Trong bài viết này sử dụng thuật ngữ **framework** để nói một bộ khung làm việc giữa các class trong một chương trình hoặc một thư viện. Framework sẽ định nghĩa các cách thức mà class tương tác tới nhau, cập nhật trạng thái, đưa ra các interface, phương thức để bên thứ 3 (người viết code khác dung framework này) có thể tận dụng để mở rộng (extend) hoặc tái sử dụng (reuse)

Theo thiết kế ban đầu thì các đối tượng đồ họa có thể kẻo thả được gồm có

* Đoạn nối thẳng (Connector)
* Đoạn nối cong (CurvedConnector
* Hình tròn (biểu thị trạng thái) (State)
* Nhãn (ConnectorLabel)

Khi xử lý các sự kiện kéo thả = code, theo đa hình (polymorphism) trong lập trình hướng đối tượng, ta định nghĩa một interface đặc trưng cho đối tượng kéo thả được, ở đây là interface Selectable mà cả 4 lớp trên đều phải thừa kế. Như vây, code xử lý kéo thả không cần quá quan tâm đến các đối tượng được xử lý thuộc loại nào trong 4 loại kể trên mà chỉ làm việc chủ yếu với interface Selectable

Interface này có 2 phương thức, thuộc tính sau:

bool IsSelected: true có nghĩa là được chọn bởi người dùng , false là ko được chọn

bool HitTest(Point pt, Graphics g): trả về true nếu điểm truyển vào (pt) nằm trong đối tượng kéo thả được, false nếu không phải. g ở đây là một đối tượng đồ họa của hệ điều hành cung cấp các phương thức dùng để vẽ hoặc tô màu.

Khi kéo thả một đối tượng, có rất nhiều khả năng đối tượng kết nối tới nó bị ảnh hưởng theo, chẳng hạn khi kéo đối tượng State dịch chuyển thì tất cả các StateConnector kết nối tới nó cũng cần phải được vẽ lại. Đến lượt StateConnector cần cập nhật lại vị trí và vẽ lại, thì ta có các ConnectorLabel cũng cần được cập nhật vị trí. Do vậy cần có cơ chế quảng bá các sự kiện thay đổi tới các đối tượng liên quan. Cơ chế này được thực hiện chủ yếu thông qua lớp BaseMouseHandler và một số phương thức buộc ghi đè bởi lớp thừa kế bao gồm

public abstract bool TrackMouse(object sender, List<BaseMouseHandler> sourceChain, MouseEventArgs e) : Theo dõi tọa độ của chuột khi chuột được nhấp trái

public abstract bool HandleMouseEvent(object sender, List<BaseMouseHandler> sourceChain, MouseEventArgs e): theo dõi tọa độ chuột trong lúc chuột đang được nhấp trái và di chuyển

Trong 2 phương thức trên thì sender là đối tượng gửi đến, sourceChain là danh sách những đối tượng chịu ảnh hưởng,và e là đối tượng chưa thông số về chuột tại thời điểm đang gọi.

*Chú ý: Thiết kế cho việc quảng bá sự kiện lấy ý tưởng từ mẫu Design Pattern: Chain of Responsibility nhưng ko hoàn toàn giống về thiết kế. Tra cứu pattern trên mạng để biết thêm chi tiết*

Mục đích của 2 phương thức abstract trên là tạo cơ hội cho các đối tượng kéo thả nhận được thông tin kéo thả từ phía người dùng, từ đó các đối tượng này sẽ cập nhật trạng thái của chúng (chẳng hạn như tọa độ, độ lệch…)

2 phương thức nói trên cũng ko được thiết kế để gọi ở bất kì nơi nào tùy ý mà được gọi trong framework. Khi viết 2 phương thức trên ở lớp thừa kế, trả về true để xác nhận đôi tượng hiện tại chịu ảnh hưởng của sự kiện chuột, trả về false nếu không phải. Nếu đối tượng A cập nhật trạng thái của mình thì nhất định phải trả về true. Ngoài ra true còn có nghĩa các đối tượng kết nối tới đối tượng A như B, C, D cũng sẽ bị gọi qua TrackMouse và HandleMouseEvent

Chi tiết về phần implementation của các lớp thừa kế về TrackMouse và HandleMouseEvent ta có thể xem trong code (đã có bổ sung chú thích)

