BÁO CÁO DOAN-01: ĐỒ HỌA 2D

Nguyễn Trần Hậu - MSSV 1612180
Nguyễn Chí Thức - MSSV 1612677

Tháng 11, 2018

Mục lục

Tóm tắt nội dung

Chương trình được viết bằng ngôn ngữ C# với IDE Visual Studio 2015. Thực hiện các chức năng vẽ đoạn thẳng, vẽ hình chữ nhật, vẽ cung tròn, đa giác , text, ... cùng phép biến hình là di chuyển, tỷ lệ và quay, ... và tô màu biên, màu bên trong, thay đổi font, ... Một số chức năng khác như đọc, lưu file và thực hiện undo/redo.

1 Phân công

Nguyễn Chí Thức:

- Yêu cầu 1: Vẽ hình bình hành, đường gấp khúc, hình ellipse, parabol, bezier, kí tự
- Yêu cầu 2: Lưu file, đọc file
- Yêu cầu 3:

  Tô màu bên trong những hình khép kín

  Chọn font chữ, size, kiểu chữ (đậm, nghiêng, gạch chân)
- Yêu cầu 5:

  Xây dựng undo, redo cho thêm hình mới, xóa hình cũ

Nguyễn Trần Hậu:

- Yêu cầu 1: Vẽ đường thẳng, hình chữ nhật, đa giác, ellipse, cung tròn
- Yêu cầu 3:

  Thay đổi thuộc tính biên
- Yêu cầu 4: Xây dựng phép biến hình: di chuyển, tỷ lệ, xoay

- Yêu cầu 5:

  Xuất ra file ảnh

  Thay đổi thứ tự của hình

  Xây dựng undo, redo cho sửa thuộc tính của hình

2 Mức độ hoàn thành

Yêu cầu	**Mức độ hoàn thành**
Yêu cầu 1	90% (thiếu hyperbol, đường gấp khúc)
Yêu cầu 2	100%
Yêu cầu 3	100%
Yêu cầu 4	100%
Yêu cầu 5 (nâng cao)	15% (chỉ làm xuất file ảnh, thay đổi thứ tự hình và undo, redo)

3 Sơ lược chương trình

3.1 Danh sách class

Chương trình chứa class MyShape là abstract class chứa các thuộc tính và phương thức chung của các class hình vẽ. Các class kế thừa từ class MyShape bao gồm 3 nhóm:

- Nhóm chỉ có thể tô màu biên: MyArc, MyBezier, MyLine, MyParabol
- Nhóm tô màu biên và tô màu bên trong: MyEllipse, MyParallelogram, MyPolygon, MyRectangle
- MyCharater là nhóm riêng

Ngoài ra còn có thêm 3 class hỗ trợ:

- Class PenAttr: Lưu các thuộc tính để tạo class Pen
- Class BrushAttr: Lưu các thuộc tính để tạo class Brush
- Class FontAttr: Lưu các thuộc tính để tạo class Font

Các thuộc tính và phương thức còn lại nằm trong class FormMain.

3.2 Ý tưởng

Sử dụng PictureBox (là hình chữ nhật màu trắng ở phía dưới các nút trong chương trình) để hiển thị các hình ảnh người dùng vẽ lên. Người dùng nhấn các nút tương ứng với các hình như đường thẳng, hình chữ nhật, ellipse, ... rồi di chuyển và click chuột trong PictureBox để vẽ. Nhận biết khi nào người dùng vẽ trên PictureBox bằng sự kiện MouseDown, MouseMove và MouseUp. Chương trình có 2 chế độ là vẽ hình và chọn (select). Để chuyển từ chế độ vẽ qua chế độ chọn, người dùng nhấn nút select. Và để đổi lại chế độ vẽ thì người dùng chỉ cần nhấn lại nút chứa các hình ảnh cần vẽ.

Ở **chế độ vẽ hình**, sau khi chọn nút có hình ảnh người dùng cần vẽ, click vào trong PictureBox di chuyển chuột sau đó click lại lần nữa để hoàn thành hình. Số lần click tùy theo loại hình mà người dùng chọn.

