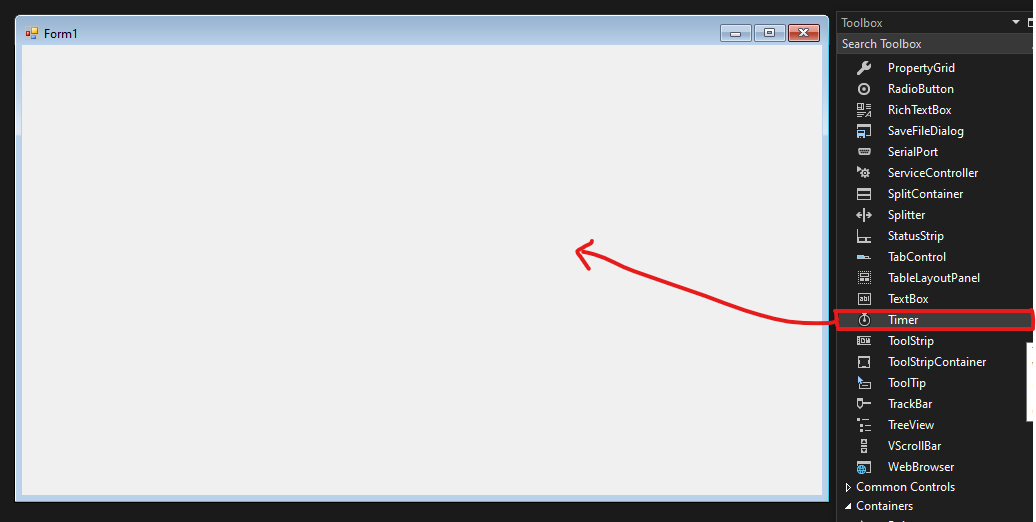
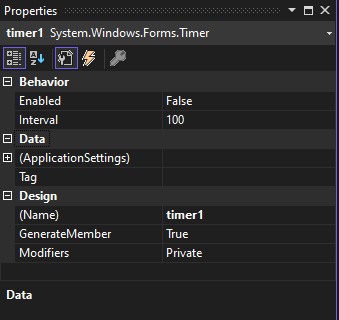
5.

"*Timer*" là một điều khiển khá hữu ích trong C# - Winform. Timer tương tự như "setInterval" hay "setTimeOut" trong JavaScript đều có chức năng thực hiện một số chức năng nhất định trong các khoảng thời gian lặp lại. Một đoạn chương trình sẽ được lặp đi lặp lại với khoảng nhất định mà không ảnh hưởng đến các tiến trình khác.  
Có nhiều loại Timer: System.Windows.Forms.Timer, System.Timers.Timer, System.Threading.Timer. Nhưng để đơn giản bây giờ chúng ta sẽ sử dụng điều khiển Timer của Windows.Forms.

Để khai báo 1 đối tượng Timer cách đơn giản là vào Toolbox tìm "Timer" rồi kéo vào Form



Đặt các thuộc tính cho Timer



Trong đó Interval là khoảng thời gian sẽ lặp lại, đơn vị tính miliseconds.  
  
Sự kiện duy nhất của Timer là Timer\_Tick cho phép ta thay đổi sự kiện mỗi khi timer lặp lại thời gian. Tức là khi timer lặp lại thì code trong hàm Timer\_Tick sẽ được thực hiện.

6

Tạo một RichTextBox

Chúng ta có thể tạo điều khiển RichTextBox bằng cách sử dụng trình thiết kế Biểu mẫu tại thời điểm thiết kế hoặc sử dụng lớp RichTextBox trong mã vào thời gian chạy.

Để tạo điều khiển RichTextBox tại thời điểm thiết kế, bạn chỉ cần kéo và thả điều khiển RichTextBox từ Hộp công cụ vào Biểu mẫu trong Visual Studio. Khi RichTextBox được thêm vào Biểu mẫu, bạn có thể di chuyển nó xung quanh và thay đổi kích thước bằng chuột cũng như đặt các thuộc tính và sự kiện cho nó.

