CSS `object-fit` and `background`

In CSS, `object-fit` is a property used to define how content (like images or videos) should be resized to fit its container while maintaining its aspect ratio. This property is commonly used when dealing with media elements like `<img>` or `<video>` to control how they scale within a specified space.

`object-fit` values:

1. **`object-fit: fill;`**
    - The content is resized to fill the container, ignoring the aspect ratio.
    - This may lead to stretching or distortion.

```
img {
  width: 300px;
  height: 300px;
  object-fit: fill;
}
```

2. **`object-fit: contain;`**
    - The content is resized to fit within the container while preserving its aspect ratio.
    - This ensures that the entire content is visible, but there may be empty space (usually letterboxing or pillarboxing).

```
img {
  width: 300px;
  height: 300px;
  object-fit: contain;
}
```

3. **`object-fit: cover;`**
   - The content is resized to completely cover the container, while maintaining the aspect ratio.
   - Parts of the content might be cropped to ensure that the container is fully covered, without leaving any empty space.

```
img {
  width: 300px;
  height: 300px;
  object-fit: cover;
}
```

4. **`object-fit: none;`**
   - The content retains its original size, regardless of the container's dimensions.
   - It may overflow or not fill the container if the original size is different.

```
img {
  width: 300px;
  height: 300px;
  object-fit: none;
}
```

5. **`object-fit: scale-down;`**
   - This is a combination of `none` or `contain`, depending on which results in a smaller image.
   - It scales the content down to fit the container, without exceeding its original size.

```
img {
  width: 300px;
  height: 300px;
  object-fit: scale-down;
}
```

Difference Between `object-fit: cover` and `object-fit: contain`:

- **`object-fit: cover;`**
  - The image is scaled to fill the container, but some parts of it may be cropped if the aspect ratio of the container differs from the image.
  - **Best for:** When you want the image to completely cover the container, like a background image that should cover the entire element.
- **`object-fit: contain;`**
  - The image is scaled down to fit inside the container, keeping its aspect ratio intact.
  - It will leave empty space around the image if the aspect ratio differs.
  - **Best for:** When you want to display the entire image without cropping, even if it leaves some space around the image.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Object-Fit Example</title>
    <style>
        .container {
            width: 300px;
            height: 300px;
            border: 2px solid black;
        }

        .cover {
            object-fit: cover;
        }

        .contain {
            object-fit: contain;
        }
    </style>
</head>
<body>
    <div class="container">
        <img src="image.jpg" class="cover" alt="Cover Example" width="300" height="300">
    </div>
    <div class="container">
        <img src="image.jpg" class="contain" alt="Contain Example" width="300" height="300">
    </div>
</body>
</html>
```

In this example, the first image will cover the container with possible cropping (`object-fit: cover`), while the second image will fit inside the container without cropping but with empty space (`object-fit: contain`).

Summary:

- **`object-fit: cover`**: Fills the container entirely but might crop parts of the image.
- **`object-fit: contain`**: Scales the image to fit within the container, leaving empty space if necessary.

background-image

The `background-image` property in CSS is used to set a background image for an element. It allows you to display an image behind the content of a block element like a `<div>`, `<body>`, or even behind text. The `background-image` property is part of the broader `background` property set, which also controls other aspects like background color, position, and size.

Basic Syntax:

```
element {
    background-image: url("image-path.jpg");
}
```

This will apply the specified image (`image-path.jpg`) as the background of the selected element.

Commonly Used Properties with `background-image`:

1. **`background-image`**:
   - Specifies the image to be used as the background.
   - Example:

     ```
     body {
       background-image: url("background.jpg");
     }
     ```

2. **`background-color`**:
   - Sets a fallback background color in case the image doesn't load or for transparent parts of the image.
   - Example:

```
body {
  background-image: url("background.jpg");
  background-color: lightgray;
}
```

3. **`background-repeat`**:
   - Controls whether the background image is repeated (tiled) and how.
   - Options:
     - `repeat` (default): Repeats the image both horizontally and vertically.
     - `repeat-x`: Repeats the image only horizontally.
     - `repeat-y`: Repeats the image only vertically.
     - `no-repeat`: No repetition, the image is shown once.
     - Example:

       ```
       body {
         background-image: url("background.jpg");
         background-repeat: no-repeat;
       }
       ```

4. **`background-position`**:
   - Defines the starting position of the background image.
   - Can use keywords like `top`, `bottom`, `center`, or use pixel or percentage values.
   - Example:

```
body {
  background-image: url("background.jpg");
```

```
  background-position: center center;
}
```

5. **`background-size`**:
   - Controls the size of the background image.
   - Options:
     - `auto`: Keeps the original image size (default).
     - `cover`: Scales the image to cover the entire element, cropping parts if necessary.
     - `contain`: Scales the image to fit within the element, leaving gaps if needed.
     - Can also use specific dimensions, like `100px` or `50%`.
     - Example:

```
body {
  background-image: url("background.jpg");
  background-size: cover;
}
```

6. **`background-attachment`**:
   - Determines if the background scrolls with the page or stays fixed.
   - Options:
     - `scroll`: The background scrolls with the content.
     - `fixed`: The background stays fixed in place while the content scrolls.
     - Example:

```
body {
  background-image: url("background.jpg");
  background-attachment: fixed;
}
```

7. **`background-clip`**:
   - Specifies how far the background extends within the element.
   - Options:
     - `border-box`: The background is clipped to the border.
     - `padding-box`: The background is clipped to the padding edge.
     - `content-box`: The background is clipped to the content box.
     - Example:

       ```
       div {
         background-image: url("background.jpg");
         background-clip: padding-box;
       }
       ```

8. **`background-origin`**:
   - Determines the positioning area of the background image.
   - Options: `border-box`, `padding-box`, `content-box`.
   - Example:

     ```
     div {
       background-image: url("background.jpg");
       background-origin: content-box;
     }
     ```

9. **`background-blend-mode`**:
    - Blends the background image with the background color or multiple background layers.
    - Options: `multiply`, `screen`, `overlay`, `darken`, `lighten`, etc.
    - Example:

    ```
    body {
      background-image: url("background.jpg");
      background-color: red;
      background-blend-mode: multiply;
    }
    ```

Multiple Background Images:

You can set multiple background images for an element by separating the image URLs with commas. Each image can be controlled individually for properties like `background-repeat`, `background-position`, etc.

```
div {
    background-image: url("image1.jpg"), url("image2.png");
    background-repeat: no-repeat, repeat;
    background-position: left top, right bottom;
}
```

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Background Image Example</title>
    <style>
        body {
            background-image: url('background.jpg');
            background-size: cover;
            background-position: center;
            background-repeat: no-repeat;
            background-attachment: fixed;
        }
    </style>
</head>
<body>
    <h1>Welcome to my website</h1>
</body>
</html>
```

In this example:

- The background image covers the entire screen (`background-size: cover`).
- The image is centered (`background-position: center`).
- The image does not repeat (`background-repeat: no-repeat`).
- The image stays fixed as the user scrolls (`background-attachment: fixed`).

Why Use `background-image` ?

- To add aesthetic design elements to a webpage.
- To overlay text or content over a visually engaging background.
- To achieve decorative effects in combination with `background-blend-mode` .
- To apply branding or thematic images to enhance user experience.