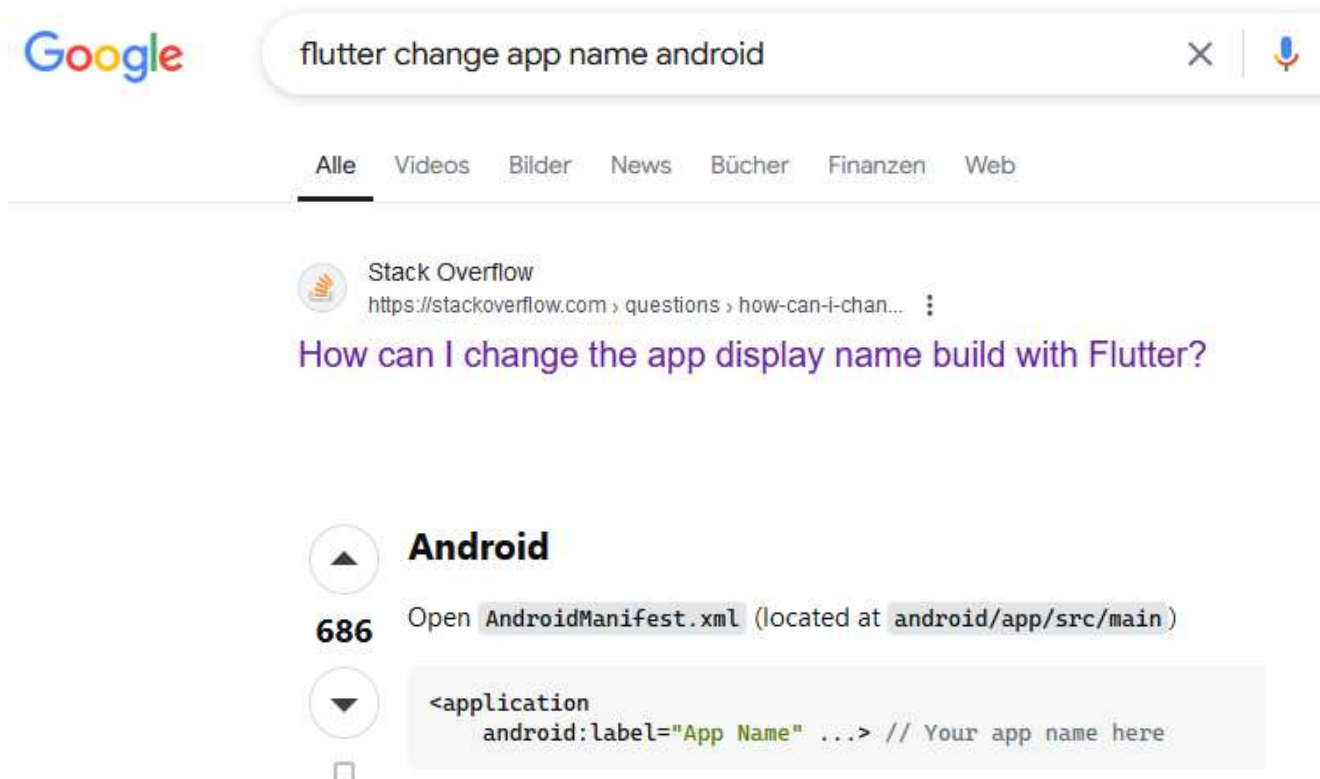# App customization and distribution

- Know how to change the app name shown in Android's home screen

- Know how to create and use an own launcher icon for your app

- Know how to build and distribute an APK file of your app

Change the display name of an app in Android

Ask Google:

Change the display name of an app in Android

Android Home Screen:

fdg_lux_meter → GS Lux Meter

Change the launcher icon of an app in Android

The "launcher icon" is shown in the home screen and when the app ist started.

