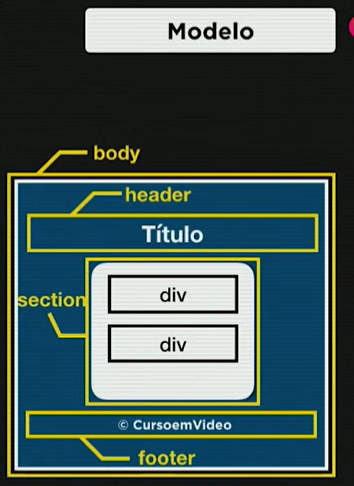
**Exemplos Práticos**

Para maior compreensão do que vimos até agora, e testarmos nossos conhecimentos, vamos realizar alguns exercícios em que as condições não serão mais simples, vão receber mais de uma “condição” em meio a sua estrutura. Dessa forma o primeiro exercício será replicar uma tela aonde traz uma imagem que corresponda ao horário do dia, ou mesmo ao período.

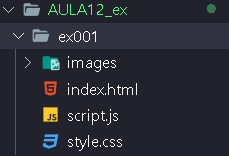


Além da imagem ser correspondente, quero uma mensagem que informa as horas naquele exato momento, sem a necessidade de nenhuma interação do usuário.



Da seguinte forma, vamos usar esse modelo para criar a nossa página, aplicando assim para os dois exercícios, assim podendo ser copiado o mesmo código **CSS**. O objetivo aqui é treinar os conhecimentos obtidos até agora em **JavaScript**.

Os arquivos necessários para o desenvolvimento das duas atividades, são os seguintes:



Para os exercícios precisamos de imagens que correspondam ao mesmo tamanho entre elas, ou imagens previamente editadas nos formatos corretos, para que não ocorram distorções quando tratarmos elas nas linhas de códigos.

Com tudo informado e esclarecidos, vamos para o **HTML** entender como o exercício será desenvolvido.



**­Linha** **7**: link para o arquivo **CSS**, a estilização será feita por meio de um arquivo esterno, deixando assim o código mais link e organizado.

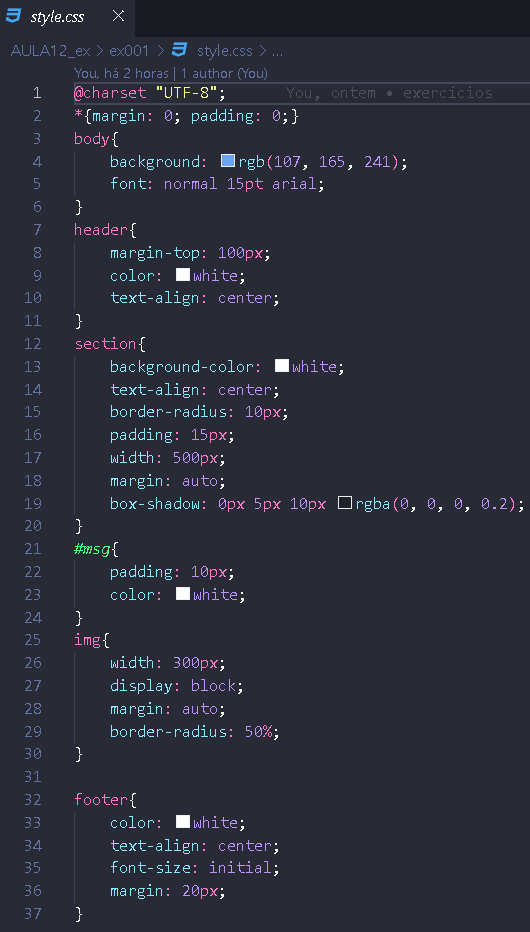
**Linha** **9**: um novo elemento é introduzido aqui, o **onload=””**. Funciona da seguinte forma, toda vez que o body for carregado a função que estará entre aspas irá ser carregada, assim eliminando a necessidade de uma futura interação do usuário.

**Linha** **14**, **15** e **16**: div com um id para que possamos identificar no **CSS** e no **JavaScript**, e um texto no seu interior para que possamos ter uma prévia visualização de como o texto irá aparecer em tela.

**Linha 17**, **18** e **19**: div como já vimos anteriormente, porém com uma imagem em seu interior. A tag img ainda leva um id para que o código possa ser estilizado por meio da **CSS** e também haja interação por meio de JavaScript.

