



# Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

## 03-07 etiquetas HTML svg

Nombre		Curso	
Apellidos		Fecha	

## Qué es SVG

SVG (Scalable Vector Graphics = Gráficos Vectoriales Escalables) es un lenguaje de marcas creado por el W3C y dirigido a la representación de gráficos vectoriales (dibujos y texto).

En la lección sobre Historia de la Web se comentan las recomendaciones de SVG publicadas o en preparación por el W3C.

En un gráfico vectorial, los elementos de la imagen están definidos como formas elementales (líneas, rectángulos, círculos, curvas, polígonos, etc.), definidas mediante etiquetas similares a las del HTML. Por ello SVG no es un formato adecuada para fotografías, pero es idóneo para cualquier tipo de dibujo, técnico o artístico.

Las ventajas de SVG son muchas:

- Las imágenes SVG se pueden ampliar a cualquier escala sin perder calidad, ya que están definidas como formas que el navegador dibuja con la precisión necesaria.
- Las imágenes SVG suelen ocupar poco espacio, ya que están definidas mediante etiquetas. El tamaño en KB de la imagen es además independiente del tamaño con el que se ve en la página web.
- Las imágenes se pueden reutilizar y combinar fácilmente ya que basta con copiar el código fuente de una imagen a otra.
- Las imágenes se pueden modificar de forma dinámica mediante hojas de estilo o javascript porque forman parte de la página web.
- Un gráfico SVG puede incluirse en una página web de dos maneras, como objeto interno o como objeto externo.

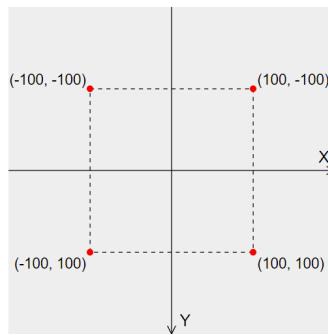
Desgraciadamente, el uso de SVG se vio frenado porque Internet Explorer no fue capaz de mostrar gráficos SVG hasta IE 9 y de forma deficiente. Actualmente (octubre de 2022), los principales obstáculos al uso generalizado de SVG siguen siendo que las implementaciones en los navegadores todavía son incompletas y los potenciales riesgos de seguridad (debido a que las imágenes SVG pueden contener código Javascript, se recomienda no incorporar imágenes SVG sin haber comprobado antes su contenido).

De todas formas, SVG se ha ido imponiendo poco a poco como formato estándar de gráficos vectoriales frente a otros formatos propietarios y numerosos programas de edición de gráficos son capaces de importar y exportar en formato SVG.

## El plano SVG

Los dibujos SVG se definen en un plano infinito en el que el eje X está orientado de izquierda a derecha, como es habitual en Matemáticas, pero en el que el eje Y está orientado hacia abajo, al contrario de lo que es habitual en Matemáticas pero como suele ser habitual en Informática. El dibujo siguiente muestra los ejes de coordenadas, cuatro puntos en rojo y sus coordenadas.

Los valores de las coordenadas SVG no tienen unidades (aunque si el ordenador necesita interpretarlas con unidades, se suelen interpretar como píxeles) y los valores pueden ser tanto enteros como decimales.



Nota: El plano SVG posee ejes de coordenadas, pero son invisibles. En el dibujo anterior se ven unos ejes de coordenadas porque se han dibujado explícitamente.

## La etiqueta <svg>

La etiqueta <svg> engloba toda la imagen SVG e incluye las formas que forman la imagen.

## Los atributos version y xmlns

El atributo `version` de la etiqueta <svg> indica la versión de SVG empleada en el dibujo. En estos apuntes se utiliza la versión 1.1. Este atributo no es obligatorio, pero está recomendado para ayudar a los navegadores (aunque en la práctica parece ser que los navegadores no tienen en cuenta este atributo).

El atributo `xmlns` de la etiqueta <svg> indica el espacio de nombres empleado en el dibujo, es siempre <https://www.w3.org/2000/svg>. Este atributo es obligatorio cuando la imagen SVG se encuentra en un fichero aparte (con la extensión .svg), pero este atributo es optativo cuando la imagen SVG está incluida en una página web.

