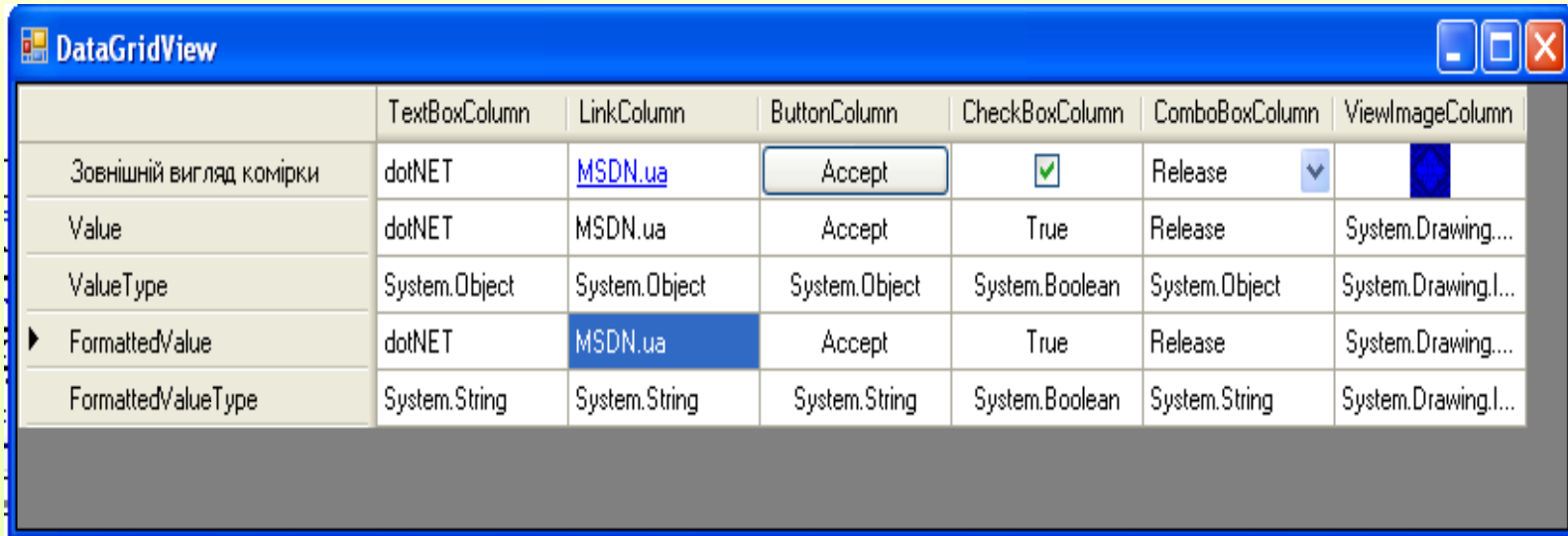



DataGridView

Клакович Л.М.

Елемент керування DataGridView

Відображення та редагування табличних даних



	TextBoxColumn	LinkColumn	ButtonColumn	CheckBoxColumn	ComboBoxColumn	ViewImageColumn
Зовнішній вигляд комірки	dotNET	MSDN.ua	Accept	<input checked="" type="checkbox"/>	Release ▼	
Value	dotNET	MSDN.ua	Accept	True	Release	System.Drawing....
ValueType	System.Object	System.Object	System.Object	System.Boolean	System.Object	System.Drawing.I...
▶ FormattedValue	dotNET	MSDN.ua	Accept	True	Release	System.Drawing....
FormattedValueType	System.String	System.String	System.String	System.Boolean	System.String	System.Drawing.I...

```
public class DataGridView : Control, ISupportInitialize
```

DataGridView підтримує три режими роботи з даними:

1. **Стандартний** – відображення даних із зовнішніх колекцій (напр., **ListView**, **DataTable**).
2. **Спеціальний** режим відображення *вільних* (**unbound**) даних, тобто дані зберігаються в самому контролі.
3. **Віртуальний** режим (**virtual mode**). Контрол генерує подію, при отриманні якої код повертає деякі дані. Дані ніде не зберігаються і тому віртуальний режим може оперувати мільйонами рядків.