Tạo điều khiển RichTextBox trong thời gian chạy chỉ là công việc tạo một phiên bản của lớp RichTextBox, thiết lập các thuộc tính của nó và thêm đối tượng RichTextBox vào bộ sưu tập Điều khiển của Biểu mẫu.

Bước đầu tiên để tạo RichTextBox động là tạo một phiên bản của lớp RichTextBox. Đoạn mã sau đây tạo một đối tượng điều khiển RichTextBox.

1. // Tạo đối tượng RichTextBox
3. RichTextBox DynamicRichTextBox =  RichTextBox **mới** ();

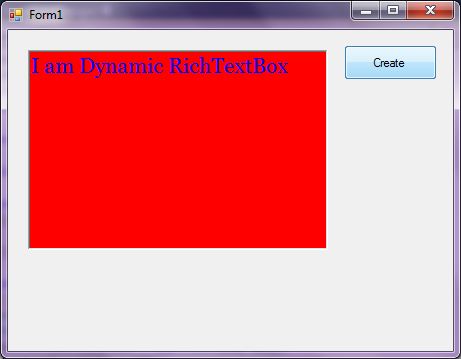
Trong bước tiếp theo, bạn có thể đặt thuộc tính của điều khiển RichTextBox. Đoạn mã sau đặt các thuộc tính kích thước, vị trí, màu nền, màu nền trước, Văn bản, Tên và Phông chữ của RichTextBox.

1. DynamicRichTextBox.Location =  Điểm **mới** (20, 20);
2. DynamicRichTextBox.Width = 300;
3. DynamicRichTextBox.Height = 200;
4. // Đặt nền và tiền cảnh
5. DynamicRichTextBox.BackColor = Color.Red;
6. DynamicRichTextBox.ForeColor = Color.Blue;
7. DynamicRichTextBox.Text =  "Tôi là Dynamic RichTextBox" ;
8. DynamicRichTextBox.Name =  "DynamicRichTextBox" ;
9. DynamicRichTextBox.Font =  Phông chữ **mới** ( "Georgia" , 16);

Khi điều khiển RichTextBox đã sẵn sàng với các thuộc tính của nó, bước tiếp theo là thêm điều khiển RichTextBox vào Biểu mẫu. Để làm như vậy, chúng tôi sử dụng phương thức Form.Controls.Add. Đoạn mã sau đây thêm điều khiển RichTextBox vào Biểu mẫu hiện tại.

1. Controls.Add(dynamicRichTextBox);

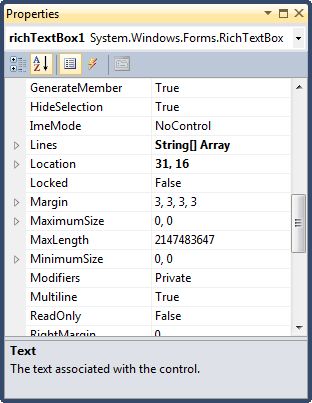
Điều khiển RichTextBox trông giống như Hình .



Đặt thuộc tính RichTextBox

Sau khi bạn đặt điều khiển RichTextBox trên Biểu mẫu, bước tiếp theo là đặt thuộc tính.

Cách dễ nhất để thiết lập thuộc tính là từ Cửa sổ Thuộc tính. Bạn có thể mở cửa sổ Thuộc tính bằng cách nhấn F4 hoặc nhấp chuột phải vào điều khiển và chọn mục menu Thuộc tính. Cửa sổ Properties trông giống như Hình 2.



Hình 2

**Vị trí, Chiều cao, Chiều rộng và Kích thước**

Thuộc tính Vị trí lấy một Điểm chỉ định vị trí bắt đầu của RichTextBox trên Biểu mẫu. Thuộc tính Size chỉ định kích thước của điều khiển. Chúng ta cũng có thể sử dụng thuộc tính Chiều rộng và Chiều cao thay vì thuộc tính Kích thước. Đoạn mã sau đặt các thuộc tính Vị trí, Chiều rộng và Chiều cao của điều khiển RichTextBox.

