**HTML5 CANVAS**

***Nhóm: Thiết kế Web***

***Lê Thị Thanh Ngân***

***Nguyễn Thị Mỹ Duyên***

***Nguyễn Thị Anh Thu***

***Phan Lê Mỹ Huyền***

***Nguyễn Huỳnh Bảo Châu***

# A. Giới thiệu về HTML5 Canvas

## **1. Canvas là gì?**

* **Phần tử <canvas>** trong HTML được sử dụng để vẽ các đối tượng đồ họa thông qua một ngôn ngữ dạng script (thường là javascript)
* **Phần tử <canvas>** sử dụng để vẽ các thành phần đồ họa từ đơn giản như các đường đến các đồ họa phức tạp (đường thẳng, hình hộp, hình tròn, viết chữ, chèn hình ảnh...).
* **<canvas>** cũng tương tự như những thẻ HTML khác, để sử dụng thì bạn phải đặt thẻ *<canvas></canvas>* tại vị trí muốn hiển thị.

## **2. Tại sao lại dùng Canvas?**

## Bởi vì Canvas có thể giúp hiển thị trực quan một số hình ảnh dễ dàng trên trình duyệt như:

* Các sơ đồ đơn giản
* Trang trí thêm cho giao diện người dùng
* Hình ảnh động
* Biểu đồ và đồ thị
* Có thể nhúng các ứng dụng vẽ
* Hoạt động tốt với những hạn chế của CSS

## **3. Ứng dụng của HTML5 Canvas:**

- Ứng dụng trong đồ họa 2D: Tạo đồ họa như biểu đồ, sơ đồ, các hiệu ứng động

- Ứng dụng trong trò chơi điện tử: Cách HTML5 Canvas sử dụng trong phát triển game, ví dụ: các trò chơi đơn giản như Tetris, Flappy Bird.

- Ứng dụng trong học tập và nghiên cứu: Các bài toán mô phỏng, nghiên cứu trực quan.

- Trong đồ họa động

-Trò chơi trực tuyến và ngoiaj tuyến

- Ảnh động

-Video và âm thanh tương tác

## **4. Ưu điểm và nhước điểm của HTML5 Canvas:**

*Ưu điểm:*

* Các sơ đồ đơn giản.
* Trang trí thêm cho giao diện người dùng.
* Hợp khi làm hình ảnh động và điều chỉnh hình ảnh, biểu đồ và đồ thị.
* Có thể nhúng các ứng dụng vẽ.
* Hoạt động tốt với những hạn chế của CSS.
* Tương thích với nhiều nền tảng.
* Hiệu quả cao với nhiều phần tử nhỏ
* Dễ dàng biến một trang web thuần túy thành một ứng dụng web động và sau đó chuyển đổi ứng dụng đó thành một ứng dụng di động để sử dụng trên điện thoại thông minh và máy tính bảng

*Nhược điểm*

* Không hiệu quả với ít phần tử lớn.
* Khó hiểu, đòi hỏi khả năng toán học cao và hiểu biết về JS.

# ***Trình duyệt hỗ trợ***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Google Chrome | Mozilla Firefox | Internet Explorer | Opera | Safari |
| *4.0* | *2.0* | *9.0* | *9.0* | *3.1* |

# B. Tọa độ trong Canvas

- Trong không gian 2D, tọa độ được xác định bởi trục X và Y:

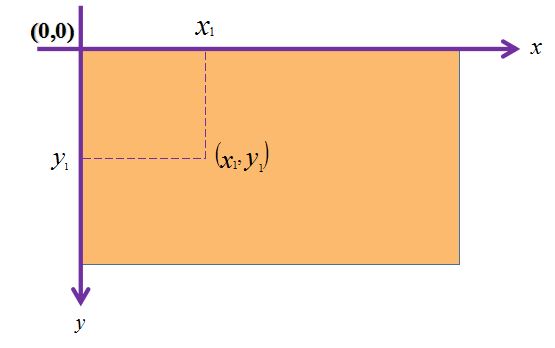
* X : tọa độ theo chiều ngang, gốc tọa độ tại x=0
* Y : tọa độ theo chiều dọc, gốc tọa độ tại y=0