Стандартна модель зв'язування даних

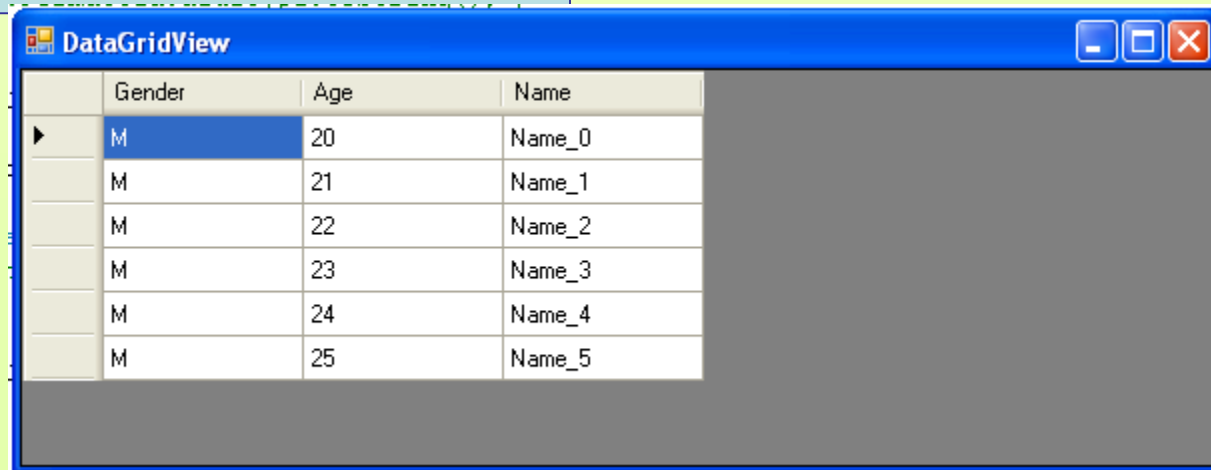
- Задати властивість **DataSource** відповідним джерелом.
- Задати властивість **DataMember** рядком, що визначає список або таблицю, яку треба під'єднати. Використовується у випадку множинних таблиць чи списків.
- **DataGridView** можна під'єднати до таких джерел :
 - клас, що реалізує **IList**, включаючи одновимірні масиви.
 - клас, що реалізує **IListSource**, такі як **DataTable** і **DataSet**.
 - клас, що реалізує **IBindingList**, такі як **BindingList**.
 - клас, що реалізує **IBindingListView**, такі як **BindingSource**.

Віддавайте перевагу компоненті **BindingSource**, як джерелу даних **DataGridView**, оскільки вона дозволяє під'єднувати різноманітні джерела даних і може розв'язати багато проблем під'єднання даних автоматично

```
public class PersonCollection : System.Collections.IEnumerable
{
    public System.Collections.IEnumerator GetEnumerator()
    {
        for(uint i = 0; i <= 5; i++)
        {
            yield return new Person("Name_" + i.ToString(), 20 + i, 'M');
        }
    }
}
```

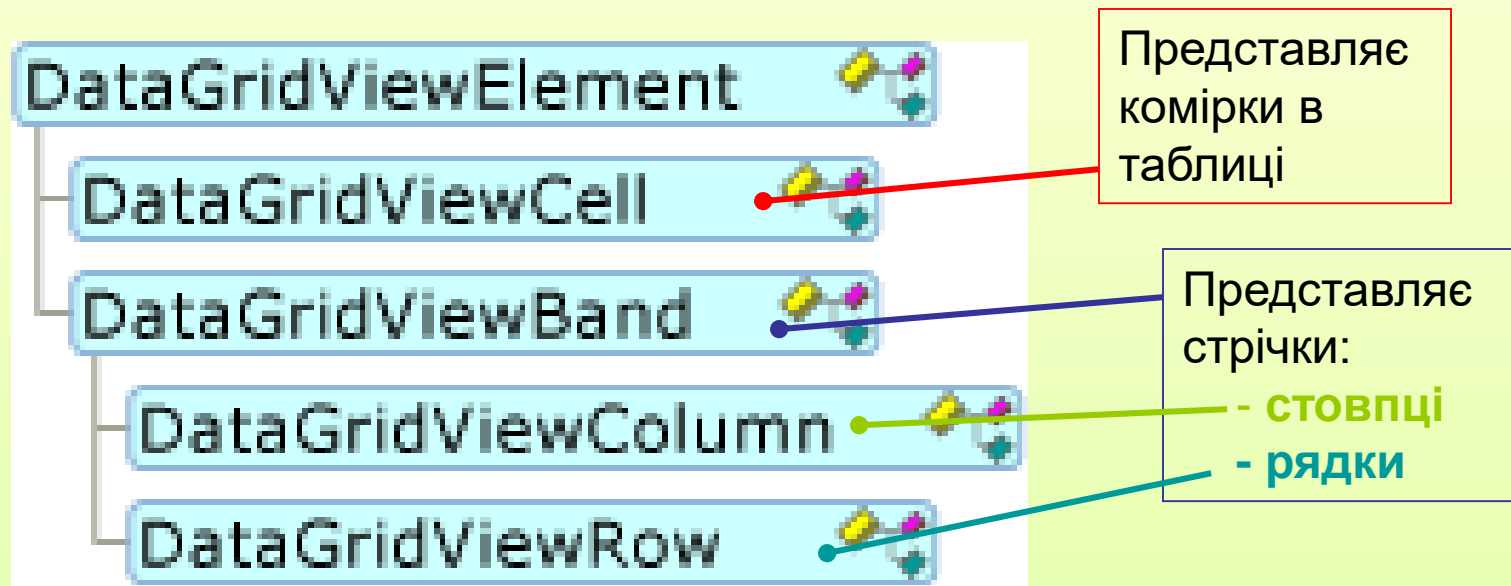
```
public class Person
{
    private string _name;
    private uint _age;
    private char _gender;
    public Person(string name,
        uint age, char gender){...}
    //властивості
    public string Name{...}
    public uint Age{...}
    public char Gender{...}
}
```

```
BindingSource biSour=new BindingSource();
biSour.DataSource = new PersonCollection();
DataGridView grid =new DataGridView();
grid.DataSource=biSour;
grid.Dock = DockStyle.Fill;
Controls.Add(grid);
```



