Atividade 1: Coleta de dados do histórico de navegação

Tempo estimado: 30 min

> Nota: esta atividade requer o navegador Chrome na versão 77 (ou posterior) e que a as configurações de idioma estão configuradas para "*English (United States)*".

> Nota: esta atividade assume que você dispõe do **código inicial** que deverá ser ajustado conforme os passos descritos a seguir. O código inicial está disponível neste repositório: https://github.com/andreplima/webmedia2019mc2

Nesta atividade, você vai:

Criar uma extensão que coleta o histórico de navegação que fica registrado no navegador. O histórico corresponde ao conjunto de endereços que o usuário digitou no omnibox ou o conjunto de endereços acessados por meio de links.

Ao concluir esta atividade, você terá:

- Carregado uma extensão não empacotada.
- Corrigido erros na extensão usando os recursos do navegador.
- Empregado a API chome.history para recuperar o histórico de navegação do usuário.
- Configurado a extensão para operar no navegador em modo incógnito.

Instruções

Carregando a extensão e corrigindo erros no manifesto e na UI

1. Configure o navegador para ativar o modo de desenvolvimento.
   Na barra de endereço (omnibox), digite "chrome://extensions"
   Na barra de controle (topo, em azul), ative o controle "Developer mode".

2. Carregue o código inicial da extensão.
   Clique no botão "Load unpacked", na barra de controle.
   No repositório, selecione a pasta ./webmedia2019mc2/Atividade1/inicial.
   Resultado: uma mensagem de erro é apresentada: "Failed to load the extension ... Could not load manifest".

3. Localize e corrija o erro na especificação da extensão.
   A falha ocorreu porque o valor do campo "manifest_version" no arquivo **manifest.json** *está errado. Consulte a documentação de extensões, corrija o erro e carregue novamente a extensão.*
   Resultado: a extensão é carregada com sucesso e seu ícone aparece na barra de controle (quadrado em cinza com o dígito "1" inscrito).

4. Execute a extensão, analise e corrija o erro de execução da extensão.
   Ao executar a extensão, a página popup com resultados aparece vazia. Note que, na página de gestão de extensões, o painel que representa esta extensão aponta um erro na execução. Você pode consultar as mensagens de erro clicando no botão "Errors". Neste caso, você vai encontrar a mensagem: "TypeError: Cannot read property 'appendChild' of null". Este erro ocorreu porque o nome do elemento que é passado à função buildUrlList, implementada no script ***urls.js*** *está errado. Corrija e atualize a extensão.*
   Resultado: ao executar novamente a extensão, a página popup ainda aparece vazia, mas nenhum erro de execução é registrado.

Ajustando chamadas à API `chrome.history`

5. Consulte o script **urls.js** e responda: qual tipo de histórico de navegação está sendo recuperado?
   Dica: analise a função de call-back passada para a chamada ao método chrome.history.getVisits(...). Consulte a documentação da API para determinar qual o tipo de histórico está sendo recuperado e veja quais tipos podem ser recuperados.

6. Ajuste as chamadas à API `chrome.history` para recuperar o histórico de navegação que corresponde aos links em que o usuário clicou. Ajuste também a janela de tempo da busca para incluir o histórico de navegação ocorrido nos últimos dez minutos. Atualize a extensão usando o botão de "*refresh" no painel da própria extensão*.

7. Faça uma busca pelo site do WebMedia 2019.
   Abra uma nova tab no navegador e busque por "webmedia 2019".
   Nos resultados da busca, clique no link que corresponde ao site do evento.
   Acione a extensão e confira que a visita ao site foi recuperada (assim como a busca iniciada pelo usuário).

8. Ajuste a chamada à API para recuperar o histórico de navegação em que o usuário digitou o endereço no omnibox e recarregue a extensão. Acione a extensão. O resultado é diferente?