1. DynamicRichTextBox.Location =  Điểm **mới** (20, 20);
2. DynamicRichTextBox.Width = 300;
3. DynamicRichTextBox.Height = 200;

**Nền, Tiền cảnh, BorderStyle**

Thuộc tính BackColor và ForeColor được sử dụng để đặt màu nền và màu nền trước của RichTextBox tương ứng. Nếu bạn bấm vào các thuộc tính này trong cửa sổ Thuộc tính, Hộp thoại Màu sẽ bật lên.

Ngoài ra, bạn có thể đặt màu nền và màu nền trước trong thời gian chạy. Đoạn mã sau thiết lập các thuộc tính BackColor và ForeColor.

1. // Đặt nền và tiền cảnh
2. DynamicRichTextBox.BackColor = Color.Red;
3. DynamicRichTextBox.ForeColor = Color.Blue;

Bạn cũng có thể đặt kiểu đường viền của RichTextBox bằng cách sử dụng thuộc tính BorderStyle. Thuộc tính BorderStyle được biểu thị bằng bảng liệt kê BorderStyle có ba giá trị – FixedSingle, Fix3D và None. Giá trị mặc định của kiểu đường viền là Fix3D. Đoạn mã sau đây đặt kiểu đường viền của RichTextBox thành FixedSingle.

1. DynamicRichTextBox.BorderStyle = BorderStyle.FixedSingle;

**Tên**

Thuộc tính Name đại diện cho tên duy nhất của điều khiển RichTextBox. Nó được sử dụng để truy cập điều khiển trong mã. Đoạn mã sau đặt và lấy tên cũng như văn bản của điều khiển RichTextBox.

1. DynamicRichTextBox.Name =  "DynamicRichTextBox" ;

**Văn bản và Độ dài văn bản**

Thuộc tính Text của RichTextBox biểu thị văn bản hiện tại của điều khiển RichTextBox. Thuộc tính TextLength trả về độ dài của nội dung RichTextBox.

Đoạn mã sau đây đặt thuộc tính Text và TextAlign và lấy kích thước của điều khiển RichTextBox.

1. DynamicRichTextBox.Text =  "Tôi là Dynamic RichTextBox" ;
2. **kích thước int** = DynamicRichTextBox.TextLength;

**Nối văn bản**

Một cách để nối văn bản vào RichTextBox chỉ đơn giản là đặt thuộc tính Text thành văn bản hiện tại cộng với văn bản mới mà bạn muốn nối thêm nội dung như thế này.

1. RichTextBox1.Text +=  " Văn bản được thêm vào" ;

RichTextBox cũng có phương thức AppendText để thực hiện tương tự. Phương thức *AppendText* nối thêm văn bản vào cuối RichTextBox. Đoạn mã sau sử dụng phương thức AppendText để nối văn bản vào nội dung RichTextBox1.

1. RichTextBox1.AppendText( " Văn bản được nối thêm" );

**Chấp nhậnTab**

Nếu điều khiển RichTextBox được đặt thành nhiều dòng, thuộc tính AcceptsTab được sử dụng để đặt điều khiển RichTextBox chấp nhận khóa TAB dưới dạng văn bản. Nếu thuộc tính này không được đặt, việc nhấn phím TAB chỉ cần di chuyển tới điều khiển tiếp theo trên Biểu mẫu. Theo mặc định, giá trị thuộc tính AcceptsTab của điều khiển RichTextBox là sai.

1. // chấp nhận phím TAB
2. DynamicRichTextBox.AcceptsTab =  **true**;

**WordWrap**

Nếu thuộc tính WordWrap là đúng, văn bản trong điều khiển RichTextBox sẽ tự động ngắt dòng sang dòng tiếp theo nếu được yêu cầu. Nếu thuộc tính này được đặt thành true thì thanh cuộn ngang sẽ không được hiển thị bất kể cài đặt thuộc tính ScrollBars.

