CSIT5400
Computer Graphics

SVG Definitions

David Rossiter

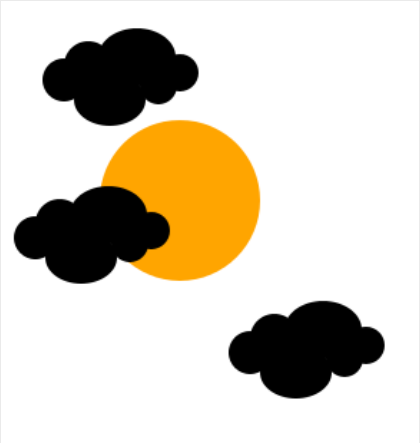
This Presentation

- There is a ‘definitions’ area of SVG in which something can be defined (once) and then used in the SVG (as many times as you want)
- These are some of the things you can define:
  - A definition of your own
  - Clipping paths
  - Patterns
  - Gradients
  - Filters

Use Example

- In this example an object is defined once and used several times

```
<svg>
   <defs>
     <g id="Cloud">
       <circle cx="24" cy="36" r="15"/>
       <circle cx="41" cy="26" r="17"/>
       <circle cx="90" cy="40" r="13"/>
       <circle cx="105" cy="31" r="13"/>
       <ellipse cx="75" cy="20" rx="27" ry="20"/>
       <ellipse cx="56" cy="50" rx="25" ry="18"/>
     </g>
   </defs>

   <circle id="Sun" cx="125" cy="140" r="56" style="fill:orange"/>
   <use id="SunCloud1" xlink:href="#Cloud" x="20" y="20" />
   <use id="SunCloud2" xlink:href="#Cloud" x="0" y="130" />
   <use id="SunCloud3" xlink:href="#Cloud" x="150" y="210" />
</svg>
```



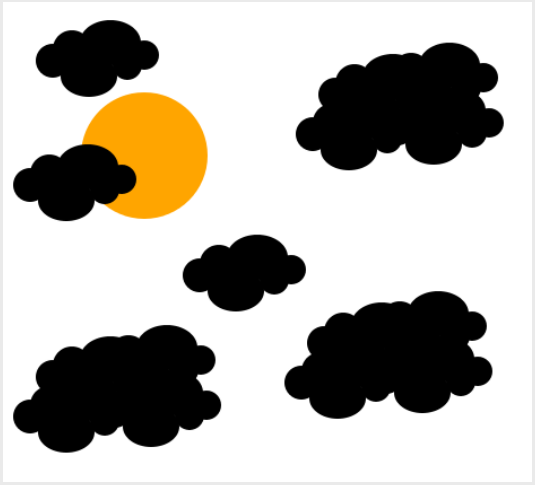
Example 1

Use Example 2

```
<svg>
   <defs>
     <g id="Cloud">
       <circle cx="24" cy="36" r="15"/>
       <circle cx="41" cy="26" r="17"/>
       <circle cx="90" cy="40" r="13"/>
       <circle cx="105" cy="31" r="13"/>
       <ellipse cx="75" cy="20" rx="27" ry="20"/>
       <ellipse cx="56" cy="50" rx="25" ry="18"/>
     </g>
     <g id="SuperCloud">
       <use xlink:href="#Cloud" x="20" y="20" />
       <use xlink:href="#Cloud" x="70" y="10" />
       <use xlink:href="#Cloud" x="0" y="55" />
       <use xlink:href="#Cloud" x="75" y="50" />
     </g>
   </defs>
```

- In this example an object is defined, then that definition is used for a new definition
- Then both definitions are used several times

Example 2, Cont.