	Gender	Age	Name
▶	M	20	Name_0
	M	21	Name_1
	M	22	Name_2
	M	23	Name_3
	M	24	Name_4
	M	25	Name_5

- В **DataGridView** значну частину функціональності, доступної користувачу, забезпечують сторонні класи. Базовим для них всіх - **DataGridViewElement**



Елемент керування DataGridView

Відображення та редагування табличних даних

TopLeftHeaderCell

Заголовки

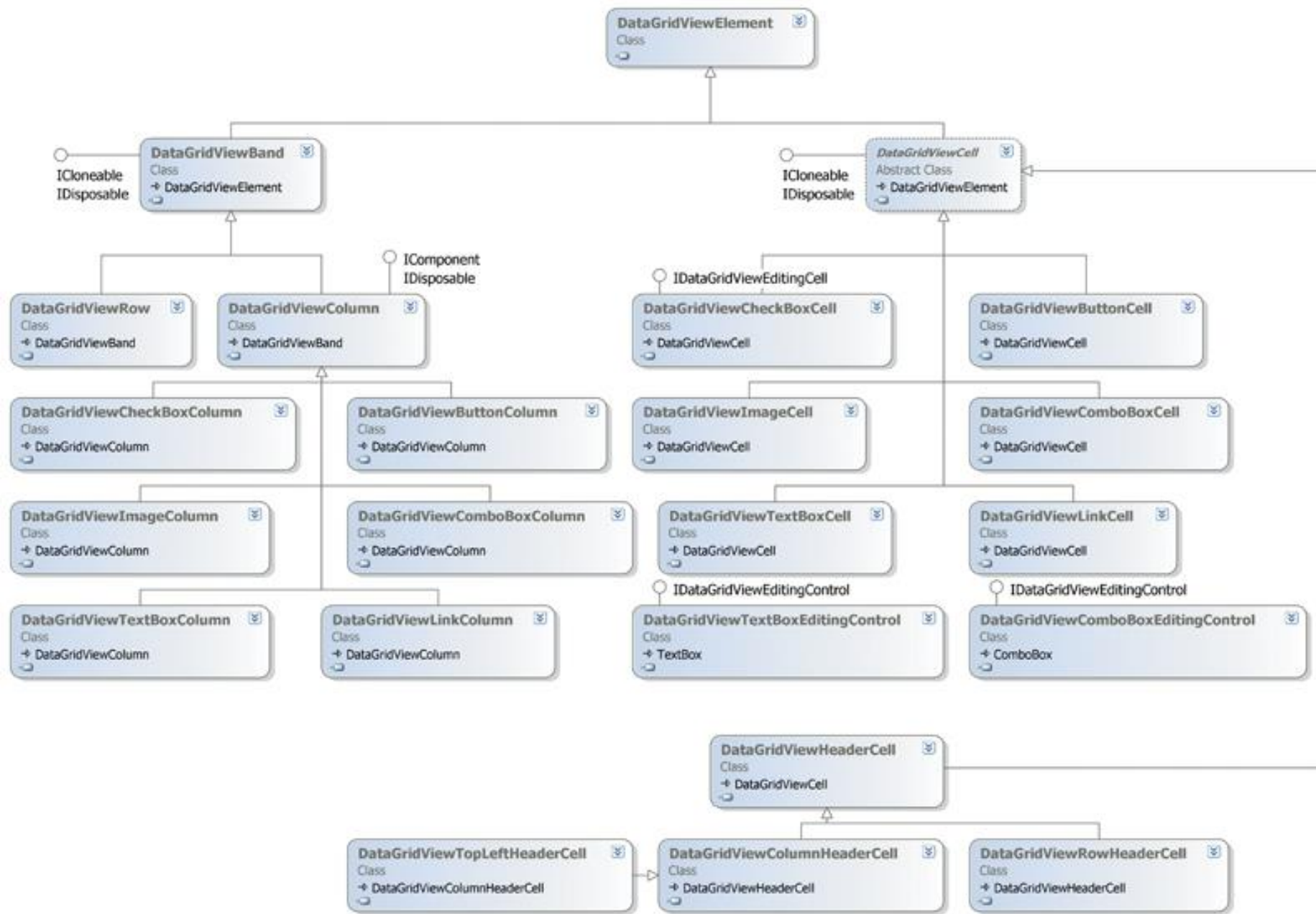
стовпців

ColumnHeaders

Заголовки
рядків
RowHeaders

	TextBoxColumn	LinkColumn	ButtonColumn	CheckBoxColumn	ComboBoxColumn	ViewImageColumn
Зовнішній вигляд комірки	dotNET	MSDN.ua	Accept	<input checked="" type="checkbox"/>	Release	
Value	dotNET	MSDN.ua	Accept	True	Release	System.Drawing....
ValueType	System.Object	System.Object	System.Object	System.Boolean	System.Object	System.Drawing.I...
FormattedValue	dotNET	MSDN.ua	Accept	True	Release	System.Drawing....
FormattedValueType	System.String	System.String	System.String	System.Boolean	System.String	System.Drawing.I...