1. // Gói từ
2. DynamicRichTextBox.WordWrap =  **true**;

**Thanh cuộn**

Điều khiển RichTextBox nhiều dòng có thể có thanh cuộn. Thuộc tính ScrollBars của điều khiển RichTextBox được sử dụng để hiển thị thanh cuộn trên điều khiển. Thuộc tính ScrollBars được biểu thị bằng bảng liệt kê RichTextBoxScrollBars có bốn giá trị – Cả hai, Dọc, Ngang và Không.

Đoạn mã sau đây làm cho cả thanh cuộn dọc và thanh cuộn ngang hoạt động trên điều khiển RichTextBox và chúng sẽ hiển thị khi cần cuộn trên điều khiển RichTextBox.

1. DynamicRichTextBox.ScrollBars = RichTextBoxScrollBars.Both;

**Nét chữ**

Thuộc tính phông chữ đại diện cho phông chữ của văn bản của điều khiển RichTextBox. Nếu kích vào thuộc tính Font trong cửa sổ Properties, bạn sẽ thấy tên Font, kích thước và các tùy chọn font khác. Đoạn mã sau đặt thuộc tính Phông chữ trong thời gian chạy.

1. DynamicRichTextBox.Font =  Phông chữ **mới** ( "Georgia" , 16);

**Chiều dài tối đa**

Bạn có thể hạn chế số lượng ký tự trong điều khiển RichTextBox bằng cách đặt thuộc tính MaxLength. Đoạn mã sau đặt độ dài tối đa của RichTextBox thành 50 ký tự.

1. DynamicRichTextBox.ReadOnly =  **true**;
2. DynamicRichTextBox.MaxLength = 50;

**Chỉ đọc**

Bạn có thể đặt điều khiển RichTextBox ở chế độ chỉ đọc (không thể chỉnh sửa) bằng cách đặt thuộc tính ReadOnly thành true. Đoạn mã sau đặt thuộc tính ReadOnly thành true.

1. DynamicRichTextBox.ReadOnly =  **true**;

**Kích hoạt và vô hiệu hóa phím tắt**

Thuộc tính Phím tắtEnabled của RichTextBox được sử dụng để bật hoặc tắt phím tắt. Theo mặc định, các phím tắt được bật. Đoạn mã sau đây vô hiệu hóa các phím tắt trong RichTextBox.

1. DynamicRichTextBox.ShortcutsEnabled =  **false**;

Thuộc tính Phím tắtEnabled áp dụng cho các tổ hợp phím tắt sau:

* CTRL+Z
* CTRL+E
* CTRL+C
* CTRL+Y
* CTRL+X
* CTRL+BACKSPACE
* CTRL+V
* CTRL+XÓA
* CTRL+A
* SHIFT+XÓA
* CTRL+L
* SHIFT+CHÈN
* CTRL+R

**Đọc nội dung RichTextBox**

Cách đơn giản nhất để đọc nội dung điều khiển RichTextBox là sử dụng thuộc tính Text. Tuy nhiên, xin lưu ý rằng thuộc tính Văn bản không có định dạng; nó chỉ có văn bản. Xem thuộc tính Rtf để biết văn bản bao gồm cả định dạng. Đoạn mã sau đọc nội dung của RichTextBox dưới dạng chuỗi.

1. **chuỗi** RichTextBoxContents = DynamicRichTextBox.Text;

Trong RichTextBox nhiều dòng, nếu nội dung RichTextBox được phân tách bằng nhiều dòng và bạn muốn đọc nội dung của RichTextBox theo từng dòng, bạn có thể sử dụng thuộc tính Lines của RichTextBox. Thuộc tính Lines trả về một mảng các chuỗi trong đó mỗi phần tử của mảng được trả về là một dòng.

