// @jamescardona11

flutter - chrome extension

content

intro

why? when?

goal?

limitations? :/

- i'm: james cardona
- role: mobile developer
- company: phononx.com
- linkedIn/github/medium/dev.to: @jamescardona11
- web: jamescardona11.com

goal

basic setup

- manifest.json
- index.html
- run a chrome extension

communication

- background.js
- contentScript.js
- popup.hmtl
- events
- chrome-api (dart)

dart-interop

- what is it?
- how to use?
- communication

basic setup

```json
{
    "name": "flutter_chrome_extension_demo",
    "short_name": "flutter_chrome_extension_demo",
    "start_url": ".",
    "display": "standalone",
    "background_color": "#0175C2",
    "theme_color": "#0175C2",
    "description": "A new Flutter project.",
    "orientation": "portrait-primary",
    "prefer_related_applications": false,
    "content_security_policy": {
        "extension_pages": "script-src 'self' ; object-src 'self'"
    },
    "action": {
        "default_popup": "index.html",
        "default_icon": "/icons/Icon-192.png"
    },
    "manifest_version": 3
}
```

manifest.json

the blueprint of your extension

index.html

the launcher

run

```
flutter build web --web-renderer html --csp
```

basic setup

```html
<!DOCTYPE html>
<html style="height: 650px; width: 350px;">

<head>
  <base href="$FLUTTER_BASE_HREF">
  <meta charset="UTF-8">
  <title>flutter_chrome_extension_demo</title>
  <link rel="manifest" href="manifest.json">
  <!-- This script adds the flutter initialization JS code -->
  <script src="flutter.js" defer></script>
</head>
<body>
  <script src="main.dart.js" type="application/javascript"></script>
</body>
</html>
```

manifest.json

the blueprint of your extension

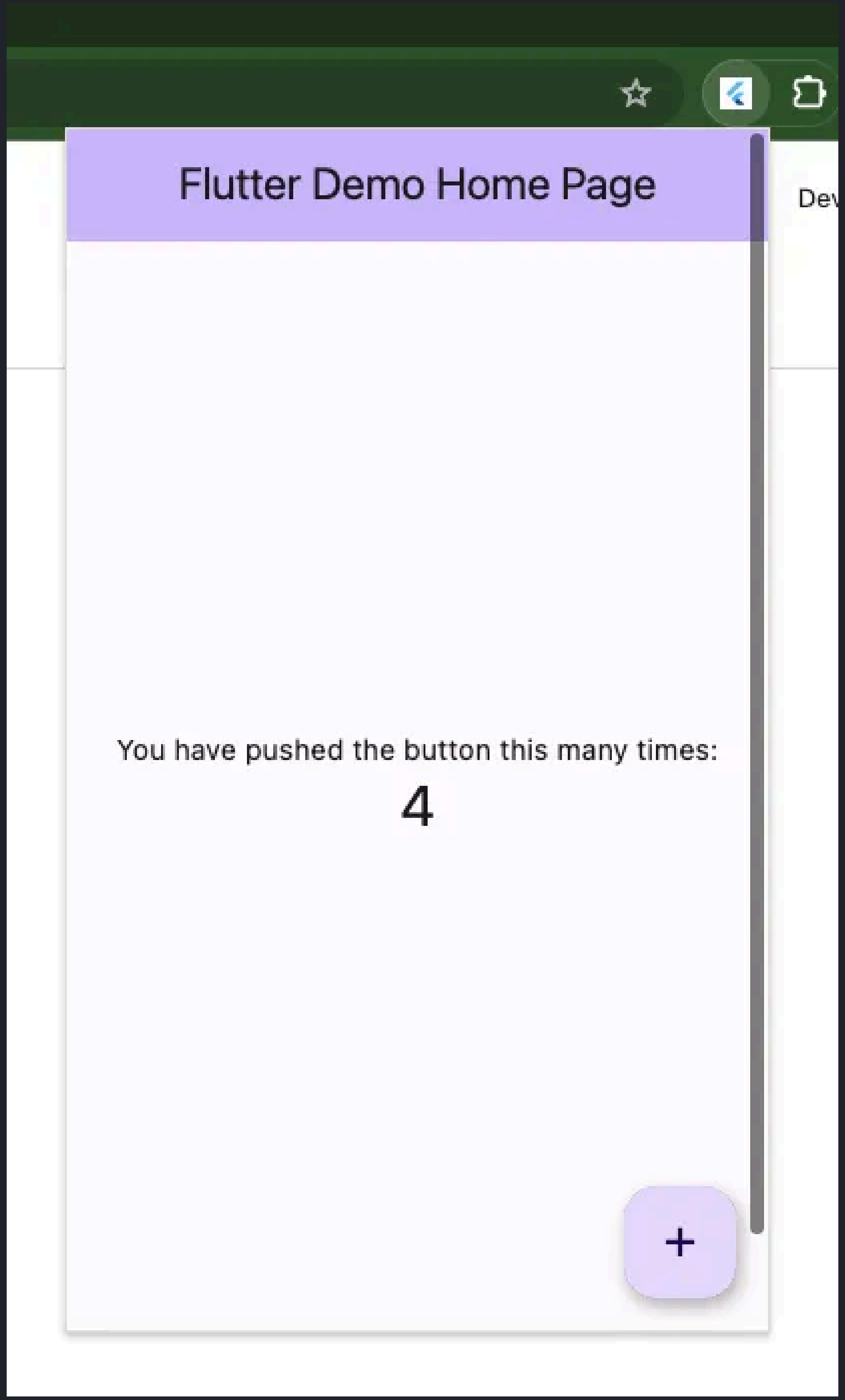
index.html

the launcher

run

```
flutter build web --web-renderer html --csp
```

basic setup


Flutter Demo Home Page

You have pushed the button this many times:

4

manifest.json

the blueprint of your extension

index.html

the launcher

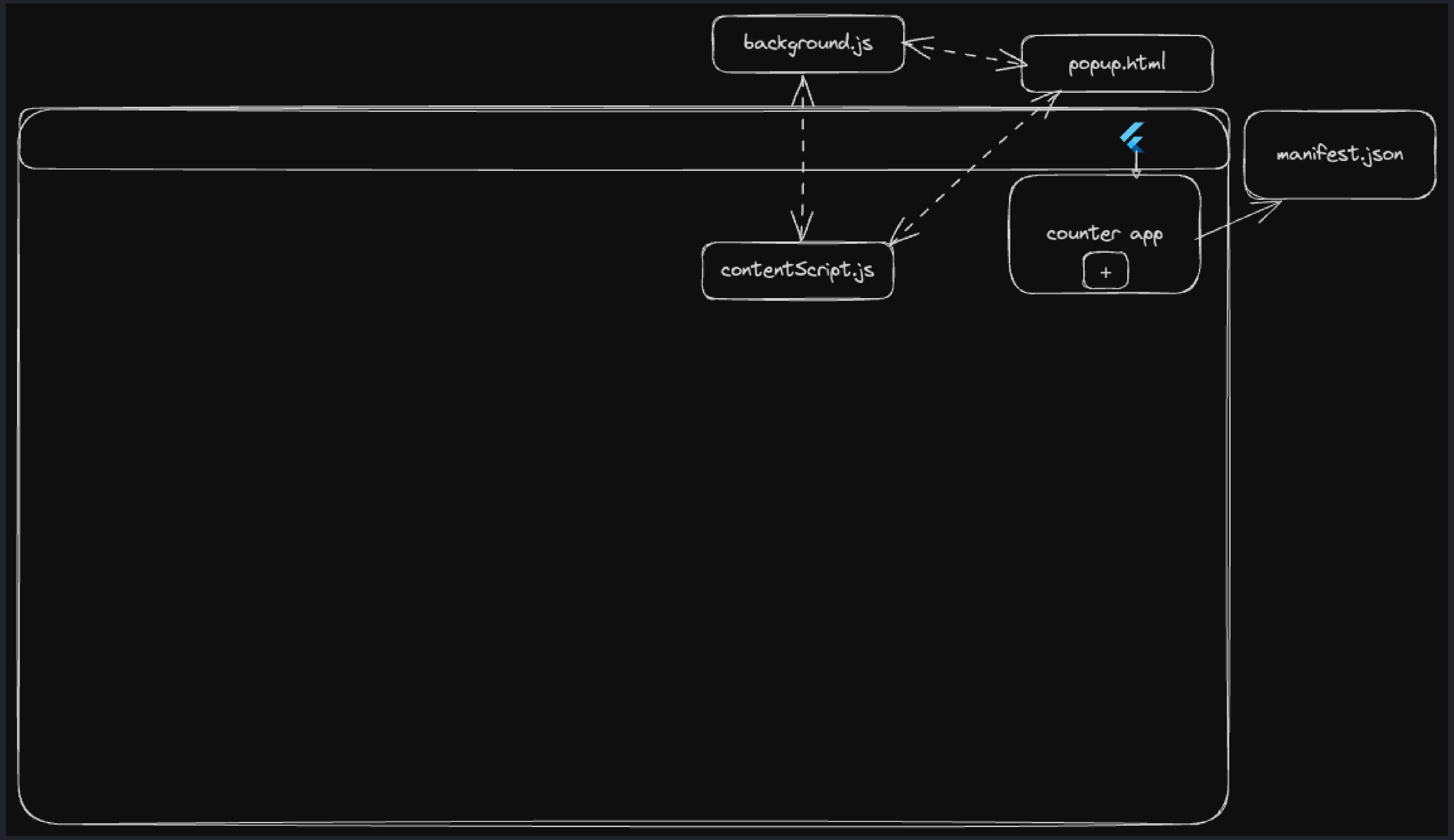
run

```
flutter build web --web-renderer html --csp
```

content

how to communicate?

a chrome extension consists of several parts that work together to provide the desired functionality.



background.js

??

