**OpenSlide HCJS 1.0**

O objetivo é facilitar a vida dos iniciantes em design web ou da queles que precisa de base lógica para fazer um pequeno sistema apresentador de imagens.

**1.0 Desenvolvimento.**

Esse sistema de slide é feito com HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) e a biblioteca para javascript jQuery. A versão do jQuery disponibilizada aqui é a 3.1.1 consulte [www.jquery.org](http://www.jquery.org/) para mais informações. Também é possível usar versões anteriores.

Cada tipo de código foi implementado o máximo possível em arquivos específicos para garantir a pureza do código.

**2.0 Entendendo o slide.js.**

Dentro do slide.js contém todo o código que dá vida ao slide, mas para isso ele conta com a ajuda do slide.css em efeitos visuais.

**2.1 Variáveis.**

* **configOS:** Variável do tipo objeto. Nela contém configurações referentes a delay de inicialização delay de troca de slides;
  + **Start\_time**: Atributo de configuração para o delay na inicialização do sistema. 0 (zero) é igual a imediato, 1000 equivale a 1 segundo;
  + **Delay\_slide**: Atributo de configuração para delay entre slides. 1000 equivale a 1 segundo, tempo padrão é de 4400. Atenção: valores abaixo de 2000 causa má experiência.
* **slides**: Variável do tipo objeto. Nela é carregado todos os slides disponíveis no HTML;
* **valSlides**: Variável do tipo inteiro. Armazena a quantidade de slides disponíveis;
* **needleSlide**: Variável do tipo inteiro. Armazena a posição do atual slide que está sendo exibido;
* **clock\_timer\_slide**: Variável do tipo número. Guarda o tempo restante do delay de slides;
* **timer\_slide**: Variável do tipo “função”. Guarda a função de inicializar o timer do delay de slides, fazendo com que fique disponível de modo publico no algoritmo;
* **boxIS**: Variável do tipo objeto. Guarda todas as entidades referente a região descritiva do slide.
* **desc\_openslide**: Variável do tipo objeto. Guarda parte das entidades referente a região descritiva do slide.

**3.0 Entendendo as funções.**

Cada função nesse algoritmo tem utilidade crucial para o funcionamento do slide. Cuidado ao modificar!

**3.1 Funções.**

* **loadOpenSlide:** Essa é uma das funções relevantes nesse algoritmo, pois ela é quem exibe o primeiro slide colocando ordem na sequência de exibição. Ela remove a classe “hidden-slide” que tem como função ocultar o slide e coloca a classe “show-slide” que tem a função de exibir o slide, em seguida passa o parâmetro de posição do slide atual para a função “alter\_info\_slide()”;
* **alter\_info\_slide**: Função com grande importância. Ela é responsável por carregar e exibir o título, descrição, url e posição referentes ao slide que está sendo atualmente exibindo. Título, descrição e url são respectivamente carregados através dos atributos “data-title”, “data-desc”, “data-url” presentes na tag “<div></div>” de classe “.box-slide” (<div class=”box-slide”></div>). Esses atributos são pegos com referência na posição passada pelo parâmetro presente na variável “s”. A posição do slide atual é exibida usando com referência a variável “needleSlide” (documentada em 2.1), sempre somada com a constante “1” para apresentar valores em um sistema de 1 a 10;
* **next\_slide**: Essa função avança na exibição de slides conforme for solicitado pelo timer de delay ou manualmente pelo botão next. Essa função analisa a variável **needleSlide** com o objetivo de saber se o valor presente é superior ao valor total de slides disponível (disponível na variável “valSlides” 2.1), caso seja, a variável é zerada (zero representa a posição do primeiro slide) caso contrário needleSlide será somada +1. Quando se trata de array ou objeto dentro de objeto ou array, a chave de identificação é definida com sistema decimal (0 à 9). Nessa função o corre também a ocultação e exibição de slides usando a mesma lógica da função loadOpenSlide. Em seguida passa o parâmetro de posição do slide atual para a função “alter\_info\_slide()”;;
* **back\_slide**: Segue a mesma lógica da função **next\_slide,** porém ao contrário. A variável **needleSlide** é analisada com o objetivo de evitar que seu valor seja inferior ao valor total de slides (presente em “valSlides’ 2.1), caso seja, needleSlide será definida para o valor representativo ao último slide, exemplo:   
  **valSlides = 3,   
  se needleSlide == -1,   
  needleSlide = valSlides -1**.  
  valSlide foi subtraído com a constante -1 para retorna o valor do último slide em decimal (0 à 9) ao em vez de (1 à 10). Essa ação é essencial para a localização do slide dentro do objeto. Em seguida passa o parâmetro de posição do slide atual para a função “alter\_info\_slide()”;
* **timer\_slide**: essa função é o cronometro de delay aplicado entre a exibição de cada slide. É utilizado a função “setInterval()” para cronometra o tempo do delay. Esse tempo é carregado a partir da variável objeto “configOS.Delay\_slide”. A cada intervalo esgotado a função “next\_slide()” é chamada para providenciar a substituição do atual slide;
* **restart\_timer\_slide**: essa função é responsável por reiniciar o timer de delay acionado dentro da função “timer\_slide()” que está carregado dentro da variável “clock\_timer\_slide”, usando a função “clearInterval()”. Para garantir a reinicialização, a função “timer\_slide()” é chamada;
* **stop\_timer\_slide**: essa função é responsável por para o timer do delay com a função “clearInterval()”, sem reiniciar o mesmo;
* **control\_infoSlide()**: Essa função cuda visualmente da região informativa do slide (titulo e descrição). Ela recebe como parâmetro na variável “op” apenas 2 valores, f ou h (full ou half). Usando um esquema condicional será determinado se a região informativa do slide está sendo exibida expandida ou pela metade. Em consequência são será aplicado efeitos visuais, manipulado as classes.