The package flutter_launcher_icons on pub.dev

flutter_launcher_icons 0.13.1

Published 15 months ago · ⊘ fluttercommunity.dev (Dart 3 compatible)

SDK | DART | FLUTTER | PLATFORM | ANDROID | IOS | LINUX | MACOS | WINDOWS

Readme | Changelog | Example | Installing | Versions | Scores

Guide

1. Setup the config file

Add your Flutter Launcher Icons configuration to your `pubspec.yaml` or create a new config file called `flutter_launcher_icons.yaml`. An example is shown below. More complex examples can be found in the example projects.

```yaml
dev_dependencies:
    flutter_launcher_icons: "^0.13.1"

flutter_launcher_icons:
    android: "launcher_icon"
    ios: true
    image_path: "assets/icon/icon.png"
    min_sdk_android: 21 # android min sdk min:16, default 21
    web:
        generate: true
        image_path: "path/to/image.png"
        background_color: "#hexcode"
        theme_color: "#hexcode"
    windows:
        generate: true
        image_path: "path/to/image.png"
        icon_size: 48 # min:48, max:256, default: 48
    macos:
        generate: true
        image_path: "path/to/image.png"
```
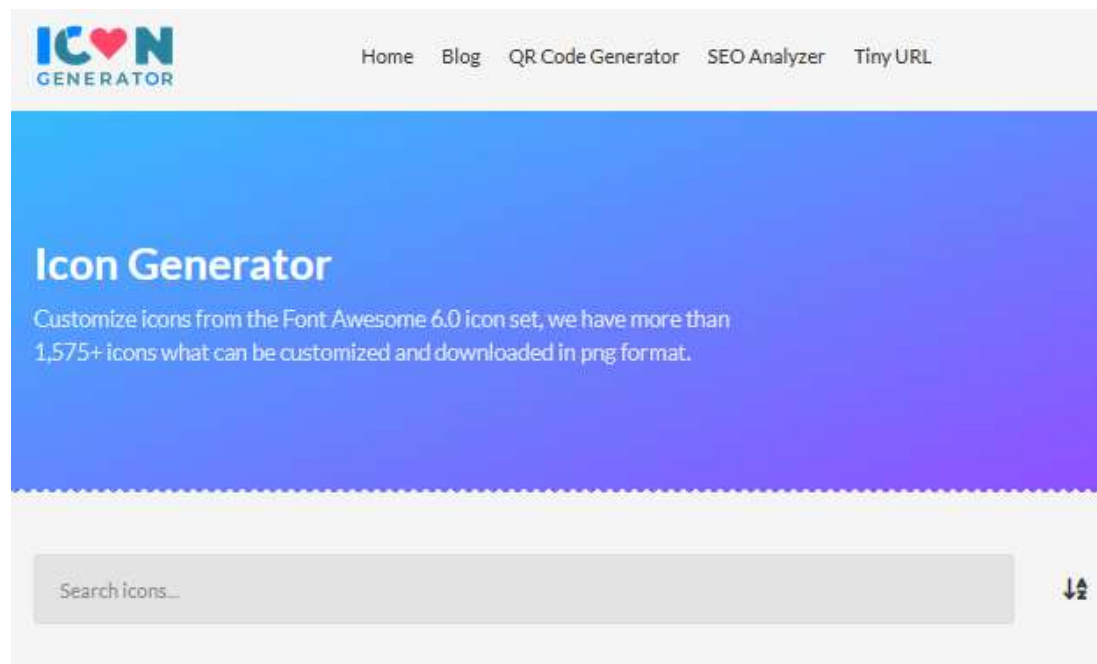
Before we copy this into our pubspec.yaml, let's first create our own icon file.

Create your own icon

There are several web sites where you can create icons.