Поточна (current)
вибрана комірка



class DataGridViewElement

```
public class DataGridViewElement: Object
```

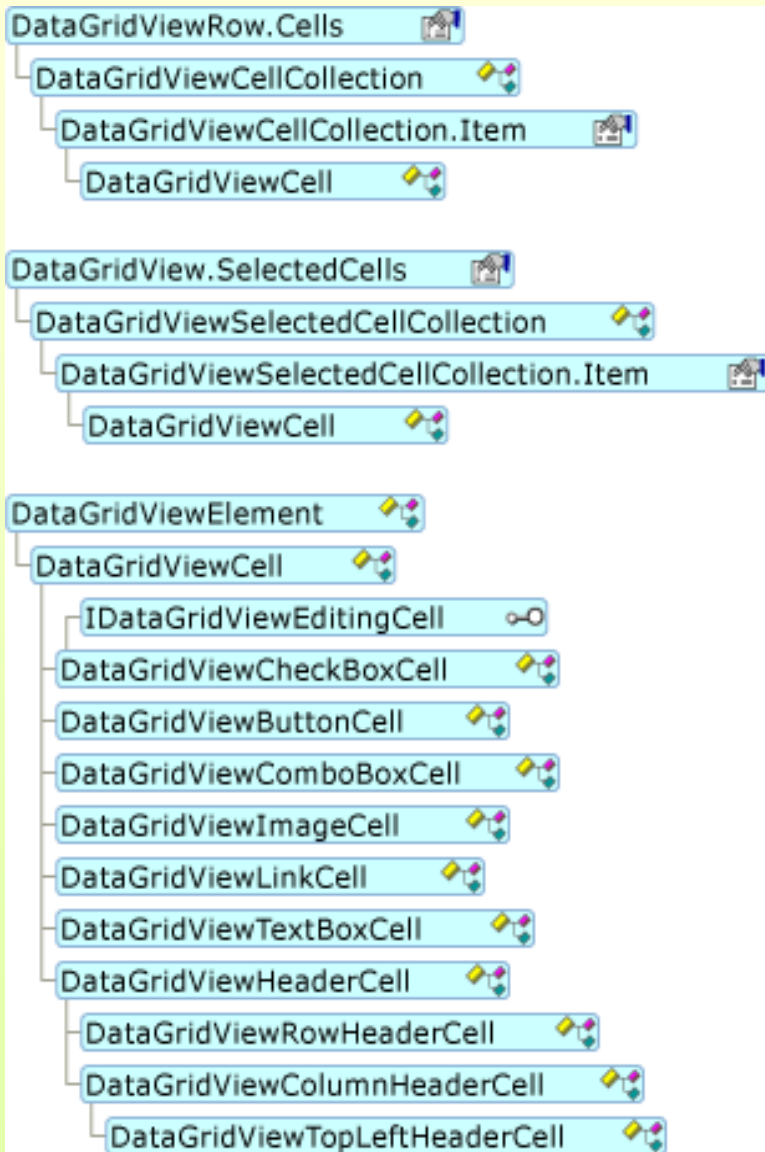
Public Properties

Name	Description
DataGridView	Отримати контрол DataGridView , до якого належить цей елементом
State	Отримати стан елемента - комбінація потенційно можливих режимів відображення (значення перелічення DataGridViewElementStates)

enum DataGridViewElementStates

Member name	Description
Displayed	Елемент відображається - поточний.
Frozen	“заморожений” - прокручування заборонено.
None	Елемент в стані за замовчуванням.
ReadOnly	Елемент не може бути змінений користувачем.
Resizable	Можна змінювати розмір.
ResizableSet	Елемент не наслідуює можливість зміни розміру.
Selected	Елемент є у виділеному стані.
Visible	Елемент видимий

Об'єктна модель **DataGridViewCell**



- **DataGridViewCell** представляє окремі комірки в **DataGridView**
- Доступ до комірок - через властивість **Cells** (класу **DataGridViewRow**), яка надає колекцію комірок.

```
grid.Rows[2].Cells[1];
```

- Доступ до виділених комірок – через властивості **SelectedCells** (класу **DataGridView**), яка надає колекцію виділених комірок.

```
for(i = 0;  
    i<grid.SelectedCells.Count); i++)  
{  
    grid.SelectedCells[i]; //...  
}
```