**Linha 22**: os caracteres e números antes do nome escrito representam o símbolo de “copyright”.

Assim completamos a primeira parte, o código **HTML** já está pronto e agora podemos seguir para o próximo passo.



Aqui vamos começar com o **CSS**, de maneira vamos resumir oque está acontecendo com cada elemento disposto na página.

Linha 1: padrão que sempre seguirmos para que os caracteres permaneçam em um padrão que não apresentem erros.

Linha 2: padroniza a pagina para que não tenha margin ou padding predefinidos, que atrapalham nas dimensões customizáveis por **CSS**.

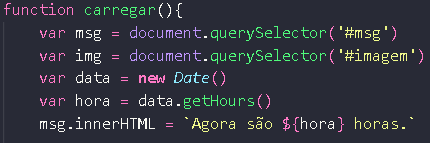
Linha 3: padroniza o fundo da tela, e a fonte padrão a ser usada, mas de forma pratica.

**font**: normal (weight) 15pt(size) arial (family);

Assim Podemos atribuir especificações de maneira mais prática, mas se não seguir a sequência nada vai acontecer.

As demais configurações não fogem do que já vimos até agora.

Agora vamos partir para o código **JavaScript**, importante dizer que dessa vez vamos partir para um código externo, ou seja, assim como o **CSS** vamos realizar o código por fora do **HTML**. No documento **HTML** na linha 25, podemos observar o comando com o atributo **src=””**, que nada mais é a localização do arquivo, mas diferente do **CSS** tem que ser feito de dentro de body, e para ficar de maneira “correta”, no final de todas as tag’s **HTML**.



Bom, para começarmos a nossa **function** mas armazenar os valores iniciais, e vamos rodar um teste logo em seguida para que possamos assegurar que o inicio já está funcionando de maneira correta.

Vamos agora entender o que está escrito em cada uma das linhas, vamos usar o método **querySelector** para capturar os elementos.

**1º** **var msg** vai receber valor capturado no elemento com o id #msg

**2º** **var img** vai receber o valor capturado no elemento com id #img

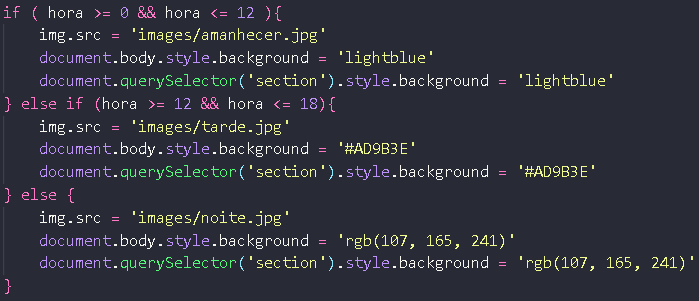
**3º** **var data** vai receber uma função própria do **JavaScript** que serve para capturarmos dados relativos a data, porem esse dado é informado pelo nosso próprio computador, ou seja, a data que ele mostrar vai ser informada para a nossa variável.

**3º.1** **var hora** vai ser alimentada pelo valor dado em var data, mas como? De maneira simples, em **JavaScript** para obtermos informações de um elemento a certo nível colocamos um ponto e depois disso existem opções para isso, nesse caso vamos usar o **getHours()**, que vai obter somente a hora da variável data.

E por fim, para testar se os dados estão sendo capturados de maneira correta, vamos escrever em tela com o método que já vimos anteriormente.

Para compreender melhor como vai ocorrer, quando a página carregar o atributo **onload** vai entrar em ação e vai na mesma hora chamar a função, que armazena os valores correspondentes e os mostra em tela. Assim que realizou esse processo, espero que tenha atingido o resultado esperado.

Agora para dar andamento na função, temos que informar as condições e também oque vai acontecer caso as informações forem o suficiente para cumprir alguma das condições.



Se forma resumida, quando o horário informado for igual a um determinado período, a imagem e as cores de fundo vão ser alteradas. Mas vamos acompanhar mais de perto para detalharmos melhor.



Se (**if**) a variável hora for maior ou igual ( **>=** ) a 0 (zero) e ( **&&** ) hora for menor ou igual ( **<=** ) a 12 (doze). Temos a primeira condição, que se entre esses valores a condição for atendida algo irá acontecer.