One of them is https://www.icongenerator.net/ :

Create you own icon

Search for "light" and select the icon:
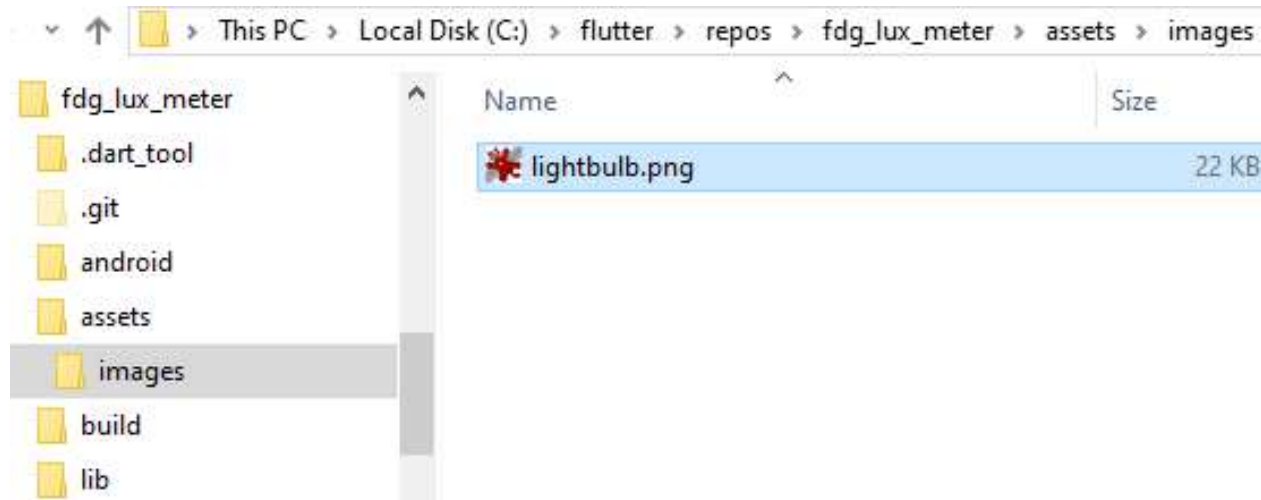
Adapt it as you want:

Store the icon file in your project

Download the icon file using the button
[⬇ Download Icon]

Store it in your project in the new created subfolder "assets/images" or "assets/icons"

Adapt your pubspec.yaml (step 1)

As described in Slide 5, copy the following lines from
https://pub.dev/packages/flutter_launcher_icons :

```
dev_dependencies:
  flutter_launcher_icons: "^0.13.1"

flutter_launcher_icons:
  android: "launcher_icon"
  ios: true
  image_path: "assets/icon/icon.png"
  min_sdk_android: 21 # android min sdk min:16, default 21
  web:
    generate: true
    image_path: "path/to/image.png"
    background_color: "#hexcode"
    theme_color: "#hexcode"
  windows:
    generate: true
    image_path: "path/to/image.png"
    icon_size: 48 # min:48, max:256, default: 48
  macos:
    generate: true
    image_path: "path/to/image.png"
```

and paste them into your pubspec.yaml file
under the section "dev_dependencies:"

```
dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^3.0.0

flutter:
  uses-material-design: true
```

Take care to also copy the 2 blanks at the beginning of the first line,
because blanks are important in yaml files !

Adapt your pubspec.yaml (step 2)

With "ios: true"
I got errors later

```
dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^3.0.0
  flutter_launcher_icons: "^0.13.1"

flutter_launcher_icons:
  android: "launcher_icon"
  ios: false
  image_path: "assets/images/lightbulb.png"
  min_sdk_android: 21 # android min sdk min:16, default 21
  # web:
  #   generate: true
  #   image_path: "path/to/image.png"
  #   background_color: "#hexcode"
  #   theme_color: "#hexcode"
```

We will work on
the web icon later.

Finally save your pubspec.yaml file.

Create the android icons in your project

Open a new Terminal in your VS Code and enter the command:

`dart run flutter_launcher_icons`

Modified files in your project

After the command is executed, you see the following modified or new files in your project:
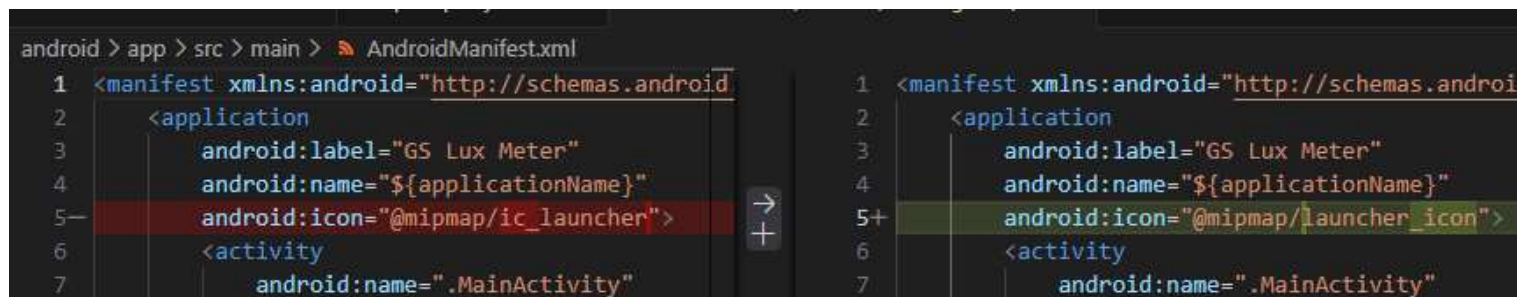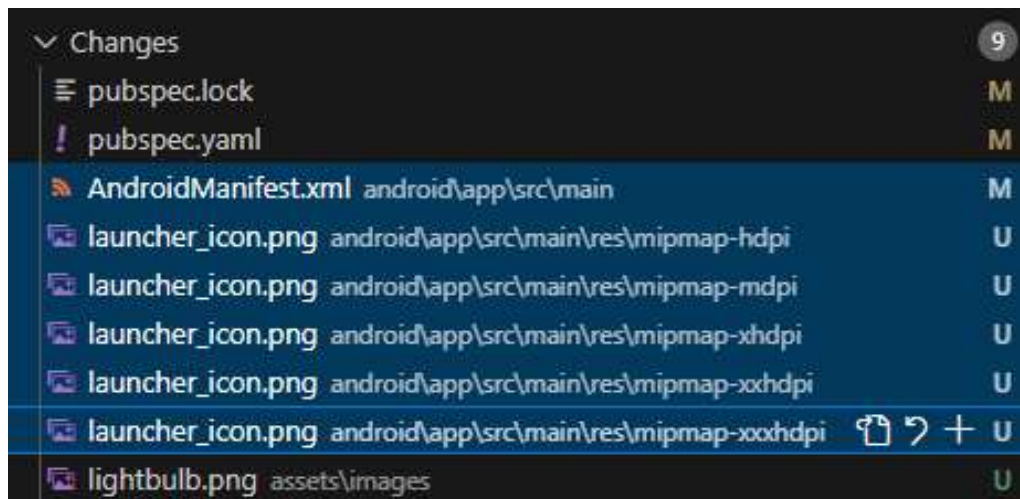
The generated icon files have different image sizes

```
v  📁 res                          Name                Size
   📁 drawable
   📁 drawable-v21                 🎴 ic_launcher.png      1 KB
   📁 mipmap-hdpi                  🎴 launcher_icon.png    3 KB
   📁 mipmap-mdpi


   📁 mipmap-mdpi                  Name                Size
   📁 mipmap-xhdpi
   📁 mipmap-xxhdpi                🎴 ic_launcher.png      2 KB
   📁 mipmap-xxxhdpi               🎴 launcher_icon.png    6 KB
```

Android: The app icon image should be provided in various sizes, including:

- mipmap-mdpi: 48x48 pixels
- mipmap-hdpi: 72x72 pixels
- mipmap-xhdpi: 96x96 pixels
- mipmap-xxhdpi: 144x144 pixels
- mipmap-xxhdpi: 192x192 pixels
- mipmap-xxxhdpi: 512x512 pixels

The text on the right was copied from
https://medium.com/@hpatilabhi10/changing-app-icon-in-flutter-a-step-by-step-guide-e2ba52c91e96

Different shapes on an Samsung A33 with Android 14

Lightbulb.png	Home screen	Program start

Square:

Circle:

Decagon:

GS Lux Meter

GS Lux Meter

GS Lux Meter

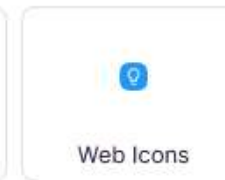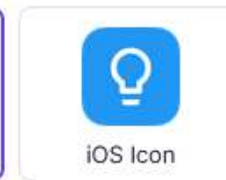# "Squircle" Icons can be created with IconKitchen

Lightbulb.png

Home screen

Program start

Link to IconKitchen: https://icon.kitchen/
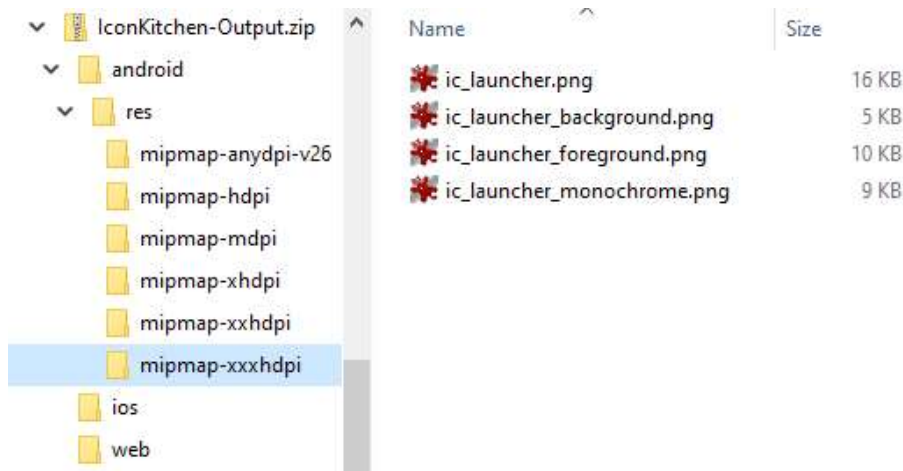
Download from IconKitchen

The download from IconKitchen is a zip file, not a single png file like from IconGenerator.

Take the png file with the highest resolution from folder "mipmap-xxxhdpi", copy it to the "assets\images" folder of your project, adapt the image name in pubspec.yaml and finally call `dart run flutter_launcher_icons` in Terminal .

Create a web icon for your app

1) Uncomment the first 3 lines of the "web-block" in your pubspec.yaml:

```
flutter_launcher_icons:
    android: "launcher_icon"
    ios: false
    image_path: "assets/images/lightbulb.png"
    min_sdk_android: 21 # android min sdk min:16, default 21
    web:
        generate: true
        image_path: "assets/images/lightbulb.png"
#       background_color: "#hexcode"
#       theme_color: "#hexcode"
# windows:
```

2) Save the pubspec.yaml file an run command `dart run flutter_launcher_icons` in Terminal.

Create a web icon for your app (continued)

3) Change the title of your MaterialApp in main.dart:
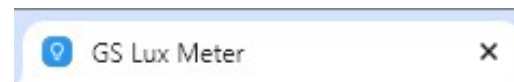
```
@override
Widget build(BuildContext context) {
  return MaterialApp(
    title: 'GS Lux Meter',
    theme: ThemeData(primarySwatch: Colors.blue),
    home: const MyHomePage(),
  ); // MaterialApp
}
```