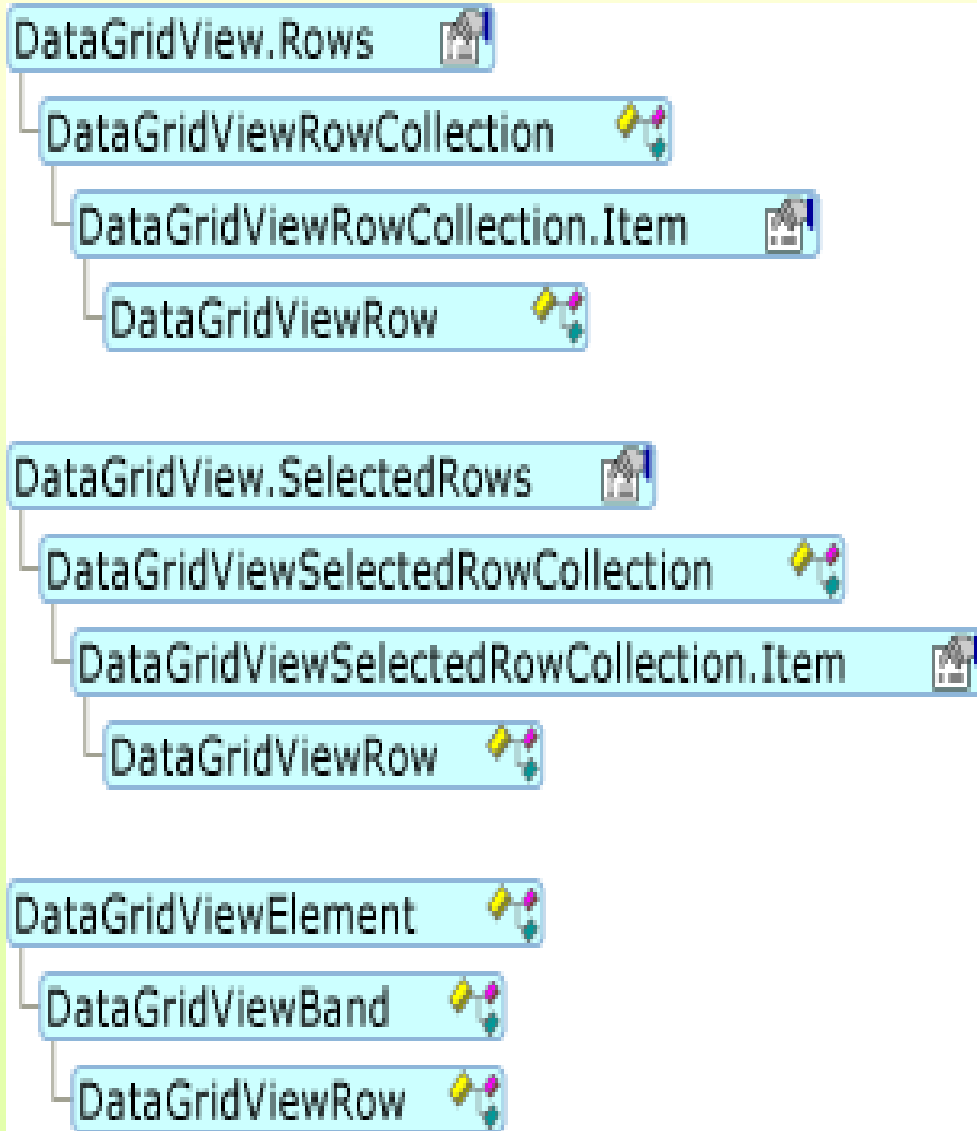
public abstract class DataGridViewCell :

DataGridViewElement, ICloneable, IDisposable

Властивість	Опис
ColumnIndex	Номер стовпця цієї комірки
ContextMenuStrip	Контекстне меню цієї комірки
DefaultNewRowValue	Значення за замовчуванням для комірки в рядку
EditType	Тип контролю в комірці.
FormattedValue	Значення комірки, форматоване для показу
IsInEditMode	Чи дану комірку почали редагувати.
OwningColumn	Стовпець, що містить дану комірку.
OwningRow	Рядок, що містить дану комірку
ReadOnly	Значення, чи комірка може бути редагована
Resizable	Значення, чи можна змінювати розмір комірки
RowIndex	Номер рядка даної комірки
Selected	Чи комірка виділена
Size	Розмір комірки
Style	Стиль комірки (типу DataGridViewCellStyle).
Value	Значення, що міститься в комірці
ValueType	Тип значення в комірці

Об'єктна модель DataGridViewRow

```
public class DataGridViewRow : DataGridViewBand
```



- Представляє рядки в **DataGridView**. Це набори даних джерела **DataGridView**
- Доступ до колекції рядків здійснюється через властивості **Rows** або **SelectedRows** контролю **DataGridView**

