button») с именем, к примеру, btnMainSets. Двойным кликом по ней создастся функция события Click. Внутри нее необходимо вызвать функции AddChartSetsWindow и ShowChartSetsWindow объекта ChartMain, передав во вторую имя данной кнопки:

|  |
| --- |
| // Кнопка "Настройка графика"  private void btnMainSets\_Click(object sender, EventArgs e)  {  ChartMain.AddChartSetsWindow();  ChartMain.ShowChartSetsWindow(btnMainSets);  } |

Одно окно настроек может настраивать несколько графиков. Для этого необходимо передать в функцию AddChartSetsWindow через запятую имена объектов типа ChartProvider, связанных с этими графиками:

|  |
| --- |
| ChartMain.AddChartSetsWindow(ChartMain, ChartSecond, ChartThird); |

**Сохранение массивов точек (замеров) в файл**

На форме необходимо создать кнопку сохранения замеров (объект типа «button») с именем, к примеру, btnMainSave. Двойным кликом по ней создастся функция события Click. Внутри нее необходимо вызвать функцию SaveToFileAs объекта ChartMain:

|  |
| --- |
| // Кнопка "Сохранить замеры как"  private void btnMainSave\_Click(object sender, EventArgs e)  {  ChartMain.SaveToFileAs();  } |

**Загрузка массивов точек (замеров) из Excel-файла**

На форме необходимо создать кнопку загрузки замеров (объект типа «button») с именем, к примеру, btnMainLoad. Двойным кликом по ней создастся функция события Click. Внутри нее необходимо вызвать функцию LoadFromExcelFile объекта ChartMain:

|  |
| --- |
| // Кнопка "Загрузить замеры из файла"  private void btnMainLoad\_Click(object sender, EventArgs e)  {  ChartMain.LoadFromExcelFile();  } |

**Удаление всех замеров**

На форме необходимо создать кнопку удаления всех замеров (объект типа «button») с именем, к примеру, btnMainDelAll. Двойным кликом по ней создастся функция события Click. Внутри нее необходимо вызвать функцию DeleteAll объекта ChartMain:

|  |
| --- |
| // Кнопка "Удалить все"  private void btnMainDelAll\_Click(object sender, EventArgs e)  {  ChartMain.DeleteAll();  } |

**Удаление одного замера**

Предположим, необходимо работать с 10-ю замерами. Для этого на форме необходимо создать 10 кнопок для удаления каждого замера (объекты типа «button») с именами, к примеру, типа btnMainDelS5. Двойным кликом по каждой кнопке создадутся функции события Click. Внутри каждой них необходимо вызвать функцию DeleteMeasurement объекта ChartMain и передать в нее номер замера, удаляемого данной кнопкой. Например, для кнопки удаления 5-го замера:

|  |
| --- |
| // Кнопка "Удалить замер 5"  private void btnMainDelS5\_Click(object sender, EventArgs e)  {  ChartMain.DeleteMeasurement(5);  } |

**Изменение доступности замера**

Если замер доступен – он отображается на графике и в текстовом поле, а также доступен для записи.

Предположим, необходимо работать с 10-ю замерами. Для этого на форме необходимо создать 10 флажков (объекты типа «checkBox») с именами, к примеру, типа checkBoxMainS5. Двойным кликом по каждому объекту создадутся функции события CheckedChanged. Внутри каждой них необходимо вызвать функцию SetMeasurementAvailability объекта ChartMain и передать в нее

1-м аргументом – номер соответствующего замера,

2-м аргументом – доступен ли замер (true/false).

Например, для изменения доступности 5-го замера:

|  |
| --- |
| // Установка доступности замера 5  private void checkBoxMainS5\_CheckedChanged(object sender, EventArgs e)  {  ChartMain.SetMeasurementAvailability(5, checkBoxMainS5.Checked);  } |

**Настройка внешнего вида графика**

Функции настройки внешнего вида графика используются в том случае, когда для этого графика не добавляется окно настроек. Кроме того, данные функции используются для сохранения и загрузки настроек графика при закрытии и открытии приложения (см. «Сохранение и загрузка настроек графика»).

**Установка значений осей**

По умолчанию, график (объект типа «chart») значения осей устанавливает автоматически. Для задания конкретных значений осей необходимо использовать функцию SetAxisesValues с именованными аргументами.