Hình	Số lần click chuột
Đoạn thẳng	2
Hình chữ nhật	2
Hình ellipse	2
Hình đa giác	(click cho đến khi trùng với vị trí click ban đầu)
Hình bình hành	3
Cung tròn	3
Bezier	4
Parabol	2
Text	1

Ở **chế độ chọn**, khi click vào hình nào thì hình đó nổi lên trên và xuất hiện những điểm neo thể hiện khung của hình. Trong chế độ chọn có tính năng xóa, là xóa một hình hay nhiều hình đang được chọn.

Có 3 phép biến hình trong chế độ chọn. Đó là phép di chuyển (move), phép tỷ lệ (scale), phép quay (rotate). Để di chuyển hình click vào điểm nằm trên biên, khung hoặc nằm trong hình rồi kéo thả. Để thực hiện phép tỷ lệ và phép quay click vào các điểm neo của hình rồi kéo thả.

Bên cạnh đó, chương trình còn cho phép người dùng chọn màu, chọn kiểu của nét vẽ, chọn kiểu tô màu đối với vẽ hình; chọn font chữ, kiểu font, kích thước font đối với text.

4 Thiết kế class MyShape và các class con kế thừa từ MyShape

4.1 Thuộc tính chính class MyShape

```
public PenAttr penAttr { get; set; }
```

`penAttr` lưu thuộc tính cho class Pen khi MyShape tạo class Pen để vẽ biên. Toàn bộ hình đều cần biên.

```
public List<Point> points { get; set; }
```

`points` lưu các tọa độ điểm mà người dùng click vào PictureBox để vẽ hình.

```
public float angleRotate { get; set; }
```

`angleRotate` lưu góc quay của hình. Mặc định góc quay = 0, tức là không quay.

```
public SizeF tyleScale { get; set; }
```

`tyleScale` lưu tỷ lệ khi scale. Mặc định tỷ lệ = 1, tức là không scale.

4.2 Phương thức chính class MyShape

Phương thức khởi tạo chính

```
public MyShape(PenAttr _penAttr, List<Point> _points
    )
public MyShape(List<Point> _points)
public MyShape(MyShape myShape)
```

Class MyShape là abstract class , không khởi tạo trực tiếp. Phương thức khởi tạo của MyShape dùng để phục vụ cho những class con kế thừa từ MyShape.

```
public abstract void draw(Bitmap _bitmap, PictureBox
    pictureBox);
public virtual void fill(Bitmap _bitmap, PictureBox
   pictureBox)
```

Vì tùy hình cụ thể mà có cách vẽ khác nhau, nhưng hình nào cũng phải vẽ được, nên phương thức `draw()` là abstract. Không phải hình nào cũng có cách tô màu bên trong nên phương thức fill() là virtual. Mặc định của phương thức `fill()` là không làm gì cả. Cài đặt cụ thể nằm ở những class kế thừa.

```
public virtual List<Point> getEdgePoints()
public virtual void drawEdgePoints(Bitmap _bitmap,
   PictureBox pictureBox)
```

Phương thức `getEdgePoints()` trả về những điểm neo của hình. Điểm neo là những điểm nằm trên biên của hình, ví dụ như 4 đỉnh của hình chữ nhật, 2 đỉnh của đoạn thẳng, ... Điểm neo dùng để scale, quay hình. Mặc định trả về points, nhưng đối với hình chữ nhật và hình ellipse thì vẽ bằng 4 điểm nhưng points chỉ gồm 2 điểm nên thêm 2 điểm còn lại vào. Phương thức `drawEdgePoints()` để vẽ những điểm neo của hình.

```
public virtual bool isPointBelongPrecisely(Point p)
public bool isPointBelong(Point p)
public virtual bool isPointInsidePrecisly(Point p)
```

Phương thức `isPointBelongPrecisely()` kiểm tra chính xác tọa độ p có nằm ở biên hình hay không. Phương thức này cần cài đặt cụ thể ở từng class kế thừa. Nhưng vì click đúng từng tọa độ là rất khó, nên phương thức `isPointBelong()` kiểm tra tọa độ p và những tọa độ xung quanh p, chỉ cần một trong những điểm đó nằm trên biên là được. Phương thức