No caso a variável img irá receber um novo atributo ( **img.src** ) , um de encontrar imagem, porém como no **HTML** isso já existe, o que vai acontecer é uma troca de valores.

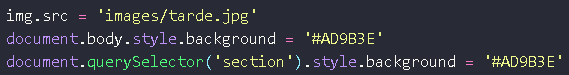


Além disso os estilos **CSS** vão receber nossos valores, e aqui vimos duas formas de fazer isso, porém as duas resultam no mesmo objetivo.

Agora, temos um incógnita temos o se ( **if** ) dentro de uma função, mas como vamos ter outra condição se o próximo passo é o senão ( **else** ) ? Para isso temos o senão se ( **else** **if** ), então vamos para a prática.

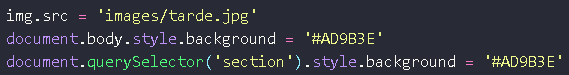


Explicado, a forma de ler e entender é a mesma do código anterior, tudo entre esses valores vão ter novas características, e elas vão ser as seguintes.



Da mesma forma que acontece na condição anterior, os mesmo valores vão ser alterados.

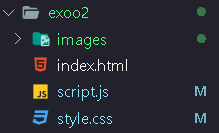
Dito isso, se as condições forem somente essas entra de fato o senão ( **else** ), que simplesmente quando nenhuma das condições é atendidas ele é acionado.



Da mesma forma que as anteriores, os mesmos valores vão receber alterações.

Se o código funcionou até aqui de maneira correta, vamos testar substituindo o valor da variável hora, por um numero que represente um horário qualquer. Assim vamos validar se o código de fato está funcionando corretamente.

Já conseguimos entender como que funciona ó código com um condição composta, ou seja, com mais de uma condição e também como alterar valores e adicionar atributos de elementos **HTML** e estilos **CSS**. Assim vamos continuar com o nosso próximo exercício que vai exigir um pouco mais de atenção.

Assim fica à disposição dos arquivos que vamos utilizar para esse novo exercício, e como no anterior vamos usar tanto o **JavaScript** e **CSS** com arquivos externos.

A construção do **HTML** vai seguir do mesmo jeito que a

anterior, mas vamos observar o exemplo para entender o que irá mudar em relação ao anterior.



Bom aqui vamos observar que temos algum elementos dos tipo input que necessitam que tenham interação com o usuário.

Então o exercício consiste em:

* Usuário colocar o ano de nascimento.
* Usuário informa o seu sexo.
* Usuário clicar em um botão para validar os dados inseridos.
* Em seguida o “sistema” calcula a idade do usuário com base no ano informado, e retorna com uma mensagem que contém o sexo e a idade e uma foto representando esses dois valores.

Primeiro vamos com o **HTML** e confirmar as mudanças que vão ocorrer e entender como que elas funcionam.



**Linha 15**: é o início da nossa primeira div, e como o documento tem que ter duas, aqui vai conter toda a estrutura de interação com o usuário.

**Linhas** **16**, **18**, **19**, **29**, **30** e **32**: note que todas essas linhas estão dotadas das tag’s <p>, e que vão servir para deixar o texto na tela de maneira mais organizada e fácil para uma estilização, e dentro de cada um desses parágrafos estão nossos elementos do tipo input.

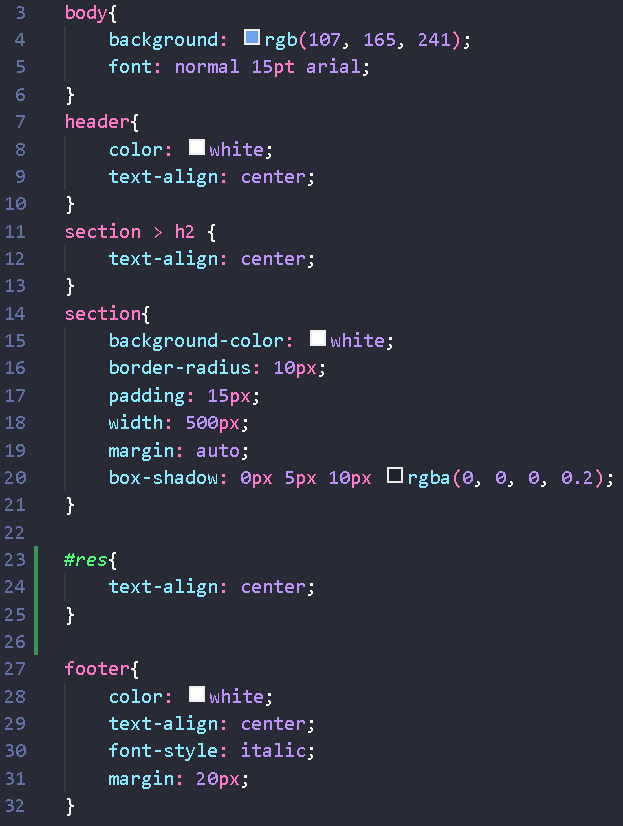
**Linha 17**: um input do tipo ***number***, e como o usuário só pode digitar números esse é o melhor tipo de input, poderia ser o ***date***, porem não é o necessário para esse exercício.

