Advanced topics with Webviews

Or when Widgets are not enough

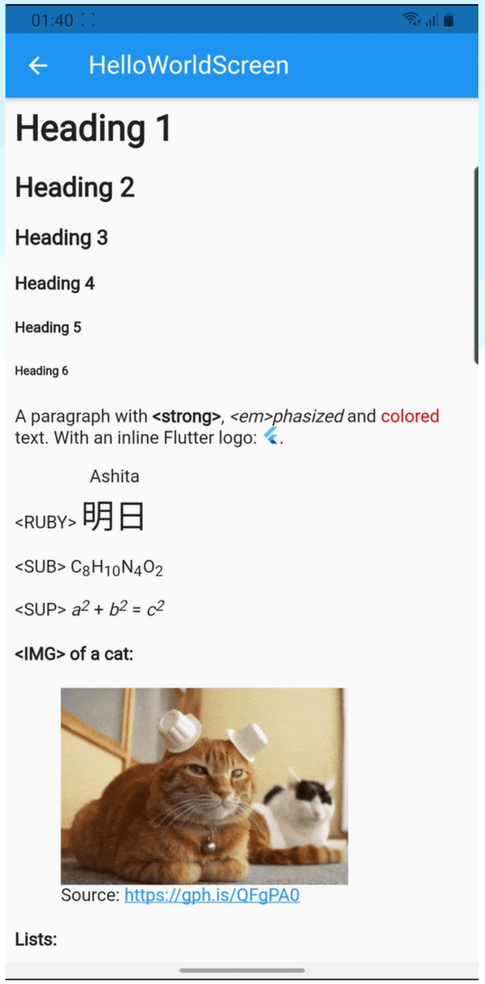
Guillaume Bernos - Flutter Connection 2023

Why use Webviews?

You can use Widgets to display HTML

flutter_widget_from_html

- You can transform HTML into Widgets
- Can be used when you want a deep integration with your app
- You control the HTML and can restrict the tags
- Basic layout



Webviews libraries overview

webview_flutter
flutter_inappwebview

webview_flutter

- Official package by the Flutter team
- Non-endorsed support for the Web
- Support preloading the page

flutter_inappwebview

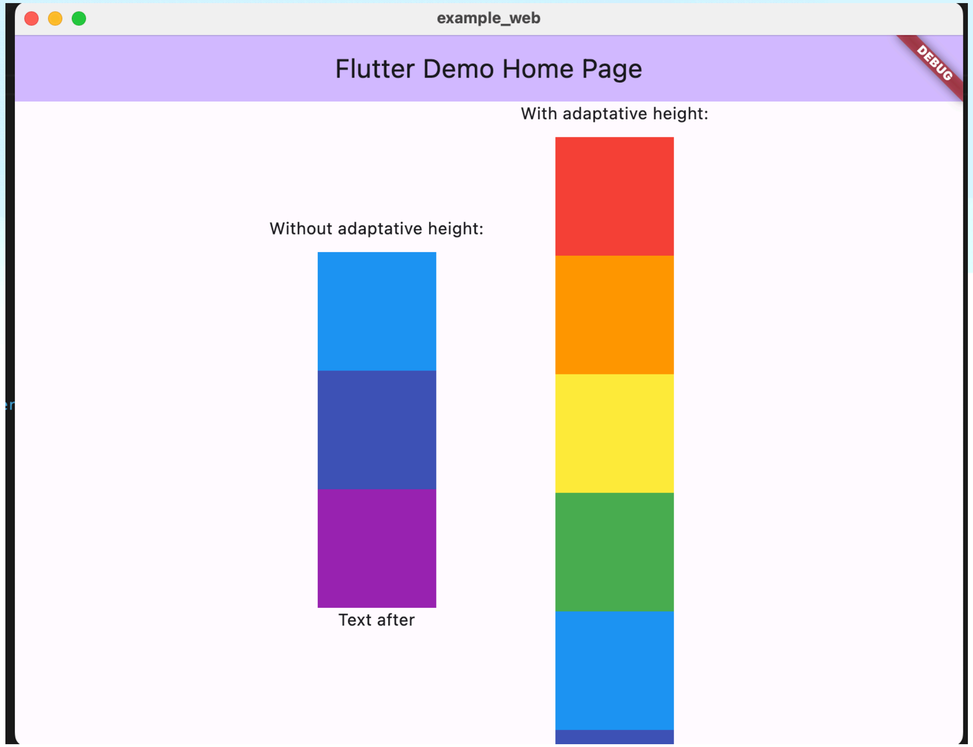
- Unofficial package
- Supports Web and macOS
- Supports preloading the page
- Supports WebRTC, Service Worker, Localhost server …

Examples

Dynamically set the height of a Webview

The problem

- You need to be able to set the height depending on the content
- Without setting the height of your PlatformView you cannot display it



