Macrobenchmarks

Performance benchmarks use either flutter drive or the web benchmark harness.

Mobile benchmarks

Cull opacity benchmark

To run the cull opacity benchmark on a device:

```
flutter drive --profile test_driver/cull_opacity_perf.dart
```

Results should be in the file `build/cull_opacity_perf.timeline_summary.json`.

More detailed logs should be in `build/cull_opacity_perf.timeline.json`.

Cubic bezier benchmark

To run the cubic-bezier benchmark on a device:

```
flutter drive --profile test_driver/cubic_bezier_perf.dart
```

Results should be in the file `build/cubic_bezier_perf.timeline_summary.json`.

More detailed logs should be in `build/cubic_bezier_perf.timeline.json`.

Backdrop filter benchmark

To run the backdrop filter benchmark on a device: To run a mobile benchmark on a device:

```
flutter drive --profile -t test_driver/run_app.dart --driver
test_driver/[test_name]_test.dart
```

Results should be in the file `build/[test_name].timeline_summary.json`.

More detailed logs should be in `build/[test_name].timeline.json`.

The key `[test_name]` can be:

- `animated_placeholder_perf`
- `backdrop_filter_perf`
- `color_filter_and_fade_perf`
- `cubic_bezier_perf`
- `cull_opacity_perf`
- `fading_child_animation_perf`
- `imagefiltered_transform_animation_perf`
- `multi_widget_construction_perf`
- `picture_cache_perf`
- `post_backdrop_filter_perf`
- `simple_animation_perf`
- `textfield_perf`

- `fullscreen_textfield_perf`

E2E benchmarks

(On-going work)

E2E-based tests are driven independent of the host machine. The following tests are E2E:

- `cull_opacity_perf.dart`
- `multi_widget_construction_perf`

These tests should be run by:

```
flutter drive --profile -t test/[test_name]_e2e.dart --driver
test_driver/e2e_test.dart
```

Web benchmarks

Web benchmarks are compiled from the same entry point in `lib/web_benchmarks.dart`.

How to write a web benchmark

Create a new file for your benchmark under `lib/src/web`. See `bench_draw_rect.dart` as an example.

Choose one of the two benchmark types:

- A "raw benchmark" records performance metrics from direct interactions with `dart:ui` with no framework. This kind of benchmark is good for benchmarking low-level engine primitives, such as layer, picture, and semantics performance.
- A "widget benchmark" records performance metrics using a widget. This kind of benchmark is good for measuring the performance of widgets, often together with engine work that widget-under-test incurs.
- A "widget build benchmark" records the cost of building a widget from nothing. This is different from the "widget benchmark" because typically the latter only performs incremental UI updates, such as an animation. In contrast, this benchmark pumps an empty frame to clear all previously built widgets and rebuilds them from scratch.

For a raw benchmark extend `RawRecorder` (tip: you can start by copying `bench_draw_rect.dart`).

For a widget benchmark extend `WidgetRecorder` (tip: you can start by copying `bench_simple_lazy_text_scroll.dart`).

For a widget build benchmark extend `WidgetBuildRecorder` (tip: you can start by copying `bench_build_material_checkbox.dart`).

Pick a unique benchmark name and class name and add it to the `benchmarks` list in `lib/web_benchmarks.dart`.

How to run a web benchmark

Web benchmarks can be run using `flutter run` in debug, profile, and release modes, using either the HTML or the CanvasKit rendering backend. Note, however, that running in debug mode will result in worse numbers. Profile mode is useful for profiling in Chrome DevTools because the numbers are close to release mode and the profile contains unobfuscated names.

Example:

```
cd dev/benchmarks/macrobenchmarks

# Runs in profile mode using the HTML renderer
flutter run --web-renderer=html --profile -d web-server lib/web_benchmarks.dart

# Runs in profile mode using the CanvasKit renderer
flutter run --web-renderer=canvaskit --profile -d web-server lib/web_benchmarks.dart
```

You can also run all benchmarks exactly as the devicelab runs them:

```
cd dev/devicelab

# Runs using the HTML renderer
../../bin/cache/dart-sdk/bin/dart bin/run.dart -t bin/tasks/web_benchmarks_html.dart

# Runs using the CanvasKit renderer
../../bin/cache/dart-sdk/bin/dart bin/run.dart -t
bin/tasks/web_benchmarks_canvaskit.dart
```

Frame policy test

File `test/frame_policy.dart` and its driving script `test_driver/frame_policy_test.dart` are used for testing `fullyLive` and `benchmarkLive` policies in terms of its effect on `WidgetTester.handlePointerEventRecord`.