**Linha 22** e **26**: input do tipo radio, que servem como opção de seleção, mas vamos entender algumas coisas sobre seu uso antes:

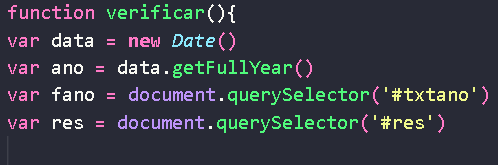
1. O ***name*** pode ser igual, mas se o ***id*** for igual não vamos conseguir selecionar uma opção diferente.
2. O atributo ***checked*** não necessariamente precisa estar sendo usado, já que apenas deixa uma das opções selecionadas por padrão.
3. A tag ***label*** é extremamente importante ser usada aqui para que seja melhor adaptado para uma versão de celular.

**Linha 31**: aqui var ser o nosso botão de ação, vai também receber um atributo que chama a função **JavaScript**, assim fazendo toda a interação com o usuário.

**Linha 34**, **35** e **36**: uma div com um texto que somente mostra se o código está funcionando, já que nessa parte aonde vai acontecer o resultado da interação do usuário, como a mensagem e a foto.

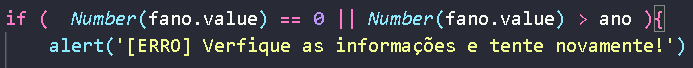


Nessa configuração de estilo **CSS**, não há nenhuma configuração ou estilização que não teríamos visto antes.



Aqui vamos nos basear no que houve no exercício anterior, da mesma maneira vamos capturar o valor do dia, mas nesse caso somente o ano, por isso ***getFullYerar()***, assim somente o ano é capturado para a variável. Logo em seguida precisamos dos valores inseridos pelo usuário no campo numérico e também da informação da div em brando que ia aparecer o resultado.

Até aqui nada é muito diferente do que vimos anteriormente, mas tem suas particularidades para funcionar da maneira que pensamos.



Bom aqui temos a primeira condição, e essa condição é para apresentar erro (se houver). Digamos que com varia condições no código é melhor que a primeira seja jogo a de validação, para que não continue.

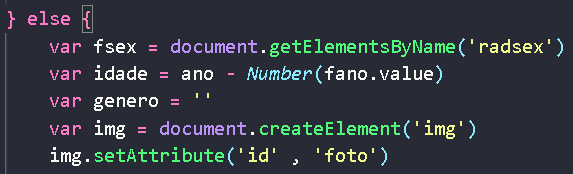
Sendo assim vamos entender que:

Se (***if***) o número digitado no campo numérico ( ***fano.value***) for igual ( ***==*** ) a zero, ou ( **||** ) o número digitado no campo numérico ( ***fano.value***) for maior que o ano capturado, a seguinte mensagem vai aparecer na tela com o elemento ***alert***.

Para garantir que o código vai entender o valor digitado como numero vamos usar o conversor de ***string*** para número, o atributo ***Number***.

Para garantir que a variável realmente tenha o seu valor na operação, vamos usar ***.value*** (***variável.value***), sendo assim somente o valor da variável vai ser mostrado para o código.

Agora que colocamos a nossa validação de erros, o que for acontecer em seguida é toda a interação código com o usuário.



Se (***if***) é a validação de erro, senão (***else***) vai ser código acontecendo de forma completa, então:

Variável ***fsex*** para capturar o elemento que contem os dados de sexo.

Temos que calcular idade também, para isso uma variável ***idade*** que recebe o valor do ano capturado do sistema e o valor inserido pelo usuário (***fano.value***). E da mesma forma realiza uma operação para gerar o valor da idade.