`isPointInsidePrecisly()` kiểm tra chính xác tọa độ p có nằm trong hình hay không. Mặc định phương thức này trả về false vì có những hình không có vùng phía trong (như đoạn thẳng, bezier, ...) Đối với những hình có vùng phía trong thì cài đặt lại trong class kế thừa.

```
public virtual void movePoints(Point p_before, Point
    p_after)
public virtual void scalePoints(Point _p_before,
   Point _p_after)
public virtual void rotatePoints(Point _p_before,
   Point _p_after)
```

3 phương thức thể hiện phép biến hình. Phương thức `movePoints()` tính độ lệch giữa điểm click kéo đi điểm sau khi thả rồi tính lại points theo độ lệch. Phương thức `scalePoints()` và `rotatePoints()` không làm thay đổi points. Phương thức `scalePoints()` tính tỷ lệ giữa 2 khoảng cách., là khoảng cách điểm khi thả ra đến trọng tâm hình và khoảng cách điểm khi click kéo đến trọng tâm của hình, rồi lưu vào `tyleScale`. Phương thức `rotatePoints()` tính góc quay giữa 2 vecto, là vecto từ trọng tâm hình đến điểm khi thả ra và vector từ trọng tâm của hình đến điểm khi click kéo đi, rồi lưu vào `angleRotate`.

4.3 Nhóm class chỉ tô màu biên

Nhóm class chỉ tô màu biên bao gồm: MyArc, MyBezier, MyLine, MyParabol.

Class MyBezier, MyLine, MyParabol không có thêm thuộc tính so với class MyShape.

Class MyArc có thêm thuộc tính mới

```
private RectangleF rect_bound;
```

```
private float startAngle;
private float sweepAngle;
```

`rect_bound` lưu hình chữ nhật bao quanh cung tròn. `startAngle` là góc so với Ox tại điểm bắt đầu cung. `sweepAngle` là góc quay của cung.

Cả 4 class đều cài đặt lại phương thức:

```
public override void draw(Bitmap _bitmap, PictureBox
    pictureBox)
public override bool isPointBelongPrecisely(Point p)
```

Phương thức `draw()` dùng các phương thức của class Graphics để vẽ. Các phương thức có sẵn của class Graphics: `DrawCurve()`, `DrawBeziers()`, `DrawLine()`, `DrawArc()` để vẽ hình; `TranslateTransform()`, `ScaleTransform()`, `RotateTransform()` để scale và xoay. Trong phương thức draw(), tự động scale và xoay luôn vì ở class MyShape đã lưu `tyleScale` và `angleRotate` rồi. Phương thức `isPointBelongPrecisely()` sử dụng class GraphicsPath có phương thức `IsOutlineVisible()`.

Class MyArc có 2 phương thức riêng:

```
public void calcBound()
public static PointF centerOfCircle(List<Point>
   _points)
```

Phương thức `centerOfCircle()` tính tọa độ của tâm hình tròn ngoại tiếp của `_points`. Phương thức calcBound() dùng để tính hình chữ nhật bo quanh hình tròn có tâm được tính từ `centerOfCircle()`

4.4 Nhóm class tô màu biên và tô màu bên trong

Nhóm class tô màu biên và tô màu bên trong: MyEllipse, MyParallelogram, MyPolygon, MyRectangle

4 class có thêm thuộc tính mới so với class MyShape:

```
public BrushAttr brushAttr { get; set; }
```

`brushAttr` dùng để khởi tạo class Brush, dùng để tô màu bên trong hình.

Class MyEllipse và MyRectangle có thêm thuộc tính mới so với class MyShape:

```
private Point mostLeft;
private Size size;
```

Ellipse nằm trong khung là hình chữ nhật. `mostLeft` lưu tọa độ điểm trái nhất, `size` lưu kích thước của hình chữ nhật.

