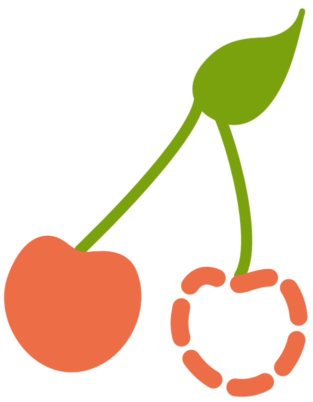
4
Fills and Strokes



`fill` and `stroke` allow us to paint to the interior and border of SVG.

"Paint" refers to the action of applying colors, gradients, or patterns to graphics through `fill` and/or `stroke`.

fill Properties

The `fill` attribute paints the interior of a specific graphical element. This fill can consist of a solid color, gradient, or pattern.

The interior of a shape is determined by examining all subpaths and specifications spelled out within the `fill-rule`.

When filling a shape or path, `fill` will paint open paths as if the last point of the path connected with the first, even though a `stroke` color on this section of the path would not render.

fill-rule

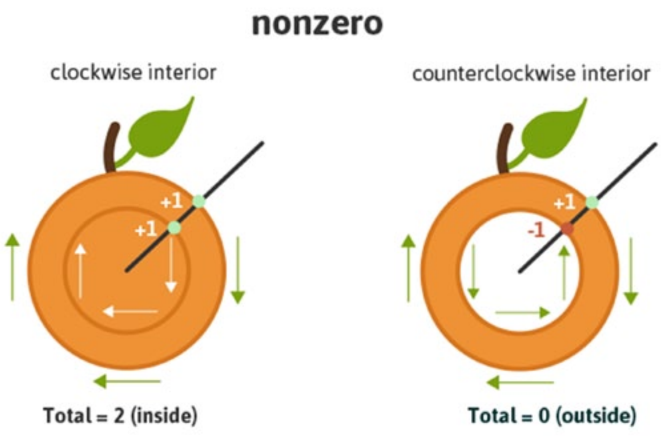
The `fill-rule` property indicates the algorithm to be used in determining which parts of the canvas are included inside the shape. This is not always straightforward when working with more complex intersecting or enclosed paths.

The accepted values here are `nonzero`, `evenodd`, `inherit`.

nonzero

A value of `nonzero` determines the inside of a point on the canvas by drawing a line from the area in question through the entire shape in any direction and then considering the locations where a segment of the shape crosses this line. This starts with zero and adds one each time a path segment crosses the line from left to right and subtracts one each time a path segment crosses the line from right to left.

If the result is zero after evaluating and counting these intersections then the point is outside the path, otherwise it is inside.



Essentially, if the interior path is drawn clockwise it will be considered as "inside", but if drawn counter-clockwise it will be considered "outside" and therefore be excluded from painting.

evenodd

A value of `evenodd` determines the inside of an area on the canvas by drawing a line from that area through the entire shape in any direction and counts the path segments that the line crosses. If this results in an odd number the point is inside, if even the point is outside.

Given the specific algorithm of the `evenodd` rule, the drawing direction of the interior shape in question is irrelevant, unlike with `nonzero`, as we are simply counting the paths as they cross the line.

While this property is not generally necessary, it will allow for greater `fill` control of a complex graphic, as mentioned.

inherit

A value of `inherit` will direct the element to take on the `fill-rule` specified by its parent.

fill-opacity

The `fill-opacity` value refers to the opacity level of the interior paint fill. A value of “0” results in complete transparency, “1” applies no transparency, and values in between represent a percentage-based level of opacity.

Stroke Attributes

There are a number of stroke-related attributes within SVG that allow for the control and manipulation of stroke details. The abilities of these attributes provide for greater control of hand-coded SVG, but also prove convenient when needing to make edits to an existing embedded graphic.

The following examples use an inline SVG of grapes. The attributes being used reside directly within the correlating shape’s element.

stroke

The `stroke` attribute defines the “border” paint of particular shapes and paths.

The following grapes image has a purple stroke: `stroke="#765373"`.

