CSS Masking and Clipping

Two commonly used operations in computer graphics are clipping and masking. Both operations hide visual portions of an element. Clipping defines the region of an element that is visible. Everything around this region does not get rendered - it gets "clipped". On masking, a mask image is composited with the element, affecting the alpha channel of this element. Portions of a masked element get fully or partially transparent.

Clipping

This property is limited to rectangular clipping with the `rect()` function taking four distance arguments for the top, right, bottom and left edges. The annoying part: The `clip` property applies to absolutely positioned elements exclusively. The property is just ignored on other elements.

CSS:

```
img {
  position: absolute;
  clip: rect(10px, 290px, 190px, 10px);}
```

HTML:

```
<img src="image.jpg" width="568">
```

The clip-path property

The clip-path property can be applied to all HTML elements, SVG graphic elements and SVG container elements. It either references a <clipPath> element or one of the basic shapes introduced with CSS Exclusions.

The <clipPath> element takes any graphical element from SVG and uses them as clipping region. Graphical elements in SVG are <rect>, <circle>, <ellipse>, <path>, <polygon>, <image> and <text>. <clipPath> allows combining multiple graphical elements as well. The union of all shapes is then used as clipping region. The following example demonstrates the use of <clipPath>:

CSS:

```
img {
  clip-path: url(#clipping);
}
```

HTML:

```
<svg>
  <defs>
    <clipPath id="clipping">
      <circle cx="284" cy="213" r="213" />
    </clipPath>
  </defs>
</svg>
<img src="image.jpg" width="568">
```

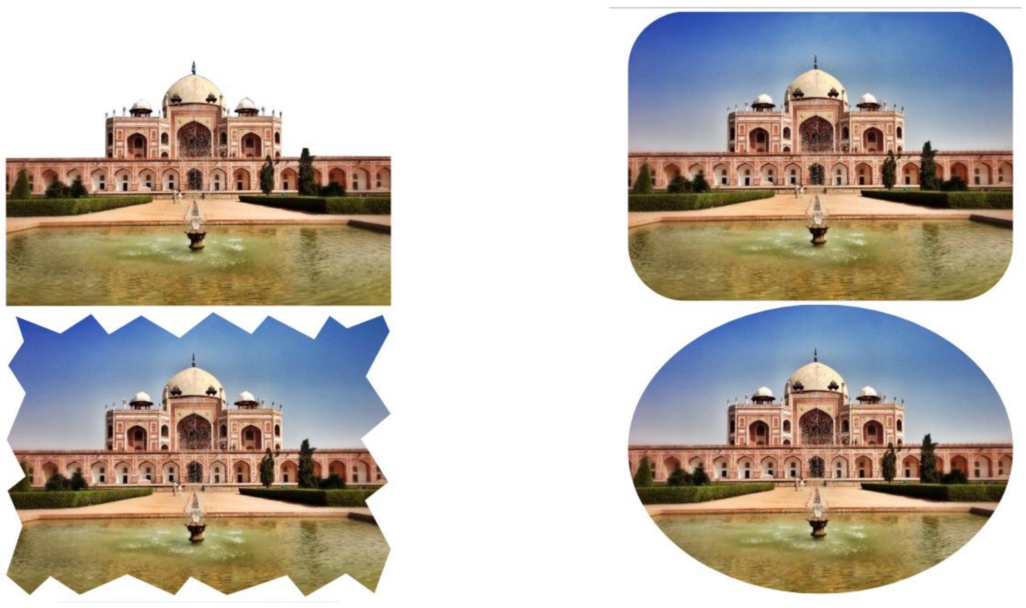
Basic shapes on the other hand do not require any SVG markup. They were added to clip-path to provide easy shorthand functions for simple clipping operations.

- inset(<top> <right> <bottom> <left> [round <border-radius>]?) defines a rectangle, similar to the rect() function of clip. The paramters describe the offsets from the top, right, bottom and left side. The function also has optional radius parameters with the syntax of the border-radius property for rounded rects.
- circle(<r>? [at <position>]?) defines a simple circle with an optional radius parameter. In addition, the optional position parameter described the center point of the circle. <position> has the same syntax as the background-position property.
- ellipse(<rx> <ry>? [at <position>]?) defines an ellipse with a horizontal and an optional vertical radius as well as a center point based on the background-position property's syntax.
- polygon(<x1> <y1>, <x2> <y2>, ..., <xn> <yn>) defines a polygon based on the point list parameters.

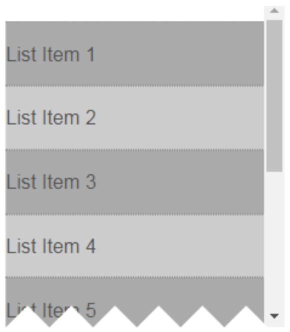
The CSS markup can look like the following example:

```
img {
  clip-path: polygon(0px 208px, 146.5px 207px, 147px 141.2px, ...);
}
```

Clipping can be very useful for the presentation of visual content. The following examples apply different clipping operations to images.



But clipping can be useful for UI design as well. In the following example a chevron indicates a continuing list. Scroll the list to see the effect.



List Item 1
List Item 2
List Item 3
List Item 4
List Item 5

Animation of clip-path

Both, basic shapes and the content of a <mask> element can be animated. The following example has a star shape animation:

Here is the source code for the animation of the basic shape:

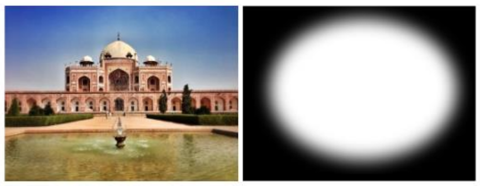
```
img:hover {
  clip-path: polygon(0px 208px, 146.5px 207px, 147px 141.2px, ...);
  animate: star 3s;
}

@keyframes star {
  0% {
    clip-path: polygon(0px 208px, 146.5px 207px, 147px 141.2px, ...);
  },
  100% {
    clip-path: polygon(0px 208px, 146.5px 207px, 147px 141.2px, ...);
  }
}
```

Masking

The second operation after clipping is masking. A mask image is used as some kind of "color mesh", to filter the visual portions of an element. In the following paragraphs I'll explain the difference between two kind of masks: the luminance mask and the alpha mask.

Luminance Mask

For luminance masks, the mask image is transformed to a rasterized gray scale image first (if it isn't in gray scale already). The "lighter" a portion of the mask image is, the more of the masked element will be visible on the same position. For instance black indicates fully transparent, white indicates fully opaque and a gray tone indicates partial transparency of the element.



Alpha Mask

Alpha masking uses the same principle as luminance masking. The difference to luminance masking: just the alpha channel of the image matters. The less opaque a portion of the mask image is, the less visible the element will be at the same position.

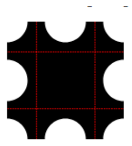
To summarize: both masking types affect the transparency level of the element. The image below is the result of both masking operations above.

The mask-border property

The mask-border property allows to divide a mask image into 9 tiles: four corners, four edges and the middle piece. These pieces may be sliced, scaled and stretched in various ways to fit the size of the mask image area. The property borrows the functionality from border-image and allows powerful masking at the edges and corners of the content. The following example demonstrates the behavior of the property:

```
img {
  -webkit-mask-box-image: url("stamp.svg") 35 repeat;
  mask-border: url("stamp.svg") 35 repeat;
}
```

The following image is used as mask image and divided into 9 parts:



These parts are now used to mask the corner and edges of the content and result in the following view: