EBU6305

Interactive Media Design and Production

CSS Animations

Dr. Xianhui Cherry Che
x.che@qmul.ac.uk

Learning Objectives

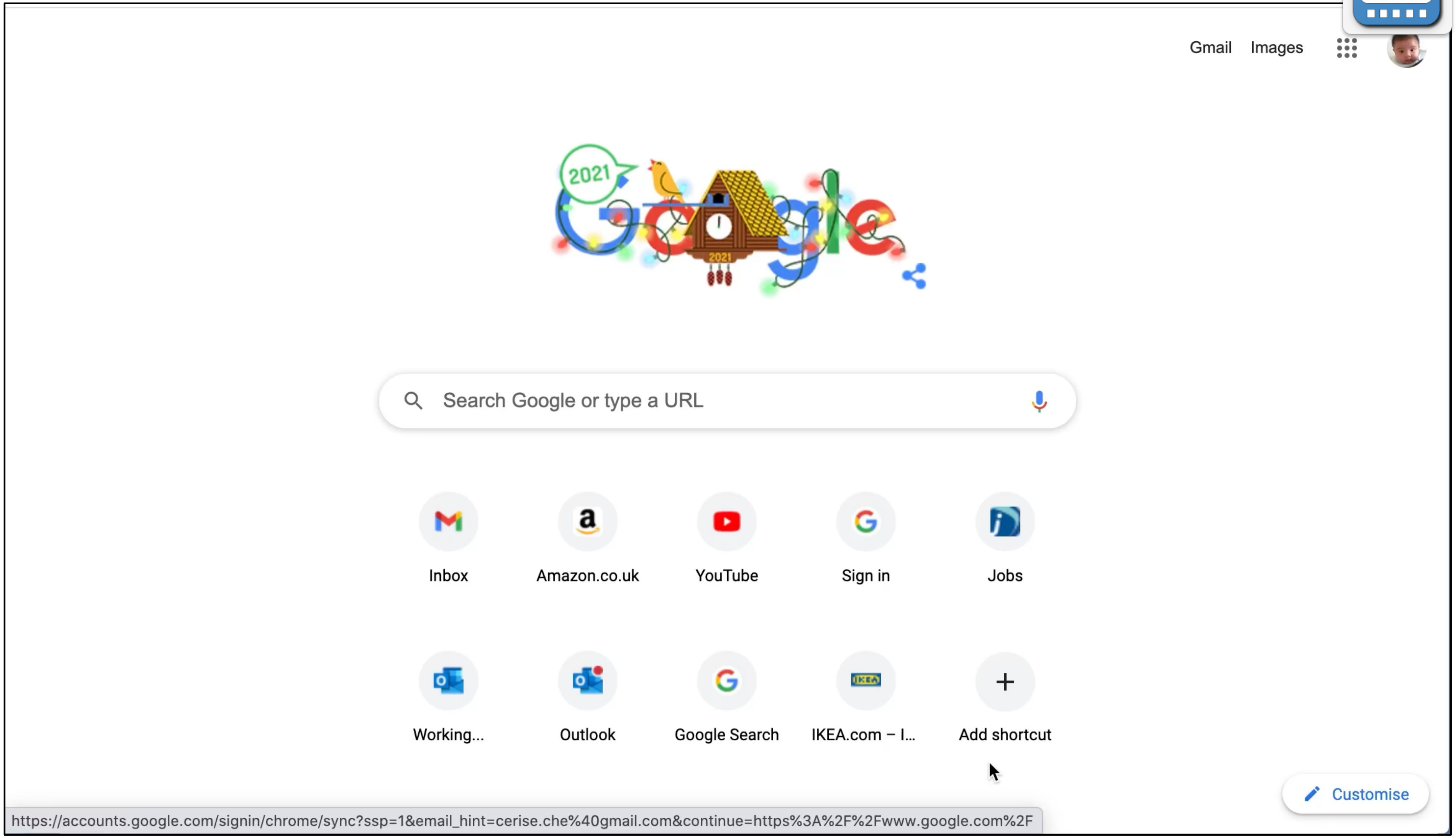
- Ability to generate an innovative design for products, systems, components or processes to fulfil new needs.
- Use creativity to establish innovative solution.

Topics

- CSS Animation
- SVG
- Perspectives for Illusion

Falling Animation

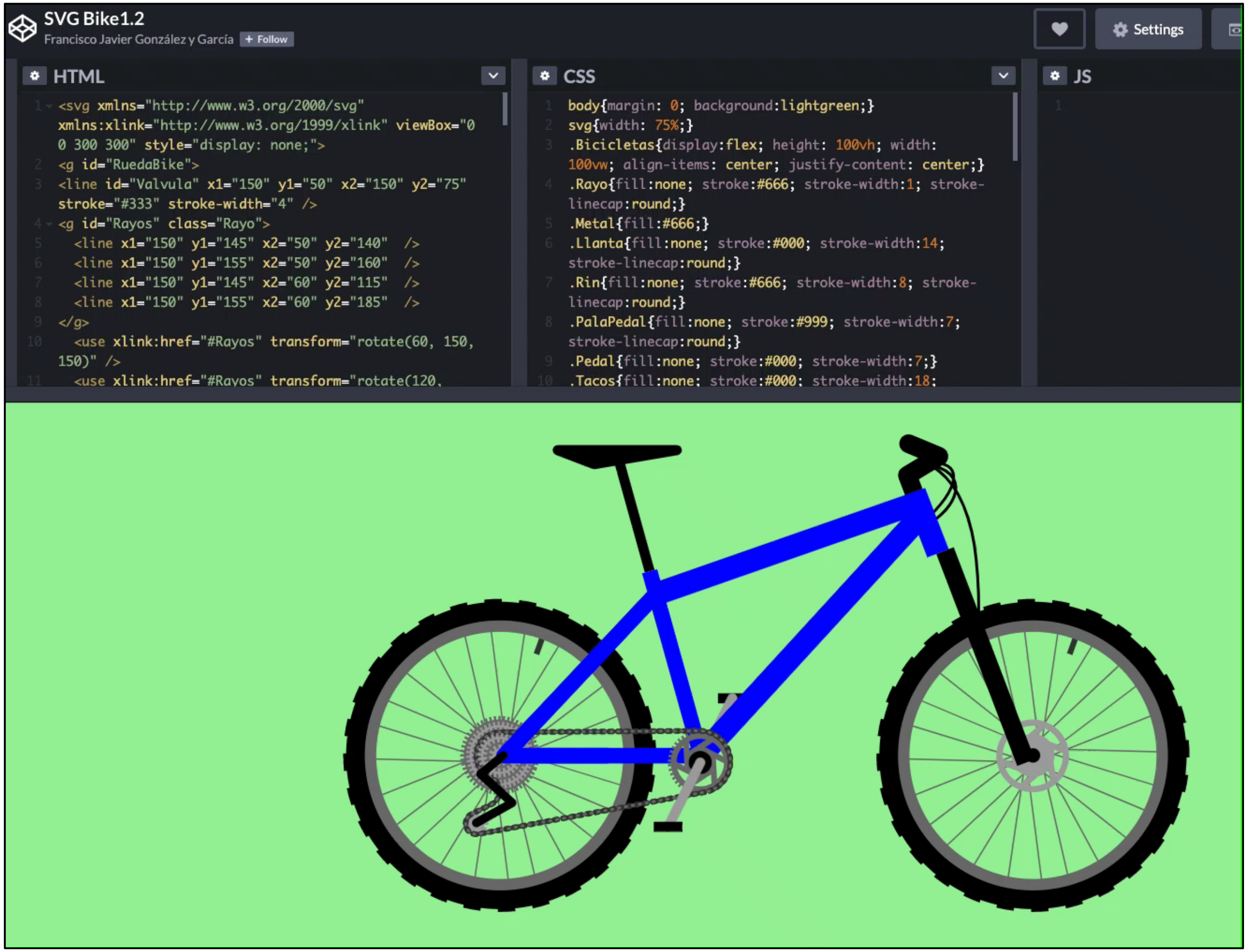
- Let’s watch a demo first:



Gmail
Images
2021
Search Google or type a URL
Inbox
Amazon.co.uk
YouTube
Sign in
Jobs
Working...
Outlook
Google Search
IKEA.com – I...
Add shortcut
Customise
https://accounts.google.com/signin/chrome/sync?ssp=1&email_hint=cerise.che%40gmail.com&continue=https%3A%2F%2Fwww.google.com%2F

Bike

- This animation is done entirely using HTML and CSS.

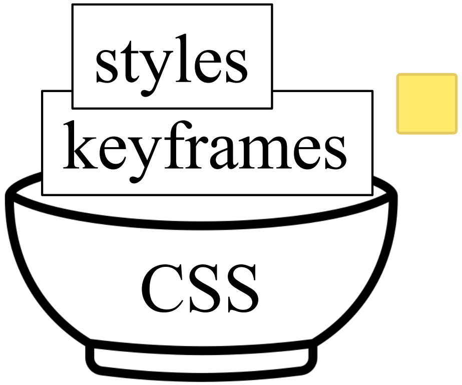


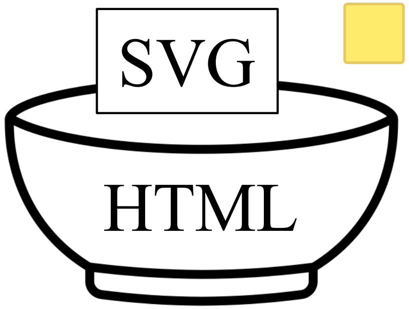
```html
<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0
0 300 300" style="display: none;">
<g id="RuedaBike">
<line id="Valvula" x1="150" y1="50" x2="150" y2="75"
stroke="#333" stroke-width="4" />
<g id="Rayos" class="Rayo">
  <line x1="150" y1="145" x2="50" y2="140"  />
  <line x1="150" y1="155" x2="50" y2="160"  />
  <line x1="150" y1="145" x2="60" y2="115"  />
  <line x1="150" y1="155" x2="60" y2="185"  />
</g>
  <use xlink:href="#Rayos" transform="rotate(60, 150,
150)" />
  <use xlink:href="#Rayos" transform="rotate(120,
```

```css
body{margin: 0; background:lightgreen;}
svg{width: 75%;}
.Bicicletas{display:flex; height: 100vh; width:
100vw; align-items: center; justify-content: center;}
.Rayo{fill:none; stroke:#666; stroke-width:1; stroke-
linecap:round;}
.Metal{fill:#666;}
.Llanta{fill:none; stroke:#000; stroke-width:14;
stroke-linecap:round;}
.Rin{fill:none; stroke:#666; stroke-width:8; stroke-
linecap:round;}
.PalaPedal{fill:none; stroke:#999; stroke-width:7;
stroke-linecap:round;}
.Pedal{fill:none; stroke:#000; stroke-width:7;}
.Tacos{fill:none; stroke:#000; stroke-width:18;
```

Key Ingredients

- One bowl of HTML code to draw the arty stuffs.
- One bowl of CSS code to define the behaviour of the artwork.