Предположим, объявлен объект ChartSecond1 типа ChartProvider, связанный с графиком, которому необходимо иметь постоянные значения оси X: минимум, максимум и интервал. Для этого необходимо вызвать функцию SetAxisesValues из данного объекта и передать в нее имя аргумента и, через двоеточие, значение этого аргумента:

|  |
| --- |
| ChartSecond1.SetAxisesValues(xmin: 0, xmax: 100, xint: 10); |

Эта функция может принимать несколько аргументов из списка: xmin, xmax, xint, ymin, ymax, yint.

**Установка автоматических значений осей**

Чтобы обратно установить автоматические значения осей, используется аналогичная функция SetAxisesAutoValues. Например, установка автоматических значений оси X:

|  |
| --- |
| ChartSecond1.SetAxisesAutoValues(xmin: true, xmax: true, xint: true); |

**Установка типа графика**

Для установки типа графика необходимо вызвать функцию SetChartType и передать в нее имя типа из списка: "Line", "Spline", "StepLine", "Column", "Candlestick", "Point". Например, установка типа графика в виде точек:

|  |
| --- |
| ChartSecond1.SetChartType("Point"); |

**Установка толщины линий графика**

Для установки толщины линий графика необходимо вызвать функцию SetChartLineWidth и передать в нее значение толщины (целое число > 0). Например, установка толщины линий равной двум:

|  |
| --- |
| ChartSecond1.SetChartLineWidth(2); |

**Установка одинакового цвета линий графика**

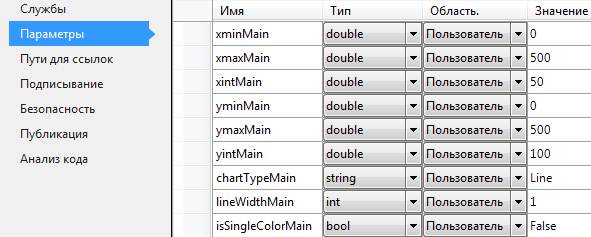
Для установки одинакового цвета линий графика необходимо вызвать функцию SetChartSingleLineColor и передать в нее true:

|  |
| --- |
| ChartSecond1.SetChartSingleLineColor(true); |

или false для установки различных цветов линий графика.

**Сохранение и загрузка настроек графика**

Во-первых, необходимо в верхнем меню Visual Studio нажать «Проект», «Свойства», «Параметры» и создать следующую таблицу сохраняемых параметров:



Здесь перечислены все имеющиеся параметры графика в классе ChartProvider. Для каждого графика необходимо создавать соответствующий набор требуемых параметров.

Далее, в окне свойств главной формы необходимо перейти во вкладку «События» и двойным кликом создать функции событий Load и FormClosing.

В функции события Load необходимо вызвать функции SetAxisesValues, SetChartType, SetChartLineWidth, SetChartSingleLineColor из ранее созданного объекта ChartMain и передать в них соответствующие параметры:

|  |
| --- |
| //Открытие формы  private void FormMain\_Load(object sender, EventArgs e)  {  //Ссылка на объект настроек (для сокращения кода)  var prop = Properties.Settings.Default;    //Загрузка настроек  ChartMain.SetAxisesValues(xmin: prop.xminMain, xmax: prop.xmaxMain,  ymin: prop.yminMain, ymax: prop.ymaxMain,  xint: prop.xintMain, yint: prop.yintMain);  ChartMain.SetChartType(prop.chartTypeMain);  ChartMain.SetChartLineWidth(prop.lineWidthMain);  ChartMain.SetChartSingleLineColor(prop.isSingleColorMain);  } |

В функции события FormClosing необходимо присвоить значения параметрам. Для этого необходимо вызвать соответствующие свойства объекта ChartMain:

|  |
| --- |
| //Закрытие формы  private void FormMain\_FormClosing(object sender, FormClosingEventArgs e)  {  //Ссылка на объект настроек (для сокращения кода)  var prop = Properties.Settings.Default;  //Сохранение настроек  prop.xminMain = ChartMain.Xmin;  prop.xmaxMain = ChartMain.Xmax;  prop.xintMain = ChartMain.Xint;  prop.yminMain = ChartMain.Ymin;  prop.ymaxMain = ChartMain.Ymax;  prop.yintMain = ChartMain.Yint;  prop.chartTypeMain = ChartMain.ChartType;  prop.lineWidthMain = ChartMain.LineWidth;  prop.isSingleColorMain = ChartMain.SingleColor;  prop.Save();  } |

|  |
| --- |
| **Список доступных свойств и функций** |

**Свойства**

Все свойства доступны только для чтения. Для установки значений свойств необходимо использовать соответствующие функции.