Example 2

```
    <circle id="Sun" cx="125" cy="140" r="56" style="fill:orange"/>
    <use id="SmallCloud1" xlink:href="#Cloud" x="20" y="20" />
    <use id="SmallCloud2" xlink:href="#Cloud" x="0" y="130" />
    <use id="SmallCloud3" xlink:href="#Cloud" x="150" y="210" />

    <use id="BigCloud1" xlink:href="#SuperCloud" x="250" y="30" />
    <use id="BigCloud2" xlink:href="#SuperCloud" x="240" y="250" />
    <use id="BigCloud3" xlink:href="#SuperCloud" x="0" y="280" />

</svg>
```

Clip Path 1/2

- A clip path is basically a ‘window’ through which the rest of the SVG can be shown

```
<svg>

        <style type="text/css">
                text {
                        font-family: Arial;
                        font-size: 120px;
                        font-weight: bold;
                }

                rect {
                        fill: black;
                        fill-opacity: 1.0;
                }
        </style>
```

- This particular example uses a style sheet
- Using a style sheet is completely optional
- It makes the code on the next slide less crowded and much easier to read

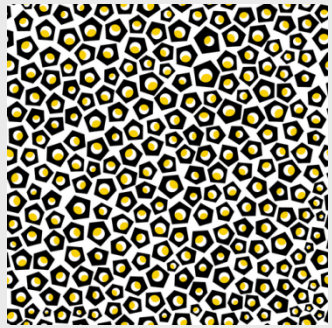
Clip Path 2/2

```
<defs>
        <clipPath id="some_text" >
                <text x="0" y="130">Clipping</text>
                <text x="10" y="220">Window</text>
        </clipPath>
</defs>

<image xlink:href="image.jpg"
   style="clip-path:url(#some_text)" width="800" height="400"/>

</svg>
```

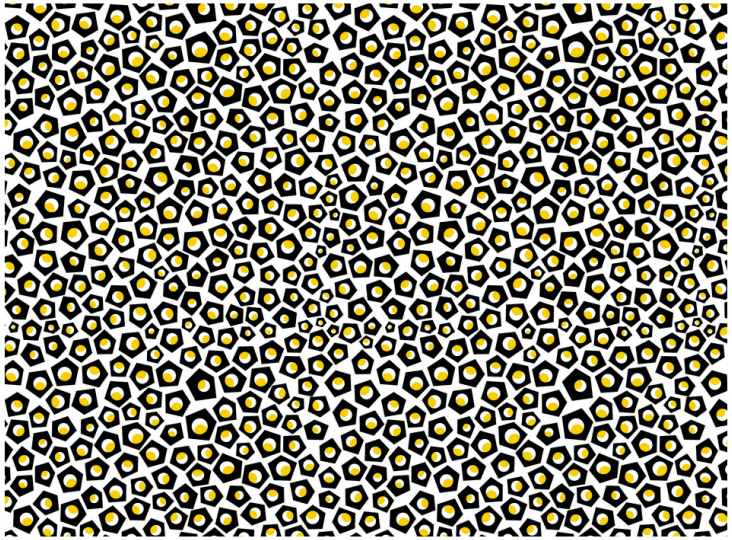
Example 3

Patterns

- This particular image is a cleverly designed tileable image



dots.png



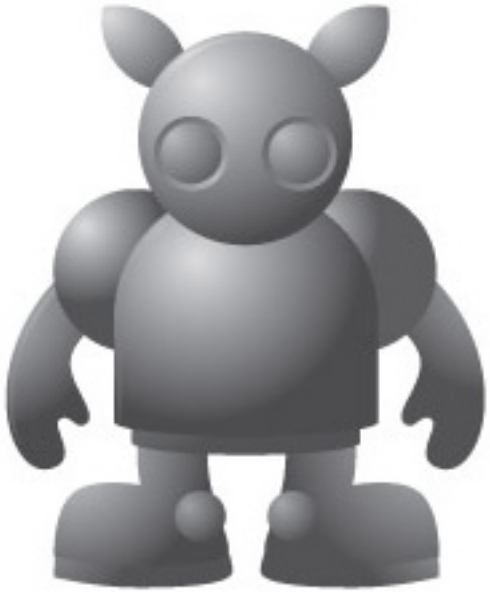
Example 4

```
<svg>

  <defs>
    <pattern id="dotspattern" x="0" y="0"
      patternUnits="userSpaceOnUse" width="495px" height="495px">
        <image xlink:href="dots.png" x="0" y="0"
         width="495px" height="495px"/>
    </pattern>
  </defs>

  <rect style="fill:url(#dotspattern)"
    width="950" height="700" x="50" y="50" />

</svg>
```

- A pattern is a way to fill an area by repeating an image many times

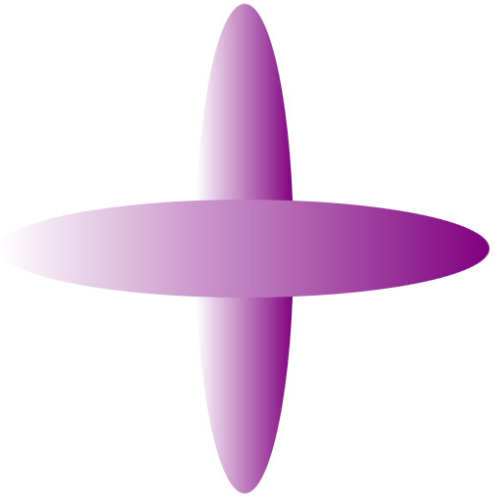
Gradients

- Gradients are simple to program but when used cleverly can be visually very powerful



Example creature made using gradients

```
<svg>

  <defs>
    <linearGradient id="disc_gradient">
        <stop offset="0" style= "stop-color:white"/>
        <stop offset="1" style= "stop-color:purple"/>
    </linearGradient>
  </defs>

  <ellipse style="fill:url(#disc_gradient)"
        cx="400" cy="400" rx="50" ry="250" />
  <ellipse style="fill:url(#disc_gradient)"
        cx="400" cy="400" rx="250" ry="50" />

</svg>
```



Example 5

Filters

- Filters can be used to create lots of different effects
- Filters can use just a few lines to define them, but the results can look fantastic
- So this is great for the Internet - very fast download speed and high impact as well
- However, there may be a time delay while the result is generated by the browser
- Some filter operations are quicker for the browser to generate than others

Defining Filters

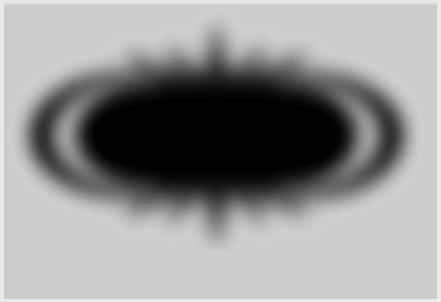
```
<svg>

  <defs>
    <filter id="cool_effect">
        Definition of filter goes here
    </filter>
  </defs>

  <text style="filter:url(#cool_effect)">
    In this example, the defined filter
    is applied to these words
  </text>

</svg>
```

Filters - Example

- The following SVG uses a series of filters which add the shadow and lighting effects

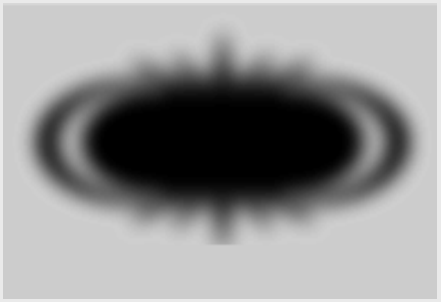
Example 6

Example - Composition Stages



1. Gaussian Blur

+



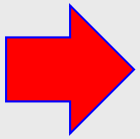
2. Offset (=move)

+

3. Specular Lighting
(specular=point source)

+

4. Limited Specular Lighting



Apply to some SVG

=

Result

The Filter Code

```
<filter id= "MyFilter" >
    <feGaussianBlur in="SourceAlpha" stdDeviation="4" result="blur"/>

    <feOffset in="blur" dx="4" dy="4" result="offsetBlurredAlpha"/>

    <feSpecularLighting in="blur" surfaceScale="5" specularConstant="0.9"
    specularExponent="20" lightColor="white" result="specularOut">
        <feDistantLight azimuth="135" elevation="30"/>
    </feSpecularLighting>

    <feComposite in="specularOut" in2="SourceAlpha" operator="in"
    result="specularOut"/>

    <feComposite in="SourceGraphic" in2="specularOut"
    operator="arithmetic" k1="0" k2="1" k3="1" k4="0" result="litPaint"/>

    <feMerge>
        <feMergeNode in="offsetBlurredAlpha"/>
        <feMergeNode in="litPaint"/>
    </feMerge>

</filter>
```

- At the moment, this code looks scary!
- We will discuss SVG filters in much more detail later