Canvas

Reference: https://developer.mozilla.org/en-US/docs/Web/API/Canvas\_API/Tutorial#:~:text=is%20an%20HTML%20element,and%20not%20so%20simple)%20animations.&text=This%20tutorial%20describes%20how%20to,graphics%2C%20starting%20with%20the%20basics.

[**<canvas>**](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/canvas) is an [HTML](https://developer.mozilla.org/en-US/docs/Web/HTML) element which can be used to draw graphics via scripting (usually [JavaScript](https://developer.mozilla.org/en-US/docs/Glossary/JavaScript)). This can, for instance, be used to draw graphs, combine photos, or create simple (and [not so simple](https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API/A_basic_ray-caster)) animations.

## The <canvas> element:

<canvas id="tutorial" width="150" height="150"></canvas>

At first sight a [<canvas>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/canvas) looks like the [<img>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img) element, with the only clear difference being that it doesn't have the src and alt attributes. Indeed, the <canvas> element has only two attributes, [width](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/canvas#attr-width) and [height](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/canvas#attr-height). The element can be sized arbitrarily by [CSS](https://developer.mozilla.org/en-US/docs/Glossary/CSS), but during rendering the image is scaled to fit its layout size: if the CSS sizing doesn't respect the ratio of the initial canvas, it will appear distorted, in this case try to set directly from the html file the width and height inside the canvas tag.

The <canvas> element can be styled just like any normal image ([margin](https://developer.mozilla.org/en-US/docs/Web/CSS/margin), [border](https://developer.mozilla.org/en-US/docs/Web/CSS/border), [background](https://developer.mozilla.org/en-US/docs/Web/CSS/background)…).

## The rendering context

The [<canvas>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/canvas) element creates a fixed-size drawing surface that exposes one or more **rendering contexts**, which are used to create and manipulate the content shown. In this tutorial, we focus on the 2D rendering context.

The canvas is initially blank. To display something, a script first needs to access the rendering context and draw on it. The [<canvas>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/canvas) element has a method called [getContext()](https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/getContext), used to obtain the rendering context and its drawing functions. getContext() takes one parameter, the type of context. For 2D graphics, such as those covered by this tutorial, you specify "2d" to get a [CanvasRenderingContext2D](https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D).

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

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

The first line in the script retrieves the node in the DOM representing the [<canvas>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/canvas) element by calling the [document.getElementById()](https://developer.mozilla.org/en-US/docs/Web/API/Document/getElementById) method. Once you have the element node, you can access the drawing context using its getContext() method.

Nota: Il DOM (Document Object Model) è una rappresentazione ad oggetti della pagina web(HTML), che può essere modificata con linguaggi di scripting come JavaScript. Il DOM non è un linguaggio di programmazione, ma senza di esso, il linguaggio JavaScript non potrebbe avere nessun modello o conoscenza sulla pagina web HTML o XML e le loro componenti (ad es. elementi). Il DOM è stato progettato per essere indipendente da qualsiasi particolare linguaggio di programmazione, rendendo la rappresentazione strutturale del documento disponibile da un'unica API coerente