4) Change the title tag in file web/index.html

```
<!-- Favicon -->
<link rel="icon" type="image/png" href="favicon.png"/>

<title>GS Lux Meter</title>
<link rel="manifest" href="manifest.json">
</head>
```
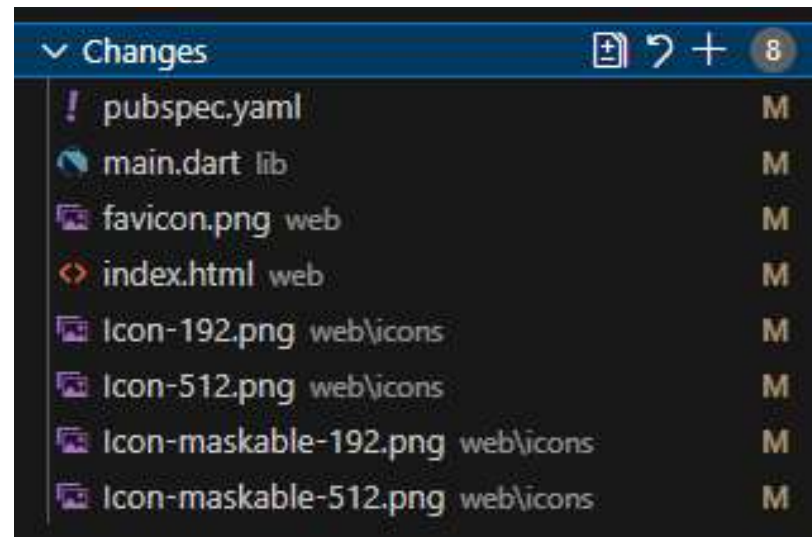
5) Debug your app on Chrome:

GS Lux Meter

Create a web icon for your app (changed files)

During the last steps, the following files have been modified in your project:

Create an APK file of your app (Android application PacKage)
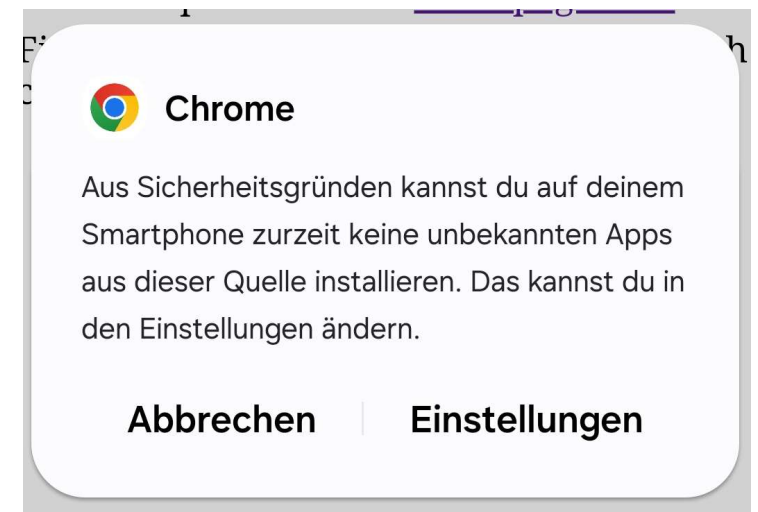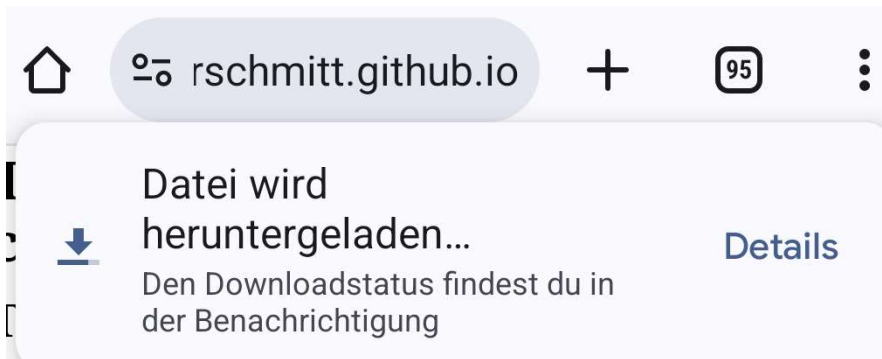
In VS Code Terminal, enter the command "`flutter build apk`"

Upload apk files to a GitHub Pages repository

1) Rename the file app_release.apk into whatever you want, e.g. fdg_lux_meter.apk

2) Create a repository on GitHub and activate Pages like we have done for our web repository where we host our web applications like flying_balls (for details see document
https://github.com/GuentherSchmitt/fdg_flutter_2023/blob/main/docs/09a%20How%20to%20host%20a%20flutter%20web%20app%20on%20GitHub%20pages.docx

3) Upload your apk file into this repo via "Drag&Drop" on the GitHub website of the repo and commit modification in GitHub.