Dynamically set the height of a Webview

Set up the page

- Create a base layout for all your HTML content
- Can be reused and customised with functions

```
<!DOCTYPE html>
  <html>
  <head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <style>
    .sharemail-content {
      min-height: ${minHeight ?? 0}px;
      min-width: ${minWidth ?? 0}px;
      overflow: auto;
    }
    ${hideScrollBar ? '''
      .sharemail-content::-webkit-scrollbar {
        display: none;
      }
      .sharemail-content {
        -ms-overflow-style: none;  /* IE and Edge */
        scrollbar-width: none;  /* Firefox */
      }
    ''' : ''}
    ${styleCSS ?? ''}
  </style>
  ${javaScripts ?? ''}
  </head>
  <body>
  <div class="sharemail-content">$content</div>
  </body>
  </html>
```

Dynamically set the height of a Webview

Get the page height

- When using flutter_inappwebview we get a controller in the `onLoadStop`
- We use this controller to run JavaScript directly in the page

```
final scrollHeight = await controller.evaluateJavascript(
  source: 'document.body.offsetHeight',
 );

 if (mounted && _webViewHeight != scrollHeight) {
  setState(() {
    _webViewHeight = scrollHeight;
  });
}
```

Dynamically set the height of a Webview

Set the height

- You can then update the Widget size accordingly
- If your HTML contains images, you should run this function regularly.

```
final scrollHeight = await controller.evaluateJavascript(
  source: 'document.body.offsetHeight',
 );

 if (mounted && _webViewHeight != scrollHeight) {
  setState(() {
    _webViewHeight = scrollHeight;
  });
}
```

Dynamically set the width of a Webview

- Width can also impact the user’s experience
- Here we work on the scale of the page to adapt it
- If you control the HTML content, you should have it fit properly any width

```
final scrollWidth = await controller.evaluateJavascript(
    source: 'document.body.offsetWidth');
if (mounted) {
  final size = MediaQuery.of(context).size;
  if (scrollWidth != null) {
    final scale = size.width / scrollWidth;
    if (scale < 1) {
      await controller.evaluateJavascript(
        source: '''document.body.style.scale = $scale;
            document.body.style.transformOrigin = "top left";''',
      );
    }
  }
}
```

Run code in the Webview

Set up a Javascript callback

- Can be injected in the script part of our setup page
- You can use window.flutter_inappwebview to get a callback in Flutter

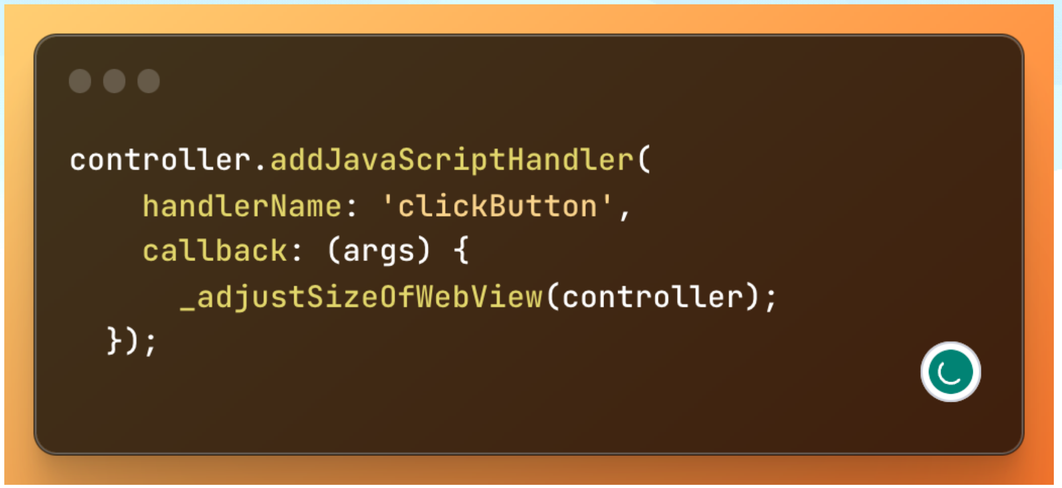
```
<script>
    var nextVisibility = "block";
    var quotes = document.getElementsByClassName("gmail_quote");
    for (var i = 0; i < quotes.length; i++) {
        quotes[i].style.display = "none";
    }

    document.getElementById("collapse_button").addEventListener("click", function(){
      var quotes = document.getElementsByClassName("gmail_quote");
      for (var i = 0; i < quotes.length; i++) {
          quotes[i].style.display = nextVisibility;
      }
      nextVisibility = nextVisibility == "block" ? "none" : "block";
      window.flutter_inappwebview.callHandler('clickButton');
    });
</script>
```

Run code in the Webview

Set up a Javascript callback

- Can be injected in the script part of our setup page
- You can use window.flutter_inappwebview to get a callback in Flutter



Webviews with Flutter Web

- Works well because it’s displayed as an Iframe
- If you place the Webview under Flutter widgets you might get issues interacting
- You need to use pointer_interceptor to get the tap properly handle