9. Acesse o site do WebMedia 2018.
   Digite o endereço do site no omnibox: https://webmedia.org.br/2018/.
   Acione a extensão e confira se a visita ao site foi recuperada, bem como a visita ao site do WebMedia 2019 realizada no passo 7. Note que a busca iniciada pelo usuário no passo 7 (links) não é apresentada.

10. Ajuste novamente a chamada à API `chrome.history` para recuperar o histórico de navegação que corresponde aos links em que o usuário clicou (como fizemos no passo 6). Atualize e acione a extensão.
    Resultado: a visita ao site do WebMedia 2018 não é recuperada.

Recuperando o histórico de navegação em modo incógnito

11. A extensão não está habilitada em modo incógnito.
    Abra uma outra instância do navegador em modo incógnito. Note que a extensão não aparece na barra de navegação.

12. Habilite a extensão para operar em modo incógnito.
    Na página de gestão da extensão, clique no botão "Details" e ative o controle "Allow in incognito".
    Abra uma outra instância do navegador, em modo incógnito, e acione a extensão. Confira se a visita ao site do WebMedia 2018 foi recuperada.

13. (Opcional) No navegador, em modo incógnito, acesse o site do WebMedia 2017.
    Na tab do navegador em modo incógnito, busque por "webmedia 2017". Nos resultados da busca, clique no link que corresponde ao site do evento. Acione a extensão e confira se a visita ao site foi recuperada.

14. (Opcional) Como você usaria os resultados dos passos 12 e 13 para convencer o colega ao seu lado de que o navegador não registra histórico de navegação quando opera em modo incógnito?

Atividade 2: Coleta de resultados de busca

Tempo estimado: 20 min

> Nota: esta atividade requer o navegador Chrome na versão 77 (ou posterior) e que a as configurações de idioma estão configuradas para "*English (United States)*". Assume também que a opção "Search engine used in the address bar" do navegador está configurada como "Google".

> Nota: esta atividade assume que você dispõe do **código inicial** que deverá ser ajustado conforme os passos descritos a seguir. O código inicial está disponível neste repositório: https://github.com/andreplima/webmedia2019mc2

Nesta atividade, você vai:

Criar uma extensão que recupera o conteúdo de uma busca pelo *Google web search engine*. A extensão abre uma nova tab no navegador, faz uma busca pela *string* "o efeito do consumo de café", parseia os resultados e os apresenta em uma janela popup.

Ao concluir esta atividade, você terá:

- Corrigido erros na extensão usando os recursos do navegador.
- Empregado a API chrome.tabs para abrir uma nova tab e realizar uma busca usando o Google web search engine.
- Empregado componentes do tipo background e content script.
- Empregado as APIs chrome.runtime e chrome.tabs para trocar mensagens entre componentes da extensão.

Instruções

Carregando a extensão e corrigindo erros no script de background

1. Configure o navegador para ativar o modo de desenvolvimento.
   Na barra de endereço (omnibox), digite "chrome://extensions"
   Na barra de controle (topo, em azul), ative o controle "Developer mode".

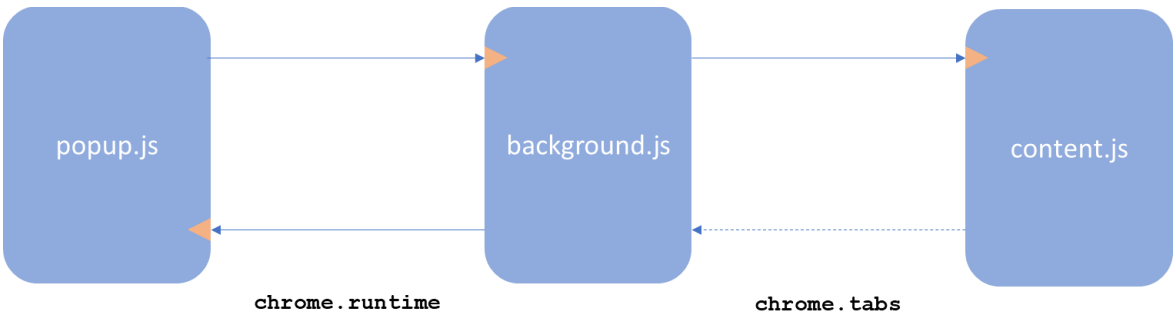
2. Carregue o código inicial da extensão.