Cả 4 class đều cài đặt lại phương thức:

```
public override void draw(Bitmap _bitmap, PictureBox
    pictureBox)
public override void fill(Bitmap _bitmap, PictureBox
    pictureBox)
public override bool isPointBelongPrecisely(Point p)
public override bool isPointInsidePrecisly(Point p)
```

Phương thức `draw()` dùng phương thức của class Graphics để vẽ. Phương thức `fill()` dùng phương thức của class Graphics để tô màu bên trong. Các phương thức có sẵn của class Graphics: `DrawRectangle(), DrawPolygon(), DrawEllipse()` để vẽ hình; `FillRectangle(), FillPolygon(), DrawEllipse()` để tô màu bên trong; `TranslateTransform(), ScaleTransform(), RotateTransform()` để scale và xoay. Trong phương thức `draw()` và `fill()`, tự động scale và xoay luôn vì ở class MyShape đã lưu `tyleScale` và `angleRotate` rồi. Phương thức `isPointBelongPrecisely()` sử dụng class GraphicsPath có phương thức `IsOutlineVisible()`. Phương thức `isPointInsidePrecisly()` sử dụng class GraphicsPath có phương thức `IsVisible()`.

4.5 Class MyCharater

Class MyCharater có thêm thuộc tính mới so với class MyShape:

```
private Size size;
public BrushAttr brushAttr { get; set; }
public FontAttr fontAttr { get; set; }
```

`size` chỉ kích thước hình chữ nhật bao quanh text, `brushAttr` là các thuộc tính của class Brush để tô màu text, `fontAttr` là các thuộc tính của class Font để tạo text.

Class MyCharater cài đặt lại phương thức:

```
public override void draw(Bitmap _bitmap, PictureBox
    pictureBox)
public override bool isPointInsidePrecisly(Point p)
public override void scalePoints(Point _p_before,
   Point _p_after)
```

Phương thức `draw()` dùng phương thức của class Graphics để vẽ. Các phương thức có sẵn của class Graphics: `DrawString()` để viết text; `TranslateTransform()`, `RotateTransform()` để scale và xoay. `isPointInsidePrecisly()` dùng class GraphicsPath có phương thức `IsVisible()`. `scalePoints()` không làm gì cả, vì phóng to thu nhỏ đã có font size.

5 Thiết kế của class phụ trợ

5.1 Class PenAttr

Thuộc tính:

```
public Color color { get; set; }
public DashStyle dashStyle { get; set; }
public int width { get; set; }
```

Dùng để lưu những dữ liệu cần thiết để khởi tạo đối tượng Pen, gồm `color` dùng để chỉ định màu tô, `dashStyle` là kiểu nét vẽ và `width` là độ dày của các nét vẽ. Dùng để tô màu, chỉ định nét vẽ và kiểu nét vẽ khi vẽ các hình đã được yêu cầu.

Phương thức khởi tạo:

```
public PenAttr(Color _color, DashStyle _dashStyle,
   int _width)
public PenAttr(PenAttr penAttr)
```

5.2 Class BrushAttr

Thuộc tính:

```
public Color color { get; set; }
public Color color2 { get; set; }
public String typeBrush { get; set; }
```

Dùng để lưu trữ các dữ liệu cần thiết để khởi tạo class Brush. Một số kiểu Brush cần 2 màu để tô nên ở đây lưu 2 giá trị `color` và `color2`, `typeBrush` dùng để chỉ định loại tô màu mà người dùng muốn chọn. Dùng để tô màu bên trong các hình vẽ có thể tô được như là đừng ellipse, đa giác, hình chữ nhật, hình bình hành.

Phương thức khởi tạo:

```
public BrushAttr(Color _color, String _typeBrush)
public BrushAttr(BrushAttr brushAttr)
```

Dùng để khởi tạo đối tượng mới cho class BrushAttr, từ các giá trị này để tạo class Brush dùng để tô màu bên trong đối tượng.