-Hệ trục tọa độ như vậy trong toán học còn được gọi là hệ trục tọa độ Cartesian

-Canvas sử dụng trục tọa độ 2 chiều, với góc trái phía trên là tọa độ gốc (0,0)

-Những vật thể có tọa độ âm sẽ không hiển thị trên trang

* X : tọa độ theo phương nằm ngang, tăng dần từ trái sang phải
* Y : tọa độ theo phương thẳng đứng, tăng dần từ trên xuống dưới



# C. Khai báo Canvas

## **1. Định nghĩa**

**Canvas** là một vùng hình chữ nhật trên một trang HTML. Theo mặc định, một canvas không đường biên và không chứa nội dung.

## **2. Khai báo**

**Code khai báo một phần tử canvas:**

**VD:** <canvas id="myCanvas" width="200" height="100"></canvas>

**Chú ý:** Luôn luôn đặt thuộc tính *id* (để có thể gọi trong script), thuộc tính ***width*** và ***height*** để đặt kích thước của canvas. Nếu không được chỉ ra kích thước, thì kích thước mặc định là *300px* chiều rộng và *150px* chiều cao.

**Để thêm đường viền, sử dụng thuộc tính style như sau:**

**VD:** <canvas id="myCanvas" width="500" height="200" style="border:1px solid #000000;"></canvas>

**Kết quả:**



# D. Các kiểu trang trí trong HTML5 Canvas

## **I. Viết chữ trong HTML5 Canvas**

## **1. Các phương thức vẽ văn bản trong Canvas**

* **fillText(text, x, y)**: Dùng để vẽ văn bản vào canvas tại vị trí (x, y) với màu sắc hiện tại của context.
* **strokeText(text, x, y)**: Vẽ văn bản dưới dạng đường viền, tại vị trí (x, y), thay vì tô đầy như fillText.
* **measureText(text)**: Đo kích thước của văn bản được vẽ (ví dụ: chiều rộng của chuỗi văn bản). Phương thức này trả về một đối tượng có thuộc tính width chứa độ dài của văn bản.

## **2. Thuộc tính:**

Để viết một đoạn văn bản, ta chú trọng đến thuộc tính và hàm sau:

### Font định kiểu chữ cho đoạn văn bản.

Để thay đổi kiểu chữ, bạn sử dụng thuộc tính **ctx.font**. Font có thể bao gồm:

* Kích thước chữ (px, em, pt, …).
* Loại phông chữ (ví dụ: "Arial", "Times New Roman").
* Độ dày chữ (normal, bold, bolder).

### FillText(text,x,y): Viết chữ trên canvas.

### StrokeText(text,x,y): Viết chữ trên canvas nhưng không tô màu chữ.

**VD:** <!DOCTYPE html>

<html>

<body>

<canvas id="myCanvas" width="500" height="200" style="background: teal"></canvas>

<script>

var c = document.getElementById("myCanvas");

var ctx = c.getContext("2d");ctx.font = "48px Arial";

ctx.fillStyle='lightsalmon';

ctx.fillText("HelloWorld",120,80);

ctx.font = '50px serif';ctx.strokeStyle='lightsalmon';

ctx.strokeText('Hello world', 120, 150);

</script>

</body>

</html>

**Kết quả:**



**II. Vẽ ảnh trong HTML5 Canvas:**

**1. Vẽ ảnh**

|  |  |
| --- | --- |
| [**drawImage():**](https://webvn.com/drawimage-trong-canvas/) | Vẽ ảnh, tranh hoặc video trong canvas |

