There are many ways to write a memory test for Flutter. In this article, we give 3 classes of example tests that are currently used by Flutter device lab. Memory performance is a high priority for Flutter so there are many new memory tools and test utilities in progress. We'll add them in this doc in the future.

MemoryTest that interacts with adb directly

These memory tests use the MemoryTest class defined in the device lab perf_tests.dart to poll adb directly before and after an overridable `useMemory` function. By default, `useMemory` will just run an app in release and wait for a "done" message to be printed in logcat.

Examples include

- `complex_layout_scroll_perf__memory`
  - device lab task file
  - main file
- `fast_scroll_large_images_memory`
  - device lab task file
  - main file

To write a new MemoryTest case `some_memory_perf` and add it to Flutter's device lab so Flutter's CI system can measure it for each Flutter commit, follow examples above to

1. Create a `main` function for the test app in a file named like `test_memory/some_memory_perf.dart`.
2. Add a `some_memory_perf` entry to manifest.yaml
3. Add a `some_memory_perf.dart` file to dev/devicelab/bin/tasks folder.

Pros

- Low overhead.
- Works in all runtime modes, including release.
- The test has complete control on when to start and stop the memory measurement.

Cons

- Only have 2 memory readings, begin and end, during the app run.
- Polling ADB may trigger collections of the Java heap.
- Only works on Android targets.
- Requires a test environment with access to ADB.
- Requires a host machine with Flutter SDK installed.

DevTools Memory Test

The memory tests use DevTools to poll adb and Dart VM during a normal Flutter driver test run, which typically measures speed performance instead of memory performance. DevToolsMemoryTest handles most of the process so a new test only needs to specify the driver test location.

Examples include

- `complex_layout_scroll_perf__devtools_memory`
  - device lab task file
- `large_image_changer_perf_android`
  - device lab task file

To write a new DevTools memory test case `some_memory_perf` and add it to Flutter's device lab so Flutter's CI system can measure it for each Flutter commit, follow examples above to

1. Write (or reuse) a normal Flutter driver test for the app in files named like `test_driver/some_memory_perf.dart` and `test_driver/some_memory_perf_test.dart`.
2. Add a `some_memory_perf` entry to manifest.yaml
3. Add a `some_memory_perf.dart` file to dev/devicelab/bin/tasks folder.

Pros

- Have finer grained measurements (~1 reading per second).
- Also have Dart VM memory info.
- Can easily turn a speed-focused driver test into a memory test.

Cons

- Don't have much control on when to start and stop the measurement.
- Polling ADB may trigger collections on the Java heap.
- Requires a test environment with access to ADB.
- Only works on Android targets.
- Not available for release mode, so may incur extra memory overhead in profile or debug mode.
- Requires a host machine with Flutter SDK installed.

iOS Memory Test

The iOS embedding of Flutter supports sampling memory usage during runtime, which then writes metrics to the Dart timeline. After recording a timeline for the relevant portion of an application's execution, the timeline can be analyzed to obtain memory related information from the profile.

Examples include

- `large_image_changer_perf_ios`
  - device lab task file

To write a new iOS memory test case `some_memory_perf` and add it to Flutter's device lab so Flutter's CI system can measure it for each Flutter commit, follow examples above to

1. Write (or reuse) a normal Flutter driver test for the app in files named like `test_driver/some_memory_perf.dart` and `test_driver/some_memory_perf_test.dart`.
2. Add a `some_memory_perf` entry to manifest.yaml
3. Add a `some_memory_perf.dart` file to dev/devicelab/bin/tasks folder that specifies `measureMemory: true`.

Pros

- Can be run on a machine that does not have the Flutter SDK installed.
- Can adjust the sampling frequency so one can have as many or as few measurements as needed.
- Each sampling has much less overhead compared to calling adb
- Flutter driver tests on iOS get memory measurements for free.

Cons

- Only works on iOS targets.
- Not available for release mode, so may incur extra memory overhead in profile or debug mode.
- Memory polling mechanism may incur additional memory overhead.