Đoạn mã sau đọc từng dòng nội dung RichTextBox.

1. **chuỗi** [] RichTextBoxLines = DynamicRichTextBox.Lines;
2. **foreach** (  dòng  **chuỗi trong** RichTextBoxLines)
3. {
4. MessageBox.Show(line);
5. }

**Lựa chọn trong RichTextBox**

Thuộc tính SelectedText trả về văn bản đã chọn trong điều khiển RichTextBox.

1. **chuỗi** đã chọnText = DynamicRichTextBox.SelectedText;

Bạn cũng có thể sử dụng các thuộc tính SelectionStart và SelectionLength để lấy và đặt văn bản đã chọn trong RichTextBox. Thuộc tính SelectionStart biểu thị chỉ mục bắt đầu của văn bản đã chọn và thuộc tính SelectionLength biểu thị số lượng ký tự được chọn sau ký tự bắt đầu. Đoạn mã sau thiết lập lựa chọn trên RichTextBox.

1. DynamicRichTextBox.SelectionStart = 10;
2. DynamicRichTextBox.SelectionLength = 20;

**Xóa, chọn tất cả và bỏ chọn tất cả**

Phương thức Clear xóa nội dung của RichTextBox. Đoạn mã sau sử dụng phương thức Clear để xóa nội dung của RichTextBox.

1. RichTextBox1.Clear();

Lớp RichTextBox cung cấp các phương thức SelectAll và DeselectAll để chọn và bỏ chọn tất cả văn bản của điều khiển RichTextBox. Đoạn mã sau đây cho biết cách sử dụng các phương thức SelectAll và DeselectAll.

1. **khoảng trống riêng tư** selectAllToolStripMenuItem\_Click( người gửi **đối tượng** , EventArgs e)
2. {
3. **nếu** (RichTextBox1.TextLength > 0)
4. RichTextBox1.SelectAll();
5. }
7. **void void** deselectAllToolStripMenuItem\_Click( người gửi **đối tượng** , EventArgs e)
8. {
9. **nếu** (RichTextBox1.TextLength > 0)
10. RichTextBox1.DeselectAll();
11. }

**Cắt, Sao chép, Dán, Hoàn tác các thao tác trong RichTextBox**

Lớp RichTextBox cung cấp các phương thức Cắt, Sao chép, Dán và Hoàn tác để cắt, sao chép, dán và hoàn tác các thao tác trong bảng tạm. Đoạn mã sau đây cho thấy cách sử dụng các phương pháp Cắt, Sao chép, Dán và Hoàn tác.

1. **riêng tư void** cutToolStripMenuItem\_Click( người gửi **đối tượng** , EventArgs e)
2. {
3. **if** (RichTextBox1.SelectionLength > 0)
4. RichTextBox1.Cut();
5. }
7. **void void** copyToolStripMenuItem\_Click( người gửi **đối tượng** , EventArgs e)
8. {
9. **if** (RichTextBox1.SelectionLength > 0)
10. RichTextBox1.Copy();
11. }
13. **void void** dánToolStripMenuItem\_Click( người gửi **đối tượng** , EventArgs e)
14. {
15. **if** (Clipboard.GetDataObject().GetDataPresent(DataFormats.Text))
16. {
17. RichTextBox1.Paste();
18. }
19. }
21. **void void** undoToolStripMenuItem\_Click( người gửi **đối tượng** , EventArgs e)
22. {
23. **nếu** (RichTextBox1.CanUndo)
24. {
25. RichTextBox1.Undo();
26. RichTextBox1.ClearUndo();
27. }
28. }

**Tải và lưu tệp RTF**

Phương thức LoadFile của điều khiển RichTextBox được sử dụng để tải tệp RTF và hiển thị nội dung của nó. Phương thức SaveFile được sử dụng để lưu nội dung của RichTextBox vào tệp RTF. Đoạn mã sau tải tệp RTF bằng OpenFileDialog và lưu lại nội dung của nó.

