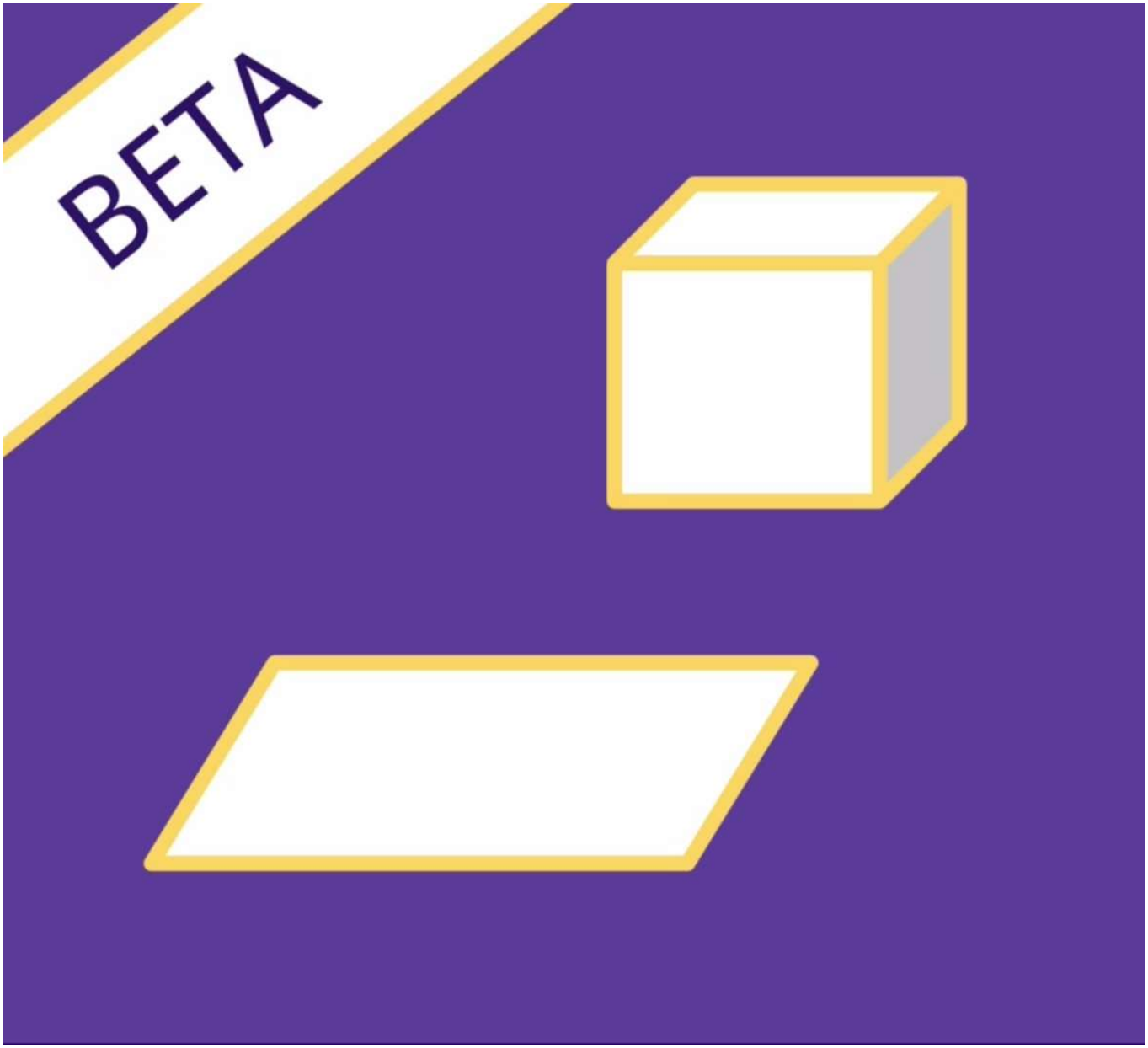
01

Transformações geométricas usando CSS3

Transcrição

Vamos continuar estudando as propriedades do CSS3. Nessa aula veremos as propriedades ***transform*** e ***perspective***, o que nos permitirá, por exemplo, inserir textos rotacionados, caixas "tortas" e objetos 3D:



A propriedade *transform*

O *transform* é uma propriedade poderosa, mas também muito simples de usar. Basta especificar o tipo de transformação que queremos fazer com o objeto:

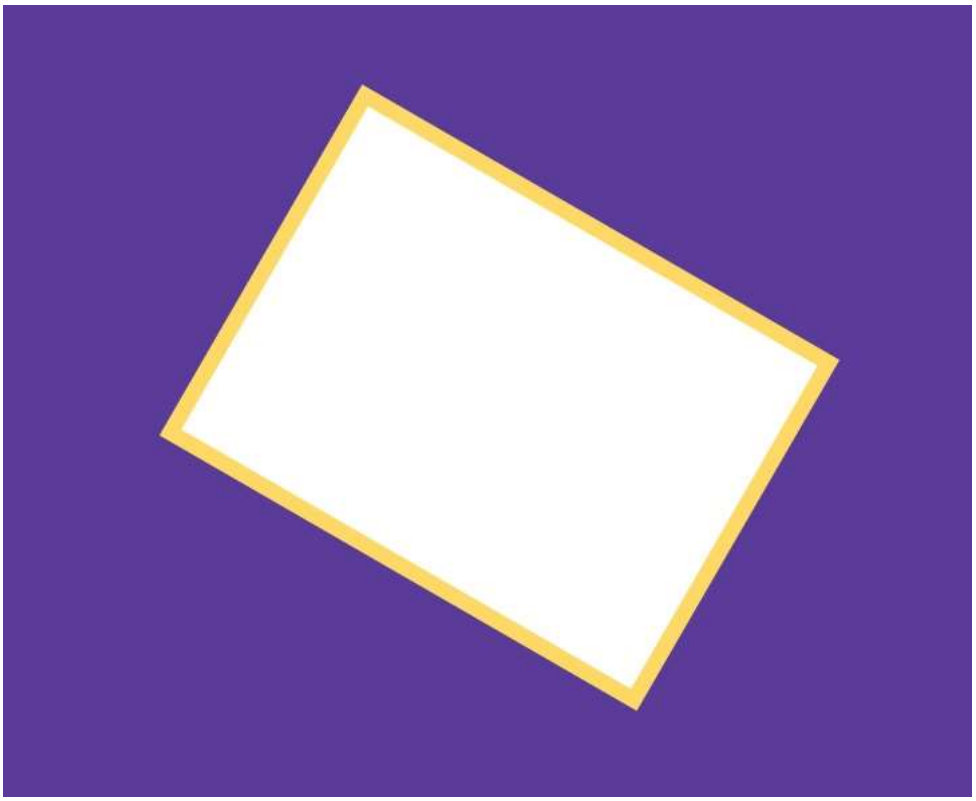
```
div {
    transform: ...;
}
```

Vamos ver alguns exemplos práticos:

- Rotacionar o objeto

Aqui o objeto está sendo rotacionado em 30 graus:

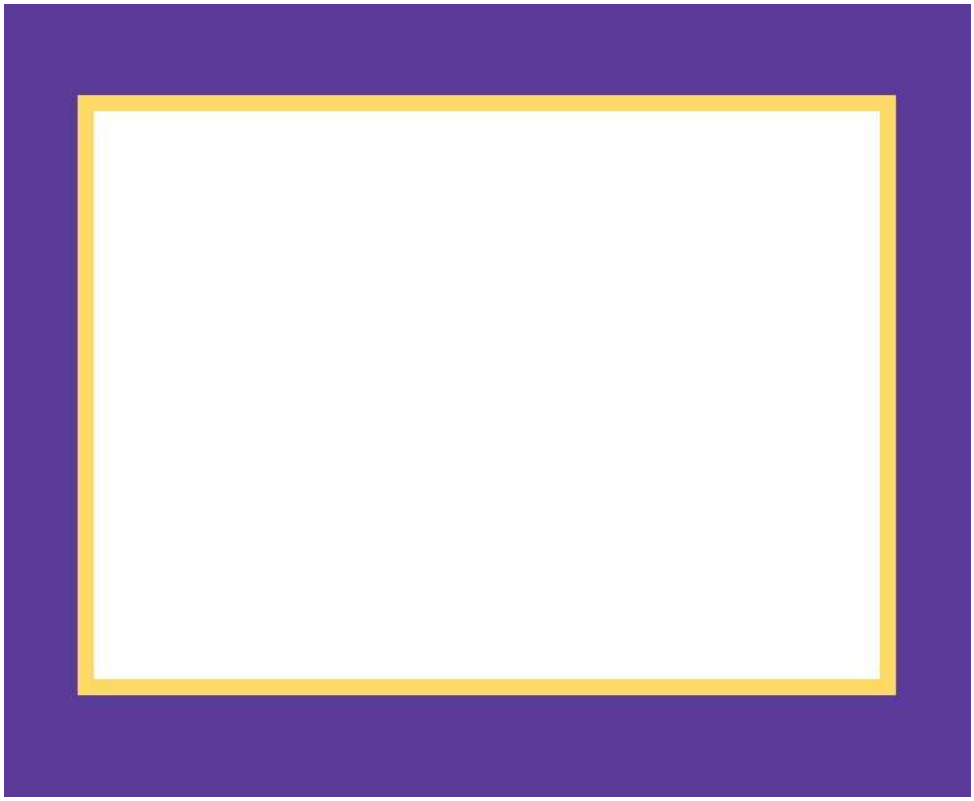
```
div {
    transform: rotate(30deg);
}
```



- Aumentar ou diminuir o tamanho do objeto

Aqui o objeto aumenta em 1,5 vezes:

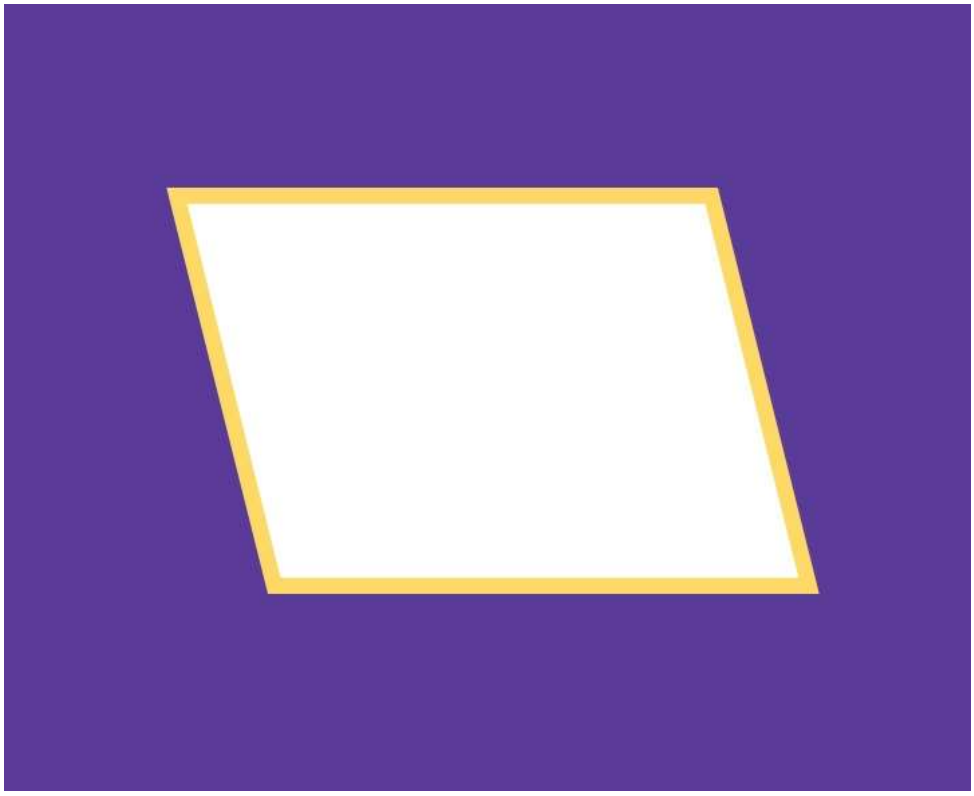
```
div {
    transform: scale(1.5);
}
```