Uma variável que vai corresponder o resultado obtido pela variável ***fsex***, e vai ficar mais fácil de entender no andamento do exercício. Essa variável recebe o valor de duas aspas para simbolizar que é uma *string* vazia.

Por fim uma variável para capturar os dados da tag img, já que precisamos trocar a foto dependendo da idade do usuário. E também como a tag img não tem id para uma melhor identificação, por JavaScript podemos fazer isso com ***setAttibute(‘id ou class’, ‘nome’)***, ficando dessa forma ***img. setAttibute(‘id’, ‘foto’)***. Essa forma fica visível somente no **JavaScript**, no **HTML** nada acontece.

Bom já que validamos o ano, e já tem a forma para calcular a idade (var idade), temos que filtrar pelo sexo também, e agora que vai entrar o primeiro se (if) em meio a condição.

Já que temos mais de uma opção selecionável o JavaScript entende que a primeira opção tem o valor 0, e para identificar esse valor em meio ao código **[0]**. E da maneira que esta é lida da seguinte maneira.

Se (**if**) variável com valor do sexo (***fsex***) for selecionado masculino **[0]*.checked***.

Assim o valor de **genero** vai receber ‘Masculino’.

O mesmo vale se for Mulher, porem como senão se (else if)

Como explicado, a opção de Mulher vai ser representada por ***[1].checked***, mudando também o valor de **genero**.

E dentro dessas condições vão existir mais condições, nas duas situações os códigos abaixo vão ser usados, alterando parâmetros para que se adequem ao tipo de usuário que virá a ser representado.



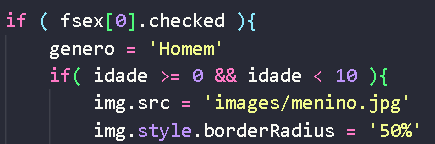
Se (***if***) a **idade** for maior ou igual ( ***>=*** ) a zero (***0***) e (***&&***) **idade** for menor ( ***<*** ) que dez (***10***), vamos ter alguma coisa acontecendo. Vale lembrar que isso ocorre assim que o código identifica o valor de **fsex** e de **idade**.~



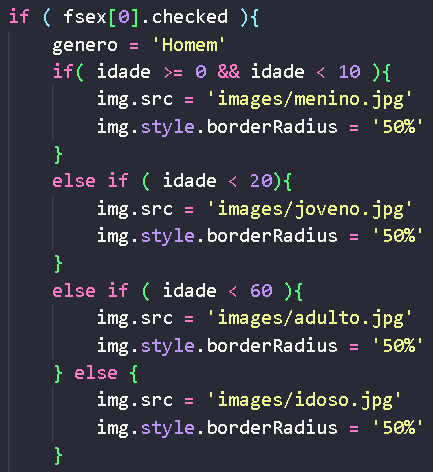
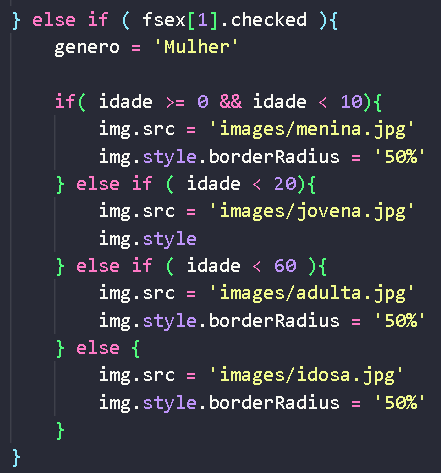
Se for masculino e a idade for entre os valores informado acima, eu vou aplicar os seguintes parâmetros em **JavaScript** que valem para o **CSS**.

***img.src = ‘’*** : localiza uma imagem para colocar no lugar do valor informado na variável, tem o mesmo peso de ***<img src=””>***.

***img.style.borderRadius = ‘50%’*** : define o valor de ***border-radius***, tem o mesmo peso do atributo **CSS**, assim fazendo a imagem ter o formato arredondado.



Para visualizar de maneira prática como ficaria o código com essa condição, mas como que fariam com mais condições para outras idades e para o gênero mulher?



Teste, faça e refaça o código. Pense em melhorias e veja como o código reage a mudanças. E a cada paço teste para sentir o que está acontecendo e mudando.