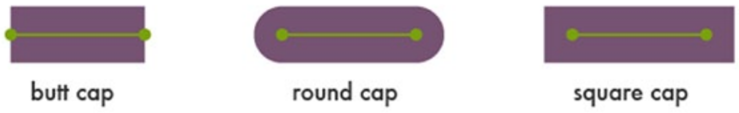
stroke-width

The `stroke-width` value establishes the width of the grape's stroke, which is set to `6px` on the grapes image.

The default value for this attribute is 1. If a percentage value is used, the value is based on the dimensions of the viewport.

stroke-linecap

`stroke-linecap` defines which shape the end of an open path will take and there are four acceptable values: `butt`, `round`, `square`, `inherit`.



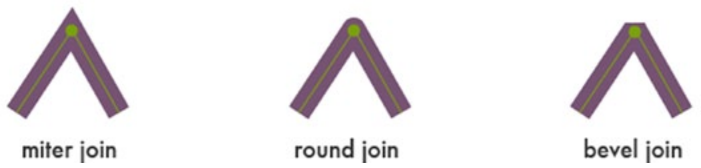
A value of `inherit` will direct the element to take on the `stroke-linecap` specified by its parent.

The stem in the following image has a `stroke-linecap` value of `square`:

```
<svg>
    <!--<path <path for grapes> />-->
    <!--<path stroke-linecap="square" <path for
stem> />-->
    <!--<path <path for leaf> />-->
</svg>
```

stroke-linejoin

`stroke-linejoin` defines how the corners of strokes will look on paths and basic shapes.



Here is a look at the grapes with a `stroke-linejoin` of `"bevel"`:

```
<svg>
    <!--<path stroke-linejoin="bevel" <path for
grapes> />-->
    <!--<path <path for stem> />-->
    <!--<path <path for leaf> />-->
</svg>
```

stroke-miterlimit

When two lines meet at a sharp angle and are set to a `stroke-linejoin="miter"`, the `stroke-miterlimit` attribute allows for the specification of how far this joint/corner extends.

The length of this joint is called the miter length, and it is measured from the inner corner of the line join to the outer tip of the join.

This value is a limit on the ratio of the miter length to the `stroke-width`.

1.0 is the smallest possible value for this attribute.

The first grape image is set to `stroke-miterlimit="1.0"`, which creates a bevel effect. The `stroke-miterlimit` on the second image is set to 4.0.

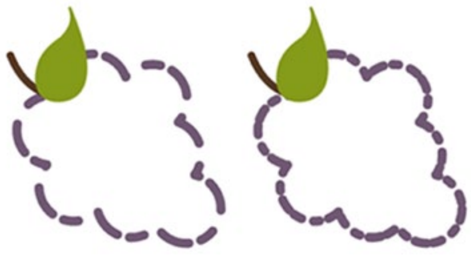
stroke-dasharray

The `stroke-dasharray` attribute turns paths into dashes rather than solid lines.

Within this attribute you can specify the length of the dash as well as the distance between the dashes, separated with commas or whitespace.

If an odd number of values are provided, the list is then repeated to produce an even number of values. For example, `8,6,4` becomes `8,6,4,8,6,4` as shown in the second grapes image below.

Placing just one number within this value results in a space between the dashes that is equal to the length of a dash.

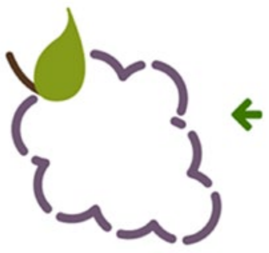


The first grapes image here shows the impact that an even number of listed values has on the grape's path: `stroke-dasharray="20,15,10,8"`.

stroke-dashoffset

`stroke-dashoffset` specifies the distance into the dash pattern to start the dash.

```
<svg>
    <!--<path stroke-dasharray="40,10"
stroke-dashoffset="35" <path for grapes> />-->
    <!--<path <path for stem> />-->
    <!--<path <path for leaf> />-->
</svg>
```



In the example above, there is a dash set to be 40px long, and a `dashoffset` of 35px. At the starting point of the path the dash will not become visible until 35px in to the first 40px dash, which is why the first dash appears significantly shorter.

stroke-opacity

The `stroke-opacity` attribute allows for a transparency level to be set on strokes.

The value here is a decimal number between 0 and 1, with 0 being completely transparent.

Buy the complete book