Clique no botão "Load unpacked", na barra de controle.
No repositório, selecione a pasta ./webmedia2019mc2/Atividade2/inicial.

3. Acione a extensão. Localize e corrija o erro.
   Ao executar a extensão, a janela popup aparece vazia e o botão "Errors" aparece no painel da extensão. Na página de extensões, o painel que representa a extensão contém uma mensagem de alerta: "Uncaught FakeError", na página auto-gerada para o script de background. Comente a linha que provoca esse erro e atualize a extensão.

Analisando o padrão de troca de mensagens entre componentes da extensão

4. Acione a extensão. Localize e corrija os erros.
   Resultado: o botão "Errors" não é apresentado, mas a janela popup aparece vazia. Esse problema ocorre porque há erros nas chamadas às APIs de troca de mensagem entre componentes da extensão. Corrija o problema e atualize a extensão.
   Dica: o diagrama a seguir ilustra as trocas de mensagens entre componentes desta extensão. Os triângulos em amarelo representam "listeners" e as setas representam chamadas à APIs de envio de mensagens. Identifique esses elementos no código.



5. Acione a extensão. Analise os resultados.

Resultado: a janela popup apresenta os resultados da busca. Considerando apenas os títulos dos documentos recuperados, responda:

a) O colega ao seu lado observou os mesmos resultados?

b) Quantos resultados sugerem efeitos positivos, negativos ou neutros?

c) Se você executasse a extensão em um navegador em modo logado, você acha que os resultados seriam os mesmos?

Explorando variações da extensão

6. No passo 4, você viu que o script de *background* envia uma mensagem para o script de conteúdo, passando uma função de *callback* para receber a resposta, ao passo que o script **popup.js** usa um outro padrão de troca de mensagens. Qual é a diferença?

7. (Opcional) No script **popup.js**, remova o *listener* de mensagem e adapte a chamada à API `chrome.runtime` para passar uma função de *callback*.
   Antes de iniciar essa modificação, salve uma cópia do código da extensão.

8. (Opcional) Substitua a autorização "tabs" por "activeTab" no arquivo **manifest.json**. Atualize e acione a extensão.
   Por que a extensão funcionou se a nova tab é aberta, mas não é configurada como a tab ativa?

9. (Opcional) Adapte o script **content.js** para coletar os endereços dos documentos recuperados, ao invés do título dos documentos.

Atividade 3: Coleta da identificação do usuário participante

Tempo estimado: 30 min

> Nota: esta atividade requer o navegador Chrome na versão 77 (ou posterior) e que a as configurações de idioma estão configuradas para "*English (United States)*".

> Nota: esta atividade assume que você dispõe do **código inicial** que deverá ser ajustado conforme os passos descritos a seguir. O código inicial está disponível neste repositório: https://github.com/andreplima/webmedia2019mc2

Nesta atividade, você vai:

Criar uma extensão que recupera diferentes formas de identificação do usuário participante: se o usuário estiver logado no navegador (*signed in*), é possível recuperar seu e-mail e seu ID da conta no Google. Alternativamente, durante o esforço de recrutamento, você pode enviar junto com o convite para participar da pesquisa um "token" de participante. Neste método, o usuário registra seu token na extensão como parte das tarefas designadas para ele.

Ao concluir esta atividade, você terá:

- Empregado a API **`chrome.identity`** para recuperar dados da identificação do usuário participante.
- Empregado a API **`chrome.storage`** para salvar dados na instância local do navegador.
- Empregado o componente "*options page*" para permitir a configuração da extensão pelo usuário.