- "Entortar" o objeto

Aqui os ângulos do objeto crescem ou diminuem em 20 graus:

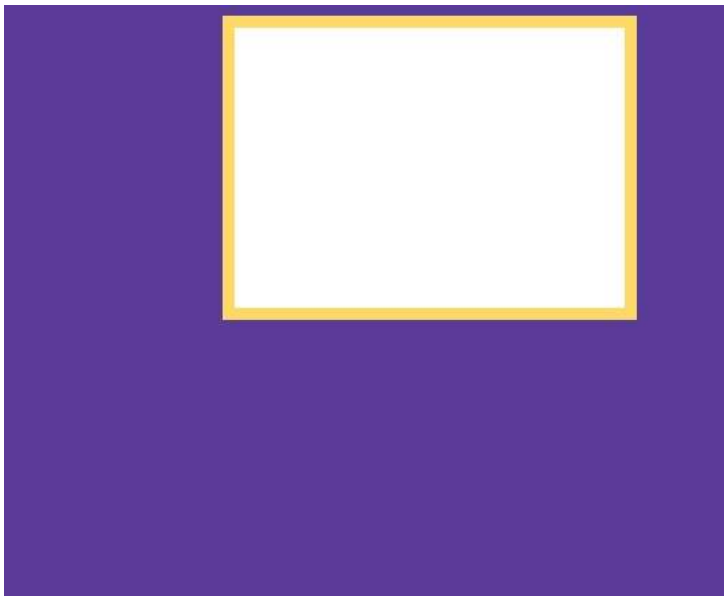
```
div {
    transform: skew(20deg);
}
```



- Transladar o objeto

Aqui o objeto foi transladado 10 pixeis para a direita e 50 para baixo:

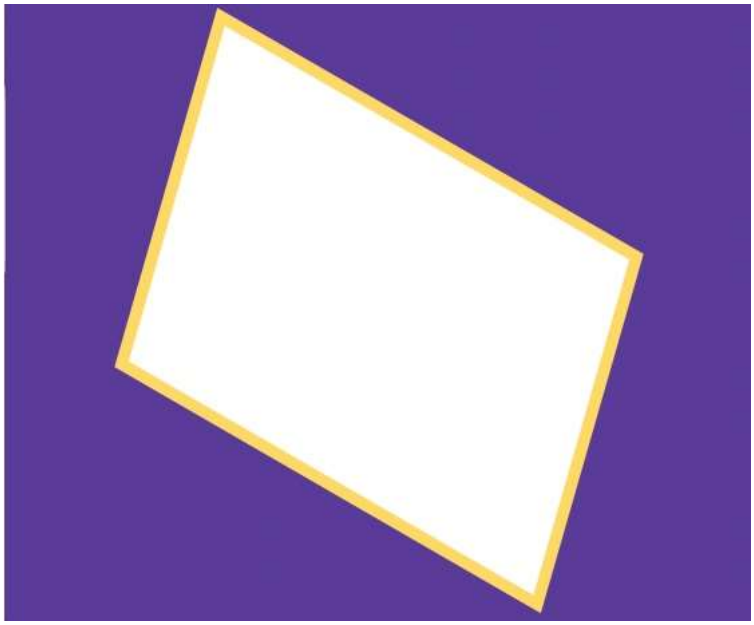
```
div {
    transform: translate(10px, 50px);
}
```



- Fazer tudo ao mesmo tempo

Aqui o objeto foi: entortado em 20 graus, rotacionado em 30 graus e aumentado em 1,2 vezes:

```
div {
    transform: skew(20deg)
        rotate(30deg)
        scale(1.2);
}
```

