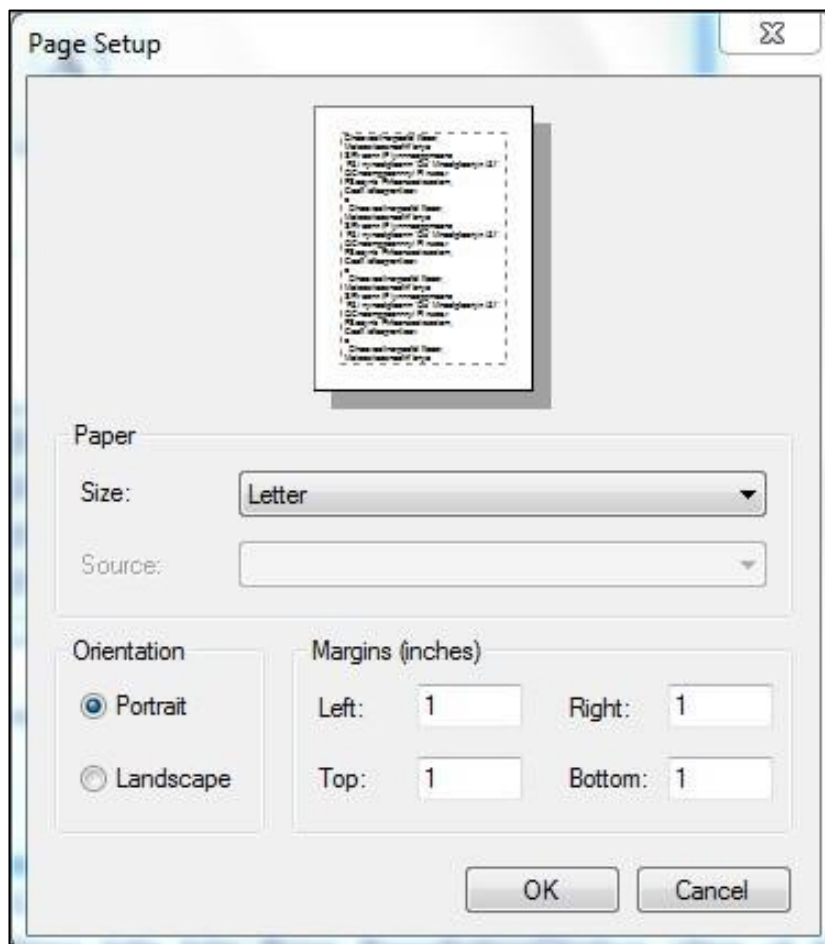


## CHƯƠNG 8: LÀM VIỆC VỚI ĐIỀU KHIỂN IN ẤN

### 8.1. Điều khiển PageSetupDialog

Điều khiển *PageSetupDialog* được hiển thị dạng hộp thoại khi chương trình thực thi. Việc hiển thị dạng hộp thoại cho phép người dùng dễ dàng thay đổi các thiết lập về trang như: canh lề, định hướng, kích thước trang, ... . Hộp thoại *PageSetupDialog* có giao diện như hình 8.1.



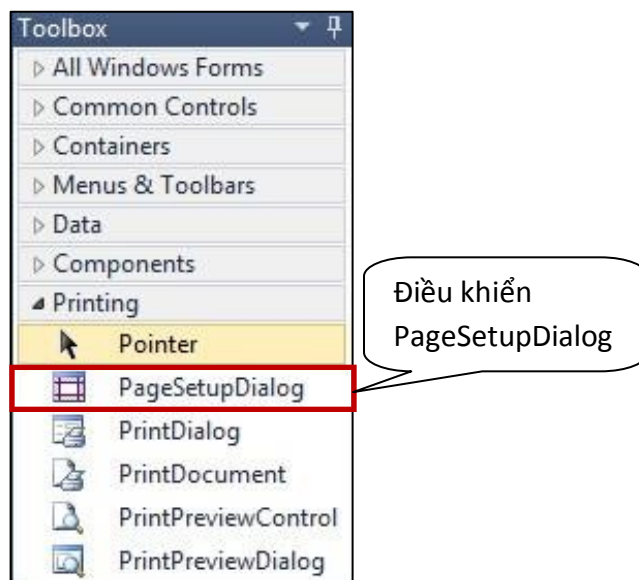
Hình 8.1: Giao diện hộp thoại *PageSetupDialog*

Những thay đổi của người dùng trên hộp thoại *PageSetupDialog* cũng sẽ làm thay đổi các giá trị tương ứng trên hộp thoại *PrintDocument*. Do đó việc thiết lập các giá trị trong hộp thoại *PageSetupDialog* cũng ảnh hưởng đến hình dạng của trang khi in ấn.

Hiển thị hộp thoại *PageSetupDialog* sử dụng phương thức *ShowDialog()*. Tuy nhiên, điểm khác biệt là để có thể gọi được phương thức *ShowDialog()* thì cần phải thực hiện một thao tác trước là gán một đối tượng thuộc lớp *PrintDocument* cho thuộc tính

*Document* của điều khiển *PageSetupDialog*. Bởi vì như đã nói ở trên là *PageSetupDialog* và *PrintDocument* có liên hệ với nhau. Những thay đổi tùy chọn trong *PageSetupDialog* đều ảnh hưởng đến hình dạng của trang khi in, mặt khác *PrintDocument* là đối tượng chính để thực hiện tiến trình in ấn. Do đó cần phải thiết lập một đối tượng thuộc lớp *PrintDocument* cho thuộc tính *Document* trước.

Điều khiển *PageSetupDialog* nằm trong nhóm Printing của cửa sổ Toolbox như hình 8.2.



Hình 8.2: Điều khiển *PageSetupDialog* trong cửa sổ Toolbox

- Một số thuộc tính thường dùng của *PageSetupDialog*:

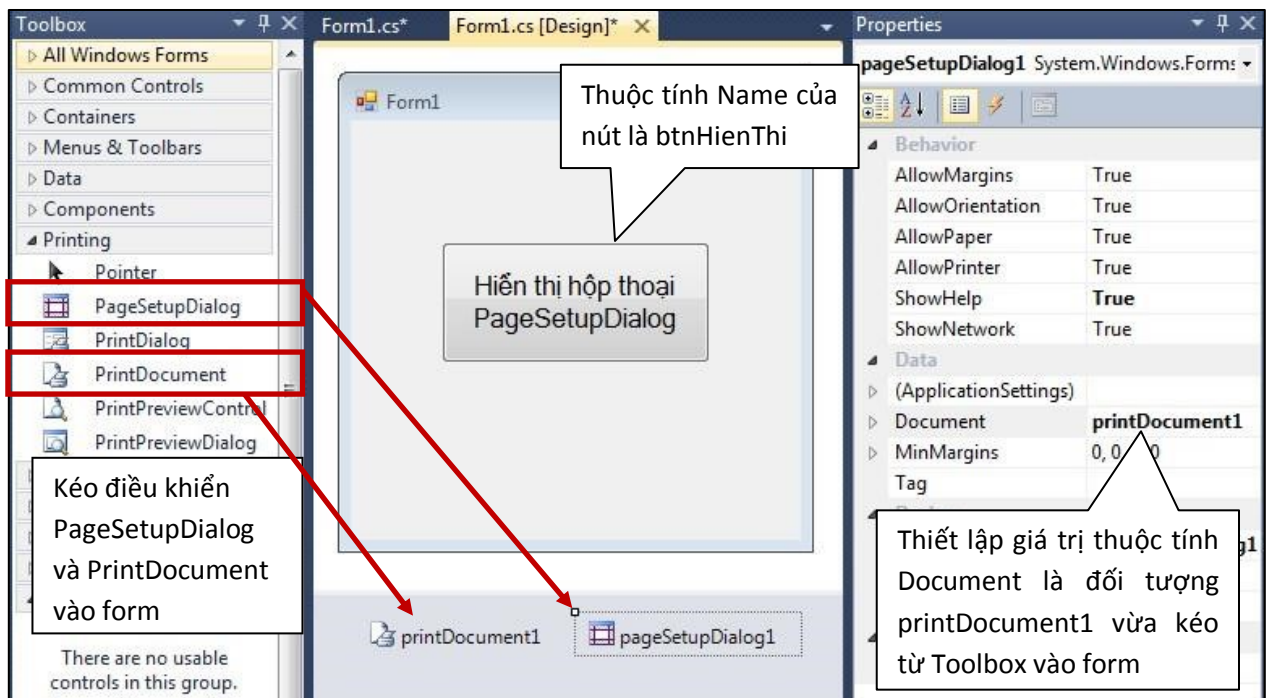
Bảng 8.1: Bảng mô tả các thuộc tính của *PageSetupDialog*

Thuộc tính	Mô tả
<i>AllowMargins</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép thiết lập độ rộng lề trên, dưới, phải và trái của trang</li> <li>- Nếu là False: Không cho phép thiết lập lề của trang</li> </ul>
<i>AllowOrientation</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép thiết lập trang hiển thị theo chiều ngang hoặc theo chiều dọc.</li> <li>- Nếu là False: Không cho phép thiết lập chiều hiển thị của trang.</li> </ul>
<i>AllowPaper</i>	Mang hai giá trị True hoặc False.

	<ul style="list-style-type: none"> <li>- Nếu là True: Cho phép thiết lập kích thước của trang. Có thể thiết lập trang kích thước A4, A3, ...</li> <li>- Nếu là False: Không cho phép thiết lập kích thước của trang</li> </ul>
<i>AllowPrinter</i>	<p>Mang hai giá trị True hoặc False.</p> <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép người dùng sử dụng chọn nút máy in. Khi người dùng nhấp vào nút Printer sẽ xuất hiện hộp thoại cho phép chọn máy in.</li> <li>- Nếu là False: Nút printer không có hiệu lực</li> </ul>
<i>Document</i>	<p>Mang giá trị là một đối tượng thuộc lớp PrintDocument.</p> <p>Lưu ý: Thuộc tính <i>Document</i> phải được gán giá trị trước khi hộp thoại <i>PageSetupDialog</i> được hiển thị bằng phương thức <i>ShowDialog()</i>.</p>
<i>MinMargins</i>	Kích thước nhỏ nhất mà người dùng có thể thiết lập để canh lề trên, dưới, trái hay phải của trang.
<i>ShowHelp</i>	<p>Mang hai giá trị True hoặc False.</p> <ul style="list-style-type: none"> <li>- Nếu là True: Trên hộp thoại xuất hiện thêm nút Help</li> <li>- Nếu là False: Không xuất hiện nút Help trên hộp thoại</li> </ul>
<i>ShowNetwork</i>	<p>Mang hai giá trị True hoặc False.</p> <ul style="list-style-type: none"> <li>- Nếu là True: Trên hộp thoại xuất hiện nút Network. Nút này cho phép người dùng có thể chọn máy in mạng.</li> <li>- Nếu là False: Không hiển thị nút Network</li> </ul>

➤ **Hiển thị hộp thoại *PageSetupDialog*:**

Lập trình viên có thể sử dụng điều khiển *PageSetupDialog* bằng cách thêm điều khiển *PageSetupDialog* từ cửa sổ Toolbox vào form. Sau đó, trước khi hiển thị hộp thoại *PageSetupDialog* bằng phương thức *ShowDialog()*, lập trình viên cần phải thiết lập thuộc tính *Document* bằng 1 đối tượng thuộc *PrintDocument* như hình 8.3.



Hình 8.3: Giao diện hiển thị hộp thoại PageSetupDialog

Sự kiện Click của nút btnHienThi:

```
private void btnHienThi_Click(object sender, EventArgs e)
{
    pageSetupDialog1.ShowDialog();
}
```

Ngoài ra, nếu không sử dụng điều khiển *PageSetupDialog* và *PrintDocument* bằng cách kéo thả vào form từ cửa sổ Toolbox, lập trình viên cũng có thể tạo đối tượng *PageSetupDialog* và hiển thị hộp thoại này bằng mã lệnh như sau:


```
private void btnHienThi_Click(object sender, EventArgs e)
{
    PageSetupDialog PageSD = new PageSetupDialog();
    System.Drawing.Printing.PrintDocument printDoc = new
        System.Drawing.Printing.PrintDocument();
    PageSD.Document = printDoc;
    PageSD.ShowDialog();
}
```

## 8.2. Điều khiển `PrintPreviewDialog`

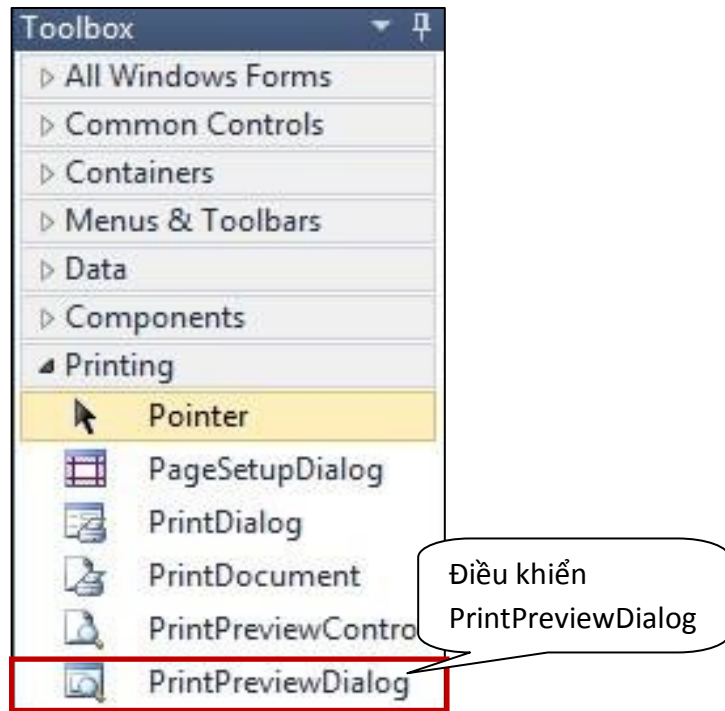
Điều khiển `PrintPreviewDialog` giúp cho người dùng có thể xem trước trang in sẽ trông như thế nào trước khi trang đó được in ra trên giấy. Lý do điều khiển `PrintPreviewDialog` có thể xem trước được trang in như vậy là vì đã gọi phương thức `Print()` của đối tượng lớp `PrintDocument` được thiết lập trong thuộc tính `Document` của `PrintPreviewDialog`. Khi đó, thay vì phương thức `Print()` thực hiện chức năng in trực tiếp trên giấy thì trang in sẽ xuất ra trên một giao diện đồ họa được biểu diễn như một trang giấy của hộp thoại Print preview hình 8.4.



Hình 8.4: Giao diện `PrintPreviewDialog` khi được hiển thị

Trên giao diện hình 8.4 của điều khiển `PrintPreviewDialog`, người dùng có thể lựa chọn trang sẽ xem, phóng to hoặc thu nhỏ trang, lựa chọn chế độ hiển thị trang (1 trang, 2 trang, 3 trang, ...). Sau khi đã xem xong, người dùng có thể in trang hiển thị trên giấy bằng cách nhấn vào biểu tượng máy in  trên hộp thoại Print preview.

Điều khiển `PrintPreviewDialog` nằm trong nhóm Printing của cửa sổ Toolbox như hình 8.5.



Hình 8.5: Điều khiển *PrintPreviewDialog* trong cửa sổ Toolbox

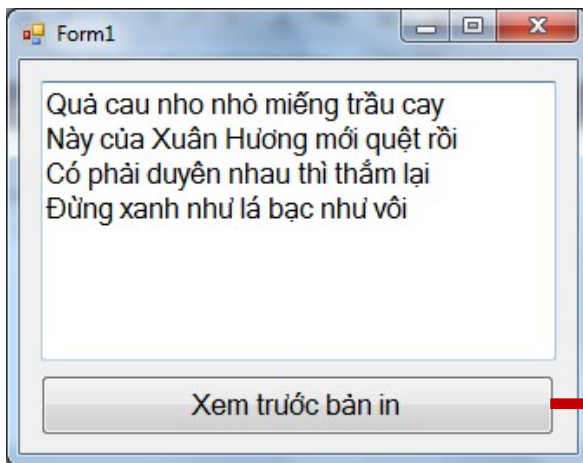
➤ **Hiện thị hộp thoại *PrintPreviewDialog*:**

Hộp thoại *PrintPreviewDialog* hiển thị bằng phương thức *ShowDialog()* hoặc phương thức *Show()*. Tuy nhiên trước khi hộp thoại được hiển thị lập trình viên cần thiết lập thuộc tính *Document* của *PrintPreviewDialog* có giá trị là một đối tượng *PrintDocument*. Lập trình viên có thể tạo đối tượng và hiển thị hộp thoại *PrintPreviewDialog* bằng mã lệnh như sau:

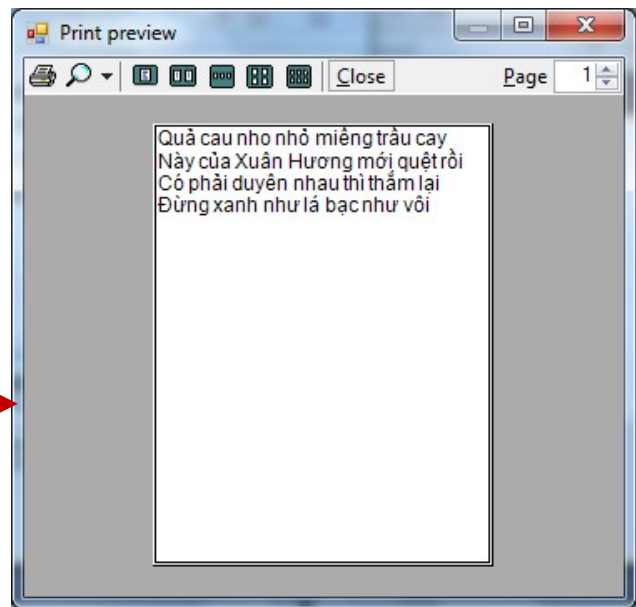
```
private void btnHienThi_Click(object sender, EventArgs e)
{
    PrintPreviewDialog preview = new PrintPreviewDialog ();
    System.Drawing.Printing.PrintDocument printDoc = new
        System.Drawing.Printing.PrintDocument();
    preview.Document = printDoc;
    preview.ShowDialog();
}
```

Ví dụ 8.1: Thiết kế chương trình có giao diện gồm *Button* và *Textbox* như hình 8.6.

Yêu cầu: Người dùng có thể nhập văn bản vào *Textbox* và khi ấn vào *Button* cho phép xem trang văn bản trước khi in như hình 8.7.



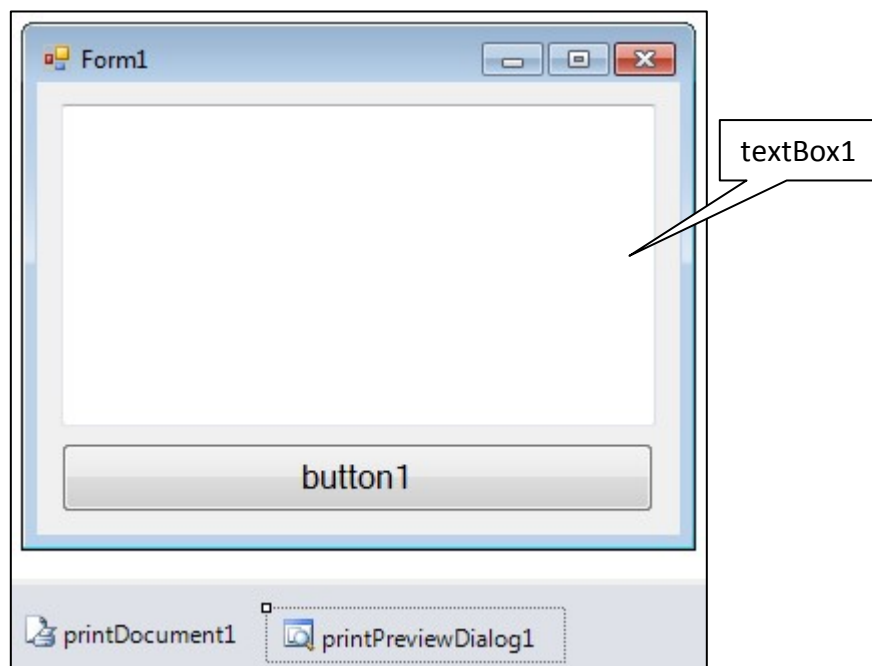
Hình 8.6: Giao diện chương trình nhập văn bản



Hình 8.7: Giao diện hộp thoại xem trang văn bản trước khi in

Hướng dẫn:

Bước 1: Thiết kế giao diện ban đầu: Thêm các điều khiển *Button*, *Textbox*, *PrintDocument* và *PrintPreviewDialog* từ cửa sổ *Toolbox* vào form như hình 8.8.



Hình 8.8: Giao diện ban đầu của form sau khi thêm điều khiển

Bước 2: Thiết lập giá trị thuộc tính cho các điều khiển trong cửa sổ *Properties*.

- `textBox1`:  
Thuộc tính *Name*: `txtNoiDung`  
Thuộc tính *Multiline*: `True`  
Thuộc tính *Size*: 297, 161
- `Button1`:  
Thuộc tính *Name*: `btnXemBanIn`  
Thuộc tính *Text*: “Xem trước bản in”  
Thuộc tính *Size*: 296, 35
- `printPreviewDialog1`:  
Thuộc tính *Document*: `printDocument1`

Bước 3: Viết mã lệnh cho các điều khiển

- Xây dựng hàm `DrawText`:

```
private void DrawText(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
{
    string text = txtNoiDung.Text;
    System.Drawing.Font printFont = new
        System.Drawing.Font("Arial", 35,
        System.Drawing.FontStyle.Regular);
    e.Graphics.DrawString(text, printFont,
        System.Drawing.Brushes.Black, 0, 0);
}
```

- Viết mã lệnh cho sự kiện *Click* nút `btnXemBanIn`:

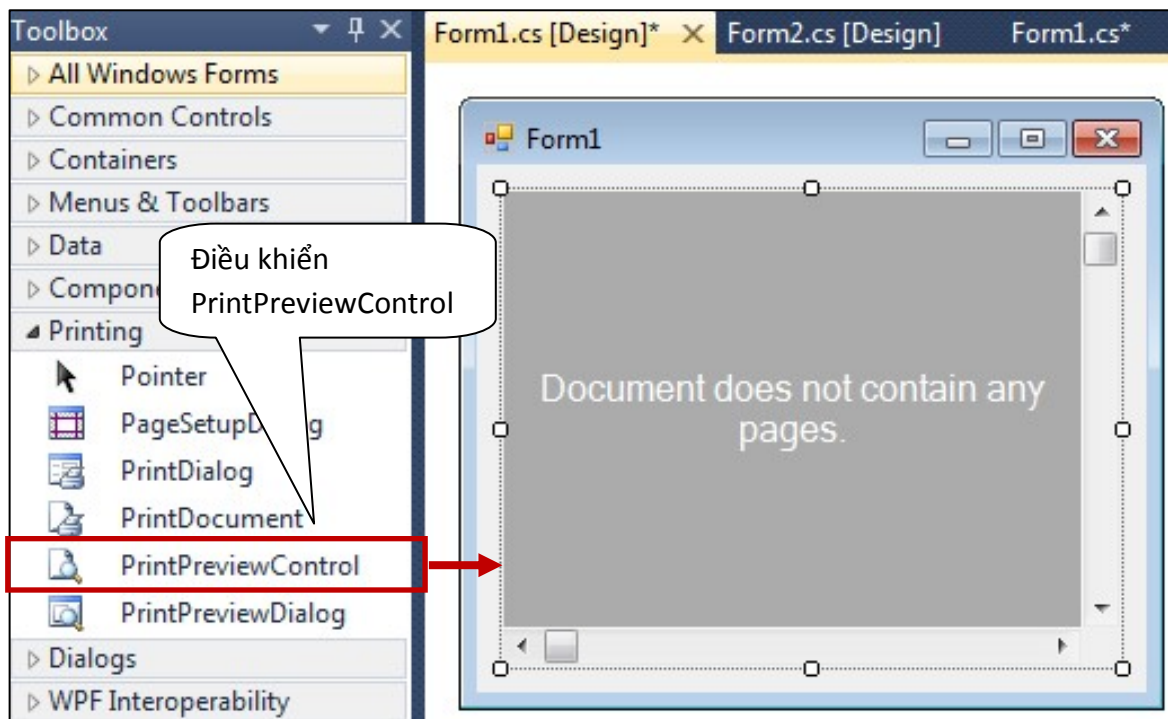
```
private void button1_Click(object sender, EventArgs e)
{
    printDocument1.PrintPage += DrawText;
    printPreviewDialog1.Document = printDocument1;
    printPreviewDialog1.ShowDialog();
}
```

### 8.3. Điều khiển `PrintPreviewControl`

Cũng như điều khiển `PrintPreviewDialog`, điều khiển `PrintPreviewControl` sử dụng để xem trước trang in trước khi in văn bản trên giấy bằng máy in. Tuy nhiên, điểm khác biệt là `PrintPreviewDialog` hiển thị ở dạng hộp thoại độc lập với form, còn `PrintPreviewControl` thì hiển thị trên form, do đó việc hiển thị `PrintPreviewControl`



không cần phải sử dụng phương thức *Show()* hay phương thức *ShowDialog()*. Lập trình viên có thể kéo *PrintPreviewControl* từ cửa sổ Toolbox vào form và bố trí cạnh các điều khiển khác như *TextBox*, *Button*, *Label*, ... Điều khiển *PrintPreviewControl* nằm trong nhóm Printing của cửa sổ Toolbox như hình 8.9.