contentscript.js

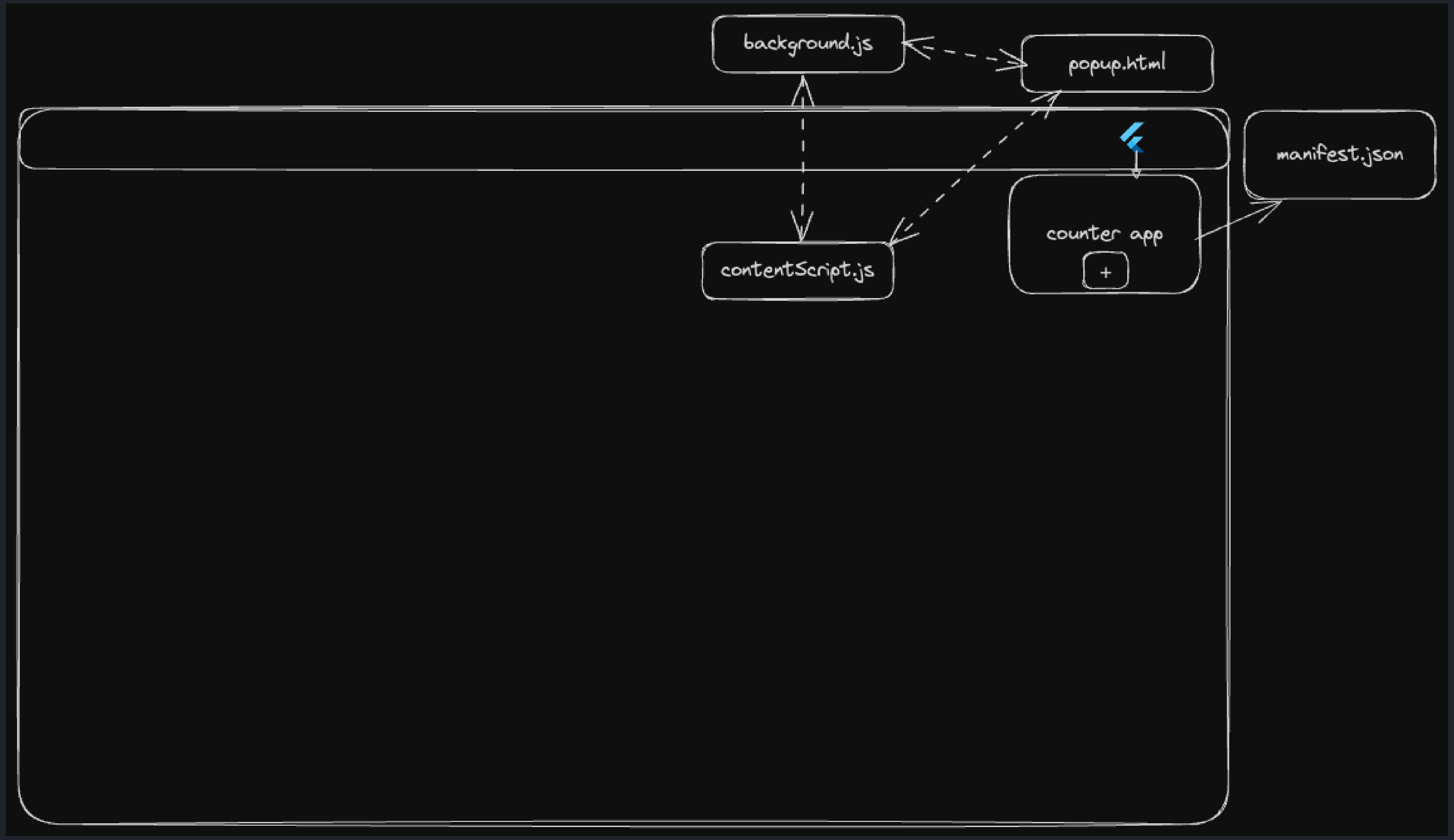
??

popup.html - flutter

??

how to communicate?

a chrome extension consists of several parts that work together to provide the desired functionality.



background.js

- always running, handles long-term tasks (listeners, API calls).
- limited access to a webpage (no direct interaction).
- communicates with contentScript for webpage actions.

contentscript.js

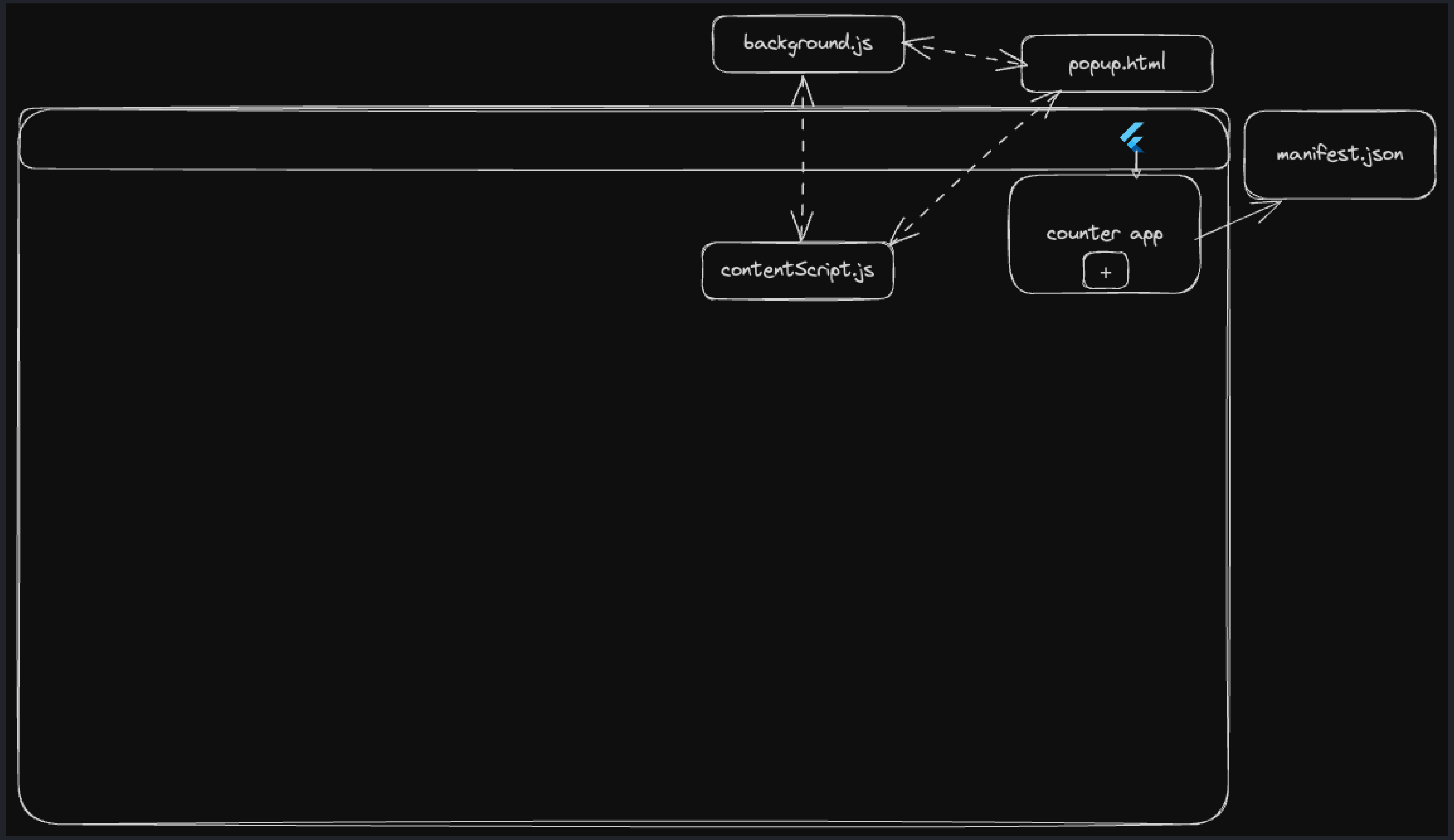
??