Instruções

Construindo componentes da extensão

1. Complemente a solução inicial para implementar as funcionalidades apresentadas no vídeo de demonstração.
   No repositório, selecione a pasta ./webmedia2019mc2/Atividade3/inicial.

*Note que os arquivos **options.js** e **popup.js** estão vazios. Você precisa implementar as funcionalidades esperadas para que a extensão opere como apresentado no vídeo de demonstração.*
Dica: explore os exemplos de código na página do Get Started e das APIs declaradas no arquivo de manifesto.

2. Com o navegador em modo logado, configure o valor do token de usuário para "user1" na página de opções da extensão. Acione a extensão.
   Ao executar a extensão, verifique que os três identificadores foram recuperados com sucesso.

3. Com o navegador em modo não logado, acione a extensão.
   Ao executar a extensão, verifique que nenhum dos identificadores foi recuperado. Por que o token do usuário não foi recuperado? Configure o valor do token de usuário para "user2" e acione a extensão.

4. Com o navegador em modo logado, acione novamente a extensão.
   Qual o valor do token de usuário foi recuperado – user1 ou user2? Com o colega ao lado, justifique.

5. (Opcional) Modifique o popup para apresentar e permitir a atualização do token do usuário.
   Dica: você pode reusar partes do código da extensão publicada como parte do seguinte artigo:

   Ronald E. Robertson, David Lazer, and Christo Wilson. 2018. Auditing the Personalization and Composition of Politically-Related Search Engine Results Pages. In Proceedings of the 2018 World Wide Web Conference (WWW '18). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 955-965. DOI: https://doi.org/10.1145/3178876.3186143

Atividade 4: Envio de dados coletados para repositório no GitHub

Tempo estimado: 30 min

> Nota: esta atividade requer o navegador Chrome na versão 77 (ou posterior) e que a as configurações de idioma estão configuradas para "*English (United States)*".

> Nota: esta atividade assume que você dispõe do **código inicial** que deverá ser ajustado conforme os passos descritos a seguir. O código inicial está disponível neste repositório: https://github.com/andreplima/webmedia2019mc2

Nesta atividade, você vai:

Criar uma extensão que recupera o conteúdo de uma busca pelo *Google web search engine* e salva os resultados da busca em um repositório do GitHub, na forma de um arquivo JSON.

Ao concluir esta atividade, você terá:

- Empregado a API do `Github` para salvar os dados coletados em um repositório.
- Empregado a API `chrome.tabs` para abrir uma nova tab e realizar uma busca usando o Google *web search engine*.
- Empregado componentes do tipo *background* e *content script*.
- Empregado as APIs `chrome.runtime` e `chrome.tabs` para trocar mensagens entre componentes da extensão.

Instruções

Carregando a extensão e corrigindo erros de sintaxe

1. Complemente a solução inicial para implementar as funcionalidades apresentadas no vídeo de demonstração.
   *No repositório, selecione a pasta ./webmedia2019mc2/Atividade4/inicial. Note que o arquivo **background.js** está incompleto. Você precisa implementar as funcionalidades esperadas para que a extensão opere como na demo. O nome do arquivo com o conteúdo da busca deve seguir o padrão <user ID>_<timestamp>.json. O arquivo deve ser salvo no*

repositório "donateyourdata". As credenciais de acesso ao repositório serão informadas pelo instrutor.
Dica: explore o código que você desenvolveu da Atividade 2 e os snippets para acionamento da API Github do slide de introdução à essa Atividade. Opcionalmente, você pode ler o artigo "Upload files on Github using Github.js".

2. Com o navegador em modo logado, acione a extensão.
   Ao executar a extensão, verifique que o conteúdo da busca é apresentado na janela de popup e confere com o conteúdo do arquivo criado no repositório.

3. (Opcional) Modifique a extensão para coletar também o histórico recente de navegação do usuário participante.
   Dica: explore o código que você desenvolveu na Atividade 1.