O estilo abaixo serve para colocar apenas uma imagem na pagina e separa-la por frames quando necessário. Pode-se fazer quantos frames precisar

.sprite {

background: url(../img/sprite.jpeg) no-repeat;

width: 162.6px;

height: 160.5px;

margin: 0 auto;

}

.frame1 { background-position: 0 0; }

Vale recapitular um conceito importante do JavaScript, o conceito de **closure**. O objeto retornado por createSprite guarda em uma de suas propriedades a função nextFrame() criada internamente em createSprite.Tanto isso é verdade que nossa intenção é usar a função desta forma:

// exemplo apenasvar sprite = createSprite('.sprite');

sprite.nextFrame();

Isso só funcionará, porque a função sprite.nextFrame() tem acesso ao contexto de createSprite() e a todas as variáveis e funções declaradas neles mesmo depois de createSprite() ter terminado de executar. Sendo assim, a chamada de sprite.nextFrame() que chama por debaixo dos panos moveFrame() e que modifica ao estado da variável current está se valendo deste recurso da linguagem JavaScript.

Quando criarmos uma função através de function declaration, mesmo que tenhamos definido a função no meio ou no final do nosso script elas serão sempre içadas para o topo do script. Se declaramos uma função dentro da outra, elas serão içadas para o início da função. Esse procedimento se chamada *function hoisting* (içamento de função). Vejamos um exemplo menos:

exibeNome();

function exibeNome() {

alert('Flávio Almeida');

}

O código acima funciona como esperado, porque antes que o interpretador execute nosso código, ele modificará a sua estrutura para:

function exibeNome() {

alert('Flávio Almeida');

}

exibeNome();

Todavia, se mudarmos a declaração do nosso código para function expression ele não funcionará:

exibeNome(); // não funcionará, exibeNome é undefined

var exibeNome = function() {

alert('Flávio Almeida');

}

Isso acontece porque o JavaScript iça também para o topo as declarações de variáveis, mas apenas sua declaração, sem qualquer inicialização. Para o interpretador JavaScript seu código ficará estará assim:

var exibeNome; // içou a declaração da variável

exibeNome(); // repare que não há qualquer valor

exibeNome = function() {

alert('Flávio Almeida');

}

Com este exemplo, fica claro a diferença entre as duas abordagens. Este instrutor prefere a segunda forma porque força o desenvolvedor a declarar as expressão de função antes do seu uso, o que é mais compreensível. Além disso, essa forma abre espaço para outras melhorias que fogem o escopo do ECMASCRIPT 5 e entram no escopo do ES2015 (ES6). No final do curso teremos um bônus no qual eu migrarei todo código para ES2015 para que o aluno veja como essa atualização do JavaScript pode ajudá-lo a escrever um código ainda melhor, além de indicar quais cursos de outras carreiras o aluno deve procurar para adiquirir esse conhecimento.

Para criar um array de uma melhor forma e colocar todos os espaços com o mesmo valor

lacunas = Array(palavraSecreta.length).fill('');

var exp = new RegExp(chute, 'gi')

g para buscar todos os resultados e não parar no primeiro encontrado, o i é de case insensitive