popup.html - flutter

??

how to communicate?

a chrome extension consists of several parts that work together to provide the desired functionality.



background.js

- always running, handles long-term tasks (listeners, API calls).
- limited access to a webpage (no direct interaction).
- communicates with contentScript for webpage actions.

contentscript.js

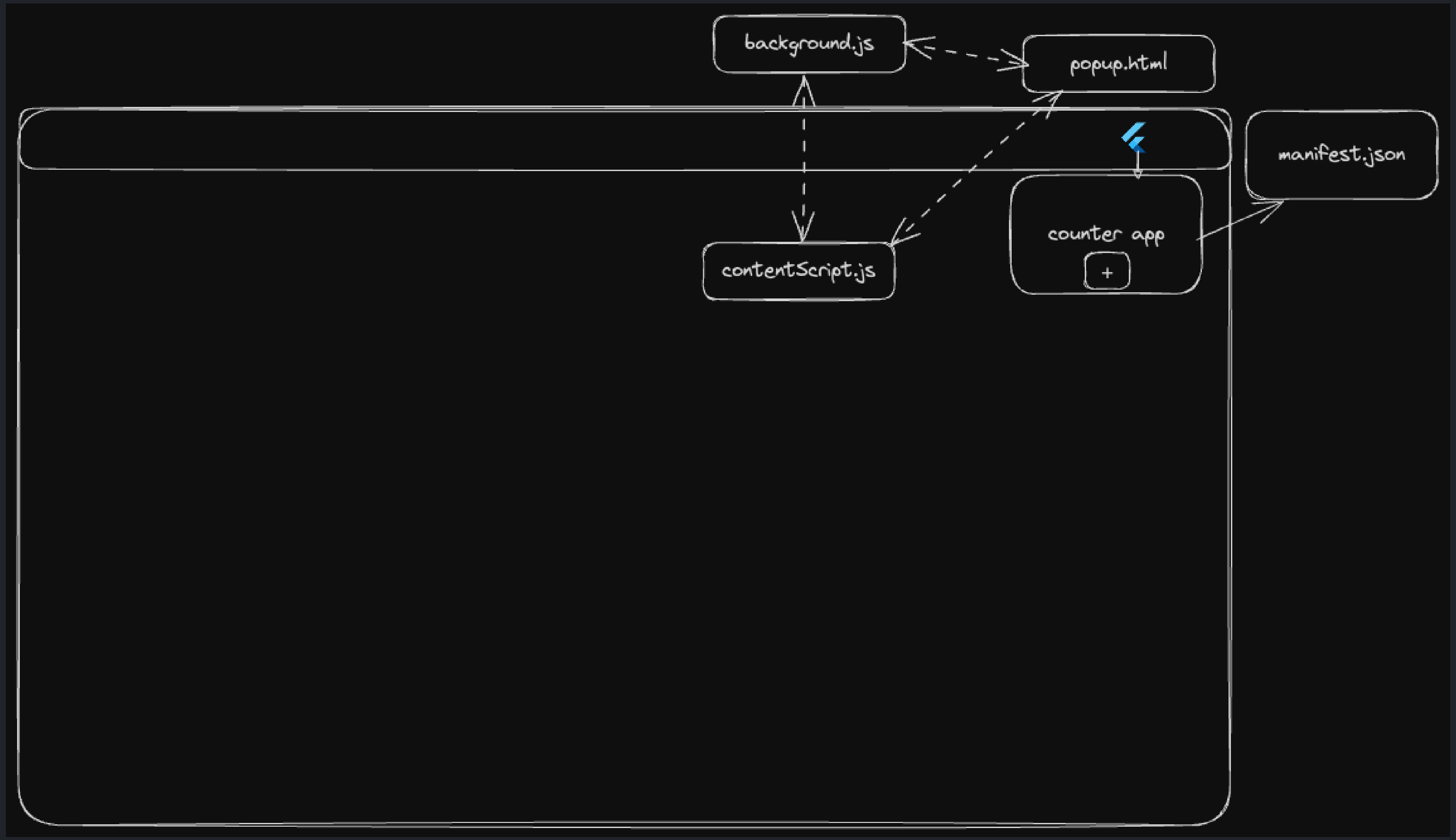
- injects into webpages.
- directly controls content (add, remove, modify).
- runs user-facing tweaks.

