



dolphin

Setup and Developer Guide

Guided walkthroughs for Flutter that remember where every user left off. Everything you need to add dolphin to your app, from installation to production.

1	Overview and how it works	2
2	Installation	3
3	Quick start	4
4	Writing step content	5
5	Customising the look	6
6	Pages, resume and journeys	7
7	Progress, storage and sync	8
8	Testing and troubleshooting	9
9	Quick reference	10
10	License and disclaimer	11

1 Overview and how it works

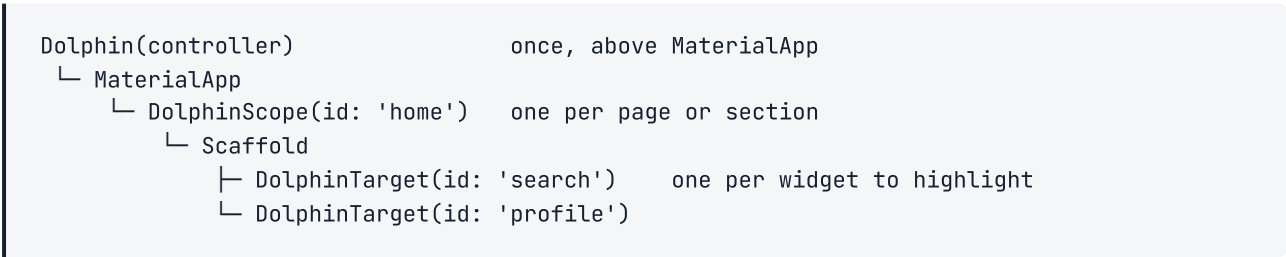
dolphin highlights any widget in your Flutter app and shows a tooltip next to it. It remembers which tips each user has seen, on every page, and picks up where they left off.

What you get

- **Automatic start per page.** A page with unseen tips starts its walkthrough by itself, pauses when covered, and resumes when the user returns.
- **No bookkeeping.** No key lists, no manager class, no "seen" flags to store. dolphin saves progress on the device for you.
- **One source of truth.** Tip text and styling can live in a single JSON file or come from your server.
- **Journeys.** Guide users through several pages in a fixed order.
- **Your data, your way.** Progress is plain JSON you can sync with your own backend.
- **Polished by default.** Six tooltip styles, eight highlight shapes, light and dark mode, on Android, iOS and web.

The scope-based approach

dolphin is built from three widgets, each with a single job:



WIDGET	ROLE
Dolphin	Wraps the app once. Holds the DolphinController , which stores progress and draws the overlay above every page.
DolphinScope	Marks a page or section. It is the unit of progress: it starts, pauses, resumes, skips and resets on its own, without touching other scopes.
DolphinTarget	Wraps a widget to highlight. It only needs an id ; content can live on the target, the scope, or central JSON.

2 Installation

dolphin has no native code of its own and needs no permissions, so installation is the same on every platform.

Requirements

ITEM	MINIMUM
Flutter	3.27
Dart	3.6
Platforms	Android, iOS and web (macOS also supported)

Steps

- 1 **Add the package.** From your project folder, run:

```
flutter pub add dolphin
```

Or add it to `pubspec.yaml` and run `flutter pub get` :

```
dependencies:  
  dolphin: ^1.0.0
```

- 2 **Import it** in the files where you use it:

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

- 3 **Platform setup.** None. Android, iOS and web work with the default project settings.

- 4 **Check it.** Run `flutter pub get` and build your app once. If it builds, dolphin is installed.

Where progress is saved. On Android, iOS and macOS, dolphin uses SharedPreferences for small settings and a local SQLite database for progress. On the web and desktop, SharedPreferences holds both. You can replace this with your own storage (section 7).

3 Quick start

Three steps add a working walkthrough to an existing app.

- 1 **Create one controller and wrap your app with `Dolphin` , above `MaterialApp` :**

```
final guide = DolphinController();

void main() => runApp(
  Dolphin(
    controller: guide,
    child: const MaterialApp(home: HomePage()),
  ),
);
```

- 2 **Mark the page with `DolphinScope` , and wrap each widget you want to explain with `DolphinTarget` :**

```
class HomePage extends StatelessWidget {
  const HomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return DolphinScope(
      id: 'home',
      child: Scaffold(
        appBar: AppBar(actions: [
          DolphinTarget(
            id: 'search',
            title: 'Search',
            description: 'Find anything from here.',
            child: IconButton(
              icon: const Icon(Icons.search),
              onPressed: () {},
            ),
          ),
        ]),
      ),
    );
  }
}
```

- 3 **Run the app.** The walkthrough starts by itself when the page opens. After the user taps *Done*, the tip is saved as seen and never shows again, until you reset it.