**2. Thao tác với điểm ảnh:**

***Thuộc tính:***

Width: trả về chiểu dài của ImageData

Height: trả về chiều cao của ImageData

Data: trả về một đối tượng chứa dữ liệu ảnh của ImageData

***Phương thức:***

CreateImageDat: Tạo một ImageData rỗng

GetImageData: trả về đối tượng ImageData chứa dữ liệu điểm ảnh đã được sao chép trong hình chữ nhật chỉ định

PutImageData: chèn dữ liệu ảnh (lấy từ imagedata) vào canvas

# E. Một số lệnh trong canvas

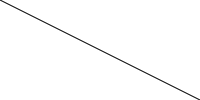
## **I. Các tính năng cơ bản của HTML5 Canvas**

## **1. Vẽ đường thẳng:**

* beginPath() :để khai báo một đường thẳng mới.
* moveTo(x,y) :để xác định điểm bắt đầu.
* lineTo(x,y) :để xác định điểm kết thúc .
* stroke() :để hiển thị đường vẽ với màu đen mặc định.

**VD:**

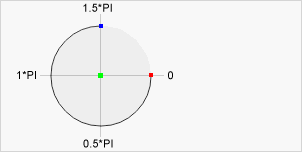
|  |
| --- |
| var c = document.getElementById("myCanvas");  var context = c.getContext("2d");  context.moveTo(0,0);  context.lineTo(200,100);  context.stroke(); |



## **2. Vẽ hình tròn:**

Phương thức arc() dùng để tạo một đường tròn/đường cong (sử dụng để tạo đường tròn hoặc một phần của đường tròn). Sau đó sử dụng thuộc tính stroke() hoặc fill() để thể hiện hình trên canvas.

**Chú ý:** Để tạo một đường tròn với arc(): Thiết lập góc bắt đầu là 0 và góc kết thúc là 2\*Math. PI.



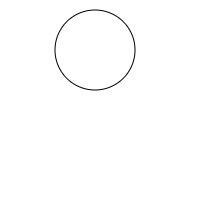
* ctx.arc(x ,y ,r ,sAngle ,eAngle ,counterclockwise) dùng để tạo một đường tròn/đường cong
* x: Tọa độ X tâm của đường tròn

y: Tọa độ Y tâm của đường tròn

* r: Bán kính đường tròn
* sAngle: Góc bắt đầu (0 là vị trí góc 3 trên cung của đường tròn)
* eAngle: Góc kết thúc
* counterclockwise: Tùy chọn. Xác định hình ngược chiều kim đồng hồ hay xuôi chiều kim đồng hồ. Mặc định là false tức là xuôi chiều kim đồng hồ, nếu là true sẽ ngược chiều kim đồng hồ.

**VD:**

|  |
| --- |
| var c=document.getElementById("myCanvas");  var ctx=c.getContext("2d");  ctx.beginPath();  ctx.arc(95,50,40,0,2\*Math.PI);  ctx.stroke(); |



## **3. Vẽ hình chữ nhật:**

Để vẽ hình chữ nhật ta sẽ dùng hai phương pháp sau:

* FillRect(x,y,width,height): vẽ hình chữ nhật với gốc tọa độ và kích thước hình chữ nhật (Dùng để vẽ hình chữ nhật được fill)
* FillStyle: đổ màu cho hình chữ nhật vừa vẽ

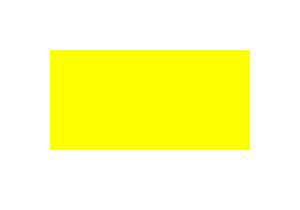
*Chú thích:*

* x và y: xác định vị trí trên canvas (liên quan tới điểm gốc) của góc trên cùng bên trái của hình chữ nhật
* width và height:là độ rộng và chiều cao của hình chữ nhật.

*Ví dụ:*

|  |
| --- |
| context.fillStyle=’yellow’;  context.fillRect(50,50,200,100); |

*Minh họa hình vẽ:*



## **4. Vẽ Text:**

* **Fill Text**

Phương thức **fillText()** sử dụng để vẽ chữ đã được đổ màu trong canvas ,mặc định là màu đen.