popup.html - flutter

??

how to communicate?

a chrome extension consists of several parts that work together to provide the desired functionality.



background.js

- always running, handles long-term tasks (listeners, API calls).
- limited access to a webpage (no direct interaction).
- communicates with contentScript for webpage actions.

contentscript.js

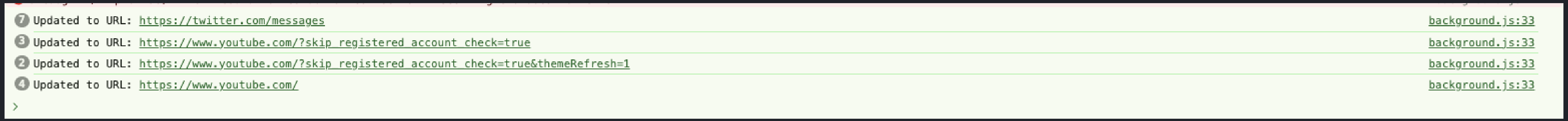
- injects into webpages.
- directly controls content (add, remove, modify).
- runs user-facing tweaks.

popup.html - flutter

- app ui
- user interaction

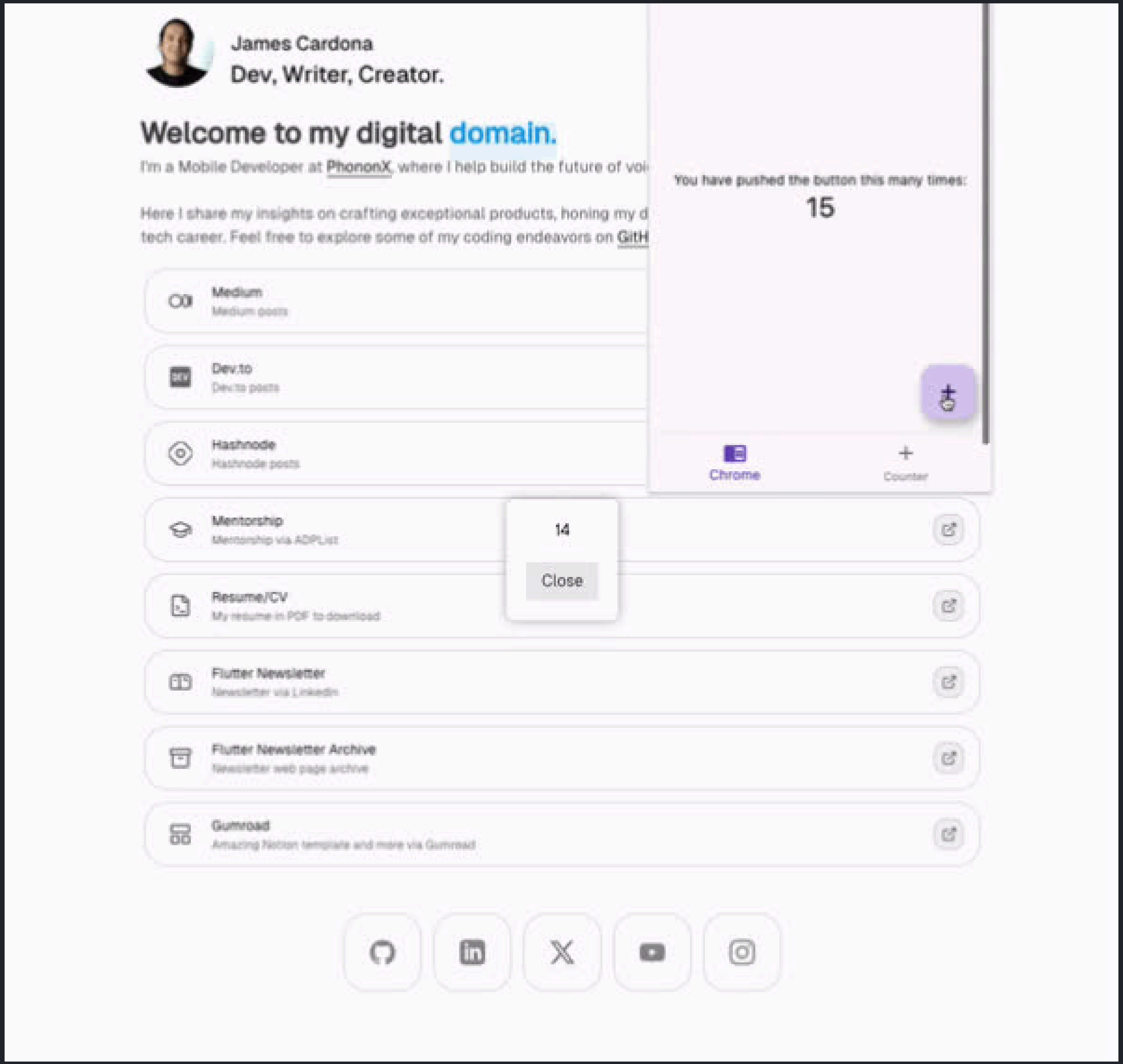
background listener

```
chrome.tabs.onUpdated.addListener(
  (tabId, changeInfo, tab) => {
    console.log('Updated to URL:', tab.url)
  }
)
```



```
7 Updated to URL: https://twitter.com/messages                                                background.js:33
3 Updated to URL: https://www.youtube.com/?skip_registered_account_check=true                 background.js:33
2 Updated to URL: https://www.youtube.com/?skip_registered_account_check=true&themeRefresh=1  background.js:33
4 Updated to URL: https://www.youtube.com/                                                    background.js:33
>
```

contentscript listener



James Cardona
Dev, Writer, Creator.

