Webdev - Módulo 2
Roteiro de aula

Aula 6 - Hard: Que dia é hoje!

Tópicos da aula:

- Datas
- Datas com Date e Day.js
- Branches
- Merge
- Fluxo de Trabalho
- Revisão do conteúdo do módulo visto até então

Objetivos da aula:

1. Compreender a necessidade de se trabalhar com datas
2. Entender como trabalhar com datas
3. Entender que algumas incompatibilidades podem ocorrer e saber proceder em situações assim, no JS.
4. Aprender a trabalhar com Day.js
5. Compreender como fazer ramificações (branches) em um fluxo de trabalho
6. Entender como fazer mesclagem (merge) em um fluxo de trabalho
7. Aplicar a dinâmica do ELI5 para revisar alguns conteúdos do módulo

Atividades da Aula

➔ **Problemas com datas**: <https://caniuse.com/?search=date>

➔ **Atividade 1: Quebra galho**
- Simular um projeto em grupo com Github;
- Trabalhar duplas ou trios;
- Escolher a pessoa que ficará responsável por aprovar as PR's;
- Essa pessoa deve criar um repositório no Github e as demais devem cloná-lo usando o comando *git clone*
- Cada integrante deve criar sua branch local usando o comando *git checkout -b <nome da branch>*

- Cada integrante deve criar um arquivo e nele escrever um código utilizando algum dos métodos de manipulação de data vistos em aula, por exemplo:
  - pegar o dia da semana;
  - o ano;
  - a data no formato de string
  - utilize a documentação para mais ideias (https://www.w3schools.com/js/js_date_methods.asp)
- Realizar o push dessa branch com o comando *git push -u origin <nome da banch>*
- Abrir uma pull request
- Cada integrante deve olhar a pull request do outro e criar comentários, se necessário
- O responsável deve então aprovar e fazer o merge de cada branch com a main/master.

Momento Aprendizagem em Pares

➔ ⚠️**Alerta Pesquisa**⚠️
- Responda a "**Pesquisa de Avaliação Quinzenal"** <https://forms.gle/jBPKKpBbbgE71s8i8>
- Obs: Sua resposta para essa pesquisa é muito importante para compreendermos como está sendo essa experiência para você!
- Responder apenas uma vez!

➔ Esse momento é dedicado para vocês desenvolverem suas demandas e entregas para o curso.

➔ Utilize esse tempo da maneira que preferir, mas atente-se às aulas que você deve realizar as entregas.
- **Dica:** Nas Propostas dos projetos, vocês encontram uma sugestão de organização para a realização das atividades.
- **Lembre-se:** O momento de Aprendizagem em Pares é justamente para fazer trocas e aprender em comunidade, aproveite seus colegas e se desenvolvam juntos!

➔ **Entregas:**
- Projeto individual: aula 5 - TECH
- Projeto em grupo: aula 11 - TECH
- Apresentação do projeto: aula 11 - TECH

Revisão da aula

➔ **Day.js** é uma ferramenta JavaScript direta para manipular, exibir e validar datas e horas em navegadores modernos. Day.js é uma das bibliotecas mais

recomendadas para **formatar data e hora em JavaScript**, pois pode ser usada na renderização do lado do cliente e do lado do servidor e funciona perfeitamente em ambos os cenários.

➔ Um fator importante da aula são as **Branches** que são recursos disponíveis na maioria dos sistemas de controle de versão modernos. Geralmente você usa as branches no Git no processo quase diário de desenvolvimento. Quando você deseja adicionar um novo recurso ou corrigir um bug - não importa quão grande ou pequeno - você gera uma nova ramificação para encapsular suas alterações. Isso torna mais difícil para o código instável ser mesclado na base de código principal e dá a você a chance de limpar o histórico do seu futuro antes de mesclá-lo na ramificação principal.

➔ **Merge** é uma prática comum para desenvolvedores que usam sistemas de controle de versão. Independentemente de as ramificações serem criadas para testes, correções de bugs ou outros motivos, o merge **confirma as alterações em outro local.** Para ser mais específico, **a mesclagem pega o conteúdo de uma ramificação de origem e os integra a uma ramificação de destino**. Nesse processo, apenas a ramificação de destino é alterada. O histórico da ramificação de origem permanece o mesmo.

➔ Você pode estar se perguntando em que **linguagem** você escreve o **compilador**? Pode ser qualquer idioma. O **trabalho do compilador é traduzir de seu código de alto nível para uma linguagem conhecida de baixo nível.** Não importa em qual linguagem o próprio compilador está escrito. Você escreve um programa em uma linguagem existente para ler e compilar a nova linguagem. Eventualmente, usando este método, você pode escrever um programa compilador na nova linguagem para a nova linguagem.

Para ajudar

Links interessantes:

- JavaScript Get Date Methods, w3 school <https://www.w3schools.com/js/js_date_methods.asp> Acesso em ago 2022
- Day.js <https://day.js.org/en/> Acesso em ago. 2022
- Importando o Day.js no html <https://day.js.org/docs/en/installation/browser> Acesso em ago. 2022

- Comandos Git mais utilizados e como configurar, geekhunter <https://blog.geekhunter.com.br/comandos-git-mais-utilizados/> Acesso em ago. 2022
- Fluxo do GitHub, Github <https://docs.github.com/pt/get-started/quickstart/github-flow> Acesso em ago. 2022
- 3.2 Branches no Git - O básico de Ramificação (Branch) e Mesclagem (Merge), Git <https://git-scm.com/book/pt-br/v2/Branches-no-Git-O-b%C3%A1sico-de-Ramifica%C3%A7%C3%A3o-Branch-e-Mesclagem-Merge> Acesso em ago. 2022
- Git Branching e merge: Guia passo a passo, Varonis <https://www.varonis.com/pt-br/blog/git-branching> Acesso em ago. 2022