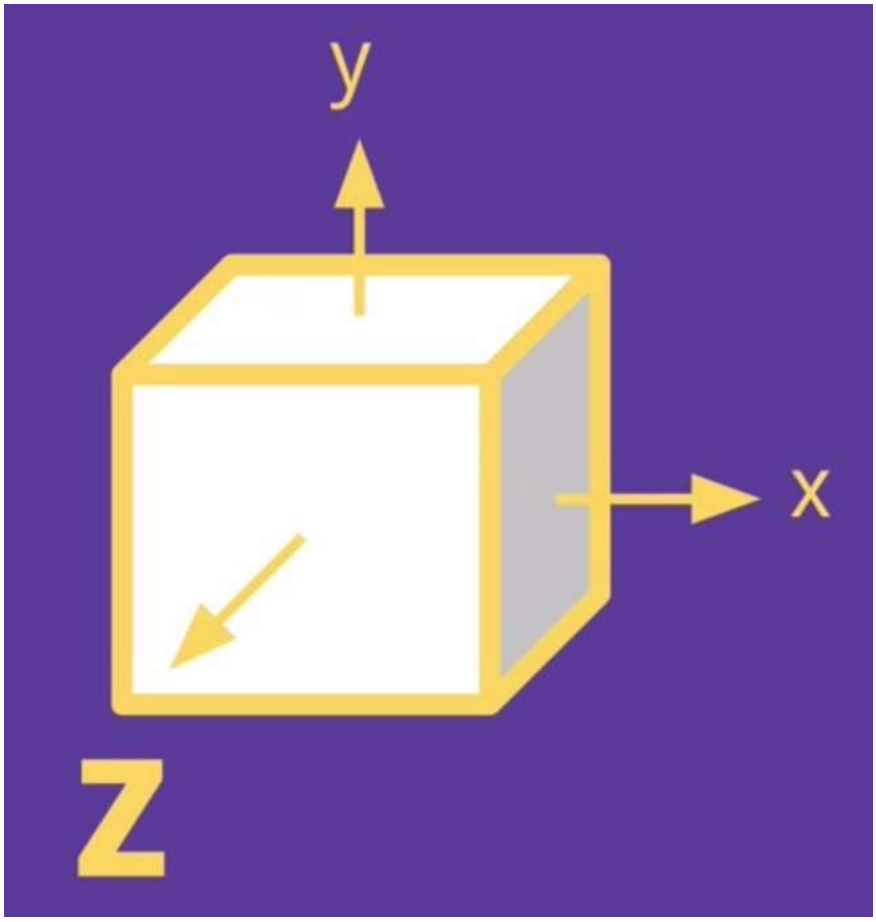


Como essa propriedade é relativamente nova, talvez seja interessante acrescentar antes dela o prefixo *BETA*, para versões mais antigas de navegadores:

```
div {
    -webkit-transform: skew(20deg)
        rotate(30deg)
        scale(1.2);
    transform: skew(20deg)
        rotate(30deg)
        scale(1.2);
}
```