1. **void void** LoadRTFButton\_Click( người gửi **đối tượng** , EventArgs e)
2. {
3. OpenFileDialog ofd =  **new** OpenFileDialog();
4. ofd.InitialDirectory =  "c:\\" ;
5. ofd.Filter =  "tệp txt (\*.txt)|\*.txt|Tất cả các tệp (\*.\*)|\*.\*" ;
6. ofd.FilterIndex = 2;
7. ofd.RestoreDirectory =  **true**;
8. **if** (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)
9. {
10. DynamicRichTextBox.LoadFile(ofd.FileName);
11. DynamicRichTextBox.Find( "Văn bản" , RichTextBoxFinds.MatchCase);
12. DynamicRichTextBox.SelectionFont =  Phông chữ **mới** ( "Verdana" , 12, FontStyle.Bold);
13. DynamicRichTextBox.SelectionColor = Color.Red;
14. DynamicRichTextBox.SaveFile(@ "C:\Junk\SavedRTF.rtf" , RichTextBoxStreamType.RichText);
15. }
16. }

**Dấu đầu dòng**

Thuộc tính BulletIndent nhận hoặc đặt thụt lề được sử dụng trong điều khiển RichTextBox khi kiểu dấu đầu dòng được áp dụng cho văn bản.

DynamicRichTextBox.BulletIndent = 10;

**Thuộc tính lựa chọn**

Đây là danh sách các thuộc tính có thể áp dụng trên văn bản được chọn hiện tại.

* SelectionAlignment - Căn chỉnh văn bản đã chọn.
* SelectionBackColor - Màu nền của văn bản được chọn.
* SelectionBullet - Đặt đúng hoặc sai nếu dấu đầu dòng được áp dụng trên văn bản đã chọn.
* SelectionCharOffset - Liệu văn bản trong điều khiển có xuất hiện trên đường cơ sở, dưới dạng chỉ số trên hay dưới dạng chỉ số dưới bên dưới đường cơ sở
* SelectionColor - Màu của văn bản được chọn.
* SelectionFont - Phông chữ của văn bản được chọn.
* SelectionHangingIndent - Khoảng cách giữa cạnh trái của dòng văn bản đầu tiên trong đoạn văn bản đã chọn và cạnh trái của các dòng tiếp theo trong cùng một đoạn văn.
* SelectionIndent - Độ dài, tính bằng pixel, của mức thụt lề của dòng nơi bắt đầu lựa chọn.
* SelectionProtected - Calue cho biết liệu lựa chọn văn bản hiện tại có được bảo vệ hay không
* SelectionTabs - Vị trí dừng tab tuyệt đối.
* SelectionType - Loại lựa chọn được biểu thị bằng bảng liệt kê RichTextBoxSelectionType với các giá trị Trống, Văn bản, Đối tượng, MultiChar và MultiObject.

Đoạn mã sau đặt các thuộc tính lựa chọn này.