5.3 Class FontAttr

Thuộc tính:

```
public String text { get; set; }
public string fontFamily { get; set; }
public int size { get; set; }
public FontStyle fontStyle { get; set; }
```

Dùng để lưu trữ dữ liệu cần thiết để khởi tạo class Font và để vẽ kí tự. `text` dùng để lưu nội dung, `fontFamily` dùng để chỉ định loại font cần vẽ kí tự, `size` lưu kích thước của hình chữ nhật bao quanh, `fontStyle` dùng để chọn loại mà mình muốn vẽ là in đậm, in nghiêng và gạch chân

Phương thức khởi tạo:

```
public FontAttr(String _text, string _fontFamily,
   int _size, FontStyle _fontStyle)
public FontAttr(FontAttr fontAttr)
```

6 Cài đặt trong FormMain

Trong FormMain, chương trình lưu tất cả các hình người dùng từng vẽ bằng `myShapes`

```
private List<MyShape> myShapes = new List<MyShape>()
   ;
```

6.1 Bắt sự kiện trong PictureBox

Chương trình bắt 3 sự kiện trong PictureBox, bao gồm MouseDown, MouseMove, MouseUp

```
private void pictureBoxMain_MouseDown(object sender,
    MouseEventArgs e)
private void pictureBoxMain_MouseMove(object sender,
    MouseEventArgs e)
```

```
private void pictureBoxMain_MouseUp(object sender,
    MouseEventArgs e)
```

Chương trình lưu những điểm người dùng đã click bằng `clickedPoints`

```
private List<Point> clickedPoints = new List<Point
    >();
```

Khi xảy ra sự kiện MouseDown, chương trình kiểm tra đang ở trong chế độ vẽ hình hay chế độ chọn.

Nếu đang ở trong chế độ vẽ hình, thêm tọa độ điểm vừa click vào `clickedPoints`. Kiểm tra xem `clickedPoints` đã đủ để vẽ hình đã chọn chưa. Nếu rồi thì vẽ hình đó ra và thêm vào `myShapes` và không bắt tiếp sự kiện MouseMove. Nếu chưa bắt sự kiện MouseMove và MouseDown tiếp theo.

Nếu đang ở trong chế độ chọn, kiểm tra tọa độ của điểm vừa chọn có nằm ở biên hay nằm trong hình nào không. Ưu tiên điểm nằm ở biên hơn điểm nằm ở bên trong vì điểm nằm ở biên khó click hơn, ưu tiên hình vừa mới vẽ. Bên cạnh đó kiểm tra đang trong trạng thái Move, Scale, Rotate để có cho phép bắt sự kiện MouseMove hay không. Nếu trong trạng thái Move, thì cho phép bắt MouseMove. Nếu trong trạng thái Scale hoặc Rotate, kiểm tra có click trúng điểm neo hay không, nếu có cho phép bắt MouseMove, nếu không thì thôi.

Khi xảy ra sự kiện MouseMove, chương trình kiểm tra đang ở trong chế độ chọn hay chế độ vẽ hình.

Nếu đang ở chế độ vẽ hình, kiểm tra có còn bắt sự kiện MouseDown nữa hay không. Nếu không còn bắt thì không làm gì cả, nếu còn bắt thì vẽ hình với điểm mà con trỏ chuột đang chỉ vào để người dùng có cảm giác trực quan là hình được vẽ hình dạng như thế nào. Lưu ý đây chỉ là vẽ tạm, vẽ thật và thêm vào hình vừa tạo vào myShapes nằm ở MouseDown.

Nếu đang ở chế độ chọn, kiểm tra có được bắt sự kiện MouseMove hay

không. Nếu không thì dừng lại không làm gì cả. Nếu có thì vẽ hình với điểm mà con trỏ chuột đang chỉ vào tùy theo trạng thái Move, Scale, Rotate. Lưu ý đây chỉ là vẽ tạm, vẽ thật và thay đổi thuộc tính hình được chọn trong myShapes nằm ở MouseUp.

