Hello!

I am Steven Dzionara

@phntm.xyz

Flutter Engineer with a passion for UI/UX.

stevendz.de

Flutter 2.0

Stable milestone for **Web**

Agenda

- Pros and Cons
- Web Renderers
- Critical limitations
- Navigation

Using Flutter Web

Pros

- Only one code for multiple platforms
- Super performant and fast (once loaded)
- Almost no UI or animation limitations

Cons

- Not really SEO friendly
- Big initial loading size for canvas kit (8mb)
- Unfamiliar behaviour (e.g. Text and Scrolling)
- Weak debugging (e.g. no hot-reload)
- Kinda complicated routing

Renderers

Web Renderers

- **HTML (DOM)**: Uses a combination of HTML elements, CSS, and Canvas elements
- **CanvasKit**: This renderer is fully consistent with Flutter mobile and desktop, has faster performance with higher widget density, but adds about 2MB in download size

HTML Renderer

Advantages

- Has a smaller bundle size than CanvasKit and therefore, loads faster
- Uses native text rendering, allowing for use of system fonts in a Flutter application

Disadvantages

- Text fidelity
- Less performant than CanvasKit
- Tricky SVG support (custom icon fonts)
- Not all methods in the canvas api work properly.

CanvasKit Renderer

Advantages

- Blazing fast and extremely performant
- Accurate text measurement and layout
- Behaves pretty much the same as Flutter for mobile/desktop (All paint methods are supported and SVGs work as normal)

Disadvantages

- Does not use native text rendering (fonts have to be shipped)
- Expensive Emoji usage (additional 9mb)
- Larger bundle size (roughly 2mb larger than the HTML Renderer)
- CORS issues

When to use which one?

- **Data Usage**: The CanvasKit bundle size is kinda huge compared to the HTML renderer
- **Loading Time**: If the loading time is important, especially on mobile, go with HTML renderer
- **Precision**: For some crazy text layouts or if you just want to be 100% sure that the your widgets are painted correctly, consider CanvasKit
- **Performance**: If your app uses a lot of dynamic graphical content, CanvasKit is the way to go

(Selectable)Text

- Developing a layout system for text was one of the biggest challenges to support Flutter on the web
- Text can be selected, copied, and pasted by using **SelectableText** and **EditableText** widgets instead of the plain **Text** widget
- **Forms** with **Textfields** kinda support autofill but this functionality is still far from perfect (set **autofillHints** and **autovalidateMode**)

Screenreader

- Flutter's web semantic features are expanded to support accessibility for Windows, macOS, and ChromeOS
- A second DOM tree called the **SemanticsNode** tree is generated in parallel to the **RenderObject** DOM tree
- Support of Narrator, VoiceOver, TalkBack, or ChromeVox screen readers can be used to navigate a Flutter web app (with **ARIA** attributes and **Semantics** widget)

Scrollability

- Flutter web **experiences should feel right**, regardless of the shape and size of your browser window
- On mobile browsers, Flutter apps already have excellent support for gestures and scrolling physics
- Content on the desktop has to display scrollbars that can be controlled by a **mouse, trackpad or keyboard**
- Need of higher default **content density**, because a mouse click is obviously **more precise** than a touch with your finger

Navigation

Navigation is the new state management!

Navigator 1.0 vs. Navigator 2.0 (Router)

Navigator 1.0

- Easy to use
- Works great on Android and iOS
- Limited support for web navigation
- Imperative / Declarative

Navigator 2.0

- Fully supports web navigation with url handling
- Hard to use (without a package)
- Declarative

Imperative or Declarative?

Run | Debug

```
void main() {
  final array = [0, 1, 2, 3, 4, 5];
  print(imperative(array)); // [0, 2, 4]
  print(declerative(array)); // [0, 2, 4]
}

// Describe exaclty HOW to solve the problem
List<int> imperative(List<int> array) {
  final newArray = <int>[]; // create a new array
  for (var element in array) {
    // iterate over the old array
    if (element % 2 == 0) {
      // check if the element is an odd number
      newArray.add(element); // add the odd number to the new array
    }
  }
  return newArray; // return the new array
}

// Describe WHAT the output should be
List<int> declerative(List<int> array) {
  return array.where((element) => element % 2 == 0).toList(); // return a new array with just odd numbers
}
```