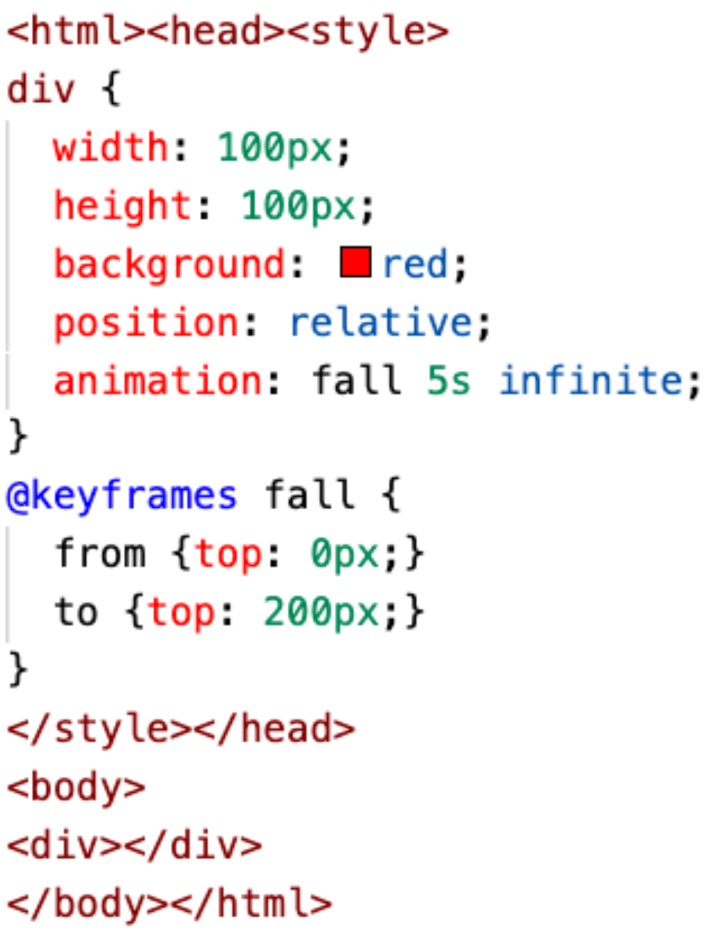




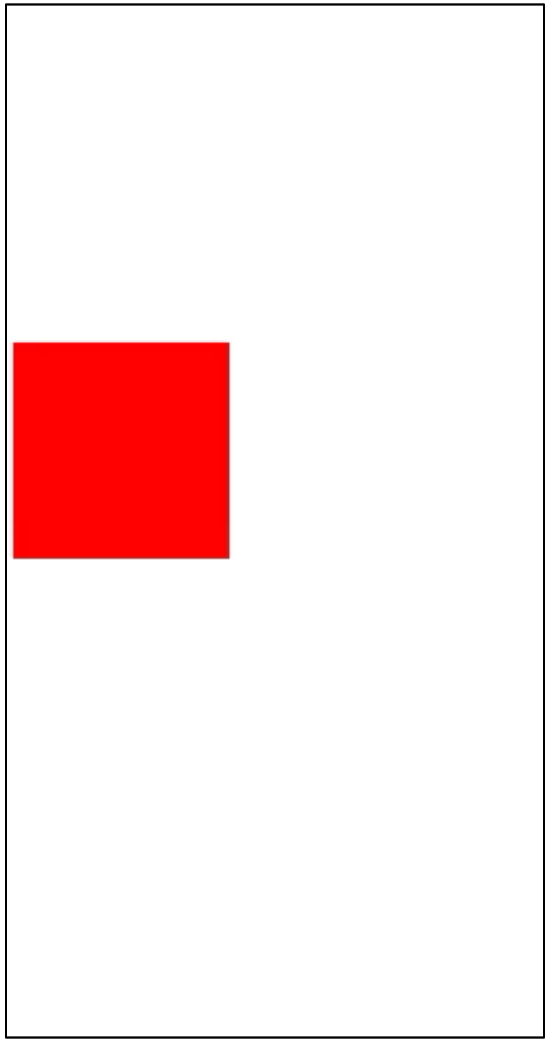
@keyframes

- The @keyframes rule specifies the animation code.
- The animation is created by gradually changing from one set of CSS styles to another.
- During the animation, the CSS style can be changed.
- Two ways to specify the change:
  - – Use the keywords "from" and "to” to define the starting and ending styles.
  - – Use 0% and 100% to define the starting and ending styles. Other timestamps in between are also accepted, such as 25% of the full duration.