***Cú pháp :***

|  |
| --- |
| **Context.fillText(text,x,y,maxWidth);** |

***Trong đó:***

• text: Nội dung để hiển thị trên Canvas.

• x: Tọa độ X điểm bắt đầu viết chữ.

• y: Tọa độ Y điểm bắt đầu viết chữ.

• maxWidth: Tùy chọn. Độ dài lớn nhất của chữ, đơn vị tính là pixels.

**Ví dụ:**

|  |
| --- |
| context.font = 'italic 40pt Calibri, sans-serif';  context.fillText('Hello World!', 50, 50); |

Minh họa hình vẽ bằng text:

|  |
| --- |
| **Hello World!** |

* **Stroke Text :** text rỗng ruột

Phương thức strokeText() sử dụng để viết chữ( sử dụng viền) trong canvas .Mặc định là màu đen

***Cú pháp:***

|  |
| --- |
| **context.strokeText(text, x, y, max Width);** |

***Trong đó:***

• text: Nội dung để hiển thị trên Canvas.

• x: Tọa độ X điểm bắt đầu viết chữ.

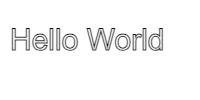
• y: Tọa độ Y điểm bắt đầu viết chữ.

• maxWidth: Tùy chọn. Độ dài lớn nhất của chữ, đơn vị tính là pixels.

*Ví dụ:*

|  |
| --- |
| var c = document.getElementById("myCanvas");  var ctx = c.getContext("2d");  ctx.font = "30px Arial";  ctx.strokeText("Hello World", 10, 50); |

Minh họa vẽ text rỗng ruột:



# **II. Các tính năng nâng cao của HTML5 Canvas**

1. **Các hiệu ứng hình ảnh và đồ họa động**
   1. Vẽ các hình ảnh động (animation) sử dụng **requestAnimationFrame**.
   2. Tạo chuyển động và các hiệu ứng đặc biệt như chuyển động lắc, hiệu ứng mờ, và các hình ảnh động cơ bản.
2. **Tính toán và xử lý ảnh trên Canvas**
   1. Sử dụng **getImageData() và putImageData()** để truy xuất và thay đổi dữ liệu pixel trên Canvas.
   2. Ví dụ về thay đổi màu sắc, độ sáng và các hiệu ứng ảnh.
   3. Phân tích dữ liệu ảnh để tạo các bộ lọc ảnh đơn giản (filtering).
3. **Canvas và sự kiện (Event Handling)**
   1. Các sự kiện cơ bản như **click, mousemove, mousedown** để tương tác với người dùng trên Canvas.
   2. Cách xử lý các sự kiện người dùng như vẽ tự do, kéo thả đối tượng, v.v.
4. **Sử dụng Canvas cho các trò chơi web (Game Development)**
   1. Tạo các trò chơi đơn giản như trò chơi bắn súng, chạy đua sử dụng HTML5 Canvas.
   2. Phối hợp với các thư viện như Phaser.js để phát triển trò chơi.

**Một vài ví dụ vẽ hình bằng canvas:**

var canvas = document.getElementById('canvas');

if (canvas.getContext) {

var ctx = canvas.getContext('2d');

ctx.fillStyle = "rgb(255,0,0)";

ctx.fillRect (0, 0, 500, 300);

ctx.beginPath();

ctx.fillStyle = "yellow";

ctx.moveTo(250,80);

ctx.lineTo(200,200);

ctx.lineTo(320,125);

ctx.lineTo(180,125);

ctx.lineTo(300,200);

ctx.lineTo(250,80);

ctx.closePath();

ctx.fill();

}

**Minh họa :**



# **F.Màu Gradient:**

## **1. Phân loại:**

**- Linear Gradient** (Gradient tuyến tính): Là gradient có sự chuyển màu từ một điểm này sang một điểm khác theo một hướng nhất định.

**- Radial Gradient** (Gradient hình tròn): Là gradient có sự chuyển màu từ một điểm trung tâm ra ngoài, thường có dạng tròn.