Aunque no sea necesario, en estos apuntes se aconseja incluir siempre los atributos `version` y `xmlns` como en el ejemplo siguiente:

```
<?xml version="1.0" encoding="utf-8" ?>
<svg xmlns="http://www.w3.org/2000/svg"> ..... />
</svg>
```

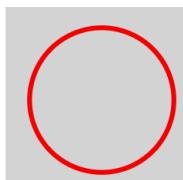
## Los atributos viewBox, width y height

Los atributos viewBox, width y height determinan la porción del plano SVG que se va a mostrar en el dibujo y el tamaño que va a tener ese dibujo en la pantalla. Aunque cada uno se encarga de un aspecto, los tres atributos están muy relacionados y en algunas situaciones el cambio de uno de ellos afecta a los otros.

Los ejemplos siguientes permiten modificar los valores de los atributos que intervienen en cada dibujo y ver cómo influyen en el resultado. En los apartados que se encuentran a continuación de estos dos ejemplos se explican estos atributos en detalle.

En este primer ejemplo, al modificar el tamaño horizontal o vertical del dibujo (width y height) automáticamente se modifica la otra dimensión para conservar la proporción del dibujo. De la misma manera, al modificar el tamaño horizontal o vertical de la porción de plano SVG (tercer y cuarto valor de viewBox) automáticamente se modifica el tamaño del dibujo para conservar la proporción del dibujo.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="200" height="200" viewBox="-100 -100 200 200"
style="background-color: lightgray">
<circle cx="0" cy="0" r="75" fill="none" stroke="red" stroke-width="5" />
</svg>
```



En este segundo ejemplo, al modificar cualquier parámetro no se modifica ningún parámetro automáticamente. Al modificar la proporcionalidad entre el tamaño del dibujo y el tamaño de la zona visible del plano SVG, el resultado es un poco más difícil de interpretar, como se explica más adelante en el apartado Atributos width y height no proporcionales a viewBox.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="200" height="200" viewBox="-100 -100 200 200"
style="background-color: lightgray">
<circle cx="0" cy="0" r="75" fill="none" stroke="red" stroke-width="5" />
</svg>
```

## El atributo viewBox

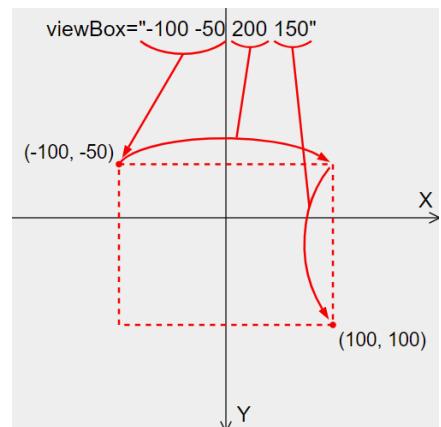
El atributo viewBox establece la porción del plano SVG que muestra la imagen. Este atributo se establece con cuatro valores:

**viewBox = (x0 y0 width height)**

**viewBox = <min-x> <min-y> <width> <height>**

- primer valor: abscisa (X) de la esquina superior izquierda de la porción del plano que muestra la imagen
- segundo valor: ordenada (Y) de la esquina superior izquierda de la porción del plano que muestra la imagen
- tercer valor: width de la porción del plano que muestra la imagen
- cuarto valor: height de la porción del plano que muestra la imagen

El ejemplo siguiente muestra con una línea roja a trazos la porción del plano SVG que se mostraría con unos valores determinados en el atributo viewBox.