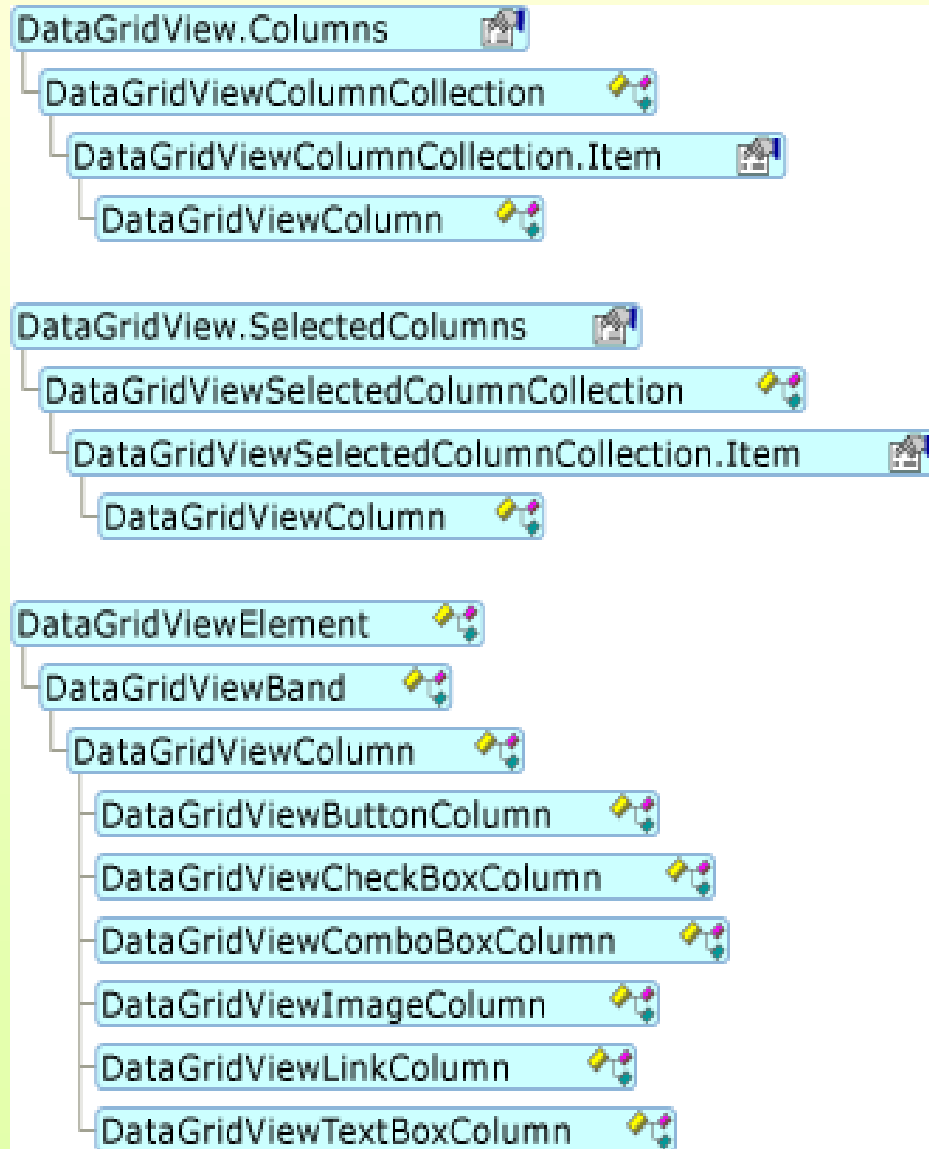
```
DataGridViewRow row = grid.Rows[0];
```

- Містить властивість **Cells** – колекцію всіх комірок в цьому рядку (на відміну від **DataGridViewColumn**).

Public Properties

Name	Description
Cells	Отримати колекцію комірок, що утворюють рядок
ContextMenuStrip	Overridden. Отримати чи встановити контекстне меню для рядка
DataBoundItem	Отримати об'єкт, що під'єднаний до рядка.
DefaultCellStyle	Overridden. Отримати чи встановити стиль за замовчуванням для рядка
Displayed	Overridden. Отримати значення, яке показує чи цей рядок видно на екрані
DividerHeight	Отримати чи встановити висоту в пікселях, роздільника рядка.
HeaderCell	Отримати чи встановити комірку-заголовок рядка.
Height	Отримати чи встановити поточну висоту рядка.
Index	Отримати відносну позицію стрічки в DataGridView
MinimumHeight	Отримати чи встановити мінімальну висоту рядка.
Selected	Overridden. Отримати чи встановити значення, що показує чи рядок є вибраним (selected).
State	Overridden. Отримати поточний стан рядка.
Visible	Overridden. Отримати чи встановити значення, що показує чи рядок є видимий (visible).

public class DataGridViewColumn : DataGridViewBand, IComponent, IDisposable



- Представляють стовпці в **DataGridView** - схему відображення даних в контролі
- Доступ до колекції стовпців – через властивість **Columns** або **SelectedColumns**.
- Якщо **DataGridView** не містить жодного стовпця - не існує.
- В основному використовується для встановлення вигляду та поведінки стовпців, таких як товщина стовпця або стиль комірки
- Стиль комірок встановлений у стовпці поширюється на всі комірки. Для окремої комірки стиль можна змінити .

Default Column Types

Class	Description
DataGridViewTextBoxColumn	Для розміщення тексту . Генеруються автоматично, коли під'єднуються до чисел або рядків.
DataGridViewCheckBoxColumn	Для булівського або CheckState типу. Генеруються автоматично, коли під'єднуються до значень цих типів.
DataGridViewImageColumn	Для відображення рисуноків . Генеруються автоматично, коли під'єднуються до масиву байтів, об'єкту Image або Icon.
DataGridViewButtonColumn	Для відображення кнопки в комірці. Автоматично не генерується. Використовується для “вільних” колонок
DataGridViewComboBoxColumn	Для відображення випадаючого списку . Автоматично не генерується. Під'єднання до даних - вручну
DataGridViewLinkColumn	Для відображення посилання в комірці. Автоматично не генерується. Під'єднання до даних - вручну
Твій власний тип стовпця	Можна створити власний тип стовпця, похідний від DataGridViewColumn для забезпечення потрібного вигляду, поведінки.

Додавання стовпців

4 ситуації:

- 1) Є джерело даних, воно доступне під час розробки, додаємо стовпці під час розробки
- 2) Нема джерела даних, але під час розробки ми знаємо склад і тип стовпців .
- 3) Є джерело даних, але воно доступне тільки під час виконання, а під час розробки невідоме ні джерело ні склад стовпців.
- 4) Нема джерела даних, а склад/тип стовпців виявляється динамічно, під час виконання, а під час розробки невідомо тип і навіть кількість стовпців.