Khi xảy ra sự kiện MouseUp, kiểm tra có đang ở chế độ chọn hay không. Nếu không thì dừng lại không làm gì cả. Nếu có thì kiểm tra tiếp có vừa trải qua sự kiện MouseMove hay không. Nếu không thì đừng lại không làm gì cả. Nếu có thì vẽ hình với điểm mà con trỏ chuột vừa thả ra tùy theo trạng thái Move, Scale, Rotate. Hình vừa vẽ được lưu lại thuộc tính mới trong myShapes. (points, tyleScale, angleRotate)

6.2 Tính năng lưu file, đọc file

Để thuận tiện cho việc lưu và đọc file, sử dụng Serialization cho tất cả các class có trong chương trình.

Sử dụng class SaveFileDialog để mở dialog lưu file. Mặc định lưu file vào ổ `C`, file lưu có dạng mở rộng là `*.bin`

Sử dụng class OpenFileDialog để mở dialog đọc file. Mặc định mở file trong ổ `C`, file đọc phải có dạng mở rộng là `*.bin`

Có hai cách để lưu là Save và Save as. Nếu file đang mở ra chưa lưu, thì Save và Save as đều mở ra SaveFileDialog. Nếu file đang mở đã có file lưu, thì Save lưu thẳng vào file lưu cũ, Save as mở ra SaveFileDialog để lưu vào file mới.

6.3 Xuất ra ảnh

Sử dụng class Bitmap với phương thức Save() để lưu dưới dạng hình ảnh. Những định dạng hỗ trợ: `*.png, *.jpeg, *.gif, *.tiff`

6.4 Thay đổi thứ tự các hình

Vì đã lưu các hình trong `myShapes` theo thứ tự thời gian. Hình vẽ đầu tiên là `myShapes[0]`, hình vẽ tiếp theo là `myShapes[1]`, ... Do đó thay đổi thứ tự của hình là thay đổi thứ tự trong `myShapes`. Vì `myShapes` khai báo dạng `List<MyShape>` nên dùng những phương thức của `List` để hỗ trợ: RemoveAt(), Insert()

6.5 Cài đặt undo, redo

Để cài đặt undo và redo, thêm 2 thuộc tính mới:

```
private int indexOfListMyShapes = -1;
private List<List<MyShape>> listMyShapes = new List<
    List<MyShape>>();
```

`listMyShapes` lưu lại danh sách `myShapes` theo thứ tự thời gian, mỗi lần thay đổi thuộc tính của 1 hình trong `myShapes`, hoặc thêm hình mới, xóa hình cũ. `indexOfListMyShapes` thể hiện vị trí của `myShapes` hiện tại trong `listMyShapes`.

Khi nhấn nút Undo, kiểm tra trước `myShapes` hiện tại trong `listMyShapes` còn `myShapes` nào hay không. Nếu còn thì thay thế `myShapes` hiện tại bằng `myShapes` trước đó.

Khi nhấn nút Redo, kiểm tra sau `myShapes` hiện tại trong `listMyShapes` còn `myShapes` nào hay không. Nếu còn thì thay thế `myShapes` hiện tại bằng `myShapes` sau đó.

Tài liệu

[1] Equation of a Circle from 3 Points (2 dimensions)
`http://paulbourke.net/geometry/circlesphere/`

[2] Using atan2 to find angle between two vectors
https://stackoverflow.com/questions/21483999/using-atan2-to-find-angle-between-two-vectors

[3] How to rotate, scale, and translate a matrix all at once in C#?
https://stackoverflow.com/questions/636081/how-to-rotate-scale-and-translate-a-matrix-all-at-once-in-c

[4] Whats the difference between Control.Invalidate, Control.Update and Control.Refresh?
https://blogs.msdn.microsoft.com/subhagpo/2005/02/22/whats-the-difference-between-control-invalidate-control-update-and-control-refresh/

[5] GraphicsPath Class
https://docs.microsoft.com/en-us/dotnet/api/system.drawing.drawing2d.graphicspath?view=netframework-4.7.2

[6] Pen Class
https://docs.microsoft.com/en-us/dotnet/api/system.drawing.pen?view=netframework-4.7.2

[7] Brush Class
https://docs.microsoft.com/en-us/dotnet/api/system.drawing.brush?view=netframework-4.7.2

[8] Font Class
https://docs.microsoft.com/en-us/dotnet/api/system.drawing.font?view=netframework-4.7.2