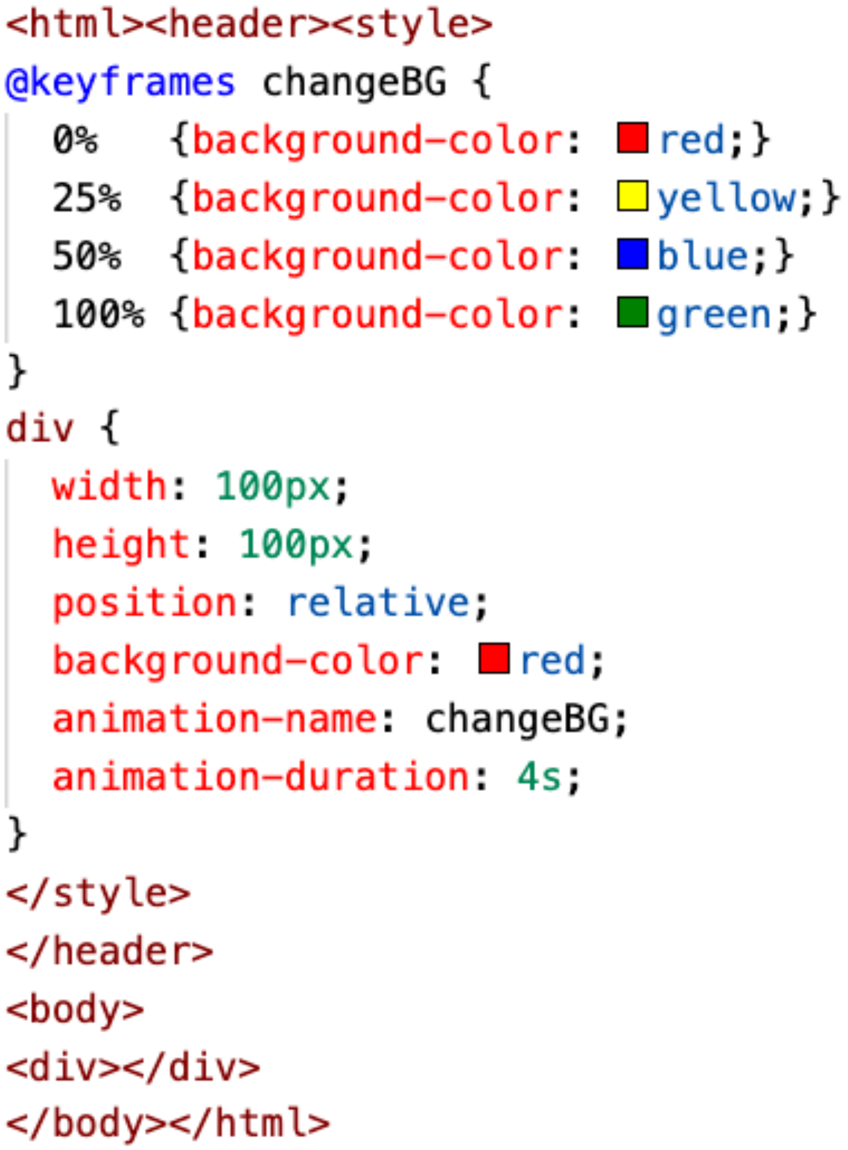
@keyframes – From and To



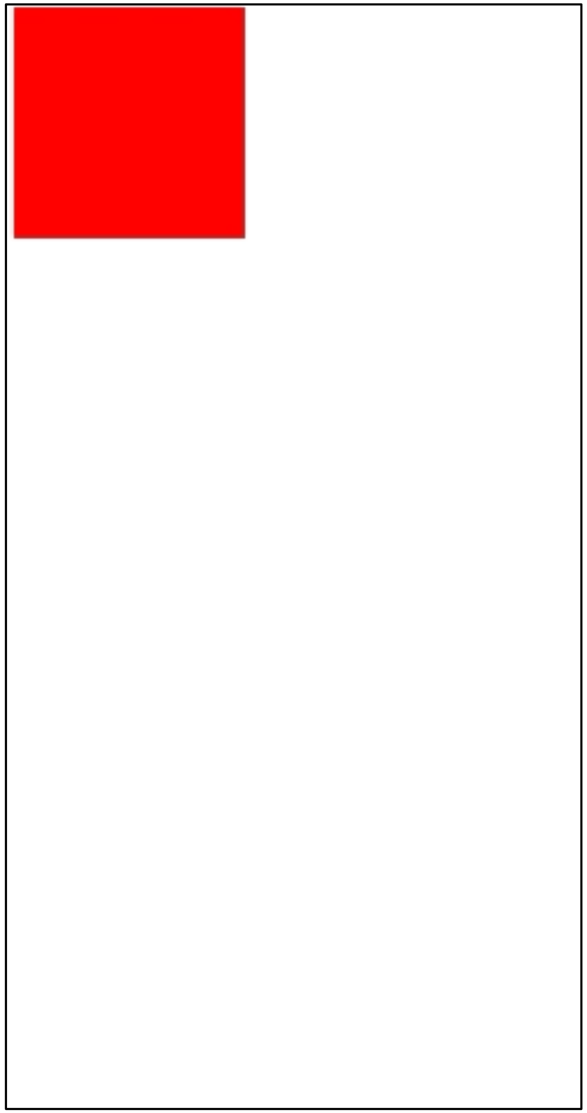
```
<html><head><style>
div {
  width: 100px;
  height: 100px;
  background: 🟥red;
  position: relative;
  animation: fall 5s infinite;
}
@keyframes fall {
  from {top: 0px;}
  to {top: 200px;}
}
</style></head>
<body>
<div></div>
</body></html>
```



@keyframes – Percentages



```
<html><header><style>
@keyframes changeBG {
   0%   {background-color: 🟥red;}
   25%  {background-color: 🟨yellow;}
   50%  {background-color: 🟦blue;}
   100% {background-color: 🟩green;}
}
div {
   width: 100px;
   height: 100px;
   position: relative;
   background-color: 🟥red;
   animation-name: changeBG;
   animation-duration: 4s;
}
</style>
</header>
<body>
<div></div>
</body></html>
```

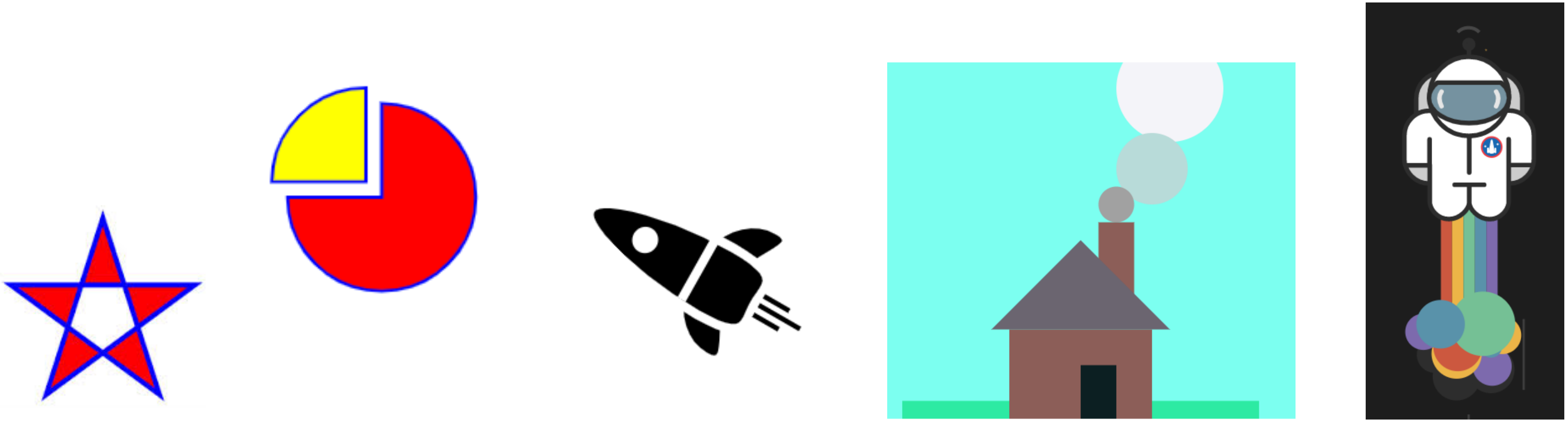


Topics

- CSS Animation
- SVG
- Perspectives for Illusion

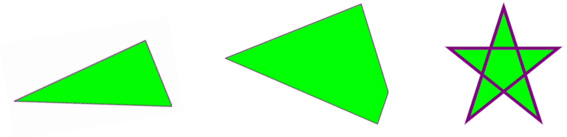
SVG

- SVG – Scalable Vector Graphics
- Can be used to draw graphics such as circle, square, line, etc.
- Every element and every attribute in SVG files can be animated
- SVG is a W3C recommendation
- These examples are all drawn using SVG:

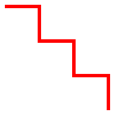


Basic Elements

- The <g> element is similar to the <div> concept in HTML
- Circle: <circle>
- Rectangle: <rect>
- Ellipse: <ellipse>
- Line: <line> (straight line only)