- La zona visible es independiente de los elementos dibujados, es decir, sólo se mostrarán los elementos (o la parte de los elementos) que se encuentren en la zona visible definida por el atributo viewBox.
- Los ejemplos siguientes muestran tres dibujos con los mismos elementos SVG: los ejes de coordenadas y un círculo de radio 100 centrado en el origen, pero con diferentes atributos viewBox
- En el primer ejemplo, se ve todo el círculo porque la zona visible definida por viewBox empieza arriba a la izquierda en el punto (-100, -100) y tiene un tamaño de 200 x 200. El tamaño de la imagen es también de 200 x 200.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="200" height="200" viewBox="-100 -100 200 200"
style="background-color: lightgray">
<circle cx="0" cy="0" r="100" fill="none" stroke="red" stroke-width="1"/>
<line x1="-200" y1="0" x2="200" y2="0" stroke="black" stroke-width="1"/>
<line x1="0" y1="-200" x2="0" y2="200" stroke="black" stroke-width="1"/>
</svg>
```

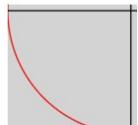
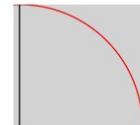
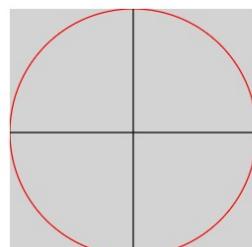
Si en ese mismo dibujo la zona visible definida por viewBox empieza en el punto (-5, -5) y tiene un tamaño de 105 x 105 sólo se verá el cuadrante inferior derecho (valores X e Y positivos). El tamaño de la imagen es también de 105 x 105 para que coincida con el tamaño de la zona visible.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="105" height="105" viewBox="-5 -5 105 105"
style="background-color: lightgray">
<circle cx="0" cy="0" r="100" fill="none" stroke="red" stroke-width="1"/>
<line x1="-200" y1="0" x2="200" y2="0" stroke="black" stroke-width="1"/>
<line x1="0" y1="-200" x2="0" y2="200" stroke="black" stroke-width="1"/>
</svg>
```

Si quisieramos mostrar el cuadrante superior izquierdo (valores X e Y negativos), la zona visible podría empezar en el punto (-100, -100) y tener un tamaño de 105 x 105. El tamaño de la imagen sería también de 105 x 105 para que coincida con el tamaño de la zona visible.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="105" height="105" viewBox="-100 -100 105 105"
style="background-color: lightgray">
<circle cx="0" cy="0" r="100" fill="none" stroke="red" stroke-width="1"/>
<line x1="-200" y1="0" x2="200" y2="0" stroke="black" stroke-width="1"/>
<line x1="0" y1="-200" x2="0" y2="200" stroke="black" stroke-width="1"/>
</svg>
```

Ejemplos anteriores



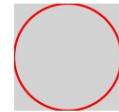
#### Los atributos width y height de <svg> con viewBox

El atributo viewBox establece la porción del plano SVG que se muestra en la imagen, mientras que los atributos width y height establecen el width y el height de la imagen en la página web. Si no se indican unidades, los valores se interpretan en px.

Así, una misma imagen en el plano SVG se puede ver en la página web más o menos grande, como muestran los ejemplos siguientes:

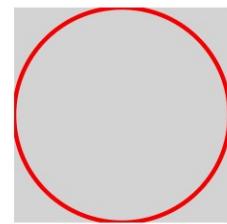
En este primer ejemplo, el tamaño de la imagen en la página web coincide con el tamaño de la zona visible del plano SVG.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="-50 -50 100 100"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="50" fill="none" stroke="red" stroke-width="2" />
</svg>
```



En este segundo ejemplo, la zona visible del plano SVG es la misma que antes, pero la imagen en la página web es cuatro veces más grande (el doble en horizontal y el doble en vertical). Al ampliar el dibujo, lógicamente el trazo de la circunferencia se ve más grueso que antes.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="200" height="200" viewBox="-50 -50 100 100"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="50" fill="none" stroke="red" stroke-width="2" />
</svg>
```



En este tercer ejemplo, la zona visible del plano SVG es la misma que antes, pero la imagen en la página web es cuatro veces más pequeña que en el primer ejemplo (la mitad en horizontal y la mitad en vertical). Al haber reducido el dibujo, lógicamente el trazo de la circunferencia se ve más fino que antes.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="50" height="50" viewBox="-50 -50 100 100"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="50" fill="none" stroke="red" stroke-width="2" />
</svg>
```





# Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

El tamaño de la imagen en la página web depende únicamente de los atributos width y height de <svg>, independientemente del tamaño que tenga el dibujo en el plano SVG:

En este primer ejemplo, el tamaño de la imagen coincide con el tamaño de la zona visible del plano SVG:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="-50 -50 100 100"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="50" fill="none" stroke="red" stroke-width="2" />
</svg>
```

En este segundo ejemplo, la imagen se ve del mismo tamaño que en el ejemplo anterior, aunque el dibujo en el plano SVG sea cuatro veces más grande (el doble en horizontal y en vertical). La circunferencia se ve más fina porque al reducir los valores de width y height es como si hiciéramos zoom alejándonos del dibujo y lógicamente la circunferencia se ve más fina.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="-100 -100 200 200"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="100" fill="none" stroke="red" stroke-width="2" />
</svg>
```