Джерело даних доступне підчас розробки

• У вікні **Properties** для **DataGridView** встановлюємо значення властивостей **DataSource** і **DataMember**. Контрол автоматично вивчає схему джерела і генерує по стовпцю для кожного стовпця таблиці (джерела даних), підбираючи не тільки заголовок стовпця, але і тип стовпця.

Редaguємо стовпці з допомогою меню **EditColumns**:

- видаляємо зайві (кнопка **Remove**),
- змінюємо заголовок стовпця (**HeaderText**),
- тип стовпця (**ColumnType**)

Edit Columns dialog details:

- Selected Columns:** ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued.
- Bound Column Properties:**
 - Appearance:** DefaultCellStyle: DataGridViewCellStyle { }, HeaderText: ProductID, ToolTipText: , Visible: True.
 - Behavior:** ContextMenuStrip: (none), MaxInputLength: 32767, ReadOnly: True, Resizable: True, SortMode: Automatic.
 - Data:** DataPropertyName: ProductID.
 - Design:** (Name): dataGridViewTextBoxColumn, ColumnType: DataGridViewTextBoxColumn.
 - Layout:** AutoSizeMode: NotSet, DividerWidth: 0, FillWeight: 100, Frozen: False, MinimumWidth: 5, Width: 100.
- ColumnType:** The DataGridViewColumn type.

DataGridView Tasks context menu:

- Choose Data Source: _biSour
- Edit Columns...
- Add Column...
- Enable Adding
- Enable Editing
- Enable Deleting
- Enable Column Reordering
- Dock in parent container
- Add Query...
- Preview Data...

DataGridView Tasks list:

- ☒ DataGridViewCheckBoxColumn
- ☐ DataGridViewButtonColumn
- ☐ DataGridViewComboBoxColumn
- ☐ DataGridViewImageColumn
- ☐ DataGridViewLinkColumn
- ☐ DataGridViewTextBoxColumn

Відсутність джерела даних в дизайн-часі

- Властивості *DataSource* і *DataMember* не встановлюємо
- Знаючи кількість і тип стовпців додаємо стовпці підчас розробки, використовуючи меню **AddColumn**

The screenshot shows the 'Add Column' dialog box with the following details:

- Title:** Add Column
- DataSource:** Databound column (radio button is unselected)
- Columns in the DataSource:** (Empty list box)
- Unbound column:** (radio button is selected)
- Name:** Column1
- Type:** DataGridViewTextBoxColumn
- Header text:** Column1
- Options:** ☒ Visible, ☐ Read Only, ☐ Frozen
- Buttons:** Add, Cancel

- Ім'я стовпця - це ім'я змінної у формі (типу DataGridViewColumn), якій присвоюється посилання на доданий стовпець.
- Тип стовпця.
- Текст, який буде показано в заголовку даного стовпця.

Джерело даних доступне підчас виконання

- **DataGridView.AutoGenerateColumns=true**
- Задаємо значення властивостей **DataSource/DataMember** підчас виконання, що приводить до автоматичного створення і додавання стовпців до **DataGridView**, згідно з джерелом даних.
- Змінюємо стовпці через властивість **Columns** типу **DataGridViewColumnCollection**:

```
grid.AutoGenerateColumns = true;  
grid.DataSource = biSour;  
grid.Columns[1].Width = 188;  
grid.Columns[1].HeaderText = "MyHeader";
```

Відсутність джерела даних підчас виконання

- Створюється власна колекція стовпців:
- Використовуючи метод **Add()** колекції `DataGridViewColumnCollection`:

Add (DataGridViewColumn) Додає заданий стовець до колекції.

Add (String, String) Додає стовець типу **DataGridViewTextBoxColumn** із заданим ім'ям стовпця і заголовком стовпця до колекції.

```
grid.Columns.Add("MyColumnName", "MyColumnHeaderText");  
grid.Columns.Add(new DataGridViewCheckBoxColumn());
```

- Додаємо стовпці зі значеннями за замовчуванням через властивість **ColumnCount**:

```
_grid.DataSource = null;  
_grid.ColumnCount = 5;
```

- Видалення стовпців і додавання “вільних” та зв’язних стовпців

Key1	Name1	City1
1	AAA1	MyCity1
2	AAA2	MyCity2
3	AAA3	MyCity3
4	AAA4	MyCity4
5	AAA5	MyCity5
6	AAA6	MyCity6
7	AAA7	MyCity7

```
grid.DataSource = this.myTbl1BindingSource;  
grid.Columns.RemoveAt(2);  
grid.Columns.Add("additionalColumn", "FreeForStart");
```

	Key1	Name1	FreeForStart
►	1	AAA1	
	2	AAA2	
	3	AAA3	
	4	AAA4	
	5	AAA5	
	6	AAA6	
	7	AAA7	

```
// Прив'яжемо стовпець до стовпця з джерела даних:  
grid.Columns["additionalColumn"].DataPropertyName = "City1";
```

	Key1	Name1	FreeForStart
►	1	AAA1	MyCity1
	2	AAA2	MyCity2
	3	AAA3	MyCity3
	4	AAA4	MyCity4
	5	AAA5	MyCity5
	6	AAA6	MyCity6
	7	AAA7	MyCity7