1. **void void** SelectionButton\_Click( người gửi **đối tượng** , EventArgs e)
2. {
3. DynamicRichTextBox.BackColor = Color.White;
4. DynamicRichTextBox.Clear();
5. DynamicRichTextBox.BulletIndent = 10;
6. DynamicRichTextBox.SelectionFont =  Phông chữ **mới** ( "Georgia" , 16, FontStyle.Bold);
7. DynamicRichTextBox.SelectedText =  "Mạng Mindcracker \n" ;
8. DynamicRichTextBox.SelectionFont =  Phông chữ **mới** ( "Verdana" , 12);
9. DynamicRichTextBox.SelectionBullet =  **true**;
10. DynamicRichTextBox.SelectionColor = Color.DarkBlue;
11. DynamicRichTextBox.SelectedText =  "Góc C#"  +  "\n" ;
12. DynamicRichTextBox.SelectionFont =  Phông chữ **mới** ( "Verdana" , 12);
13. DynamicRichTextBox.SelectionColor = Color.Orange;
14. DynamicRichTextBox.SelectedText =  "VB.NET Heaven"  +  "\n" ;
15. DynamicRichTextBox.SelectionFont =  Phông chữ **mới** ( "Verdana" , 12);
16. DynamicRichTextBox.SelectionColor = Color.Green;
17. DynamicRichTextBox.SelectedText =  ".Góc Longhorn"  +  "\n" ;
18. DynamicRichTextBox.SelectionColor = Color.Red;
19. DynamicRichTextBox.SelectedText =  ".NET Heaven"  +  "\n" ;
20. DynamicRichTextBox.SelectionBullet =  **false**;
21. DynamicRichTextBox.SelectionFont =  Phông chữ **mới** ( "Tahoma" , 10);
22. DynamicRichTextBox.SelectionColor = Color.Black;
23. DynamicRichTextBox.SelectedText =  "Đây là danh sách các trang web của Mạng lưới Mindcracker.\n" ;
24. }

Văn bản đầu ra trong RichTextBox trông giống như Hình.

**Làm lại và CanRedo**

Phương pháp làm lại có thể được sử dụng để áp dụng lại thao tác hoàn tác cuối cùng cho điều khiển.

Thuộc tính CanRedo thể hiện liệu có hành động nào đã xảy ra trong RichTextBox có thể được áp dụng lại hay không.

1. **if** (dynamicRichTextBox.CanRedo ==  **true**)
2. {
3. **if** (dynamicRichTextBox.RedoActionName !=  "Xóa" )
4. DynamicRichTextBox.Redo();
5. }

**Phát hiệnUrl**

Nếu được đặt đúng, thuộc tính DetectUrls sẽ tự động định dạng Bộ định vị tài nguyên thống nhất (URL) khi nó được nhập vào điều khiển.

**BậtTự độngKéoThả**

Điều khiển RichTextBox hỗ trợ các thao tác kéo và thả cho phép chúng ta kéo và thả văn bản, hình ảnh và các dữ liệu khác. Thuộc tính EnableAutoDragDrop cho phép thao tác kéo và thả trên văn bản, hình ảnh và dữ liệu khác.

1. DynamicRichTextBox.EnableAutoDragDrop =  **true**;

**RightMargin, AutoWordSelection và ZoomFactor**

Thuộc tính RightMargin biểu thị kích thước của một dòng văn bản trong điều khiển RichTextBox.

Thuộc tính AutoWordSelection biểu thị liệu một từ có được chọn tự động hay không khi văn bản được nhấp đúp trong điều khiển RichTextBox.

ZoomFactor biểu thị mức thu phóng hiện tại của RichTextBox. Giá trị 1.0 có nghĩa là không áp dụng thu phóng trên điều khiển.

1. **void riêng tư** ZoomButton\_Click( người gửi **đối tượng** , EventArgs e)
2. {
3. DynamicRichTextBox.AutoWordSelection =  **true**;
4. DynamicRichTextBox.RightMargin = 5;
5. DynamicRichTextBox.ZoomFactor = 3.0f;
6. }

**Rtf và Rtf được chọn**

Thuộc tính Rtf được dùng để lấy và đặt văn bản có định dạng văn bản đa dạng thức (RTF) trong điều khiển RichTextBox. Thuộc tính SelectedRtf được sử dụng để lấy và đặt văn bản đã chọn trong điều khiển. Văn bản RTF là văn bản bao gồm định dạng.

## tóm tắt

Điều khiển RichTextBox chấp nhận dữ liệu đầu vào của người dùng trên Biểu mẫu và cung cấp các tính năng văn bản đa dạng thức. Trong bài viết này, chúng tôi đã thảo luận về cách tạo điều khiển RichTextBox trong Windows Forms tại thời điểm thiết kế cũng như thời gian chạy. Sau đó, chúng ta đã biết cách sử dụng các thuộc tính và phương thức khác nhau.