Transformações 3D / propriedade *perspective*

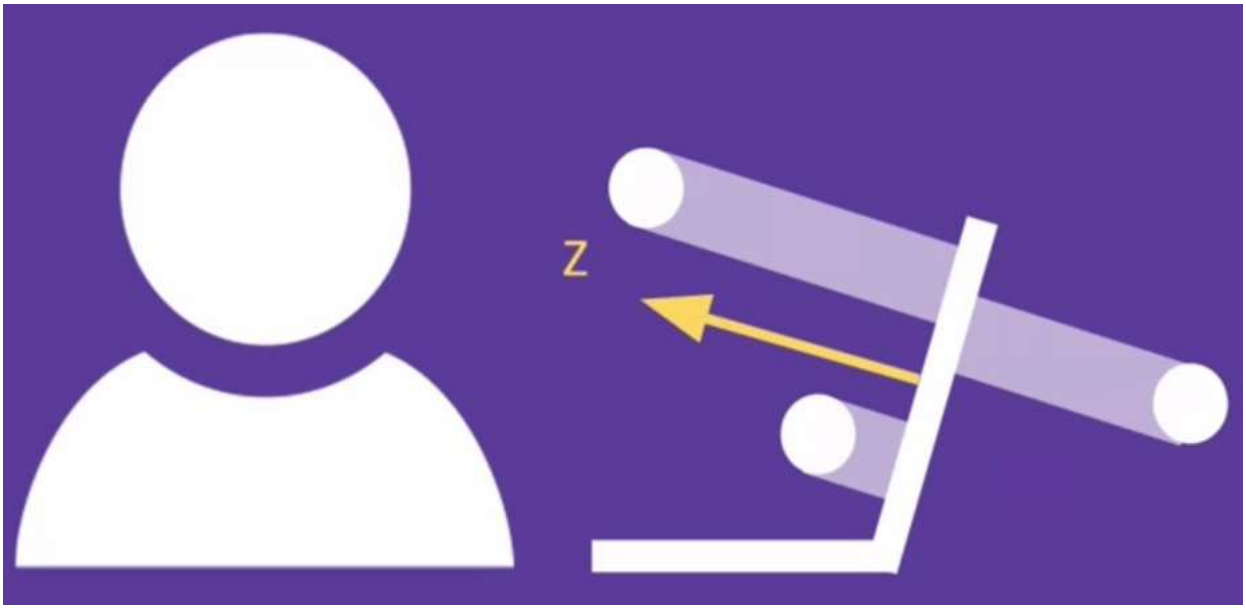
Nas transformações em 3D trabalhamos com três eixos para definir o tamanho dos nossos objetos: *x*, *y* e *z*. Este último é novidade, ele indica a profundidade do objeto. Os dois primeiros são eixos que já vínhamos trabalhando.



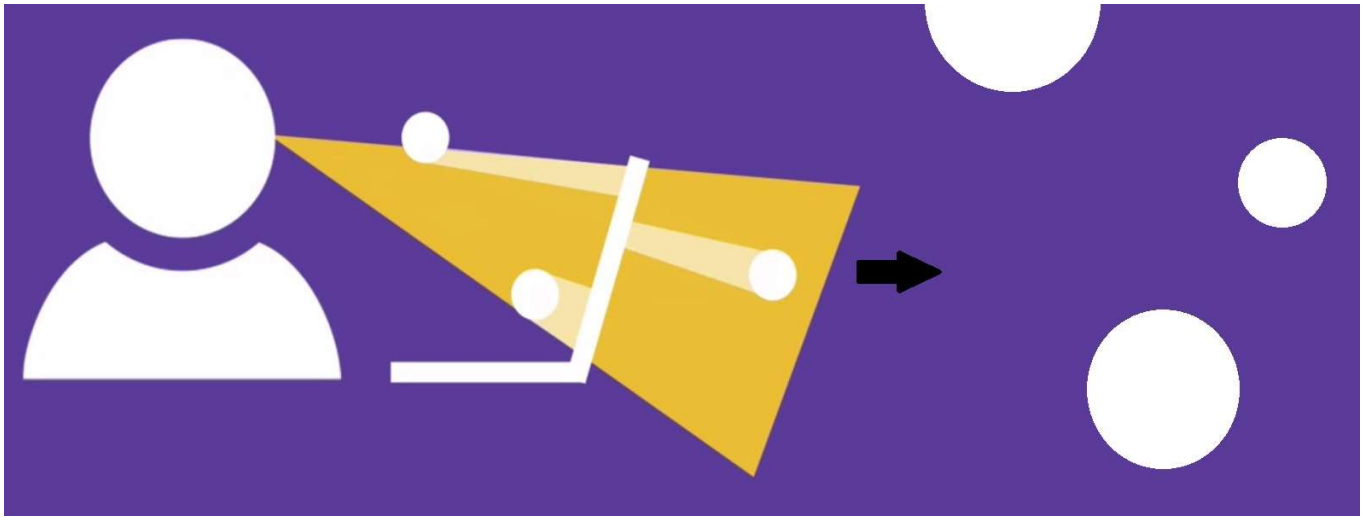
Agora especificamos para cada eixo suas transformações:

- translateX
- translateY
- translateZ

Se quisermos, por exemplo, que o objeto pareça estar saindo "para fora" da tela, mexemos no eixo *z*. Para isso usamos um efeito de projeção de luz e sombras, que dá essa impressão de tridimensionalidade.



