PROJETO INDIVIDUAL

Módulo 2 – Mensagem Oculta

RESILIA | Senac

CONTEXTO

Lembra daqueles filmes com mensagens codificadas?

O tema “Cifra de César” é utilizado em entrevistas por contemplar um conceito importante para desenvolvedores que é aprender a trabalhar e manipular dados recebidos do usuário.

Neste projeto iremos trabalhar com codificação de mensagens.

O objetivo deste projeto é explorar os conceitos de manipulação do DOM aprendidos nas aulas, assim como reforçar e desenvolver conceitos de lógica, funções e manipulação de arrays usando Javascript.

O QUE É PARA FAZER?

Criar uma mensagem codificada e escolher o algoritmo a ser utilizado.

Criar uma codificação de mensagens onde a pessoa usuária deverá ser capaz de inserir uma mensagem a ser codificada ou decodificada, escolher o algoritmo a ser utilizado e receber o retorno da mensagem.

COMO FAZER?

Requisitos

Sua aplicação deve ser capaz de codificar e decodificar mensagens utilizando tanto base64 quanto cifra de césar. Para isso, ela deve conter um formulário com:

- Um campo de entrada textual, da mensagem que será codificada ou decodificada;
- Um campo de seleção, com as opções "cifra de césar" e "base64" com o seguinte comportamento:
  - Com "cifra de césar" ativo na seleção, um novo campo deve surgir no formulário para que seja possível fornecer o incremento utilizado na cifra
  - Dois campos radiais (radio buttons) com os textos: "codificar" e "decodificar"
  - Um botão que, com "codificar" selecionado exibe o texto "Codificar Mensagem!" e com "decodificar" selecionado exibe "Decodificar Mensagem!"
  - Um outro campo textual deve ser utilizado na página para que o resultado da codificação e decodificação possa ser exibido para a pessoa usuária.

EXTRAS

Requisitos Extras

- Na Cifra de César ser capaz de diferenciar maiúsculas e minúsculas, ou seja, se a letra for maiúscula, a resposta deve conter uma letra maiúscula incrementada e se for minúscula, uma minúscula. Ex: “Oi” com 2 de incremento vira “Qk”.
- Conseguir ignorar qualquer caractere que não for uma letra, ou seja, se encontrar um caractere especial (por exemplo: /, !, $, %) ele deve ser ignorado. Ex: “oi, eu sou um dev!” -> “qk, gw uqw wo fgx!”
- Ter um texto descritivo contextualizando a pessoa usuária sobre o projeto apresentado e ensinando a usá-lo.

COMO FAZER?

Dicas

- Divida seu projeto em partes como por exemplo:
  - Criar uma função que resolve a base64;
  - Criar uma função que resolve a cifra de César;
  - Desenvolver o frontend com o forms necessário;
  - Usar o DOM para pegar as informações do forms;
  - Checar se o forms está funcionando corretamente e você está conseguindo pegar os eventos corretamente;
  - Aplicar as funções criadas nos dados recebidos do DOM.

COMO FAZER?

Dicas

- O algoritmo de base64 pode ser muito complexo para ser implementado do zero. Ainda bem que o javascript já traz um método nativo que faz isso por nós!
- Comece pelo básico! Tente primeiro resolver o problema da maneira mais básica possível, depois implemente melhorias.
- Pesquise o máximo possível! Esse é um algoritmo comum no mundo da programação, quem sabe você não encontre dicas que te ajudem a resolvê-lo!

COMO FAZER?

Documentação e referências

- Sobre a cifra de César: <https://pt.wikipedia.org/wiki/Cifra_de_C%C3%A9sar#:~:text=%C3%89%20um%20tipo%20de%20cifra,E%2C%20e%20assim%20por%20diante.>
- Sobre a base64: <https://marquesfernandes.com/self/o-que-e-base64-para-que-serve-e-como-funciona/>
- Referências de Eventos, MDN <https://developer.mozilla.org/pt-BR/docs/Web/Events>
- Trabalhando com forms <https://developer.mozilla.org/pt-BR/docs/Learn/Forms>

F.A.Q.

Posso utilizar bibliotecas?

Sim, desde que os demais requisitos não sejam afetados por conta disso.

Preciso utilizar manipulação do DOM?

Sim, esse requisito é obrigatório e as informações devem ser exibidas no HTML.

MÃO NA MASSA

Momento 1 - Início

Planejamento é a parte mais importante de um projeto

- Comece criando um plano de ação!
- Utilize o tempo descobrir o que você já sabe e o que ainda falta aprender;
- Pesquise sobre os algoritmos e pense na aplicação. Como você pretende criar as funções que vão resolvê-los?
- Aproveite para começar a rascunhar como vai ser sua página. Pode ser até em um pedaço de papel!

MÃO NA MASSA

Momento **2** - Desenvolvimento

Hora de colocar o planejamento em ação!

Comece pela parte que você se sente mais confortável, seja pelo desenvolvimento da tela em html/css seja pela criação das funções.

Sempre procure testar seu código conforme for desenvolvendo para evitar surpresas no final.

MÃO NA MASSA

Momento 3 - Aplicação

Agora é a hora de fazer a parte que você deixou para depois.

- Se a página já estiver pronta, vamos trabalhar nas funções e vice-versa.
- Continue testando cada parte.
- Quando as duas partes estiverem funcionando, é hora de juntar tudo! Teste! Teste! Teste!

MÃO NA MASSA

Momento 4 - Final de projeto

Toques finais! Verifique se seu projeto está atendendo todos requisitos necessários.

Sobrou tempo? Vale ver se tem algum extra que pode ser aplicado. Cuidado para não quebrar o que já está funcionando.

Não esqueça de subir seu código para o Github e ativar o Github Pages. Vale conferir se ele está funcionando bem lá também.

RUBRICA

Conteúdo	Habilidades
Organização do Código	1. O projeto está organizado em pastas 2. O conteúdo de HTML/CSS/JS está separado em arquivos diferentes.. 3. As imagens utilizadas estão separadas em uma pasta para elas. 4. O projeto não apresenta erros ao ser executado no navegador. 5. Nenhum código CSS/JS está inline em arquivos HTML.
JavaScript	1. O código executa corretamente. 2. Foram aplicados conceitos de POO e de reutilização/reaproveitamento de código. 3. O código não apresenta problema de sintaxe. 4. As variáveis estão de acordo com o ES6 (sem utilização de var). 5. Os eventos foram utilizados de maneira correta aplicando funções anônimas.
CSS (utilização de seletores)	1. Foi utilizado CSS no projeto e o código foi entregue. 2. Seletores avançados no CSS e código CSS otimizado. 3. O Projeto não contém erros de sintaxe. 4. O projeto está responsivo.. 5. Foi utilizado manipulação de estilo do CSS junto ao DOM.

RUBRICA

Conteúdo	Habilidades
Resolução de Problemas	1. A página possui um forms com os campos solicitados.. 2. Foram implementadas as duas formas de codificação de maneira correta. 3. Foram implementadas as duas formas de decodificação de maneira correta. 4. Caracteres especiais não são alterados na codificação.. 5. A cifra de César é sensível (age de maneira diferente) a letras maiúsculas e minúsculas.
Git/GitHub	1. Entregou o link do repositório no Github. 2. O código está completo/funcionando no Github. 3. Além do código, foi colocado um arquivo READ.me explicando do que se trata e como pode ser executado o projeto. 4. O código foi enviado em commits por etapas. 5. As descrições dos commits/PRs estão bem redigidas e apresentam bem as mudanças realizadas.

Até a próxima e
#confianoprocesso

RESILIA | Senac