Para que el trazo de la circunferencia se viera del mismo grosor que en el primer ejemplo, deberíamos aumentar el grosor del trazo en la misma proporción que hemos aumentado el tamaño del dibujo en el plano SVG.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="-100 -100 200 200"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="100" fill="none" stroke="red" stroke-width="4" />
</svg>
```

En este tercer ejemplo, el círculo se ve del mismo tamaño que en el primer ejemplo, aunque el dibujo en el plano SVG sea cuatro veces más pequeño que el del primer ejemplo (la mitad en horizontal y en vertical). La circunferencia se ve más gruesa porque al aumentar los valores de width y height es como si hiciéramos zoom acercándonos al dibujo y lógicamente la circunferencia se ve más gruesa.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="-25 -25 50 50"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```

Para que el trazo de la circunferencia se viera del mismo grosor que en el primer ejemplo, deberíamos reducir el grosor del trazo en la misma proporción que el tamaño del dibujo en el plano SVG.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="-25 -25 50 50"
      style="background-color: lightgray">
  <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="1" />
</svg>
```



# Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

## Los atributos width y height de <svg> sin viewBox

Por precaución se recomienda escribir siempre el atributo viewBox, pero no es un atributo obligatorio. **Si no se incluye el atributo viewBox, los atributos width y height establecen tanto la porción del plano SVG que se muestra en la imagen como el width y el height de la imagen en la página web.** La porción de plano empieza en el origen del plano SVG (0, 0) y su tamaño son los valores de width y height. Si no se indican unidades, los valores se interpretan en px.

Si los elementos del dibujo se encuentran en esa zona, el dibujo se podría ver correctamente, aunque no haya atributo viewBox:

con viewBox:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="0 0 100 100"
      style="background-color: lightgray">
  <circle cx="50" cy="50" r="50" fill="none" stroke="red" stroke-width="1" />
</svg>
```

sin viewBox:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100"
      style="background-color: lightgray">
  <circle cx="50" cy="50" r="50" fill="none" stroke="red" stroke-width="1" />
</svg>
```

En el ejemplo siguiente, el círculo se encuentra más a la derecha en el plano SVG que en los ejemplos anteriores, pero como el viewBox también está desplazado a la derecha, el círculo se ve en el centro del dibujo:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="50 0 100 100"
      style="background-color: lightgray">
  <circle cx="100" cy="50" r="50" fill="none" stroke="red" stroke-width="1" />
</svg>
```

Si en el ejemplo anterior se elimina el atributo viewBox, el navegador muestra la zona cuadrada desde el origen, por lo que el dibujo sólo incluye la mitad del círculo:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100"
      style="background-color: lightgray">
  <circle cx="100" cy="50" r="50" fill="none" stroke="red" stroke-width="1" />
</svg>
```

## Atributos width y height no proporcionales a viewBox

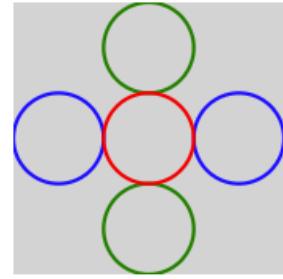
Lo lógico es que las proporciones de la imagen sean las mismas que las proporciones de la porción del plano SVG que se muestra, es decir, que la proporción entre el atributo width y el tercer valor del atributo viewBox sea la misma que entre el atributo height y el cuarto valor del atributo viewBox.

**Es una situación parecida a lo que ocurre al mostrar imágenes mediante la etiqueta img, en el que los atributos width y height deben tener la misma proporción que la imagen que se está mostrando. Pero así como en el caso de la etiqueta img si los valores no tienen la misma proporción la imagen se deforma, en el caso de la etiqueta svg la imagen no se deforma, sino que se modifica la zona mostrada.**

El criterio que siguen los navegadores es dibujar una zona del plano SVG con la proporción de los atributos width y height que incluya la zona definida por el atributo viewBox.