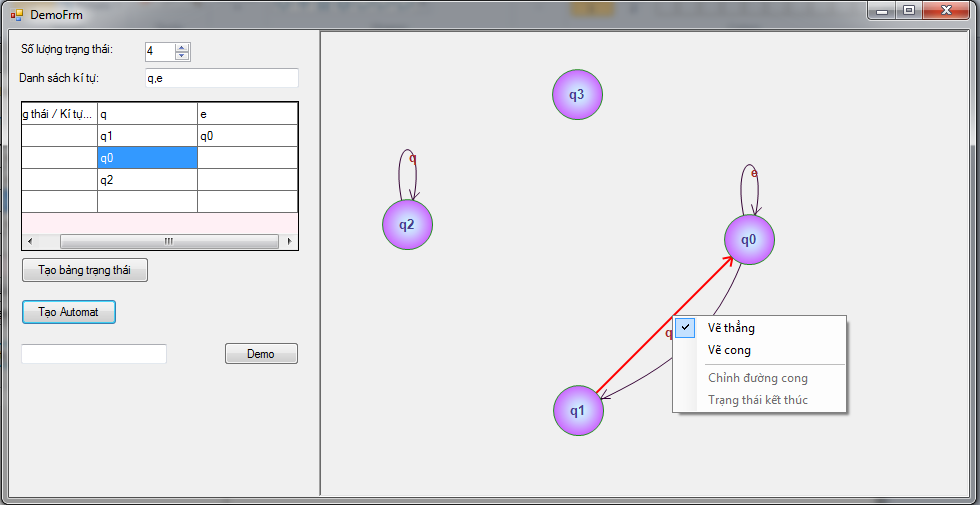
**Đôi điều về đường cong Bezier**

Thông tin về đường cong Bezier mọi người hãy tra cứu trên google

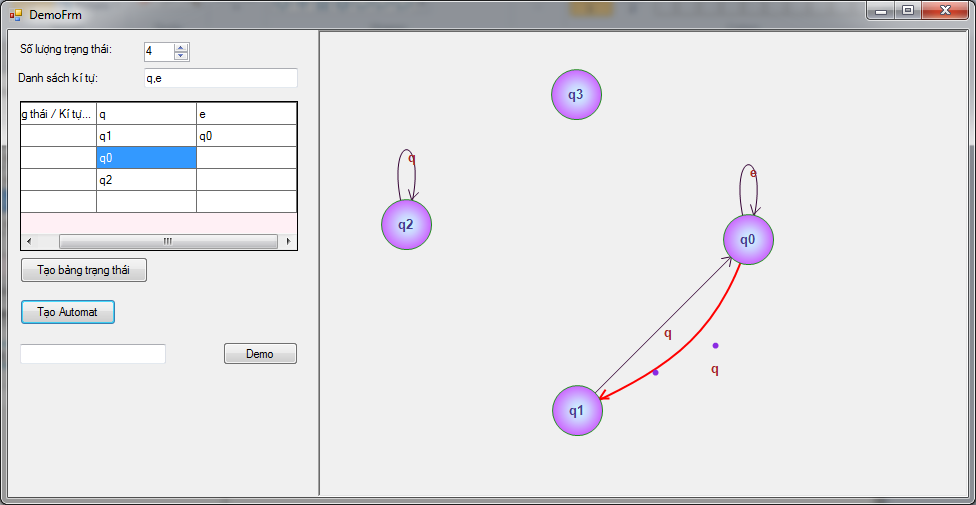
Đây là đường cong rất nổi trong xử lý đồ họa và dễ thấy nhất chính là công cụ Pen trong Photoshop.

Theo thiết kế ban đầu, việc cho phép người dùng chỉnh đường cong là điều cần thiêt để automata trong được đẹp hơn. Vẽ có cũng cần thiết khi có sự chuyển dịch giữa 2 trạng thái. Ở đây đường cong Bezier sử dụng là đường cong bậc 3, và có 2 điểm để chỉnh đường cong

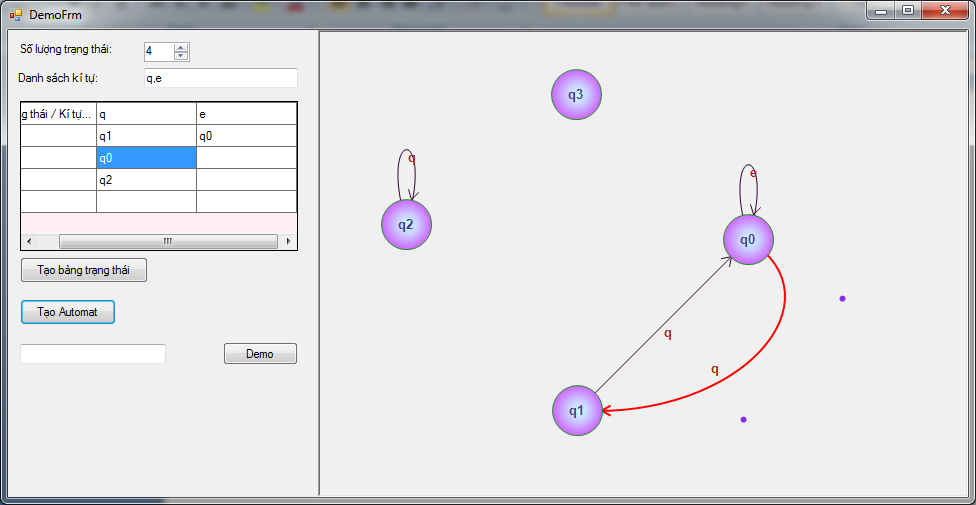
Để biến một đường thẳng thành một đường con, chọn đường thẳng đó sau đó bấm chuột phải, nhấp “Vẽ cong”



Sau đó, nhấp chuột vào 2 chỗ trống bất kí để tạo 2 điểm điều khiển. Hình của 2 điểm điều khiển sẽ là chấm xanh như hình vẽ:



Kéo thả 2 điểm điều khiển này sẽ thay đổi đường cong Bezier:



Để kết thúc chỉnh đường cong thì nhấp chuột phải chọn “Vẽ thẳng” rồi nhấp chuột vào một chỗ trống để ko chọn đường cong đó nữa (bug !!!!). Phần xứ lý context menu nằm trong file DemoFrm.cs, function connectorContextMenu\_ItemClicked

**Về GraphicsPath**

Để xác định được điểm người dùng nhấp chuột có thuộc vào đối tượng kéo thả không, trong đề tài này sử dụng GraphicsPath khá nhiều. GraphicsPath làm đơn giản hóa việc xác định một điểm có thuộc path hay ko. Chẳng hạn để phát hiện điểm nhấp chuột có thuộc đường cong Bezier, ta gọi GraphicsPath.AddBezier rồi gọi GraphicsPath.IsVisisble(pt) để kiểm tra pt có thuộc path hay ko.