Optimise loading time of Webviews

- You can preload the Webview by constructing the controller early in webview_flutter
- You can also use AutomaticKeepAlive to keep a Webview ready if you are going to trigger it multiple times

```
controller = WebViewController()
  ..setJavaScriptMode(JavaScriptMode.unrestricted)
  ..setBackgroundColor(const Color(0x00000000))
  ..setNavigationDelegate(
    NavigationDelegate(
      onProgress: (int progress) {
        // Update loading bar.
      },
      onPageStarted: (String url) {},
      onPageFinished: (String url) {},
      onWebResourceError: (WebResourceError error) {},
      onNavigationRequest: (NavigationRequest request) {
        if (request.url.startsWith('https://www.youtube.com/')) {
          return NavigationDecision.prevent;
        }
        return NavigationDecision.navigate;
      },
    ),
  )
  ..loadRequest(Uri.parse('https://flutter.dev'));
```

Optimise loading time of Webviews

- When preloading a Webview or removing a webview from the Widget Tree we can get artifacts during transitions
- Using CupertinoPageTransitionsBuilder helps with smooth transitions

```
controller = WebViewController()
  ..setJavaScriptMode(JavaScriptMode.unrestricted)
  ..setBackgroundColor(const Color(0x00000000))
  ..setNavigationDelegate(
    NavigationDelegate(
      onProgress: (int progress) {
        // Update loading bar.
      },
      onPageStarted: (String url) {},
      onPageFinished: (String url) {},
      onWebResourceError: (WebResourceError error) {},
      onNavigationRequest: (NavigationRequest request) {
        if (request.url.startsWith('https://www.youtube.com/')) {
          return NavigationDecision.prevent;
        }
        return NavigationDecision.navigate;
      },
    ),
  )
  ..loadRequest(Uri.parse('https://flutter.dev'));
```

Inject fonts in the Webview

- Can be injected in the script part of the page template

```
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Inter">
<style>
    body {
        height: fit-content;
        width: fit-content;
        font-family: 'Inter', mono;
    }
</style>
```

Inject fonts in the Webview

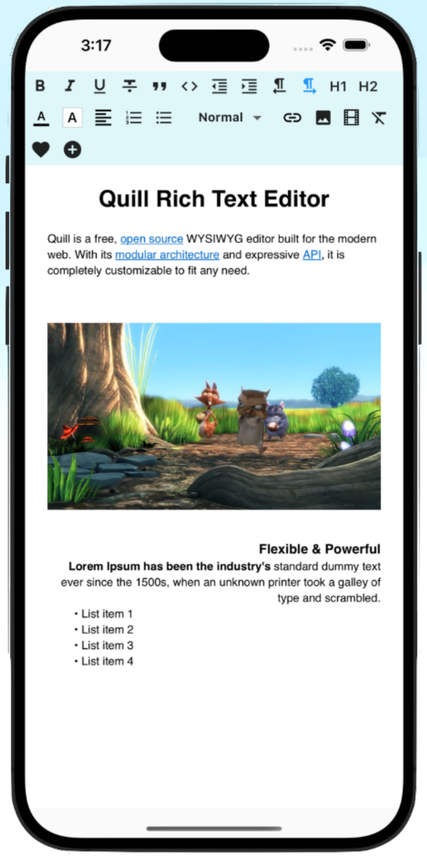
- Can be transformed into base64 to inject it into the Webview

```
@font-face {
    font-family: 'myfont';
    src: url(base64string);
}
<style>
    body {
        height: fit-content;
        width: fit-content;
        font-family: 'myfont', mono;
    }
</style>
```

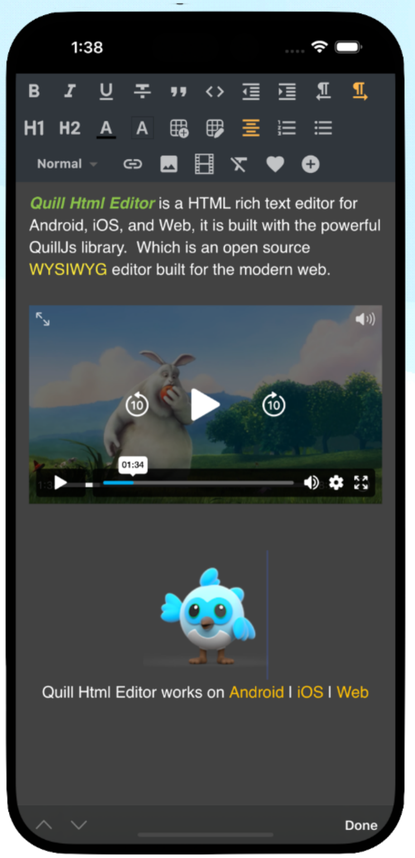
Editing a HTML file

Lot of librairies

- html_editor_enhanced
- quill_html_editor
- flutter_minimal_html_editor
- …



Light Mode



Dark Mode

Still needs improvements

- I used enough_html_editor
- Still needed a lot of modifications to fit my needs

Questions? 😄