Consideremos la siguiente imagen:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="150" height="150" viewBox="-75 -75 150 150"
      style="background-color: lightgray">
    <circle cx="-50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
    <circle cx="50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
    <circle cx="0" cy="-50" r="25" fill="none" stroke="green" stroke-width="2" />
    <circle cx="0" cy="50" r="25" fill="none" stroke="green" stroke-width="2" />
    <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```



Vamos a mostrar una parte de esta imagen (sobre esta imagen modificaremos los atributos en los ejemplos siguientes):

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="50" viewBox="-50 -25 100 50"
      style="background-color: lightgray">
    <circle cx="-50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
    <circle cx="50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
    <circle cx="0" cy="-50" r="25" fill="none" stroke="green" stroke-width="2" />
    <circle cx="0" cy="50" r="25" fill="none" stroke="green" stroke-width="2" />
    <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```

Si ampliamos solamente el width width de la imagen, podemos comprobar como el navegador aumenta la zona mostrada aunque no hayamos modificado el atributo viewBox:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="150" height="50" viewBox="-50 -25 100 50"
      style="background-color: lightgray">
    <circle cx="-50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
    <circle cx="50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
    <circle cx="0" cy="-50" r="25" fill="none" stroke="green" stroke-width="2" />
    <circle cx="0" cy="50" r="25" fill="none" stroke="green" stroke-width="2" />
    <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```



# Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

Si ampliamos solamente el height height de la imagen, podemos comprobar como el navegador aumenta la zona mostrada aunque no hayamos modificado el atributo viewBox:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="100" viewBox="-50 -25 100 50"
      style="background-color: lightgray">
  <circle cx="-50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="0" cy="-50" r="25" fill="none" stroke="green" stroke-width="2" />
  <circle cx="0" cy="50" r="25" fill="none" stroke="green" stroke-width="2" />
  <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```

Si reducimos solamente el width width de la imagen, podemos comprobar como el navegador aumenta la zona mostrada de manera que tenga la proporción de la imagen (1:1, cuadrada) aunque no hayamos modificado el atributo viewBox:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="50" height="50" viewBox="-50 -25 100 50"
      style="background-color: lightgray">
  <circle cx="-50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="0" cy="-50" r="25" fill="none" stroke="green" stroke-width="2" />
  <circle cx="0" cy="50" r="25" fill="none" stroke="green" stroke-width="2" />
  <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```

Si reducimos solamente el height height de la imagen, podemos comprobar como el navegador aumenta la zona mostrada de manera que tenga la proporción de la imagen (4:1) aunque no hayamos modificado el atributo viewBox:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="25" viewBox="-50 -25 100 50"
      style="background-color: lightgray">
  <circle cx="-50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="0" cy="-50" r="25" fill="none" stroke="green" stroke-width="2" />
  <circle cx="0" cy="50" r="25" fill="none" stroke="green" stroke-width="2" />
  <circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```

Para evitar problemas cuando queremos mostrar únicamente la porción del plano SVG definida por el atributo viewBox pero a un tamaño distinto, la mejor solución es indicar únicamente uno de los atributos width o height, ya que en ese caso el navegador ajusta el atributo faltante de acuerdo con la proporción definida por el atributo viewBox.

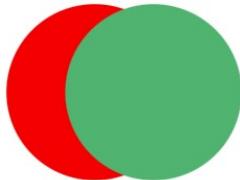
Consideremos la misma imagen del ejemplo anterior:

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="100" height="50" viewBox="-50 -25 100 50"
      style="background-color: lightgray">
  <circle cx="-50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="50" cy="0" r="25" fill="none" stroke="blue" stroke-width="2" />
  <circle cx="0" cy="-50" r="25" fill="none" stroke="green" stroke-width="2" />
  <circle cx="0" cy="50" r="25" fill="none" stroke="green" stroke-width="2" />
```

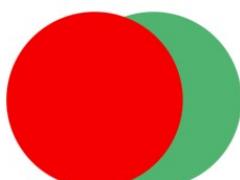
```
<circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="2" />
</svg>
```

## Elementos superpuestos

Los elementos de un dibujo SVG se van dibujando en el mismo orden en que se encuentran en el código fuente. Si dos elementos se superponen, uno se dibujará sobre el otro, como muestran los ejemplos siguientes.

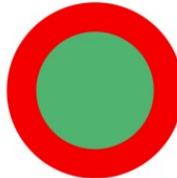