Tip. Ids only need to be unique within a scope. Use short, stable names such as `search` or `profile` , since saved progress is keyed by them.

4 Writing step content

Tip content can come from three places. Use one, or mix them.

SOURCE	WHERE	BEST FOR
On the target	<code>DolphinTarget(title:, description:)</code>	Quick, local tips
On the page	<code>DolphinScope(steps: [DolphinStep(...)])</code>	Keeping a page's tips together
Central JSON	<code>guide.loadConfigJson(json)</code>	One source for the whole app, changeable from a server

Central JSON

With central data, widgets carry only an id:

```
final raw = await rootBundle.loadString('assets/walkthrough.json');
guide.loadConfigJson(jsonDecode(raw) as Map<String, dynamic>);

DolphinTarget(id: 'search', child: searchButton) // text comes from JSON
```

```
{
  "scopes": {
    "home": {
      "sequence": ["search", "profile"],
      "steps": {
        "search": { "title": "Search", "description": "Find anything." },
        "profile": { "title": "Your profile", "shape": "circle" }
      }
    }
  }
}
```

When a step has data in more than one place, fields merge one by one, and central JSON wins by default. Fetch a new config from your server at any time with `guide.setConfig(...)` or `guide.mergeConfig(...)`; visible pages update immediately.

Showing a changed tip again

Increase a step's `version` (inline or in JSON). Users who saw the old version will see the new one once.

5 Customising the look

Everything visual lives in `DolphinTheme`. Themes layer from the app down to a single step, and each layer changes only the fields it sets.

```
DolphinController(  
  theme: const DolphinTheme(  
    tooltipStyle: 'card',          // card, bubble, minimal, banner, glass, compact  
    primaryColor: Colors.indigo,  
    shape: HighlightShape.rounded(radius: 16),  
    highlightPadding: EdgeInsets.all(8),  
    barrierOpacity: 0.7,  
    pointer: PointerStyle.arrowAndDots(sizes: [8, 6, 4]),  
    counterFormat: '{current} of {total}',  
  ),  
  darkTheme: const DolphinTheme(primaryColor: Color(0xFFB4BEFF)),  
);
```

OPTION	WHAT IT DOES
Tooltip styles	Six presets, or your own widget with <code>registerTooltipBuilder</code> .
Highlight shapes	Rect, rounded, circle, oval, stadium, diamond, beveled or a custom path, with solid, dashed or gradient borders.
Pointer	Arrow, dots, arrow and dots, line, none, or a custom painter. Each dot's size, colour and shape can be set.
Light and dark	Follows the app's theme by default. Use <code>themeMode</code> to follow the phone or force one mode.
Overlay	<code>barrierColor</code> , <code>barrierOpacity</code> and <code>barrierBlur</code> .
Labels	<code>nextLabel</code> , <code>previousLabel</code> , <code>skipLabel</code> , <code>finishLabel</code> , and the counter format.

Match your app. `DolphinTheme.fromMaterial(Theme.of(context))` picks up your app's colour scheme in one line.

6 Pages, resume and journeys

How a page behaves

- A scope starts when its page is on screen, has finished animating in, and has unseen steps.
- When another page opens on top, the scope pauses. When the user comes back, it resumes at the same step.
- *Skip* marks only the current page's remaining steps as skipped. *Dismiss* closes without marking anything, so the tip returns next visit.
- Start a page by hand with `guide.start('home')`, or show everything again with `guide.replay('home')`.

Tap-through steps

For a "tap this button" tip that opens another page, set `targetTap: passThroughAndAdvance`. The user taps the real button, the new page runs its own tips, and the first page resumes when they return.

Journeys across pages

```
"journeys": {  
  "tour": [  
    { "scope": "home",    "steps": ["search", "profile"], "route": "/" },  
    { "scope": "details", "route": "/details" }  
  ]  
}
```

```
final guide = DolphinController(navigatorKey: navigatorKey);  
await guide.startJourney('tour'); // resumes where it stopped
```

dolphin opens each leg's route for you with `pushNamed`. With any other router, pass `journeyNavigator` and navigate yourself.