Outro efeito importante para dar essa impressão é modificar algumas dimensões do objeto e suas sombras, pois apenas utilizando efeitos de luz, ainda teremos a sensação de que o objeto tem o mesmo tamanho. As sombras devem ter tamanhos diferentes.



Teremos esse resultado acima se projetarmos as sombras corretamente.

Porém, se nos afastarmos muito da tela do computador, essa diferença some e os círculos parecerão ter o mesmo tamanho. Tudo isso está relacionado à *perspectiva*.

Então, antes de especificarmos a cena 3D, devemos primeiro especificar a *perspectiva* que queremos usar: a que distância o usuário estará da tela para que possamos fazer o cálculo da projeção da cena 3D.

Conseguimos isso por meio da propriedade ***perspective***, que usa o sistema de cone de luz. Além, é claro, de nos utilizarmos das transformações já vistas.

Um outro detalhe importante é a angulação com que o usuário está olhando para a tela, o que causa efeitos diferentes de perspectiva. Para lidar com essas diferenças de posição do usuário, usamos a propriedade ***perspective-origin***.

Implementando o 3D no código

A cena que vimos com os círculos poderia ser um HTML desse tipo:

```
<div class="bolas">
    <div class="bola1"></div>
    <div class="bola2"></div>
    <div class="bola3"></div>
</div>
```

O primeiro passo para criarmos uma cena 3D com essas bolinhas é deslocá-las conforme a distância (profundidade) que queremos que elas estejam da tela:

```
.bola1 {
    transform: translateZ(2px);
}

.bola2 {
    transform: translateZ(-1px);
}
```

As translações positivas trazem o objeto para "perto" da tela e as negativas para longe.

A perspectiva é implementada no elemento pai, ou seja, em "bolas":

```
.bolas {
    perspective: 4px;
}

.bola1 {
    transform: translateZ(2px);
}

.bola2 {
    transform: translateZ(-1px);
}
```

Os "4px" não significa que o usuário estará visualizando a cena a 4 pixeis de distância, mas apenas para facilitar os cálculos, para ter uma ideia do que está perto e do que está longe de um ponto de referência.

Nós podemos ter várias cenas 3D independentes, cada uma com sua perspectiva:

```
<main>
    <section>
        ...
        perspective: 8px;
        ...
    </section>
    <section>
        ...
        perspective: 80px;
        ...
    </section>
    <section>
        ...
        perspective: 42px;
```