Welcome to my digital domain.

I'm a Mobile Developer at PhononX, where I help build the future of voi

Here I share my insights on crafting exceptional products, honing my d
tech career. Feel free to explore some of my coding endeavors on GitH

Medium
Medium posts

Dev.to
Dev.to posts

Hashnode
Hashnode posts

Mentorship
Mentorship via ADPList

Resume/CV
My resume in PDF to download

Flutter Newsletter
Newsletter via LinkedIn

Flutter Newsletter Archive
Newsletter web page archive

Gumroad
Amazing Notion template and more via Gumroad

You have pushed the button this many times:
15

14

js dart package

Use this package when you want to call JavaScript APIs from Dart code, or vice versa.

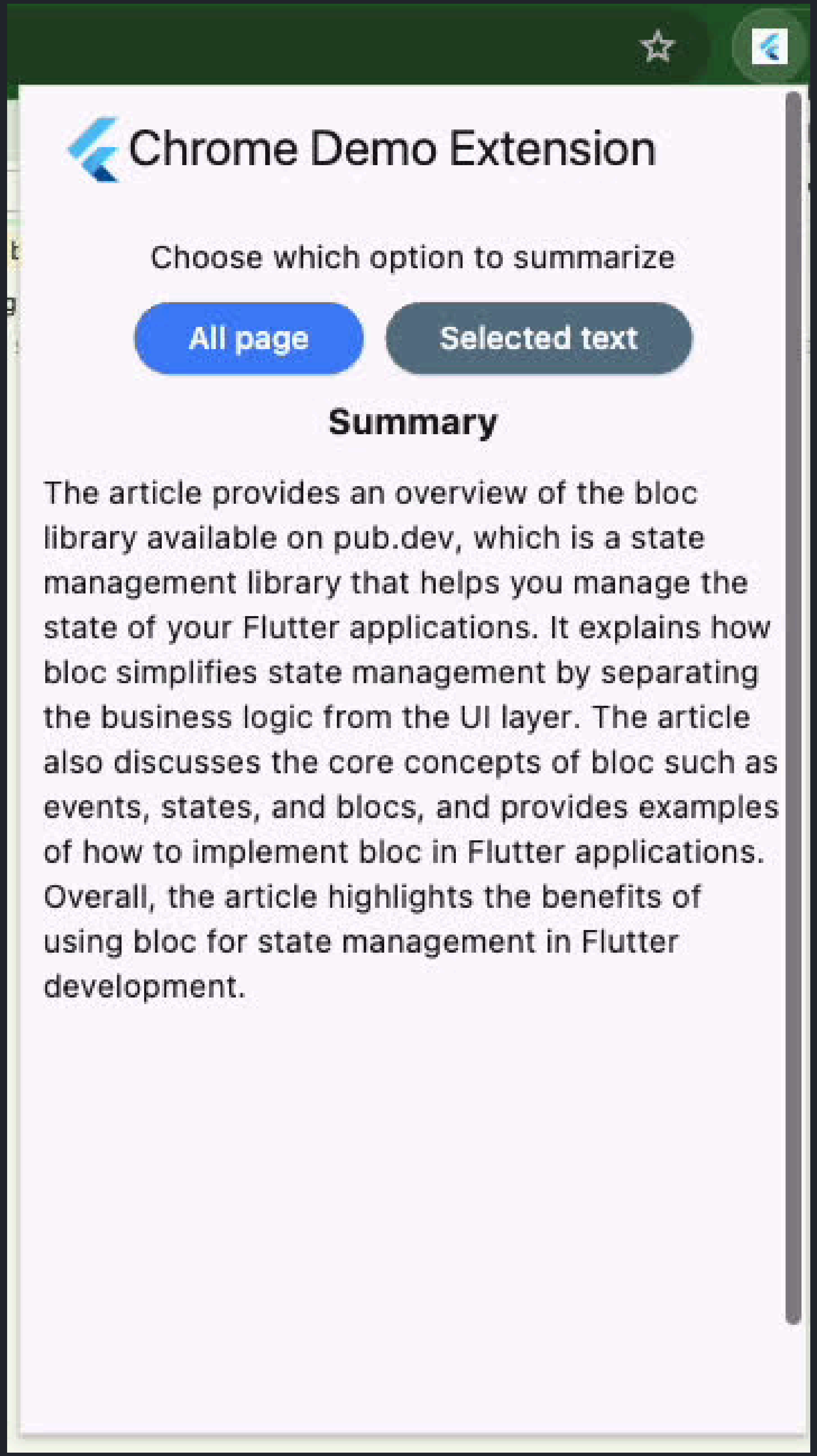
```dart
@JS('chrome')
library main; // library name can be whatever you want

import 'package:js/js.dart';
@JS('runtime.sendMessage')

external sendMessage(ParameterSendMessage parameterSendMessage);
@JS()
@anonymous
class ParameterSendMessage {
  external String get type;
  external String get data;
  external factory ParameterSendMessage({String type, String data});
}
```