```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="220" height="170" viewBox="-10 -10 220 170">
<circle cx="75" cy="75" r="75" fill="red" />
<circle cx="125" cy="75" r="75" fill="mediumseagreen" />
</svg>
```

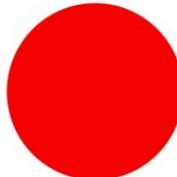


```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="220" height="170" viewBox="-10 -10 220 170">
<circle cx="125" cy="75" r="75" fill="mediumseagreen" />
<circle cx="75" cy="75" r="75" fill="red" />
</svg>
```

Un elemento puede así ocultar completamente a otro, como muestran los ejemplos siguientes.



```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="220" height="170" viewBox="-10 -10 220 170">
<circle cx="100" cy="75" r="75" fill="red" />
<circle cx="100" cy="75" r="50" fill="mediumseagreen" />
</svg>
```



```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="220" height="170" viewBox="-10 -10 220 170">
<circle cx="100" cy="75" r="50" fill="mediumseagreen" />
<circle cx="100" cy="75" r="75" fill="red" />
</svg>
```

Una manera de no ocultar los objetos superpuestos es utilizar el relleno transparente (**fill="none"**), como muestran los ejemplos siguientes. El elemento con relleno transparente deberá tener color de trazo (**stroke="color"**) para que sea visible.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="220" height="170" viewBox="-10 -10 220 170">
<circle cx="100" cy="75" r="75" fill="white" stroke="red" />
<circle cx="100" cy="75" r="50" fill="mediumseagreen" />
</svg>
```

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
      width="220" height="170" viewBox="-10 -10 220 170">
<circle cx="100" cy="75" r="50" fill="mediumseagreen" />
<circle cx="100" cy="75" r="75" fill="white" stroke="red" />
</svg>
```

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
```



# Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

```
width="220" height="170" viewBox="-10 -10 220 170">
<circle cx="100" cy="75" r="50" fill="mediumseagreen" />
<circle cx="100" cy="75" r="75" fill="none" stroke="red" />
</svg>
```

## Márgenes en viewBox

Para asegurarse de que todo el dibujo sea visible, es conveniente elegir un viewBox un poco más grande que la zona ocupada por los elementos del dibujo.

En el ejemplo siguiente los centros de los círculos está situado en la esquina superior izquierda y en la esquina inferior derecha de la zona visible y por eso sólo puede verse la cuarta parte de ambos círculos.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="100" height="100" viewBox="0 0 100 100">
<circle cx="0" cy="0" r="10" fill="red" />
<circle cx="100" cy="100" r="10" fill="mediumseagreen" />
</svg>
```

Haciendo que la zona visible empiece más arriba a la izquierda y termine más abajo a la derecha, los círculos se verían completos. En este caso, hemos ampliado la zona 10 unidades por los cuatro lados, por lo que el tamaño de la zona visible pasa de 100 a 120.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="120" height="120" viewBox="-10 -10 120 120">
<circle cx="0" cy="0" r="10" fill="red" />
<circle cx="100" cy="100" r="10" fill="mediumseagreen" />
</svg>
```

## SVG con en HTML5

### 1. Cómo utilizar un SVG con <img>

Este método es la forma más sencilla de añadir imágenes SVG a una página web. Para utilizar este método, añade el elemento <img> a tu documento HTML y haz referencia a él en el atributo src, así:

```
<img src = "happy.svg" alt="Mi SVG feliz"/>
```

Suponiendo que hayas descargado la imagen SVG de unDraw y la hayas renombrado como happy.svg, puedes seguir adelante y añadir el fragmento de código anterior en tu documento HTML.

Si lo has hecho todo correctamente, tu página web debe tener el mismo aspecto que la siguiente demostración.

Cuando se añade una imagen SVG mediante la etiqueta <img> sin especificar el tamaño, se asume el tamaño del archivo SVG original.

Por ejemplo, en la demostración anterior, no modifiqué el tamaño de la imagen SVG, por lo que asumió su tamaño original (que era un width de 915.11162px y una height de 600.53015px).



# Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<meta http-equiv="X-UA-Compatible" content="ie=edge" />
</head>

</html>
```

ejemplo.svg

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
width="200" height="200" viewBox="-100 -100 200 200"
style="background-color: lightgray">
<circle cx="0" cy="0" r="75" fill="none" stroke="red" stroke-width="5" />
</svg>
```

Aunque podemos cambiar el tamaño de las imágenes SVG añadidas a través de la etiqueta `<img>` siguen existiendo algunas restricciones si quieras hacer cambios de estilo importantes en la imagen SVG.

## 2. utilizar imágenes SVG inline

Las imágenes SVG pueden escribirse directamente en el documento HTML mediante la etiqueta `<img> </img>`.

Para ello, abre la imagen SVG en VS code o en tu IDE preferido, copia el código y pégalo dentro del elemento `<body>` en tu documento HTML.

```
<body>
<img version="1.1" xmlns="http://www.w3.org/2000/svg"
width="200" height="200" viewBox="-100 -100 200 200"
style="background-color: lightgray">
<circle cx="0" cy="0" r="75" fill="none" stroke="red" stroke-width="5" />
<circle cx="0" cy="0" r="50" fill="none" stroke="red" stroke-width="5" />
<circle cx="0" cy="0" r="25" fill="none" stroke="red" stroke-width="5" />
</img>
</body>
```

Si lo has hecho todo correctamente, tu página web debe ser exactamente igual que la demostración de abajo.

Cuando se utiliza el SVG inline en el documento HTML, se reduce el tiempo de carga porque sirve como una solicitud HTTP.

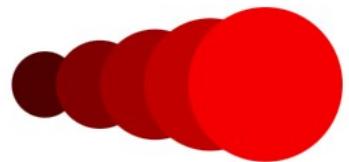
## Practica 03-07 circulos y viewBox

**Ejercicio 1.** Crea cada uno de los ejemplos que aparecen en la practica

**Ejercicio 2.** Crea un svg que aparezca un circulo de tamaño 300 dentro de otro tamaño 200. Con un width 300x300 y un viewport de 250x250

**Ejercicio 2a.** Crea un archivo independiente de svg con el archivo anterior.

**Ejercicio 3.** Crea un svg que aparezcan cinco circulos de colores degradados de rojo (fill:#550000, fill:880000, fill:AA0000, fill:CC0000, fill:FF0000) cuyo rx se vaya desplazando 25 y su radio vaya creciendo de 5 en 5.



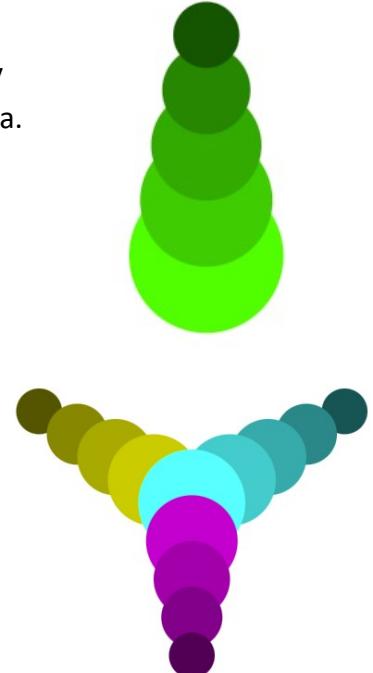
**Ejercicio 4.** Modifica el anterior para que el degradado sea en verdes y se incremente sobre el eje cy y se desplace de 25 en 25m de manera que el más oscuro salga encima de la pila y su radio vaya creciendo de 5 en 5

**Ejercicio 5.** Modifica el anterior para que el degradado sea en amarillos, cyan y purpura y se incremente sobre el eje cy y el cx de manera que salga esta figura.

(ejemplo de códigos de circle)

Este muestra el degradado de los cyanes.

```
cx="225" cy="15" r="15" fill="#005555"
cx="200" cy="30" r="20" fill="#008888"
cx="175" cy="45" r="25" fill="#00AAAA"
cx="150" cy="60" r="30" fill="#00CCCC"
cx="125" cy="75" r="35" fill="#00FFFF"
```



**Ejercicio 6.** Modifica los viewBox para que se vea solo esto . (puedes usar otra figura)

```
width="150" height="150" viewBox="-150 -150 500 500"
width="100" height="100" viewBox="-200 -200 600 600"
```