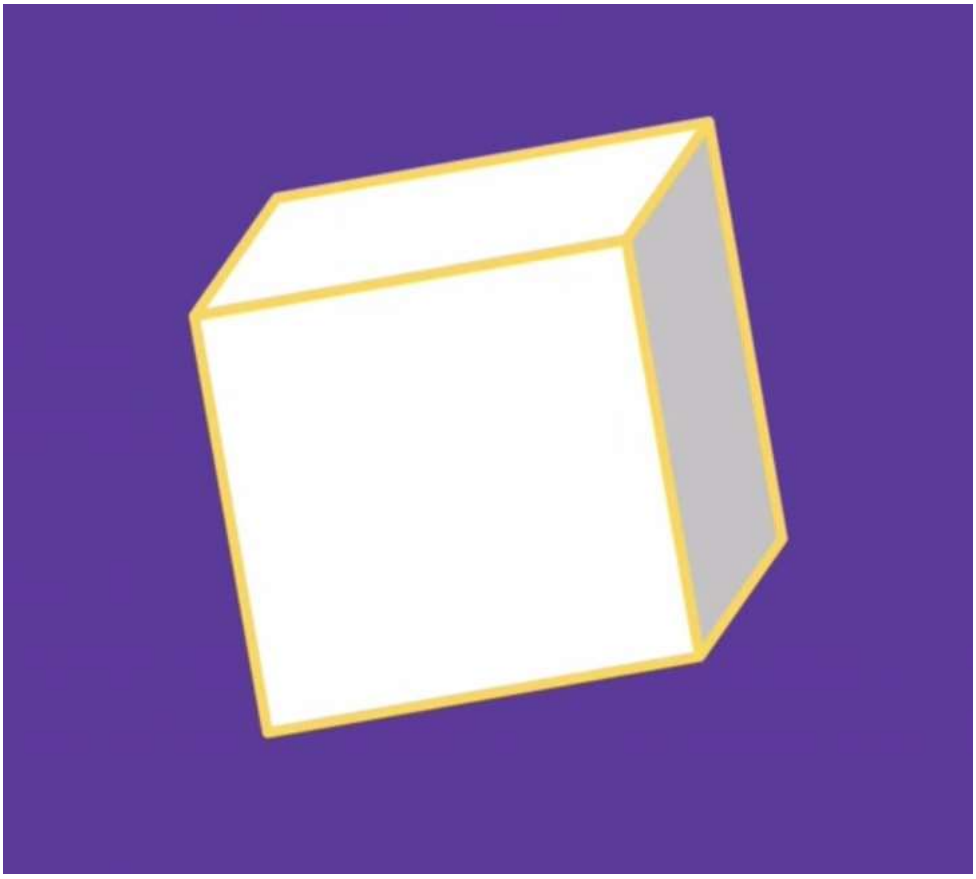
```
        ...
    </section>
</main>
```

E, dentro de cada cena, com sua própria perspectiva, também não precisamos nos limitar a um elemento. Podemos, por exemplo, definir propriedades para cada face de um cubo. Configuramos a *perspectiva* da cena, *transformamos* o objeto fazendo uma rotação ou deslocamento e *transformamos* cada face do cubo, rotacionando junto com o cubo, por exemplo:

```
<main>
    <section> <- [perspective: ...]
        <div class="cubo"> <- [transform: ...]
            <span class="face1">...</span> <- [transform: ...]
            <span class="face2">...</span>
            <span class="face3">...</span>
            <span class="face4">...</span>
            <span class="face5">...</span>
            <span class="face6">...</span>
        </div>
    </section>
    ...
</main>
```

O elemento *section* precisa pegar a transformação de dentro do cubo também. Para isso vamos utilizar a propriedade ***transform-style: preserve-3d*** dentro da classe "cubo", para que o navegador use as transformações feitas com cubo nos elementos dentro dele. Dessa forma preserva-se o efeito 3D de todos os elementos. Sem isso as faces não aparecem com esse efeito conjunto. O cubo rotacionaria, mas as faces continuariam com uma impressão achatada.

Nos utilizando de todas essas ferramentas, chegamos a um resultado como este:



Mais para frente de nosso curso, veremos como fazer para animar esse objeto.

Barra de rolagem para cenas em 3D / propriedade *overflow*

Como vimos, podemos ter diversas cenas 3D em nossa página. Cada uma delas pode ter sua própria barra de rolagem. Para tal, utilizamos a propriedade ***overflow*** com o valor *scroll*. O conteúdo desse elemento vai aparecer cortado e o usuário poderá rolar para visualizá-lo

```
<main>
    <section>
        ...
        overflow: scroll
        ...
    </section>
    <section>
        ...
        overflow: scroll
        ...
    </section>
    <section>
        ...
        overflow: scroll
        ...
    </section>
</main>
```

Se tivéssemos uma dessas *sections* com *10px* de altura e, dentro dela, um cubo de *100px* de altura, o usuário veria uma barra de rolagem e poderia movê-la para visualizá-lo.

É importante notar que esse *scroll* influencia na *perspective-origin*, ou seja, ao rolarmos a barra em uma cena 3D é como se estivéssemos nos deslocando, mudando a perspectiva. Os objetos que estiverem "mais perto" se deslocarão mais rápido que aqueles que estiverem "mais longe", dando o efeito de uma movimentação. Chamamos esse efeito de *Paralaxe*.