contentscript listener

```
chrome.runtime.onMessage.addListener(function (message, sender, sendResponse) {
  if (message.type == "notifications") {
    create_popup(message.data);
  }
});
```

```
function create_popup(message) {
  // content for the popup
  var popupContainer = document.createElement("div");
  popupContainer.style.cssText = "position: fixed; top: 50%; left: 50%; transform: translate(-50%,
-50%); background-color: #ffffff; box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1); padding: 20px;
border-radius: 8px; text-align: center; z-index: 1000;";

  // create the message element
  var popupMessage = document.createElement("div");
  popupMessage.textContent = message;
  popupMessage.style.marginBottom = "20px";

  // create the close button
  var closeButton = document.createElement("button");
  closeButton.textContent = "Close";
  closeButton.style.backgroundColor = "#eeeeee";
  closeButton.style.color = "#333333";
  closeButton.style.border = "none";
  closeButton.style.padding = "8px 16px";
  closeButton.style.borderRadius = "4px";
  closeButton.style.cursor = "pointer";
  closeButton.onclick = function () {
    popupContainer.remove();
  };

  // add elements to the popup container
  popupContainer.appendChild(popupMessage);
  popupContainer.appendChild(closeButton);

  // add the popup container to the body
  document.body.appendChild(popupContainer);

  setTimeout(function () {
    popupContainer.remove();
  }, 3000);
}
```

demo #1



Summary

The article provides an overview of the bloc library available on pub.dev, which is a state management library that helps you manage the state of your Flutter applications. It explains how bloc simplifies state management by separating the business logic from the UI layer. The article also discusses the core concepts of bloc such as events, states, and blocs, and provides examples of how to implement bloc in Flutter applications. Overall, the article highlights the benefits of using bloc for state management in Flutter development.

getPageURL

Create a function using `js` package to get the currentPageURL

use chatgpt

create a call to chatGPT to get summary

demo #1

```dart
@JS('chrome')
library main; // library name can be whatever you want

import 'package:js/js.dart';

@JS('tabs.query')
external Future<List<Tab>> query(ParameterQueryTabs parameterQueryTabs);

@JS()
@anonymous
class Tab {
  external factory Tab({String url});

  external String get url;
}

@JS()
@anonymous
class ParameterQueryTabs {
  external factory ParameterQueryTabs({
    bool active,
    bool lastFocusedWindow,
  });

  external bool get active;

  external bool get lastFocusedWindow;
}
```

getPageURL

Create a function using `js` package to get the currentPageURL

```dart
Future<String> selectUrl() async {
    List tab = await promiseToFuture(
      query(ParameterQueryTabs(active: true, lastFocusedWindow: true)),
    );
    return tab[0].url;
  }
```

content

what is dart interop?

this library facilitates smooth interaction between JavaScript (JS) and Dart by providing a comprehensive JS interop solution.

index.js

create a new file to interop the functions

raw_interop.dart

create the definition of JS functions

js_interop.dart

implement the call and result

use toDart

method that is used to convert the JS type to a Dart type.

what is dart interop?

```
async function getPageUrl() {
  console.log("getPageUrl -- web/index.js");
  const tabs = await chrome.tabs.query({ 'active': true });
  console.log("Return from chrome.tabs.query", tabs[0].url);
  return tabs[0].url;
}
```

~~index.js~~

create a new file to interop the functions

raw_interop.dart

create the definition of JS functions

js_interop.dart

implement the call and result

use toDart

method that is used to convert the JS type to a Dart type.

what is dart interop?

```dart
@JS()
library flutter_medellin_extension;

import 'dart:js_interop';

@JS()
external JSPromise<JSString> getPageUrl();
```

~~index.js~~

create a new file to interop the functions

~~raw_interop.dart~~

create the definition of JS functions

js_interop.dart

implement the call and result

use toDart

method that is used to convert the JS type to a Dart type.

what is dart interop?

```dart
import 'dart:js_interop';

import 'raw_interop.dart' as interop;

abstract class JsInterop {
  static Future<String> getPageUrl() async {
    return (await interop.getPageUrl().toDart).toDart;
  }
}
```

~~index.js~~

create a new file to interop the functions

~~raw_interop.dart~~

create the definition of JS functions

~~js_interop.dart~~

implement the call and result

~~use toDart~~

method that is used to convert the JS type to a Dart type.