Navigator 1.0

Imperative:

```
onTap: () {
  Navigator.of(context).push(MaterialPageRoute(builder: (context) => const NewPage()));
},
```

Declarative:

```
onTap: () {
  Navigator.of(context).pushNamed('Name of the Route');
},
```

Back:

```
onTap: () {
  Navigator.of(context).pop();
},
```

Imperative:

```
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      title: 'Navigator 1.0 Demo',
      home: HomePage(),
    ); // MaterialApp
  }
}
```

Declarative:

```
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Navigator 1.0 Demo',
      initialRoute: '/',
      routes: {
        '/': (context) => const HomePage(),
        '/page1': (context) => const Page1(),
        '/page2': (context) => const Page2(),
        '/page3': (context) => const Page3(),
      },
    ); // MaterialApp
  }
}
```

navigator > lib > navigator_1.0 > routes.dart > ...

```
const String kListRoute = '/';
const String kDetailsRoute = '/details';

```

```
class BookDetailsScreen extends StatelessWidget {
  static const String name = '/details';
  const BookDetailsScreen({Key? key, required this.book}) : super(key: key);
  final Book book;
```

```dart
return MaterialApp(
  title: 'Navigator 1.0 Demo',
  onGenerateRoute: (settings) {
    if (settings.name == '/') {
      return MaterialPageRoute(builder: (context) => BooksListScreen(books: books));
    }
    final uri = Uri.parse(settings.name!);
    if (uri.pathSegments.length == 2 && uri.pathSegments.first == 'details') {
      // '/details/12'
      final id = int.tryParse(uri.pathSegments[1]);
      if (id == null) {
        return MaterialPageRoute(
          settings: settings,
          builder: (context) => const UnknownScreen(), // 404
        ); // MaterialPageRoute
      }
      return MaterialPageRoute(
        settings: settings,
        builder: (context) => BookDetailsScreen(book: books[id]),
      ); // MaterialPageRoute
    }
  },
```

Navigator 2.0

GoRouter

Still Navigator 2.0 but simplified :)

pubspec.yaml

```yaml
dependencies:
  flutter:
    sdk: flutter
  go_router: ^2.3.1
```

Declarative:

```
onTap: () {
  GoRouter.of(context).push('/details/12');
},
```

```
onTap: () {
  GoRouter.of(context).go('/details/12');
},
```

Back:

```
onTap: () {
  context.pop();
},
```

```dart
@override
Widget build(BuildContext context) {
  return MaterialApp.router(
    title: 'Navigator Beamer Demo',
    routeInformationParser: _router.routeInformationParser,
    routerDelegate: _router.routerDelegate,
  );
}
```

```dart
final _router = GoRouter(
  routes: [
    GoRoute( // GoRoute ⋯
    GoRoute( // GoRoute ⋯
  ],
  errorPageBuilder: (BuildContext context, GoRouterState state) {⋯
); // GoRouter
```

Simple route:

```
final _router = GoRouter(
  routes: [
    GoRoute(
      path: BooksListScreen.name,
      pageBuilder: (context, state) {
        return MaterialPage<void>(
          key: state.pageKey,
          child: BooksListScreen(books: books),
        ); // MaterialPage
      },
    ), // GoRoute
    GoRoute( // GoRoute …
  ],
  errorPageBuilder: (BuildContext context, GoRouterState state) {…
); // GoRouter
```

Not existing URL:

```
final _router = GoRouter(
  routes: [
    GoRoute( // GoRoute …
    GoRoute( // GoRoute …
  ],
  errorPageBuilder: (BuildContext context, GoRouterState state) {
    return MaterialPage<void>(
      key: state.pageKey,
      child: const UnknownScreen(), // 404
    ); // MaterialPage
  },
); // GoRouter
```