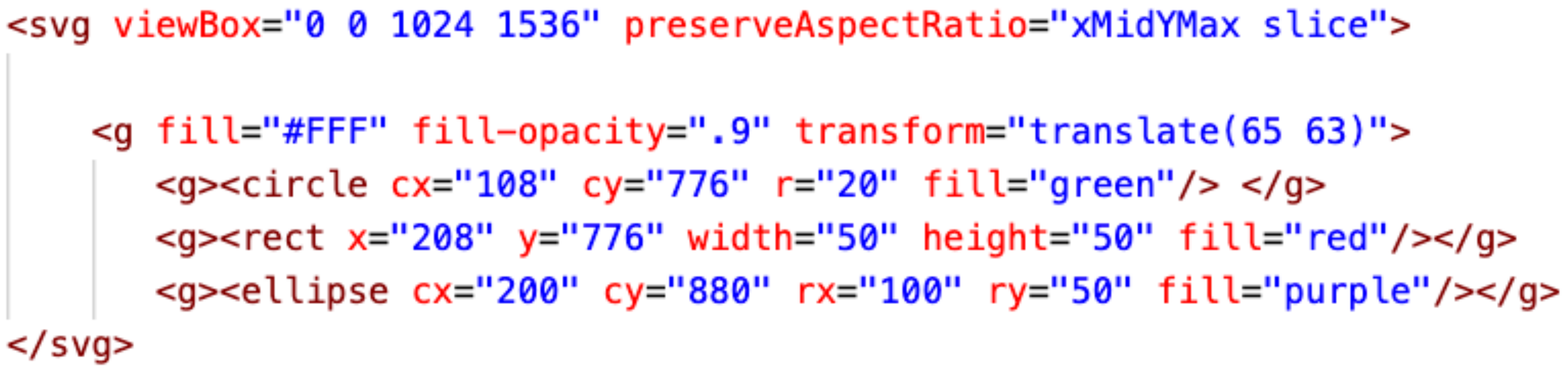


- Polygon: <polygon> (a graphic that contains at least three sides with straight lines)
- Polyline: <polyline> (a shape connected by multiple straight lines)

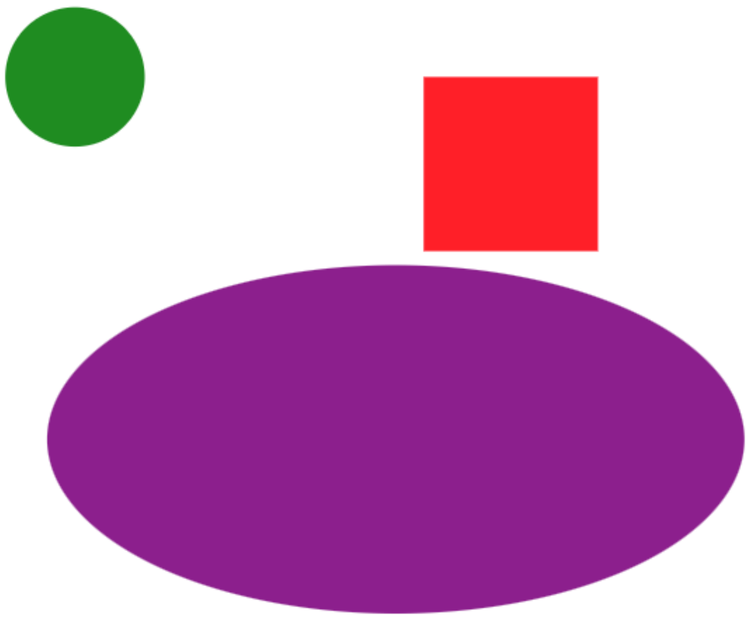


- Path: <path> (can be used to draw any line or shape)
- Text: <text>

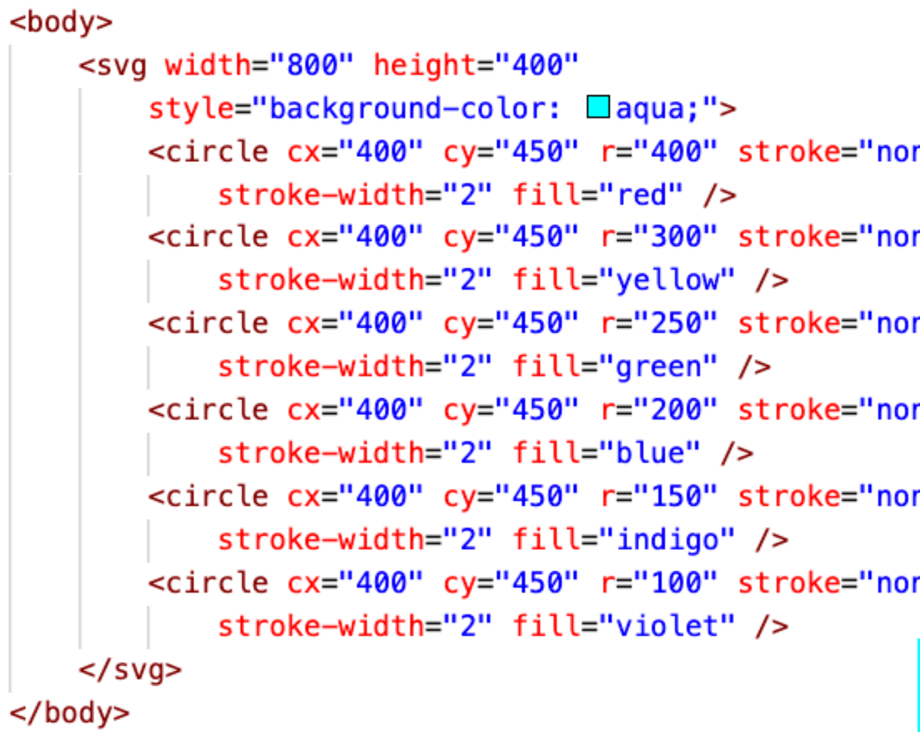
Example 1



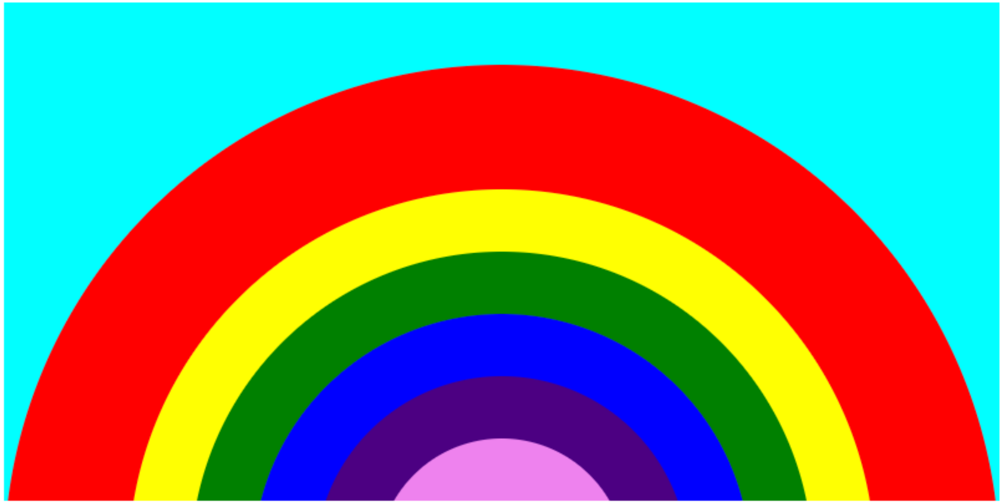
```
<svg viewBox="0 0 1024 1536" preserveAspectRatio="xMidYMax slice">

    <g fill="#FFF" fill-opacity=".9" transform="translate(65 63)">
       <g><circle cx="108" cy="776" r="20" fill="green"/> </g>
       <g><rect x="208" y="776" width="50" height="50" fill="red"/></g>
       <g><ellipse cx="200" cy="880" rx="100" ry="50" fill="purple"/></g>
</svg>
```