demo #2

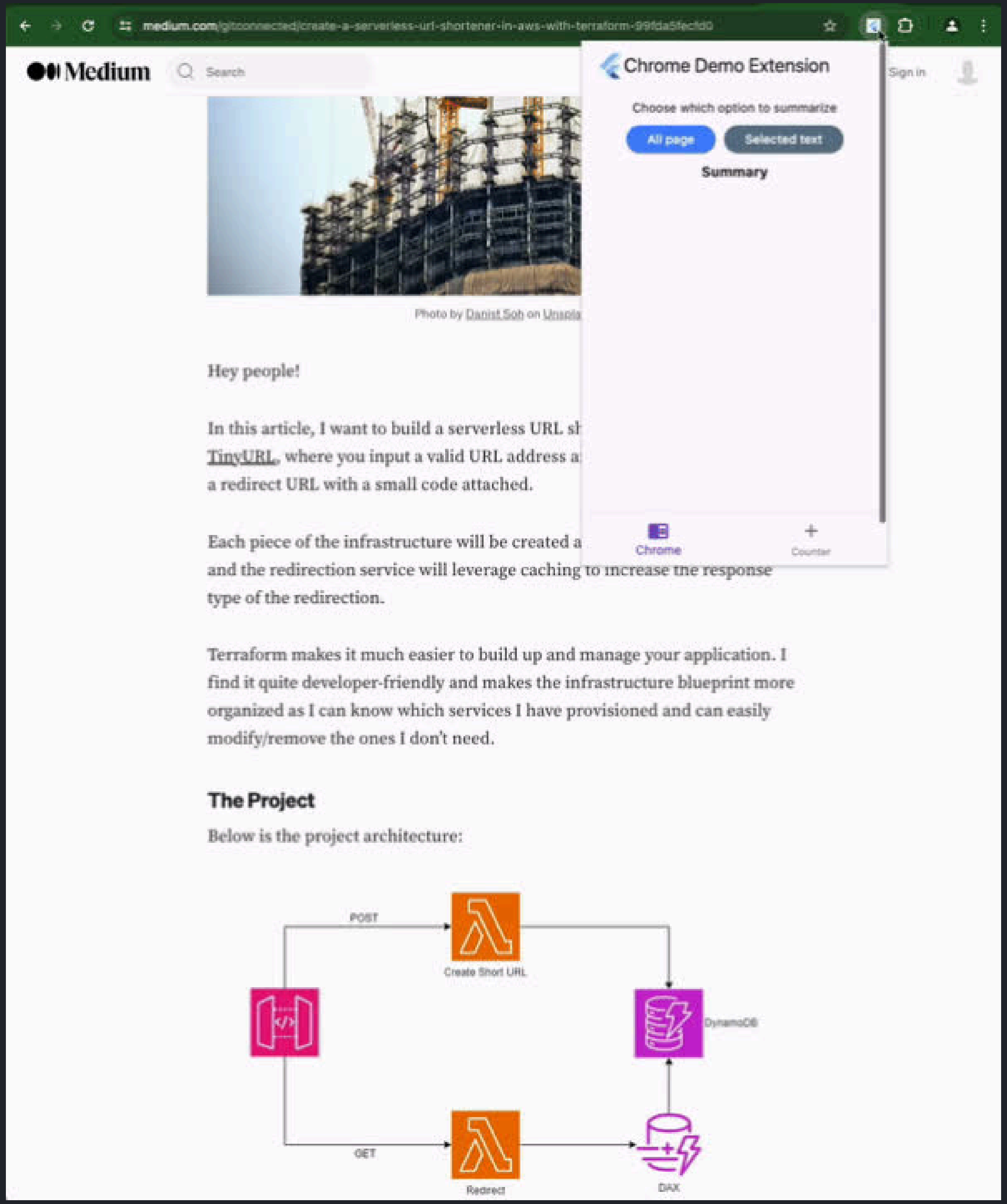

medium.com/gitconnected/create-a-serverless-url-shortener-in-aws-with-terraform-99fda5fecfd0
Medium
Search
Sign in
Chrome Demo Extension
Choose which option to summarize
All page
Selected text
Summary
Chrome
Counter
POST
Create Short URL
DynamoDB
GET
Redirect
DAX

Photo by Danist Soh on Unspla

Hey people!

In this article, I want to build a serverless URL sh
TinyURL, where you input a valid URL address a
a redirect URL with a small code attached.

Each piece of the infrastructure will be created a
and the redirection service will leverage caching to increase the response
type of the redirection.

Terraform makes it much easier to build up and manage your application. I
find it quite developer-friendly and makes the infrastructure blueprint more
organized as I can know which services I have provisioned and can easily
modify/remove the ones I don't need.

The Project

Below is the project architecture:

index.js

create a function to call the
selected text

background

receive a message from index.js

use promise

- we need to use a promise to wait
  for the result
- we can't use `await` on
  background

return information to Flutter

demo #2

```
async function getSelectedText() {
  console.log("selectedText -- web/index.js");

  const promise = new Promise(function (resolve, reject) {
    chrome.runtime.sendMessage({ type: "selectedText" }, function (response) {
      resolve(response);
    });

  })

  const selection = await promise;
  if (selection) {
    return selection[0].result ?? '';
  }
  return '';
}
```

~~index.js~~

create a function to call the selected text

background

receive a message from index.js

use promise

- we need to use a promise to wait for the result
- we can’t use `await` on background

return information to Flutter

demo #2

```
if (message.type === "selectedText") {
      const promise = new Promise(function (resolve, reject) {
        chrome.tabs.query({ active: true, currentWindow: true }, async function (tabs) {
          const tabId = tabs[0].id;
          const text = await chrome.scripting.executeScript({
            target: { tabId: tabId },
            function: () => getSelection().toString()
          });
          resolve(text);
        });
      })

      promise.then((response) => {
        sendResponse(response);
      });
      return true;
    }
```

~~index.js~~

create a function to call the selected text

~~background~~

~~receive a message from index.js~~

~~use promise~~

- we need to use a promise to wait for the result
- we can't use `await` on background

return information to Flutter

demo #2

```
String selectedText = await JsInterop.getSelectedText();
    print('Selected Text: $selectedText');

    setState(() {
      isLoading = true;
    });

    summary = await summaryApiClient.getTextSummary(selectedText) ?? 'Error fetching summary';

    setState(() {
      isLoading = false;
    });
```

~~index.js~~

create a function to call the selected text

~~background~~

~~receive a message from index.js~~

~~use promise~~

- we need to use a promise to wait for the result
- we can’t use `await` on background

~~return information to Flutter~~

// @jamescardona11

Gracias {}

- code/presentation:https://github.com/jamescardona11/isolates
- posts: https://medium.com/@jamescardona11
- linkedIn/github/medium/dev.to: @jamescardona11
- web: jamescardona11.com