## **2. Cách tạo Gradient:**

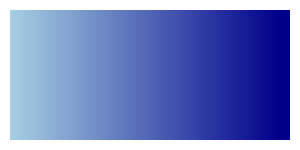
* **createLinearGradient** **(x0, y0, x1, y1)** với tham số x0, y0 là điểm bắt đầu và x1, y1 là điểm kết thúc.
* **createRadialGradient(x0, y0, r0, x1, y1, r1)** với tham số x0, y0, r0 là điểm và bán kính của vòng tròn bắt dầu và x1, y1, r1 là điểm và bán kính của vòng tròn kết thúc.

## **3. Ví dụ**

* Dùng **createLinearGradient()**:

|  |
| --- |
| const c = document.getElementById("myCanvas"); const ctx = c.getContext("2d");  // Create linear gradient const grad=ctx.createLinearGradient(0,0, 280,0); grad.addColorStop(0, "lightblue"); grad.addColorStop(1, "darkblue");  // Fill rectangle with gradient ctx.fillStyle = grad; ctx.fillRect(10,10, 280,130); |

* Minh họa **Linear Gradient**



* Dùng **createRadialGradient()**:

|  |
| --- |
| var c=document.getElementById("myCanvas");  var ctx=c.getContext("2d");  // Create gradient  var grd=ctx.createRadialGradient(75,50,5,90,60,100);  grd.addColorStop(0,”red”);  grd.addColorStop(1,”white”);  // Fill with gradient  ctx.fillStyle=grd;  ctx.fillRect(10,10,150,80); |

Minh họa **Radial Gradient**:



### **Các phương thức quan trọng khi làm việc với Gradient:**

* **addColorStop(position, color)**: Thêm một điểm dừng màu vào gradient. position là tỷ lệ (từ 0 đến 1) xác định vị trí của điểm dừng, và color là màu sắc bạn muốn tại vị trí đó.
* **fillStyle**: Chỉ định màu hoặc gradient cho các thao tác vẽ, như fillRect, fillText, v.v.
* **strokeStyle**: Tương tự fillStyle, nhưng dùng cho việc vẽ các đường viền (outline) như strokeRect, strokeText, v.v.

# **G. Một số thư viện phổ biến**

[**http://raphaeljs.com/**](http://raphaeljs.com/) hỗ trợ xuất ra SVG và Canvas

[**http://paperjs.org/**](http://paperjs.org/) hỗ trợ lập trình canvas rất tốt, có canvascript đặc thù để tạo nhanh các đối tượng đồ hoạ

[**http://fabricjs.com/**](http://fabricjs.com/) rất tốt, chạy nhanh hơn raphaeljs. Xem <http://fabricjs.com/raphael-vs-fabric-complex/>

[**http://kineticjs.com/**](http://kineticjs.com/) không những là một thư viện đồ hoạ mà nó còn hỗ trợ cả lập trình games, hoạt hình nhiều lớp với

[**http://d3js.org/**](http://d3js.org/) thư viện hiển thị mô hình hoá dữ liệu (data visualisation) rất phù hợp để vẽ đồ thị phức tạp trong các ứng dụng web reporting, data mining, statistic

|  |
| --- |
| **H. Những loại ứng dụng nào bạn có thể tạo với HTML5 Canvas?** |

* **Gaming**: Với tập hợp những tính năng của canvas bạn có thể tạo mọi thể loại game 2D và 3D.
* **Quảng cáo**: Canvas là một sự thay thế tuyệt vời cho Banner hoặc quảng cáo dựa trên Flash để thu hút người mua.
* **Biểu diễn dữ liệu**: Dựa trên tập dữ liệu đã được thu thập bạn có thể tạo nên những biểu đồ với element canvas.
* **Giáo dục và đào tạo**: Canvas có thể sử dụng để tạo nên trải nghiệm học tập thú vị và hiệu quả bằng sự kết hợp giữa text, ảnh, video và audio.
* **Nghệ thuật và trang trí**: với canvas bạn có thể vẽ mọi loại hình họa nghệ thuật và trang trí.

# **K. Tương lai của HTML5 Canvas và các công nghệ liên quan**

1. **Canvas và WebGL**
   1. Giới thiệu về WebGL và sự kết hợp giữa Canvas và WebGL để tạo ra các ứng dụng 3D trong trình duyệt.
2. **Các cải tiến trong HTML5 Canvas**
   1. Các tính năng mới và dự báo về sự phát triển của Canvas trong tương lai, bao gồm các công nghệ như WebAssembly và WebVR.
3. **Ứng dụng Canvas trong tương lai**
   1. Các xu hướng mới, như việc sử dụng Canvas trong các ứng dụng thực tế ảo (VR) và tăng cường (AR).

***Triển vọng và tương lai của HTML5 Canvas*:**

* Với sự phát triển mạnh mẽ của công nghệ web, HTML5 Canvas có tiềm năng phát triển mạnh mẽ trong tương lai. Cùng với sự xuất hiện của các công nghệ bổ sung như WebAssembly và WebGL 2.0, Canvas sẽ trở nên mạnh mẽ và hiệu quả hơn trong việc xử lý đồ họa phức tạp. Sự tiến bộ của công nghệ AI và machine learning cũng mở ra những khả năng mới cho việc tích hợp Canvas vào các ứng dụng thông minh, giúp tự động hóa việc tạo ra và chỉnh sửa hình ảnh.
* Hơn nữa, sự gia tăng trong việc sử dụng các thiết bị di động và các ứng dụng web sẽ thúc đẩy sự phát triển của Canvas, bởi vì nó hỗ trợ tốt trên nhiều nền tảng mà không yêu cầu cài đặt thêm plugin. Các trình duyệt ngày càng được tối ưu hóa để hỗ trợ tốt hơn cho HTML5 và Canvas, giúp việc phát triển ứng dụng trở nên dễ dàng và tiện lợi hơn.

# **L. Kết luận**

* ***HTML5 Canvas:*** là một công nghệ mạnh mẽ và quan trọng trong phát triển web hiện đại, mở ra nhiều khả năng sáng tạo và linh hoạt cho các nhà phát triển. Việc sử dụng Canvas trong việc tạo ra các đồ họa 2D và 3D, vẽ hình ảnh, tạo hiệu ứng động, và xây dựng các ứng dụng tương tác không chỉ cải thiện trải nghiệm người dùng mà còn giúp tăng tính tương tác, tính trực quan và khả năng sử dụng trên mọi thiết bị.
* ***Tầm quan trọng của việc học và sử dụng HTML5 Canvas*:** Với tất cả những tính năng mạnh mẽ và ứng dụng đa dạng mà nó mang lại, việc học và sử dụng HTML5 Canvas trở nên vô cùng quan trọng đối với các nhà phát triển web hiện nay. Canvas không chỉ là công cụ để vẽ đồ họa, mà còn là chìa khóa để mở rộng khả năng sáng tạo và phát triển các ứng dụng web hiện đại. Các nhà phát triển có thể tận dụng Canvas để cải thiện trải nghiệm người dùng, xây dựng các ứng dụng thú vị, sáng tạo và hiệu quả.
* Tóm lại, HTML5 Canvas đã và đang trở thành một công cụ vô cùng quan trọng trong việc phát triển ứng dụng web hiện đại. Nó không chỉ giúp các nhà phát triển vẽ đồ họa 2D và 3D mà còn tạo ra các trải nghiệm tương tác thú vị, cải thiện hiệu suất và tính linh hoạt của ứng dụng. Mặc dù còn một số hạn chế, nhưng với sự phát triển liên tục của công nghệ và các công cụ hỗ trợ, HTML5 Canvas chắc chắn sẽ có một tương lai sáng lạn và đóng góp to lớn vào sự phát triển của web và các ứng dụng trực tuyến.