Example 2

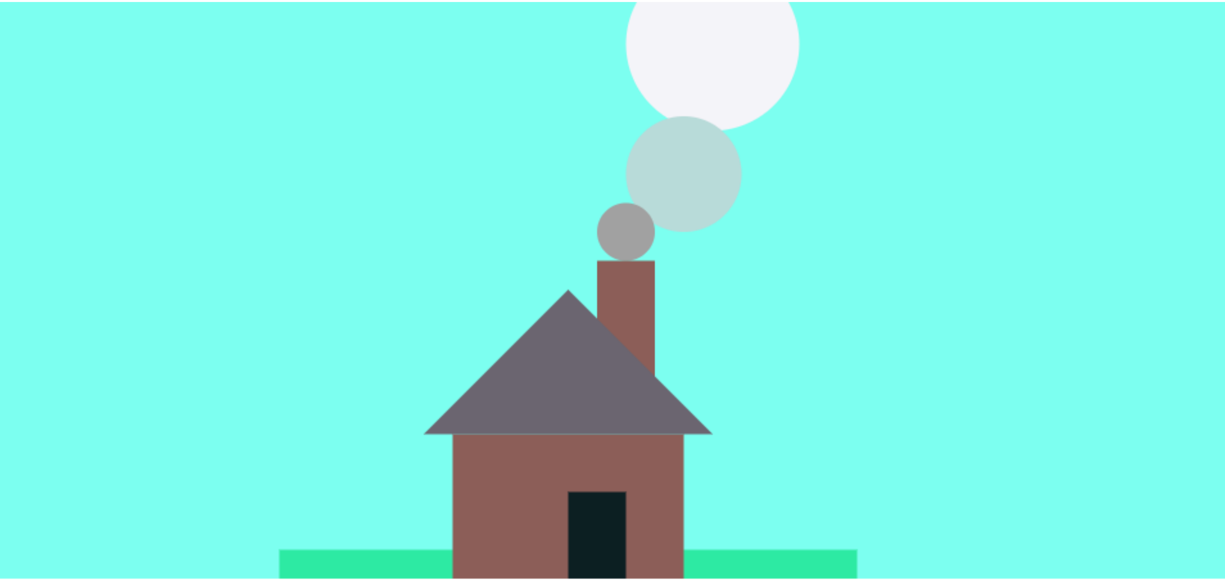


```
<body>
    <svg width="800" height="400"
        style="background-color: 🟦aqua;">
        <circle cx="400" cy="450" r="400" stroke="none"
            stroke-width="2" fill="red" />
        <circle cx="400" cy="450" r="300" stroke="none"
            stroke-width="2" fill="yellow" />
        <circle cx="400" cy="450" r="250" stroke="none"
            stroke-width="2" fill="green" />
        <circle cx="400" cy="450" r="200" stroke="none"
            stroke-width="2" fill="blue" />
        <circle cx="400" cy="450" r="150" stroke="none"
            stroke-width="2" fill="indigo" />
        <circle cx="400" cy="450" r="100" stroke="none"
            stroke-width="2" fill="violet" />
    </svg>
</body>
```



Example 3

```
<html> <head><style>
svg {
  width: 100%;
}
.house {
  background: 🟦#7CFEF0;
}
rect,
circle,
.circlegroup,
polygon {
  transition: all 0.5s ease;
}
polygon:hover,
rect:hover,
circle:hover,
.circlegroup:hover {
  fill: 🟪#624CAB;
}
</style></head>
<body>
    <svg width="450" height="450" viewBox="0 0 200 200" class="house">
        <g>
          <circle fill="#F4F4F9" cx="150" cy="15" r="30" />
          <circle fill="#B8DBD9" cx="140" cy="60" r="20" />
          <circle fill="#a1a1a1" cx="120" cy="80" r="10" />
          <rect fill="#8C5E58" width="20" height="50" x="110" y="90" />
          <polygon fill="#6B6570" points="100,100 150,150 50,150" />
          <rect fill="#2CEAA3" width="200" height="10" x="0" y="190" />
          <rect fill="#8C5E58" width="80" height="50" x="60" y="150" />
          <rect fill="#0D1F22" width="20" height="30" x="100" y="170" />
        </g>
      </svg>
</body> </html>
```