Hình 8.9: Điều khiển *PrintPreviewControl* trong cửa sổ Toolbox

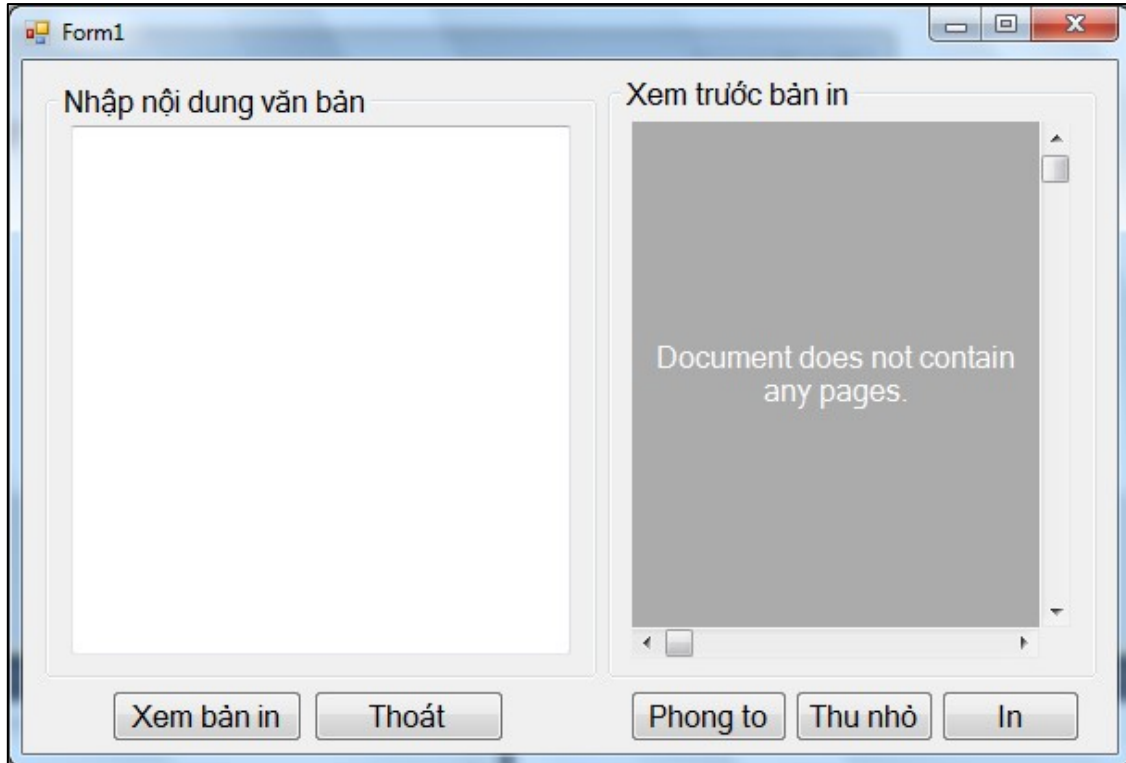
- Một số thuộc tính thường dùng của *PrintPreviewControl*:

Bảng 8.2: Bảng mô tả các thuộc tính của *PrintPreviewControl*

Thuộc tính	Mô tả
<i>AutoZoom</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép trang hiển thị tự động phóng to hay thu nhỏ vừa với điều khiển khi kích thước bị thay đổi.</li> <li>- Nếu là False: Kích thước trang sẽ cố định. Nếu kích thước điều khiển nhỏ hơn kích thước trang sẽ hiển thị thanh trượt.</li> </ul>
<i>Columns</i>	Thiết lập số trang hiển thị theo chiều ngang trên màn hình
<i>Document</i>	Thiết lập tài liệu sẽ được xem trước.

	Lưu ý: Giá trị thiết lập là một đối tượng thuộc lớp PrintDocument
<i>Rows</i>	Thiết lập số trang hiển thị theo chiều dọc trên màn hình.
<i>StartPage</i>	Thiết lập trang sẽ hiển thị đầu tiên khi xem trước trang in.
<i>UseAntiAlias</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép người dùng loại bỏ răng cưa trên trang hiển thị. Giúp trang hiển thị trông mượt hơn. Tuy nhiên, sử dụng chế độ này sẽ làm cho chương trình bị chậm.</li> <li>- Nếu là False: Không sử dụng chế độ loại bỏ răng cưa.</li> </ul>
<i>Zoom</i>	Thiết lập kích thước hiển thị của trang.

Ví dụ 8.2: Viết chương trình có giao diện như hình 8.10. Chương trình bao gồm các điều khiển TextBox, GroupBox, Button, PrintDocument, PrintPreviewControl



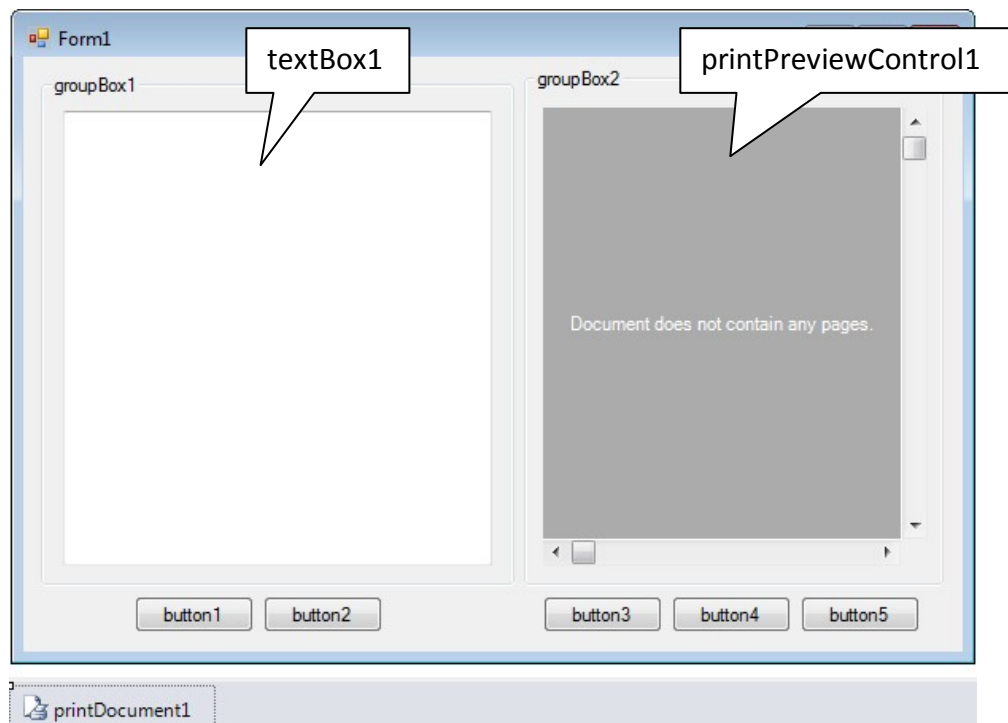
Hình 8.10: Giao diện chương trình xem trước trang in bằng PrintPreviewControl

Yêu cầu:

Người dùng nhập đoạn văn bản vào TextBox. Khi nhấn nút “Xem bản in” thì sẽ hiển thị trước trang in trên PrintPreviewControl. Ngoài ra người dùng có thể sử dụng nút “Phóng to” và nút “Thu nhỏ” để thay đổi kích thước trang hiển thị. Sử dụng nút “In” để in trang hiển thị ra giấy trên máy in.

Hướng dẫn:

Bước 1: Thiết kế giao diện ban đầu: Thêm các điều khiển TextBox, GroupBox, Button, PrintDocument, PrintPreviewControl từ Toolbox vào form như hình 8.11.