4) Now everybody can download this apk file via the link
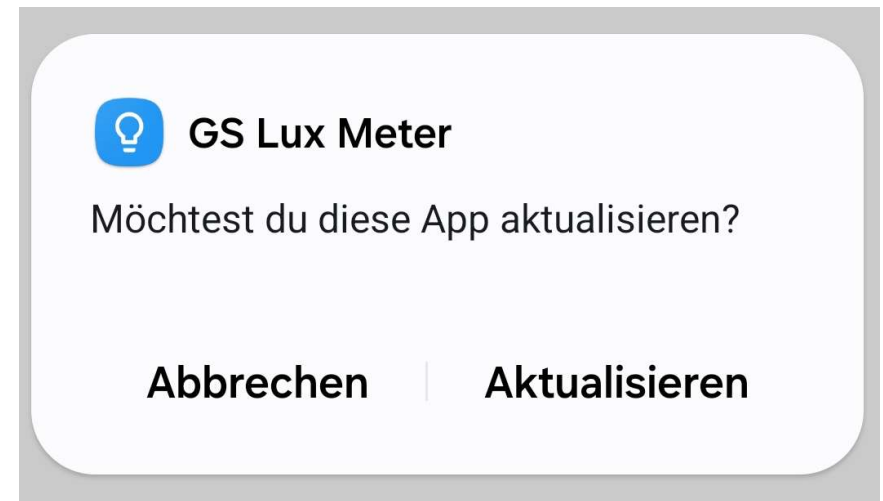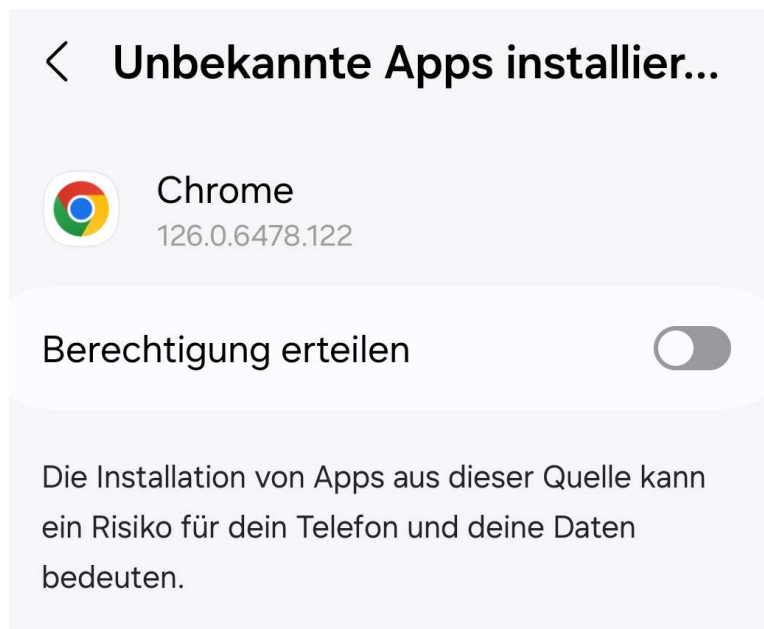https://fdg2324.github.io/apk/fdg_lux_meter.apk

Download and install the apk file

Enter the URL of the apk in Chrome in Android:

Download and install the apk file (continued)

Download and install the apk file (continued)

Google Play Protect

App-Scan empfohlen

> ⚪ GS Lux Meter

Play Protect kennt diese App noch nicht. Sende zum Schutz deines Geräts und deiner Daten einige Informationen zur App an Google, damit sie vor der Installation gescannt werden kann.

Weitere Details ⌄

App scannen

Nicht installieren

Google Play Protect

Diese App ist anscheinend sicher

> ⚪ GS Lux Meter

✓

Du kannst mit der Installation fortfahren

Installieren

Nicht installieren