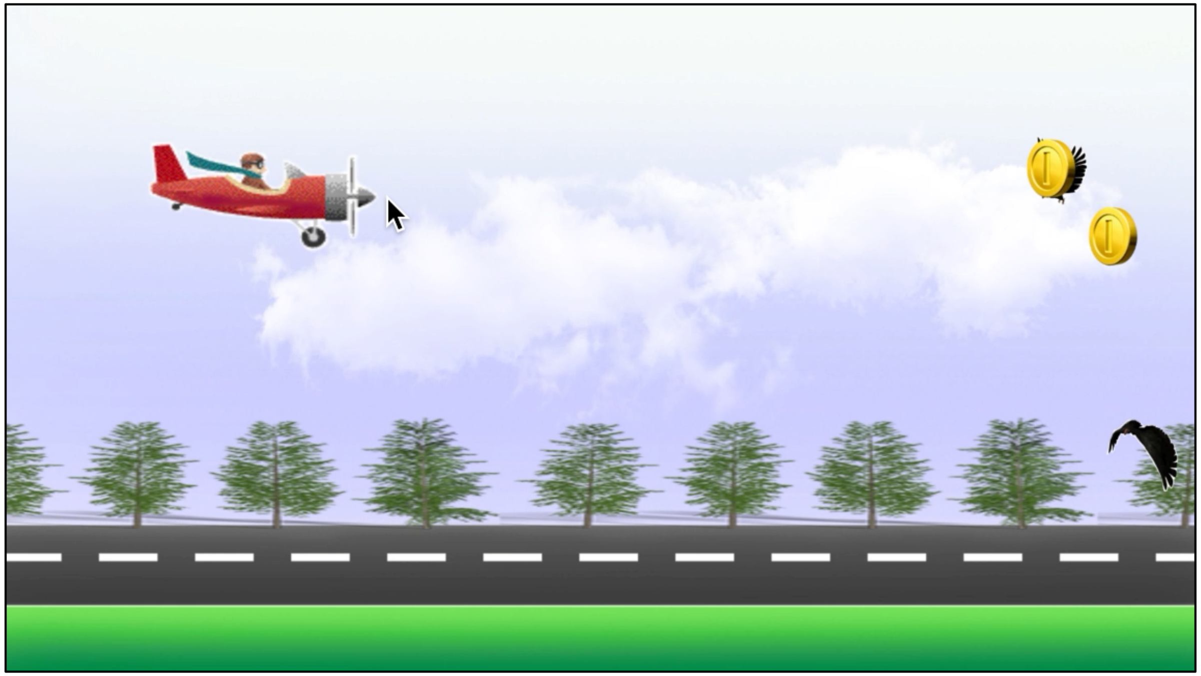


Topics

- CSS Animation
- SVG
- Perspectives for Illusion

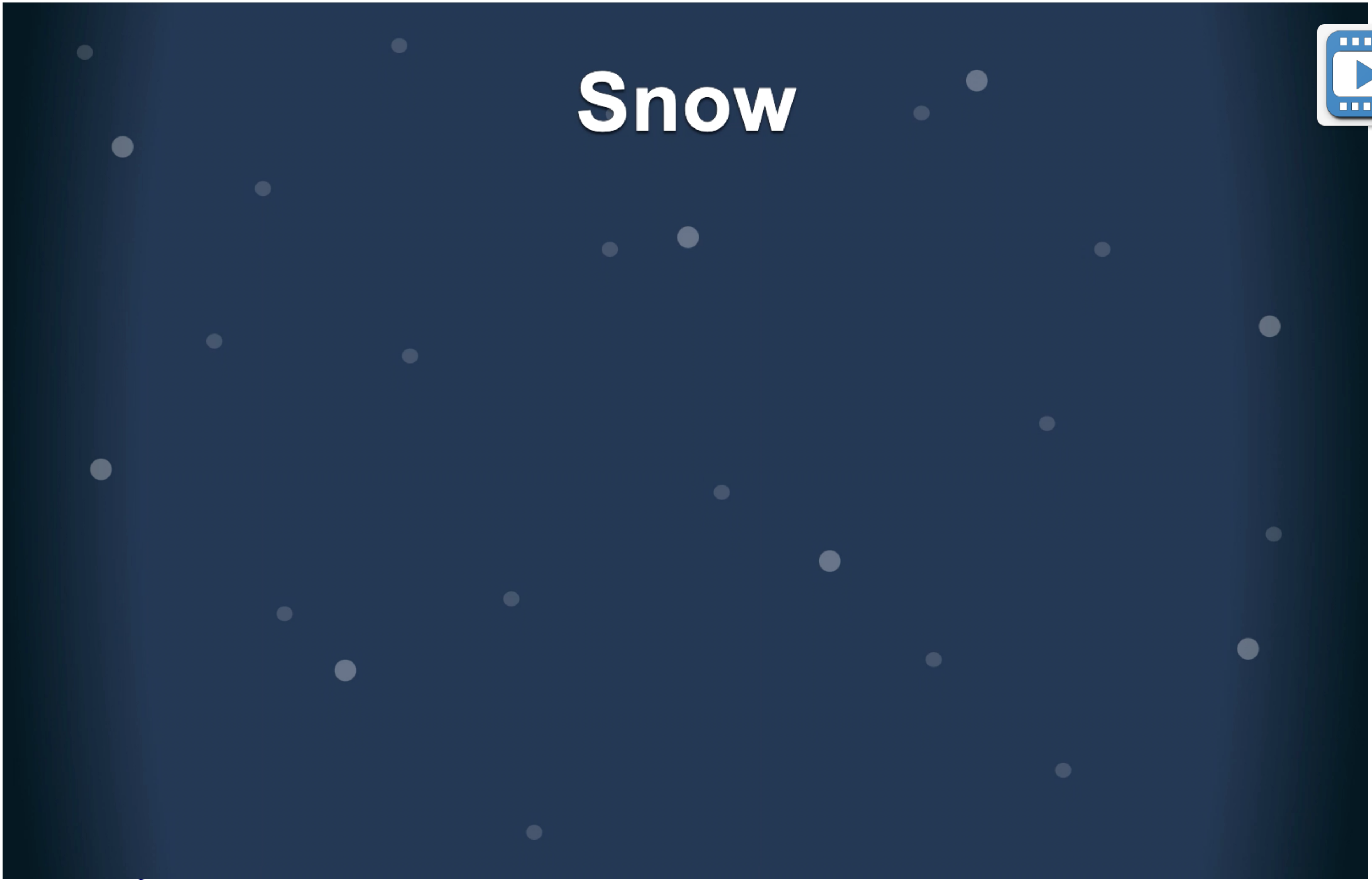
Perspectives for Illusion

- Quite often you can use multiple animations to create different perspectives so as to give users an illusion.