Hình 8.11: Giao diện thiết kế form khi thêm điều khiển

Bước 2: Thiết lập giá trị thuộc tính cho các điều khiển trong cửa sổ Properties

- Form1:  
Thuộc tính Size: 607, 412
- groupBox1:  
Thuộc tính Text: “Nhập nội dung văn bản”
- groupBox2:  
Thuộc tính Text: “Xem trước bản in”
- textBox1:  
Thuộc tính Name: txtVanBan  
Thuộc tính Multiline: True

Thuộc tính Size: 270, 286

- printPreviewControl1:

Thuộc tính AutoZoom: True

Thuộc tính Document: printDocument1

- button1:

Thuộc tính Name: btnXemBanIn

Thuộc tính Text: “Xem bản in”

- button2:

Thuộc tính Name: btnThoat

Thuộc tính Text: “Thoát”

- button3:

Thuộc tính Name: btnPhongTo

Thuộc tính Text: “Phóng to”

- button4:

Thuộc tính Name: btnThuNho

Thuộc tính Text: “Thu nhỏ”

- button5:

Thuộc tính Name: btnIn

Thuộc tính Text: “In”

Bước 3: Viết mã lệnh cho các điều khiển

- Xây dựng hàm DrawText:

```
private void DrawText(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
{
    string text = txtVanBan.Text;
    System.Drawing.Font printFont = new
        System.Drawing.Font("Arial", 35,
        System.Drawing.FontStyle.Regular);
    e.Graphics.DrawString(text, printFont,
        System.Drawing.Brushes.Black, 0, 0);
}
```

- Sự kiện Click nút btnXemBanIn:

```
private void btnXemBanIn_Click(object sender, EventArgs e)
{
    PreView();
}
```

- Xây dựng hàm PreView:

```
private void PreView()
{
    printDocument1.PrintPage += new
        System.Drawing.Printing.PrintPageEventHandler(DrawText);
    printPreviewControl1.Document = printDocument1;
}
```

- Sự kiện *Click* nút btnThoat:

```
private void btnThoat_Click(object sender, EventArgs e)
{
    Close();
}
```

- Sự kiện *Click* nút btnPhongTo:

```
private void btnPhongTo_Click(object sender, EventArgs e)
{
    printPreviewControl1.Zoom += 0.01;
    PreView();
}
```

- Sự kiện *Click* nút btnThuNho:

```
private void btnThuNho_Click(object sender, EventArgs e)
{
    printPreviewControl1.Zoom -= 0.01;
    PreView();
}
```

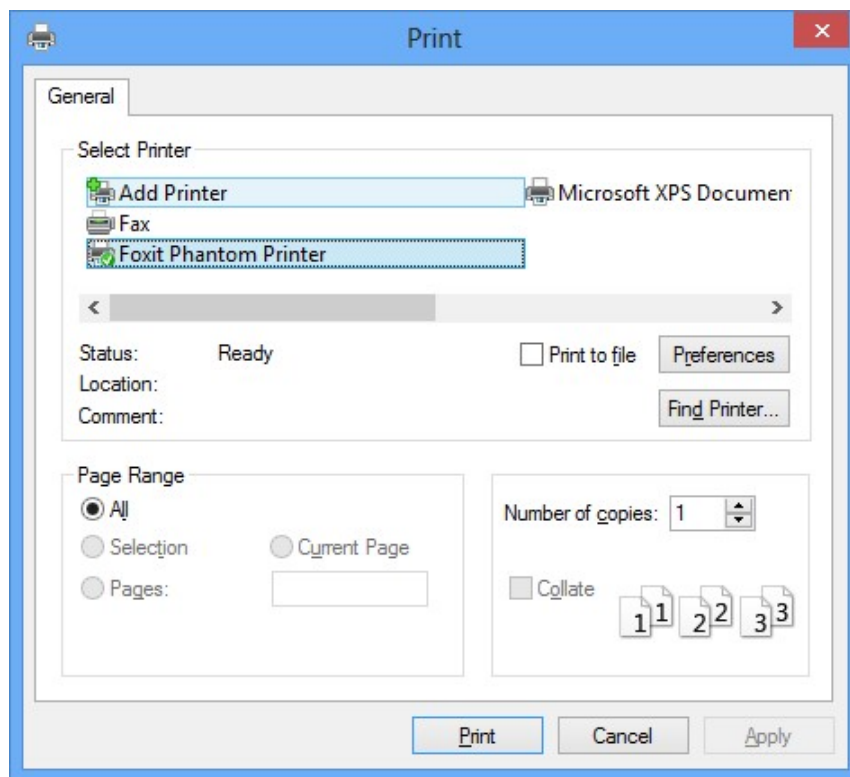
- Sự kiện *Click* nút btnIn:

```
private void btnIn_Click(object sender, EventArgs e)
{
    printDocument1.Print();
}
```

## 8.4. Điều khiển PrintDialog

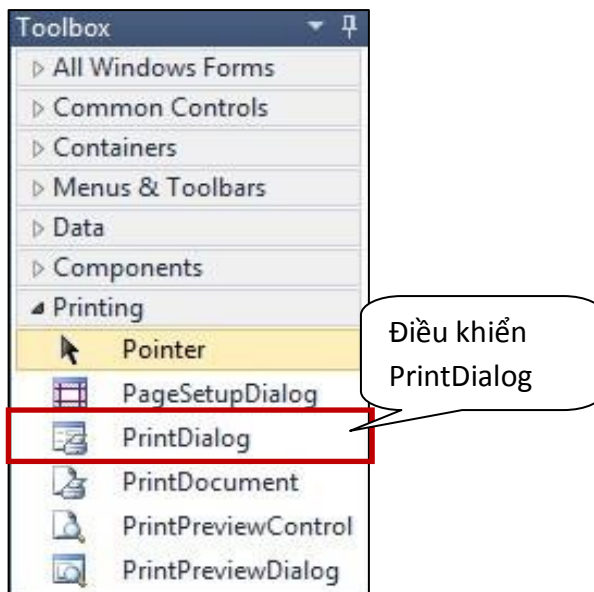
Điều khiển PrintDialog cho phép hiển thị dạng hộp thoại như hình 8.12 để người dùng chọn máy in và in văn bản. Cũng như các điều khiển thực hiện công việc in ấn khác,

hộp thoại `PrintDialog` được hiển thị bằng phương thức `ShowDialog()` nhưng trước khi gọi phương thức `ShowDialog()` lập trình viên cần thiết lập giá trị cho thuộc tính `Document`.



Hình 8.12: Giao diện hộp thoại `PrintDialog`

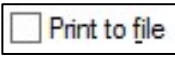
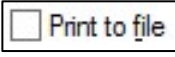
Điều khiển `PrintDialog` nằm trong nhóm `Printing` của cửa sổ `Toolbox` như hình 8.13.

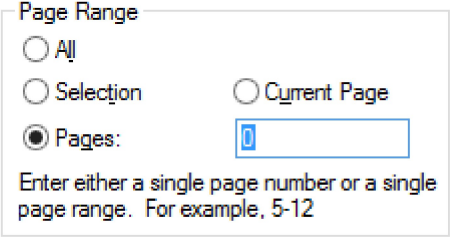


Hình 8.13: Điều khiển `PrintDialog` trong cửa sổ `Toolbox`

- Một số thuộc tính thường dùng của *PrintDialog*:

Bảng 8.2: Bảng mô tả các thuộc tính của *PrintDialog*

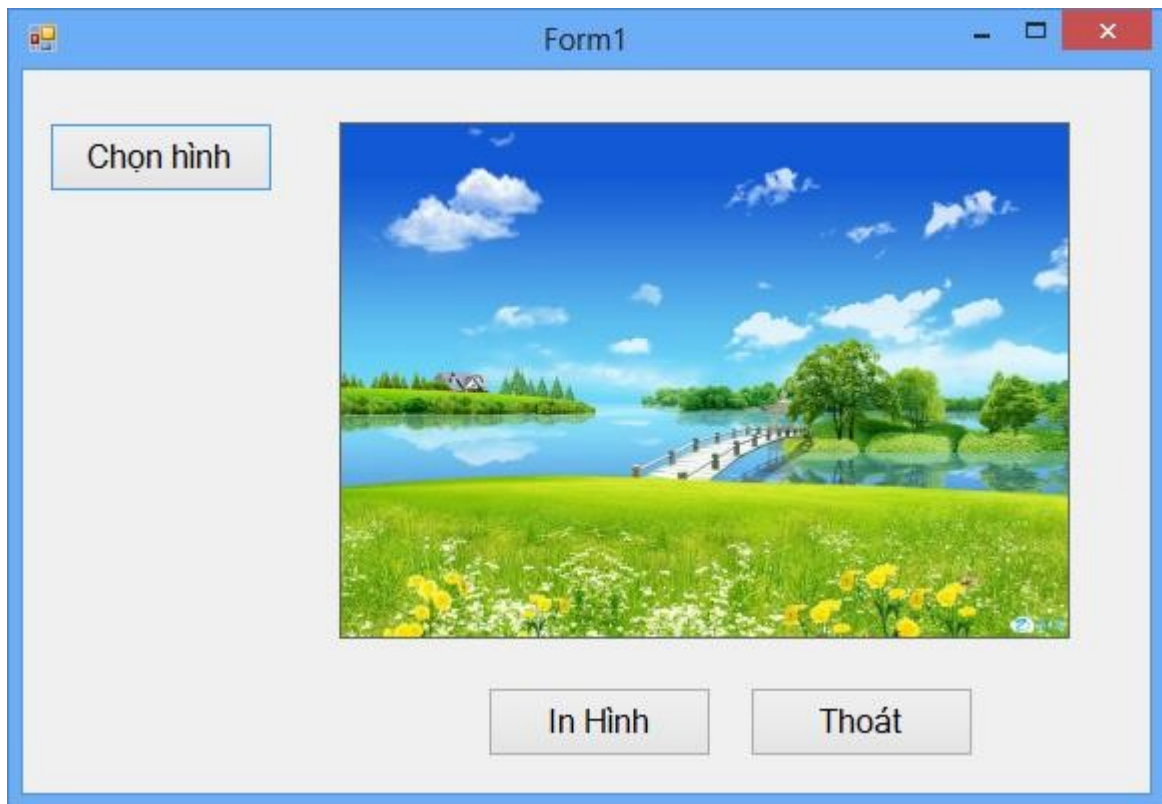
Thuộc tính	Mô tả
<i>AllowCurrentPage</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép chọn trang hiện hành để in.</li> <li>- Nếu là False: Không cho phép in chọn in trang hiện hành.</li> </ul>
<i>AllowPrintToFile</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép hiển thị Checkbox  người dùng có thể in ra thành file thay vì in ra giấy trên máy in</li> <li>- Nếu là False: Không hiển thị Checkbox </li> </ul>
<i>AllowSelection</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép hiển thị RadioButton Selection để người dùng chọn trang sẽ in. <div data-bbox="795 1113 1247 1354" data-label="Image"> </div> </li> <li>- Nếu là False: Không hiển thị RadioButton Selection</li> </ul>
<i>AllowSomePages</i>	Mang hai giá trị True hoặc False. <ul style="list-style-type: none"> <li>- Nếu là True: Cho phép hiển thị RadioButton Pages để người dùng chọn in trang thứ i</li> </ul>