**Значения осей графика**

Возвращает значение минимума оси X

public double Xmin

Возвращает значение максимума оси X

public double Xmax

Возвращает значение минимума оси Y

public double Ymin

Возвращает значение максимума оси Y

public double Ymax

Возвращает значение интервала оси X

public double Xint

Возвращает значение интервала оси Y

public double Yint

Для установки данных значений необходимо использовать функцию SetAxisesValues.

**Настройки внешнего графика**

Возвращает значение типа графика

public string ChartType

Для установки значения необходимо использовать функцию SetChartType.

Возвращает значение толщины линий графика

public int LineWidth

Для установки значения необходимо использовать функцию SetChartLineWidth.

Возвращает true если линии графика одного цвета

public bool SingleColor

Для установки значения необходимо использовать функцию SetChartSingleLineColor.

**Функции**

В квадратных скобках указаны необязательные аргументы.

В круглых скобках указаны допустимые значения для аргументов.

Конструктор класса ChartProvider

public ChartProvider(Chart chart, int numberOfMeasurements = 10)

Chart chart Связанный график

[int numberOfMeasurements = 10] Количество отображаемых замеров (> 0)

**Основные функции для работы с массивами данных (замерами)**

Добавление textBox для вывода значений замеров

public void AddTextDisplaying(TextBox textBox, string xName = "X", string yName = "Y")

TextBox textBox Связанный textBox

[string xName = "X"] Имя измеряемой величины по оси X

[string yName = "Y"] Имя измеряемой величины по оси Y

Добавление горизонтальной линии

public void AddHorizontalLine(NumericUpDown horLineInput, Color horLineColor)

NumericUpDown horLineInput Связанное поле ввода

Color horLineColor Цвет линии

Отображение замера

public void Display(double[] xPoints, double[] yPoints, int? numberOfPoints = null)

double[] xPoints Массив X-координат точек (Длина > 0)

double[] yPoints Массив Y-координат точек (Длина > 0)

[int? numberOfPoints = null ] Количество отображаемых точек (> 0)

Удаление всех замеров

public void DeleteAll()

Удаление одного замера

public void DeleteMeasurement(int num)

int num Номер замера (нумерация с единицы) (0 < num <= кол-во замеров)

Установка доступности замера

public void SetMeasurementAvailability(int num, bool state)

int num Номер замера (нумерация с единицы) (0 < num <= кол-во замеров)

bool state Доступен ли замер

Установка доступности увеличения графика мышью

public void SetScalingByMouseAvailability(bool state)

bool state Доступно ли увеличение графика мышью

Сохранение графика и/или значений замеров в файл

public void SaveToFileAs()

Загрузка значений замеров из Excel-файла

public void LoadFromExcelFile()

Добавление окна настроек графика

public void AddChartSetsWindow()

Добавление окна настроек для нескольких графиков

public void AddChartSetsWindow(params ChartProvider[] chartProviders)

params ChartProvider[] chartProviders Объекты класса СhartProvider через запятую

Открытие окна настроек графика/графиков

public void ShowChartSetsWindow(Button chartSetsWindowButton = null)

[Button chartSetsWindowButton = null] Кнопка открытия/закрытия окна

**Функции настройки внешнего вида графика**

Установка значений осей графика

public void SetAxisesValues(double? xmin = null, double? xmax = null,

double? ymin = null, double? ymax = null,

double? xint = null, double? yint = null)

[double? xmin = null] Минимум оси X

[double? xmax = null] Максимум оси X

[double? ymin = null] Минимум оси Y

[double? ymax = null] Максимум оси Y

[double? xint = null] Интервал оси X

[double? yint = null] Интервал оси Y

Установка автоматических значений осей графика

public void SetAxisesAutoValues(bool xmin = false, bool xmax = false,

bool ymin = false, bool ymax = false,

bool xint = false, bool yint = false)

[bool xmin = false] Минимум оси X

[bool xmax = false] Максимум оси X

[bool ymin = false] Минимум оси Y

[bool ymax = false] Максимум оси Y

[bool xint = false] Интервал оси X

[bool yint = false] Интервал оси Y

Установка типа графика

public void SetChartType(string chartType)

string chartType Тип графика ("Line", "Spline", "StepLine", "Column", "Candlestick", "Point")

Установка толщины линий графика

public void SetChartLineWidth(int lineWidth)

int lineWidth Значение толщины линий (> 0)

Установка одинакового цвета линий графика

public void SetChartSingleLineColor(bool isSingleColor)

bool isSingleColor Одинакового ли цвета линии графика