Thiết kế xử lý code folding với richtextbox

# Giới thiệu

Code folding là chức năng giúp người dùng ẩn một số đoạn văn bản. Nhờ vậy người dùng có thể ẩn một số đoạn code để tập trung tới một khu vực

Code foldeing bao gồm 2 thao tác chính:

* Nhận dạng đoạn code có thể gập
* Gập / mở gập đoạn code

# Các xử lý chính

## Nhận dạng đoạn code có thể gập

Đối với ngôn ngữ lập trình C, C++. C# một đoạn code có thể gập bắt đầu từ giữa hai dấu “{“ và “}”. Tức là những gì nằm giữa hai dấu đó có thể gập đi. Cần lưu ý là đoạn code có thể gập này có thể chứa đoạn code có thể gập khác.

Để nhận dạng ta sử dụng Regex. Regex là chuỗi ký tự đặc biệt dùng để làm mẫu, nhờ đó ta có thể tìm kiếm được. Để nhận dạng đoạn code có thể gập trong C, C++, C#, regex cần dùng là “(({) | (}))”.Regex trên có ý nghĩa là tìm ký tự “{“ hoặc “}”.

Để lưu trữ vị trí của đoạn code có thể gập, ta tạo một lớp mới gồm 2 số nguyên : vị trí bắt đầu và vị trí kết thức

|  |
| --- |
| public class SectionPosition  {  public int Start;  public int End;  } |

Thuật toán :

1. Tạo một danh sách SectionPosition rỗng
2. Tạo một stack số nguyên để lưu vị trí của ký tự “{“
3. Dùng Regex để Scan văn bản từ trên xuống dưới,mỗi khi tìm thấy:
   1. Nếu là “{“, thêm vị trí của “{“ vào stack
   2. Nếu là “}”
      1. Nếu stack rỗng: bỏ qua
      2. Nếu stack không rỗng, pop stack lấy vị trí của “{“, vị trí của “{“ và vị trí của “}” tạo thành một SectionPosition, thêm SectionPosition đó vào danh sách SectionPosition.
4. Trả về danh sách SectionPosition.

Code:

|  |
| --- |
| public List<SectionPosition> FindFirst(string code, List<SectionDelimiter> delimiters, int start = 0,  int end = -1)  {  List<SectionPosition> positions = new List<SectionPosition>();  Stack<SectionStackItem> stack = new Stack<SectionStackItem>();  int regexGroupIndex;  bool isStartToken;  SectionDelimiter matchedDelimiter;  SectionStackItem currentItem;  Regex scanner = RegexifyDelimiters(delimiters);  foreach (Match match in scanner.Matches(code, start))  {  // the pattern for every group is that 0 corresponds to SectionDelimter, 1 corresponds to Start  // and 2, corresponds to End.  regexGroupIndex =  match.Groups.Cast<Group>().Select((g, i) => new {  Success = g.Success,  Index = i  })  .Where(r => r.Success && r.Index > 0).First().Index;  matchedDelimiter = delimiters[(regexGroupIndex - 1) / 3];  isStartToken = match.Groups[regexGroupIndex + 1].Success;  if (isStartToken)  {  stack.Push(new SectionStackItem()  {  Delimter = matchedDelimiter,  Position = new SectionPosition() { Start = match.Index }  });  }  else  {  if (stack.Count == 0)  continue;  currentItem = stack.Pop();  if (currentItem.Delimter == matchedDelimiter)  {  currentItem.Position.End = match.Index + match.Length;  positions.Add(currentItem.Position);  return positions;  // if searching for an end, and we've passed it, and the stack is empty then quit.  if (end > -1 && currentItem.Position.End >= end && stack.Count == 0) break;  }  else  {  }  }  }  //if (stack.Count > 0) throw new Exception("Not enough closing symbols.");  return positions;  } |

## Gập, mở gập đoạn code

Khi người dùng click chuột vào một dòng cụ thể ở Number\_margin (thanh đánh số dòng), lúc đó là sẽ thực hiện thao tác gập / mở gập.

Khi đoạn code được gập nó sẽ có dạng “{folded1}”, trong đó “folded” cho biết đoạn code đã gập, 1 là chỉ số để lưu.

Để lưu đoạn code đã gập, ta thạo một lớp gồm chỉ số và chuỗi

|  |
| --- |
| class FoldedState: System.IEquatable<FoldedState>  {  public int Header;  public string Content;  public bool Equals(FoldedState other)  {  return this.Header == other.Header;  }  } |

Để lưu danh sách đoạn code đã gập. ta tạo một danh sách FoldedState

|  |
| --- |
| SortedDictionary<int, FoldedState> foldedList = new SortedDictionary<int, FoldedState>(); |

Code

|  |
| --- |
| private void NumberMargin\_MouseClick(object sender, MouseEventArgs e)  {  //blockRecordingUndo = true;  string nestedText;  FoldedState currentFolded;  //get the clicked line  int clickedLine = GetLineFromMouseClick(new Point(1, e.Y));  //find nested start with the line  SectionPosition section = nestedList.Find((x) => { return GetLineFromCharIndex(x.Start) == clickedLine; });  int foldedIndex;  if (section != null)  {  //select the nested code  Select(section.Start + 1, section.End - section.Start - 2);  nestedText = SelectedText;  //Check if it is folded  if (nestedText.StartsWith("folded"))  {  //check if folded index is a number  if (int.TryParse(nestedText.Substring(6), out foldedIndex))  {  //foldedIndex may not be in foldedList so we use try catch for safety  try  {  currentFolded = foldedList[foldedIndex];  blockAllAction = true;  SelectedText = currentFolded.Content;  blockAllAction = false;  foldedList.Remove(foldedIndex);  return;  }  catch  { }  }  }  // it is unfold so fold it  {  NewFolded(section);  }  }  //blockRecordingUndo = false;  } |