	 <p>- Nếu là False: Không hiển thị RadioButton Pages</p>
<i>Document</i>	<p>Thiết lập tài liệu sẽ được in.          Lưu ý: Giá trị thiết lập là một đối tượng thuộc lớp PrintDocument</p>
<i>PrintToFile</i>	<p>Mang hai giá trị True hoặc False.</p> <ul style="list-style-type: none"> <li>- Nếu là True: Checkbox PrintToFile trên hộp thoại được chọn.</li> <li>- Nếu là False: Checkbox PrintToFile trên hộp thoại không được chọn.</li> </ul>
<i>PrintSettings</i>	<p>Thuộc tính PrintSettings có nhiều tùy chọn bên trong giúp thiết lập các công việc như:</p> <ul style="list-style-type: none"> <li>- PrintSettings.PrinterName = &lt;Tên máy in&gt;: Chọn máy in</li> <li>- PrintSettings.PrintToFile = &lt;True/ False&gt;: Cho phép hoặc không cho phép in ra tập tin.</li> <li>- PrintSettings.FromPage = &lt;Số nguyên&gt;: Thiết lập bắt đầu in từ trang thứ mấy.</li> <li>- PrintSettings.ToPage = &lt;Số nguyên&gt;: Thiết lập trang cuối cùng sẽ in</li> <li>- PrintSettings.Size: Lấy kích cỡ giấy mà máy in hỗ trợ.</li> <li>- PrintSettings.Resolutions: Lấy tất cả độ phân giải mà máy in hỗ trợ.</li> <li>- PrintSettings.Duplex = &lt;True/ False&gt;: Thiết lập chế độ in một mặt hay hai mặt của trang giấy.</li> <li>- PrintSettings.CanDuplex: Trả về giá trị True hoặc False, cho biết máy in có hỗ trợ in hai mặt của</li> </ul>



	trang giấy hay không.
--	-----------------------

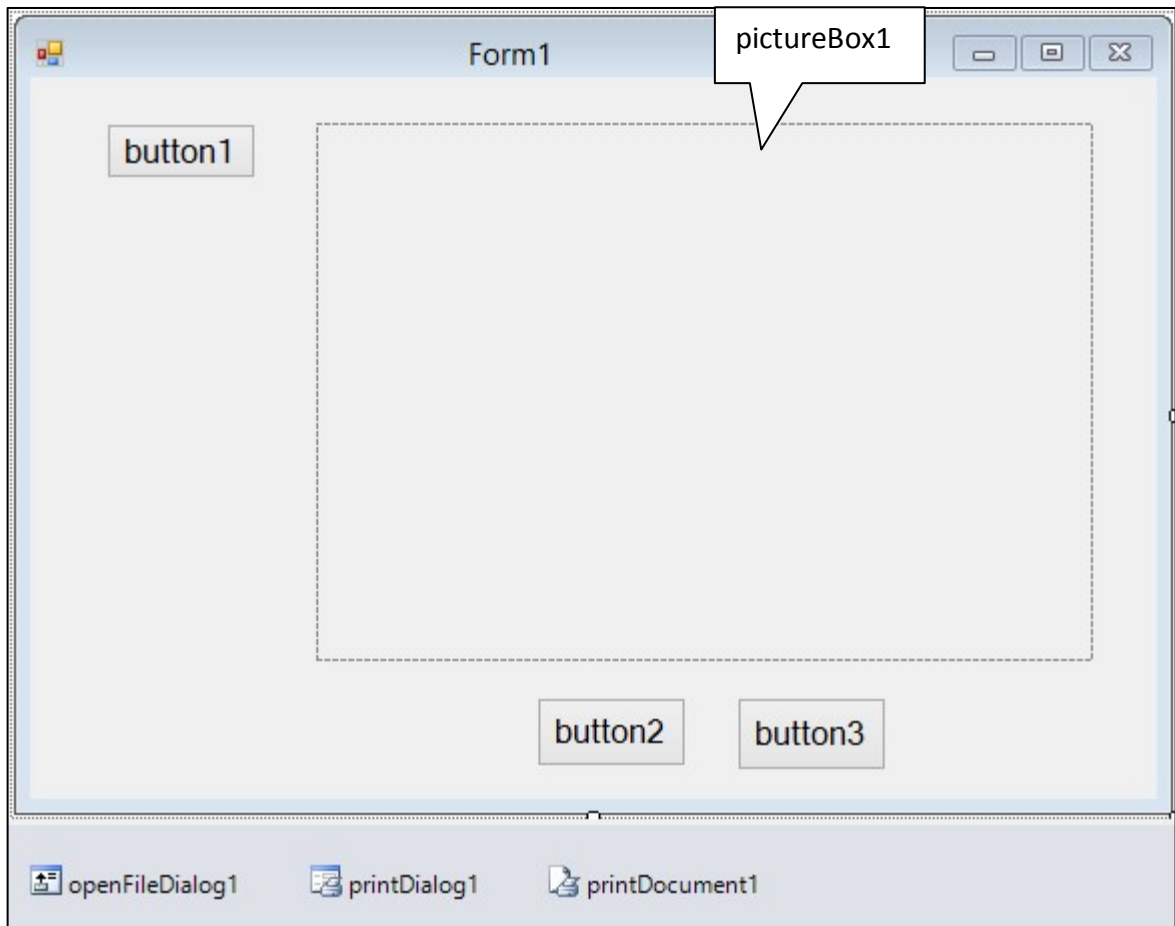
Ví dụ 8.3: Viết chương trình in hình ảnh. Chương trình có giao diện như hình 8.14 cho phép người dùng chọn 1 hình ảnh trên máy tính và hiển thị trên PictureBox. Người dùng có thể sử dụng Button “In Hình” để gọi hộp thoại PrintDialog và in hình ra giấy bằng máy in.



*Hình 8.14: Giao diện chương trình in hình ảnh*

Hướng dẫn:

Bước 1: Thiết kế giao diện ban đầu: Thêm các điều khiển Button, PictureBox, PrintDialog, PrintDocument và OpenFileDialog từ cửa sổ Toolbox và form như hình 8.15.



*Hình 8.15: Giao diện ban đầu form in hình ảnh*

Bước 2: Thiết lập giá trị thuộc tính cho điều khiển trong cửa sổ Properties.