Route with parameter(id):

```
final _router = GoRouter(
  routes: [
    GoRouter( // GoRoute ⋯
    GoRouter(
      path: '/details' + '/:bookId',
      pageBuilder: (context, state) {
        final id = int.tryParse(state.params['bookId']!);
        return MaterialPage<void>(
          key: state.pageKey,
          child: BookDetailsScreen(id: id),
        ); // MaterialPage
      },
    ), // GoRoute
  ],
  errorPageBuilder: (BuildContext context, GoRouterState state) {⋯
); // GoRouter
```

```
class BookDetailsScreen extends StatelessWidget {
  const BookDetailsScreen({Key? key, required this.id}) : super(key: key);
  final int? id;

  @override
  Widget build(BuildContext context) {
    if (id != null && id! < books.length && id! >= 0) {
      return Scaffold(
        body: Center(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.center,
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              SelectableText(books[id!].id.toString(), style: Theme.of(context).textTheme.subtitle1),
              SelectableText(books[id!].title, style: Theme.of(context).textTheme.headline6),
              SelectableText(books[id!].author, style: Theme.of(context).textTheme.subtitle1),
            ],
          ), // Column
        ), // Center
      ); // Scaffold
    } else {
      return const Text('Id is not valid');
    }
  }
}
```

Get rid of the

http://localhost:60510/#/details/1

pubspec.yaml

```yaml
dependencies:
  flutter:
    sdk: flutter
  go_router: ^2.3.1
```

pubspec.yaml

```yaml
dependencies:
  flutter_web_plugins:
    sdk: flutter
  go_router: ^2.3.1
```

navigator > lib > main.dart > ...

```dart
import 'package:flutter/material.dart';
import 'navigator_1.0/my_app.dart';
import 'package:flutter_web_plugins/flutter_web_plugins.dart';

Run | Debug | Profile
void main() {
  setUrlStrategy(PathUrlStrategy());
  runApp(const MyApp());
}

```

```
Launching lib/main.dart on macOS in debug mode...                                   lib/main.dart:1
--- xcodebuild: WARNING: Using the first of multiple matching destinations:
{ platform:macOS, arch:x86_64, id:70123539-824A-5788-AB1D-982298065119 }
{ platform:macOS, name:Any Mac }
: Error: Not found: 'dart:html'                                navigation/js_url_strategy.dart:13/…/..
import 'dart:html' as html;
       ^
: Error: Not found: 'dart:html'                                    navigation/url_strategy.dart:6/…/..
import 'dart:html' as html;
       ^
: Error: Not found: 'dart:html'                                          navigation/utils.dart:5/…/..

import 'dart:html';
       ^
: Error: Not found: 'dart:js'                                                      lib/js.dart:8/…/..
export 'dart:js' show allowInterop, allowInteropCaptureThis;
^
: Error: Type 'html.EventListener' not found.                  navigation/js_url_strategy.dart:36/…/..

typedef _AddPopStateListener = ui.VoidCallback Function(html.EventListener);
                                                        ^^^^^^^^^^^^^^^^^^
: Error: Type 'html.EventListener' not found.                  navigation/js_url_strategy.dart:80/…/..
  external ui.VoidCallback addPopStateListener(html.EventListener fn);
                                               ^^^^^^^^^^^^^^^^^^
: Error: Type 'html.EventListener' not found.                     navigation/url_strategy.dart:34/…/..

  ui.VoidCallback addPopStateListener(html.EventListener fn);
                                      ^^^^^^^^^^^^^^^^^^
: Error: Type 'html.EventListener' not found.                     navigation/url_strategy.dart:97/…/..
  ui.VoidCallback addPopStateListener(html.EventListener fn) {
```

navigator > lib >  url_strategy_html.dart > ...

```dart
import 'package:flutter_web_plugins/flutter_web_plugins.dart';

urlStrategy() {
  setUrlStrategy(PathUrlStrategy());
}

```

navigator > lib >  url_strategy_io.dart > ...

```dart
urlStrategy() {
  // no op
}

```

navigator > lib > main.dart > ...

```dart
import 'package:flutter/material.dart';
import 'navigator_1.0/my_app.dart';
import 'package:navigator/url_strategy_io.dart'
    if (dart.library.html) 'package:navigator/url_strategy_html.dart';

Run | Debug | Profile
void main() {
  urlStrategy();
  runApp(const MyApp());
}

```

Use Flutter web?

If you already have a working flutter app, then give it a try! For a complete new and only web based project stick to a JS Framework :)

Any Questions?

Feel free to contact me:

stevendz.de

Btw: we are hiring :)

phntm.xyz