7 Progress, storage and sync

dolphin saves progress on the device automatically. You can also read it, reset it, and keep it in sync with your own server.

Reading progress

```
guide.isSeen('home', 'search'); // true or false
guide.pendingSteps('home');     // ['profile']
guide.summary('home');          // steps, seen, pending, status
guide.exportProgress();          // plain JSON for your server
```

Rewatch and reset

```
await guide.resetAll();           // "Rewatch walkthrough" in settings
await guide.resetScope('home');  // one page
await guide.resetSteps('home', ['search']); // single steps
```

Syncing with your backend

```
// Save every change
DolphinController({
  onProgressChanged: (p) => api.saveGuideProgress(p.toJson()),
});

// Restore on a new device
await guide.syncWithServer({
  fetch: () => api.getGuideProgress(),
  push: (merged) => api.saveGuideProgress(merged),
});
```

Merging uses *union* by default: a tip seen on any device counts as seen, so users who change phones only see what they haven't seen yet.

Privacy. dolphin stores only which tips were seen or skipped, and when. It makes no network requests; syncing happens only through the callbacks you write.

8 Testing and troubleshooting

Widget tests

```
final guide = DolphinController(  
  store: MemoryDolphinStore(),  
  startDelay: Duration.zero,  
  theme: const DolphinTheme(animation: DolphinAnimation.none()),  
);
```

Turn animations off in tests: the highlight pulse repeats forever, so `pumpAndSettle` would never finish while it runs.

Common questions

IF THIS HAPPENS	CHECK THIS
The walkthrough never starts	<code>Dolphin</code> wraps <code>MaterialApp</code> , the page has a <code>DolphinScope</code> , and its steps are not already seen. Call <code>guide.resetAll()</code> while developing.
A step is skipped	Its target was not on screen. Items in lazy lists only exist once built; use <code>waitForTarget</code> or target a visible widget.
The tip shows the wrong text	Central JSON wins over inline text by default. Set <code>dataPrecedence: DataPrecedence.inlineWins</code> to flip it.
Changed text is not shown again	Increase the step's <code>version</code> .
I need to see what's happening	Set <code>debugLogging: true</code> , or listen to <code>guide.events</code> .

9 Quick reference

NEED	CALL
Start or continue a page	<code>start(scope)</code>
Show a page again	<code>replay(scope)</code>
Move through steps	<code>next()</code> , <code>previous()</code> , <code>skipScope()</code> , <code>dismiss()</code>
Turn tips on or off	<code>setEnabled(bool)</code> , <code>setScopeEnabled(scope, bool)</code>
Order and hide steps	<code>setSequence</code> , <code>exclude</code> , <code>include</code>
Reset	<code>resetAll</code> , <code>resetScope</code> , <code>resetSteps</code> , <code>resetJourney</code>
Journeys	<code>startJourney</code> , <code>stopJourney</code> , <code>resumeActiveJourney</code>
Progress out	<code>exportProgress</code> , <code>progress</code> , <code>onProgressChanged</code>
Progress in	<code>importProgress</code> , <code>syncWithServer</code>
Reports	<code>isSeen</code> , <code>pendingSteps</code> , <code>scopeStatus</code> , <code>summary</code>
Config	<code>setConfig</code> , <code>mergeConfig</code> , <code>loadConfigJson</code> , <code>registerTooltipBuilder</code> , <code>registerCondition</code>

The complete reference, including every JSON key, is in the README on [pub.dev](#) and [GitHub](#).

10 License and disclaimer

dolphin is free to use in personal and commercial apps under the BSD 3-Clause License. The author's name, the dolphin name and its logo may not be used to promote or publish a copy or a modified version as the original.

dolphin is provided "as is", without warranty of any kind. The author is not responsible for any bug, crash, misbehaviour or loss caused by using it in your project, or for any data in your app, including the walkthrough progress stored on the device and anything you sync with your own servers. Test it in your own app before you release it to users.

Links

Package	pub.dev/packages/dolphin
Source and issues	github.com/sanket-manohar/dolphin
Website	sanket-manohar.github.io/dolphin
Author	sanketmanohar.com

© 2026 Sanket Santosh Vidya Manohar. dolphin Setup and Developer Guide, version 1.0.0.