- button1:  
Thuộc tính *Name*: btnChonHinh  
Thuộc tính *Text*: “Chọn hình”
- button2:  
Thuộc tính *Name*: btnIn  
Thuộc tính *Text*: “In hình”
- button3:  
Thuộc tính *Name*: btnThoat  
Thuộc tính *Text*: “Thoát”
- pictureBox1:  
Thuộc tính *BorderStyle*: FixedSingle  
Thuộc tính *SizeMode*: StretchImage

- openFileDialog1:  
Thuộc tính *Filter*: “jpg images|\*.jpg”
- PrintDialog:  
Thuộc tính *AllowCurrentPage*: True  
Thuộc tính *AllowPrintToFile*: True  
Thuộc tính *AllowSelection*: True  
Thuộc tính *AllowSomePages*: True  
Thuộc tính *Document*: printDocument1  
Thuộc tính *PrintToFile*: True  
Thuộc tính *ShowHelp*: True  
Thuộc tính *ShowNetwork*: True

Bước 3: Viết mã lệnh cho các điều khiển

- Viết hàm printPicture\_PrintPage:

```
void printPicture_PrintPage(object sender, PrintPageEventArgs
e)
{
    try
    {
        if (pictureBox1.Image != null)
        {
            Bitmap bitmap = new Bitmap(pictureBox1.Image);
            if (bitmap != null)
            {
                e.Graphics.DrawImage(bitmap, 10, 10);
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
```

- Sự kiện *Click* nút btnThoat:

```
private void btnThoat_Click(object sender, EventArgs e)
{
    Close();
}
```

- Sự kiện *Click* nút btnChonHinh:

```
private void btnChonHinh_Click(object sender, EventArgs e)
{
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
        pictureBox1.ImageLocation = openFileDialog1.FileName;
    }
}
```

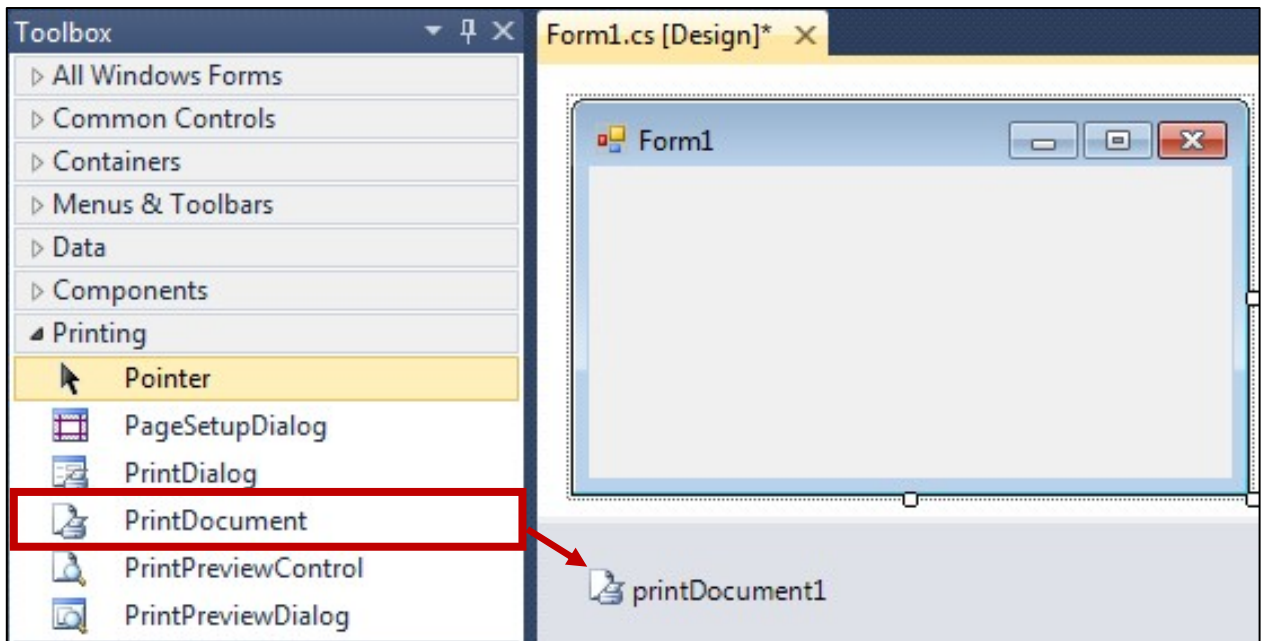
- Sự kiện *Click* nút btnIn:

```
private void btnIn_Click(object sender, EventArgs e)
{
    printDocument1.PrintPage += new
        PrintPageEventHandler(printPicture_PrintPage);
    printDialog1.Document = printDocument1;
    if (printDialog1.ShowDialog() == DialogResult.OK)
    {
        printDocument1.PrinterSettings =
            printDialog1.PrinterSettings;
        printDocument1.Print();
    }
}
```

## 8.5. Điều khiển PrintDocument

Lớp PrintDocument có không gian tên là System.Drawing.Printing. PrintDocument là lớp thường xuyên lớp nắm giữ nội dung cần in ra tập tin hay máy in, do đó PrintDocument được sử dụng nhiều trong việc in ấn.

Đối tượng tạo ra từ PrintDocument được sử dụng cùng với các điều khiển PrinDialog, PrintPreviewDialog, PrintPreviewControl, ... Lập trình viên có thể tạo ra đối tượng PrintDocument bằng cách kéo điều khiển PrinDocument từ cửa sổ Toolbox vào form như hình 8.16.



Hình 8.16: Điều khiển PrintDocument trong cửa sổ Toolbox

Hoặc có thể tạo đối tượng PrintDocument bằng mã lệnh như sau:

```
PrintDocument prtDC= new PrintDocument();
```

➤ Sử dụng điều khiển PrintDocument:

Trong công việc in ấn, để in nội dung mà PrintDocument nắm giữ ra máy in hoặc ra tập tin, lập trình viên sử dụng phương thức Print() của lớp PrintDocument:

```
prtDC.Print();
```

Khi gọi phương thức Print thì sự kiện PrintPage được phát sinh để tiến hành vẽ các nội dung PrintDocument nắm giữ trên các trang giấy. Như vậy, lập trình viên cần phải định nghĩa các công việc cần phải làm khi sự kiện PrintPage phát sinh, thông thường công việc này là vẽ các nội dung PrintDocument nắm giữ lên trang giấy. Để đơn giản lập trình viên sẽ xây dựng một hàm độc lập thực hiện công việc vẽ này và gọi hàm thực hiện trong sự kiện PrintPage.

- Hàm vẽ văn bản:

```
private void DrawText(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
{
    string text = "Khoa Công nghệ thông tin";
    System.Drawing.Font printFont = new
        System.Drawing.Font("Arial", 35,
        System.Drawing.FontStyle.Regular);
    e.Graphics.DrawString(text, printFont,
        System.Drawing.Brushes.Black, 0, 0);
}
```

- Hàm vẽ hình chứa trong pictureBox1 ra trang giấy:

```
private void DrawPicture(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
{
    try
    {
        if (pictureBox1.Image != null)
        {
            Bitmap bitmap = new Bitmap(pictureBox1.Image);
            if (bitmap != null)
            {
                e.Graphics.DrawImage(bitmap, 10, 10);
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
```

- Thực hiện gọi hàm vẽ văn bản hoặc vẽ hình ảnh lên trang giấy in hoặc lên tập tin trong sự kiện PrintPage:

```
private void btnIn_Click(object sender, EventArgs e)
{
    //khai báo đối tượng lớp PrintDocument
    PrintDocument printDC = new PrintDocument();
    //Sử dụng hàm vẽ văn bản trong sự kiện PrintPage. Nếu muốn vẽ
    //hình thì thay Drawtext bằng DrawPicture
    printDC.PrintPage += new
        PrintPageEventHandler(DrawText);
    //Khai báo đối tượng lớp PrintPreviewDialog
    PrintPreviewDialog printPrev = new PrintPreviewDialog();
    printPrev.Document = printDC;
    //Hiển thị hộp thoại PrintPreviewDialog để xem trước trang in
    if (printPrev.ShowDialog() == DialogResult.OK)
    {
        //Nếu người dùng nhấn nút in thì phương thức Print sẽ
        //thực hiện và khi đó sẽ phát sinh sự kiện PrintPage thực
        //hiện công việc vẽ văn bản hoặc vẽ hình ảnh
        printDC.Print();
    }
}
```