Додавання рядків у DataGridView

- Додавати рядки в дизайн-часі неможливо
- Рядки автоматично додадуться при підключенні до джерела даних
- Можна додавати рядки методом **Add()**:

int Add()

– додає один рядок, заповнюючи його значеннями за замовчуванням

int Add(int count)

- додає count рядків, заповнюючи їх значеннями за замовчуванням

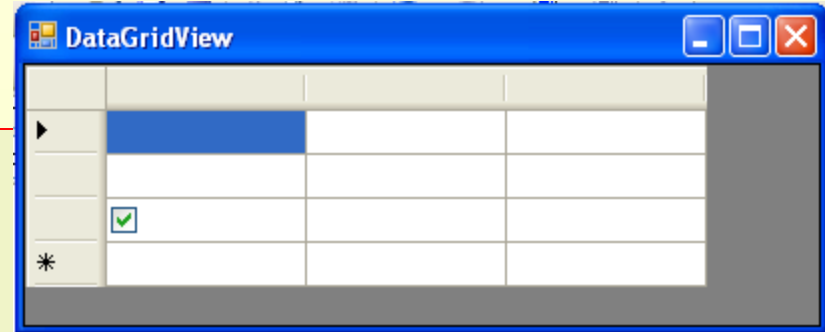
int Add(params object[] values) - додає один рядок, заповнюючи його значеннями з масиву values

int Add(DataGridViewRow row)

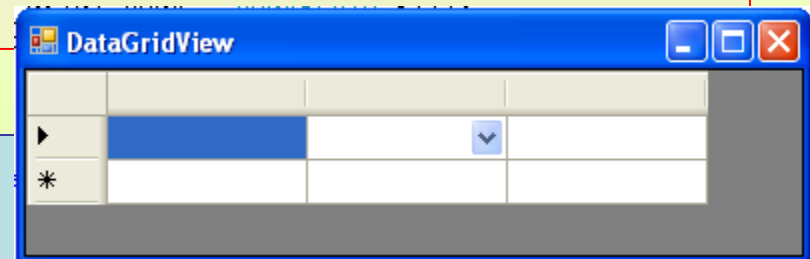
- додає наперед створений рядок

- **DataGridView** допускає розміщення в одному стовпці комірок різних типів

```
grid.DataSource = null;  
grid.ColumnCount = 3;  
grid.Rows.Add();  
grid.Rows.Add();  
DataGridViewRow newRow = new DataGridViewRow();  
DataGridViewCheckBoxCell checkCell = new DataGridViewCheckBoxCell();  
checkCell.Value = true;  
newRow.Cells.Add(checkCell);  
newRow.Cells.Add(new DataGridViewTextBoxCell());  
newRow.Cells.Add(new DataGridViewTextBoxCell());  
grid.Rows.Add(newRow);
```



```
grid.DataSource = null;  
grid.ColumnCount = 3;  
DataGridViewRow row = new DataGridViewRow();  
row.CreateCells(grid);  
row.Cells.RemoveAt(1);  
row.Cells.Insert(1, new DataGridViewComboBoxCell());  
grid.Rows.Add(row);
```



Додавання даних в комірки

//Отримуємо “вільну” текстбокс-комірку (4,2) з grid

```
DataGridViewTextBoxCell txtCell =  
    (DataGridViewTextBoxCell)grid.Rows[4].Cells[2];
```

//заповнюємо її

```
txtCell.Value = "Great!";
```

//Отримуємо “зв’язану” текстбокс-комірку (2,1) з _grid

```
DataGridViewCell txtCell2 = grid.Rows[2].Cells[1];
```

//змінюємо її значення

```
txtCell2.Value = "Also good!";
```

	Key1	Name1	FreeForStart
►	1	AAA1	
	2	AAA2	
	3	Also good!	
	4	AAA4	
	5	AAA5	Great!
	6	AAA6	
	7	AAA7	

Сортування в **DateGridView**

[Sort\(IComparer\)](#)

Сортує вміст [DataGridView](#) використовуючи реалізацію інтерфейсу.

[Sort\(DataGridViewColumn, ListSortDirection\)](#)

Сортує вміст [DataGridView](#) у зростаючому або спадаючому порядку, порівнюючи значення у вказаному стовпці .

Властивість `SortMode` класу `DataGridViewColumn`:

`DataGridViewColumnSortMode` `SortMode`

`NotSortable`

Стовпець може бути відсортований тільки програмно. Без значка в заголовку стовпця.

`Automatic`

Користувач може сортувати стовпець, клікнувши на заголовку стовпця

`Programmatic`

Стовпець може бути відсортований тільки програмно. Значок сортування в заголовку стовпця.

DataGridViewTextBoxColumn

- Містить комірки типу **DataGridViewTextBoxCell**, в які поміщається контрол типу **DataGridViewTextBoxEditingControl**, якому передається значення комірки, який керує редагуванням
- Після редагування, коли змінюється фокус з даної комірки, генерується одна з двох подій **DataGridView.CellParsing** або **DataGridView.CellValuePushed**

DataGridViewLinkColumn

Для обробки клацання по посиланню треба створити і підключити обробник події **CellContentClick** grid-у.

DataGridViewButtonColumn

Для обробки клацання по кнопці треба створити і підключити обробник події **CellClick**