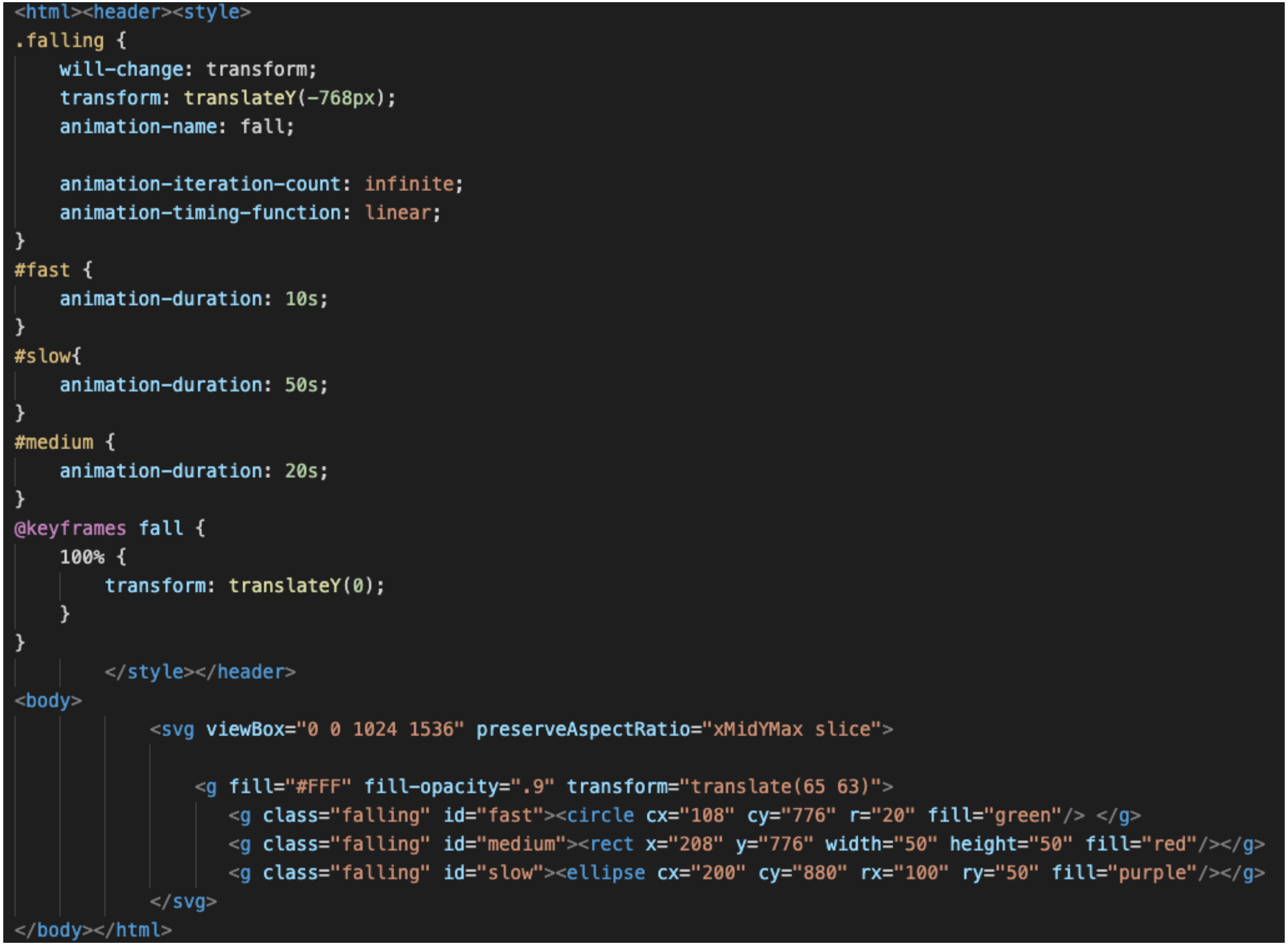


Perspectives for Illusion

One more example: (Try to observe how many different types of animations have been defined.)

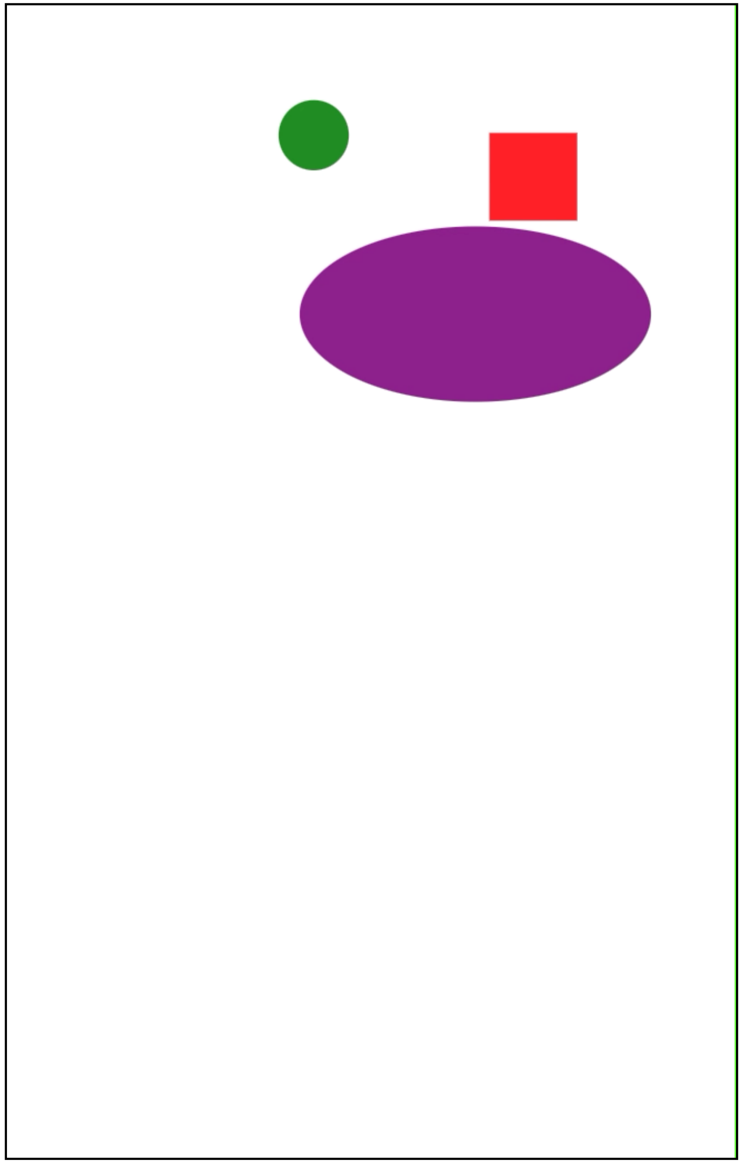


Example



```
<html><header><style>
.falling {
    will-change: transform;
    transform: translateY(-768px);
    animation-name: fall;

    animation-iteration-count: infinite;
    animation-timing-function: linear;
}
#fast {
    animation-duration: 10s;
}
#slow{
    animation-duration: 50s;
}
#medium {
    animation-duration: 20s;
}
@keyframes fall {
    100% {
        transform: translateY(0);
    }
}
        </style></header>
<body>
            <svg viewBox="0 0 1024 1536" preserveAspectRatio="xMidYMax slice">

                <g fill="#FFF" fill-opacity=".9" transform="translate(65 63)">
                   <g class="falling" id="fast"><circle cx="108" cy="776" r="20" fill="green"/> </g>
                   <g class="falling" id="medium"><rect x="208" y="776" width="50" height="50" fill="red"/></g>
                   <g class="falling" id="slow"><ellipse cx="200" cy="880" rx="100" ry="50" fill="purple"/></g>
            </svg>
</body></html>
```

Example



Questions?

x.che@qmul.ac.uk