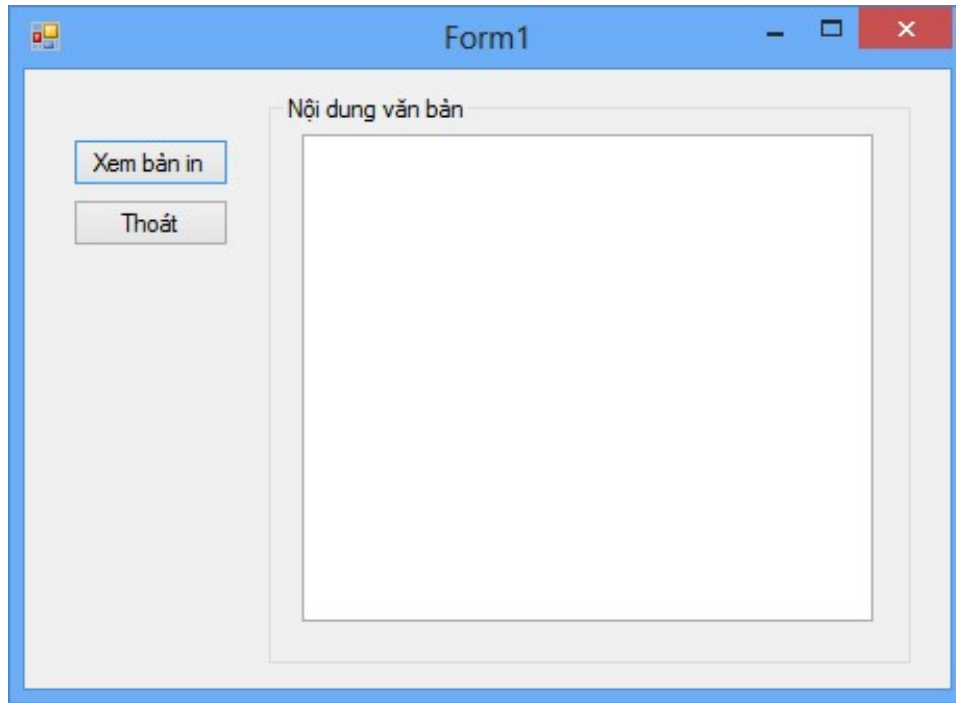
➤ In nội dung trên nhiều trang:

Có vấn đề phát sinh với việc in ấn khi nội dung cần in quá nhiều, nhiều hơn kích thước của một trang giấy và nếu xử lý bằng những các câu lệnh trên thì người dùng sẽ chỉ in được một trang. những nội dung vượt quá kích thước trang sẽ bị mất đi. Vấn đề này xảy ra bởi vì PrintDocument không có ký tự ngắt dòng và cũng không thể tự động ngắt trang. Do đó, cần phải viết mã lệnh thực hiện các công việc ngắt dòng và ngắt trang này.

Để có thể ngắt dòng in, lập trình viên cần biết được kích thước dòng và tính ra số ký tự tối đa có thể in trên một dòng của trang. Tương tự, muốn ngắt trang thì lập trình viên cũng phải tính toán được kích thước trang từ đó lấy ra thông tin số dòng có thể in trên một trang giấy.

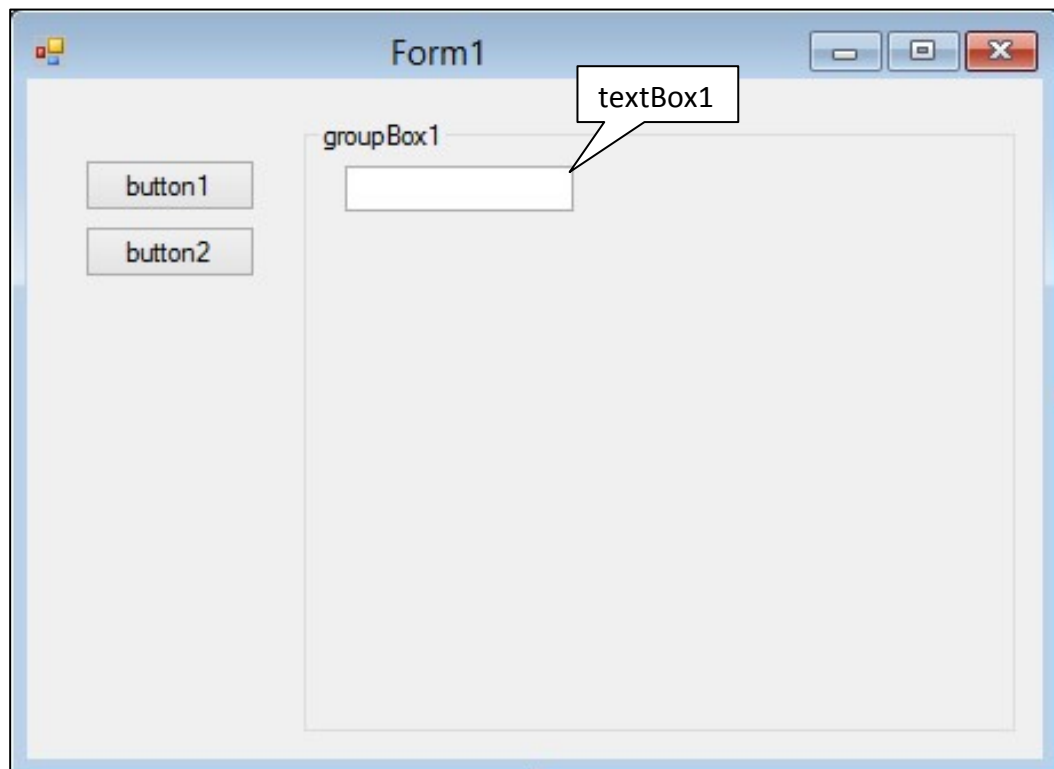
Ví dụ 8.4: Viết chương trình in văn bản có giao diện như hình 8.17. Chương trình gồm các điều khiển Button, TextBox, GroupBox.

Yêu cầu: Người dùng nhập văn bản vào TextBox. Khi nhấn nút “Xem bản in” sẽ hiển thị trước trang in. Trang hiển thị sẽ tự động ngắt dòng xuống hàng và sang trang mới nếu số lượng ký tự của văn bản vượt kích thước trang in.



*Hình 8.17: Giao diện chương trình in văn bản*

Bước 1: Thiết kế giao diện ban đầu cho điều khiển: Thêm các điều khiển Button, TextBox, GroupBox trong cửa sổ Toolbox vào form như hình 8.18.



*Hình 8.18: Giao diện form sau khi thêm điều khiển*



Bước 2: Thiết lập giá trị thuộc tính cho điều khiển trong cửa sổ Properties.

- button1:  
Thuộc tính Name: btnIn  
Thuộc tính Text: “Xem bản in”
- button2:  
Thuộc tính Name: btnThoat  
Thuộc tính Text: “Thoát”
- textBox1:  
Thuộc tính Name: txtVanBan  
Thuộc tính Multiline: True  
Thuộc tính Size: 274, 234
- groupBox1:  
Thuộc tính Text: “Nội dung văn bản”  
Thuộc tính Size: 308, 273

Bước 3: Viết mã lệnh cho các điều khiển

- Khai báo biến:

```
string stringtoPrint = "";
```

- Sự kiện Click nút btnIn:

```
private void btnIn_Click(object sender, EventArgs e)
{
    stringtoPrint = txtVanBan.Text;
    PrintDocument printDC = new PrintDocument();
    printDC.PrintPage += new PrintPageEventHandler(DrawText);
    PrintPreviewDialog printPrev = new PrintPreviewDialog();
    printPrev.Document = printDC;
    if (printPrev.ShowDialog() == DialogResult.OK)
    {
        printDC .Print();
    }
}
```

- Sự kiện Click nút btnThoat:

```
private void btnThoat_Click(object sender, EventArgs e)
{
    Close();
}
```

- Mã lệnh hàm DrawText:

```
private void DrawText(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
{
    int sokyту, sodong;
    string chuoiin;
    //khai báo font chữ
    System.Drawing.Font printFont = new
        System.Drawing.Font("Arial", 35,
        System.Drawing.FontStyle.Regular);
    //khai báo chuỗi định dạng
    StringFormat chuoidinhđang=new StringFormat();
    //khai báo vùng sẽ in trên trang
    System.Drawing.RectangleF vungin = new
        RectangleF(e.MarginBounds.Left,
        e.MarginBounds.Top, e.MarginBounds.Width,
        e.MarginBounds.Height);
    //khai báo kích thước
    SizeF kichthuoc = new SizeF(e.MarginBounds.Width,
        e.MarginBounds.Height - printFont.GetHeight(e.Graphics));
    chuoidinhđang.Trimming = StringTrimming.Word;
    //Sử dụng phương thức MeasureString để lấy số ký tự trên một
    //dòng và số lượng dòng văn bản
    e.Graphics.MeasureString(stringtoPrint, printFont, kichthuoc,
        chuoidinhđang,out sokyту,out sodong);
    //Trích các ký tự sẽ hiển thị trên một dòng
    chuoiin = stringtoPrint.Substring(0, sokyту);
    e.Graphics.DrawString(chuoiin, printFont,
        Brushes.Black, vungin, chuoidinhđang);
    if (sokyту < stringtoPrint.Length)
    {
        stringtoPrint = stringtoPrint.Substring(sokyту);
        e.HasMorePages = true;
    }
    else
    {
        e.HasMorePages = false;
        stringtoPrint = txtVanBan.Text;
    }
}
```