Compositing with `feComposite`

> The code for this example is taken from Sara Souedain's SVG Filters 101

Compositing is the combining of visual elements from separate sources into single images, often to create the illusion that all those elements are parts of the same scene (From Wikipedia)

The `feComposite` primitive has an operator attribute which is used to specify which composite operation we want to use. See the feComposite definition and Sara Souedain's Compositing and blending in CSS for more information about `feComposite`.

We use multiple filter primitives to accomplish the goal. We'll look at what each filter does and how to use the composite filer with an image.

The first filter primitive is `feGausianblur`. It creates the blur of the alpha channel of the image (in this case it will produce a blured black element).

The next step is to create the color flood using `feFlood`.

The `feComposite` primitive requires two inputs to operate on, `in` and `in2` which represent the two layers that we want to composite. In this case we use our Gausian blur and color as the inputs and call the result shadow.

`feOffset` moves the input by the required ammount in bot x and y axes and produces a result. In addition, this primitive takes two main attributes: dx and dy, which determine the distance by which you want to offset the layer along the x and y axes, respectively.

The last step on the filter is to merge the drop shadow with the source image using `feMerge`; one `mergeNode` will take our drop shadow as input, and another `mergeNode` will layer the source image using `SourceGraphic` as input.

We use the filter by referencing it in the image using the `filter` attribute. Notice that we use the `url()` syntax rather than use `xlink:href`

```
<svg width="600" height="400" viewBox="0 0 850 650">
  <filter id="drop-shadow">

    <feGaussianBlur in="SourceAlpha"
                    stdDeviation="10"
                    result="drop"></feGaussianBlur>

    <feFlood flood-color="#bbb" result="color"></feFlood>

    <feComposite  in="color"
                  in2="drop"
                  operator="in"
                  result="shadow"></feComposite>

    <feOffset   in="shadow"
                dx="20" dy="20"
                result="dropshadow"></feOffset>

    <feMerge>
        <feMergeNode in="dropshadow"></feMergeNode>
        <feMergeNode in="SourceGraphic"></feMergeNode>
    </feMerge>
  </filter>

  <image  xlink:href="images/image.jpg"
          x="0" y="0"
          width="100%" height="100%"
          filter="url(#drop-shadow)"></image>
</svg>
```

We used drop shadow as an example of how to create complex filters. It is not the only way or the easiest way to create drop shadows using CSS but I think it's an interesting example of how to compose filters with multiple primitives.

The field is open for you to experiment with filters in different combinations and with different parameters. The sky is the limit :)