ОНЛАЙН-ОБРАЗОВАНИЕ

Меня хорошо видно && слышно?

Ставьте + , если все хорошо
Напишите в чат, если есть проблемы

Flutter Mobile Developer

Тема 21. Анализ работы приложения. Сохраняем 60 fps в секунду

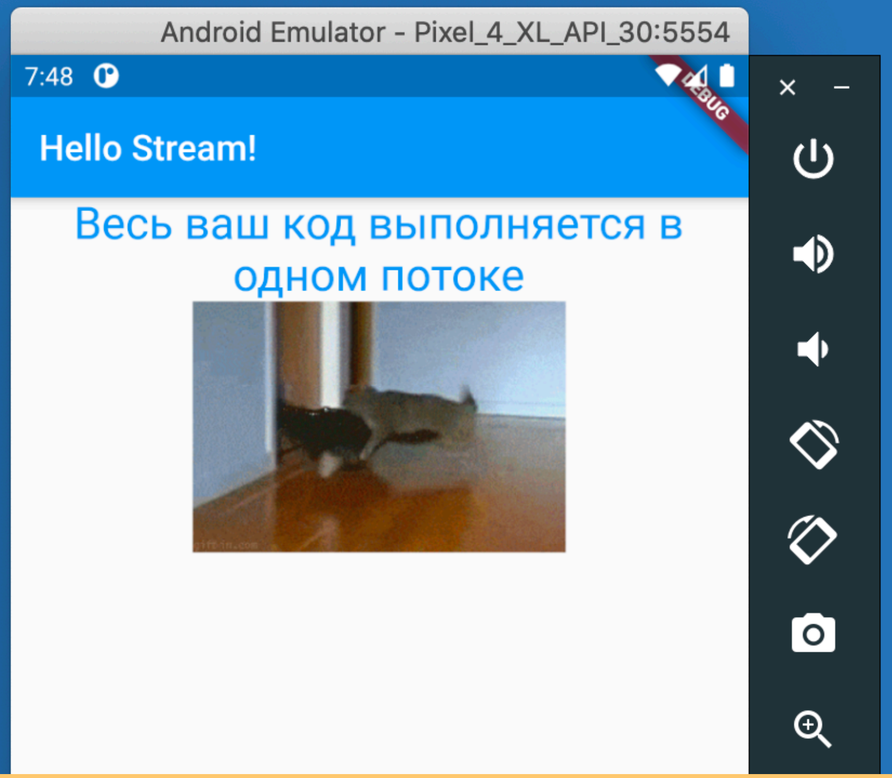
Цель урока

- Научиться анализировать производительность
- Находить слабые места в приложении и исправлять их

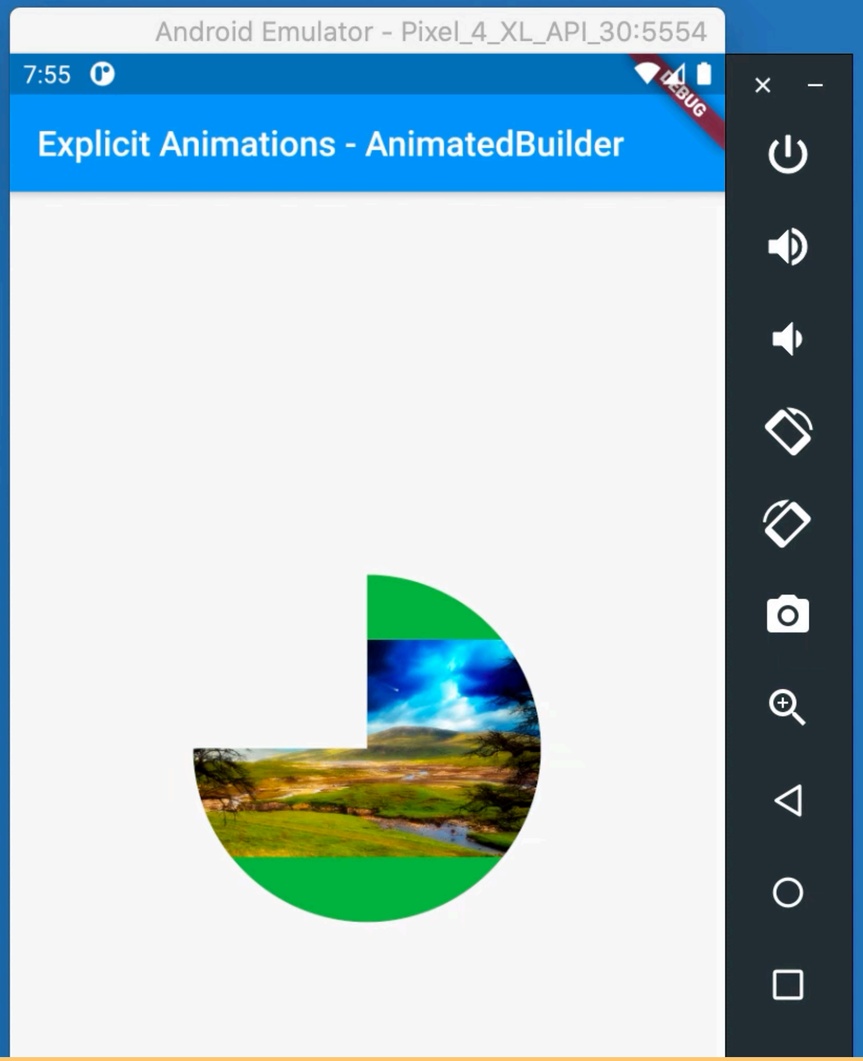
О чем будем говорить

- Немного теории
- Tooling и поиск проблем - Observatory
- Tooling - DevTools

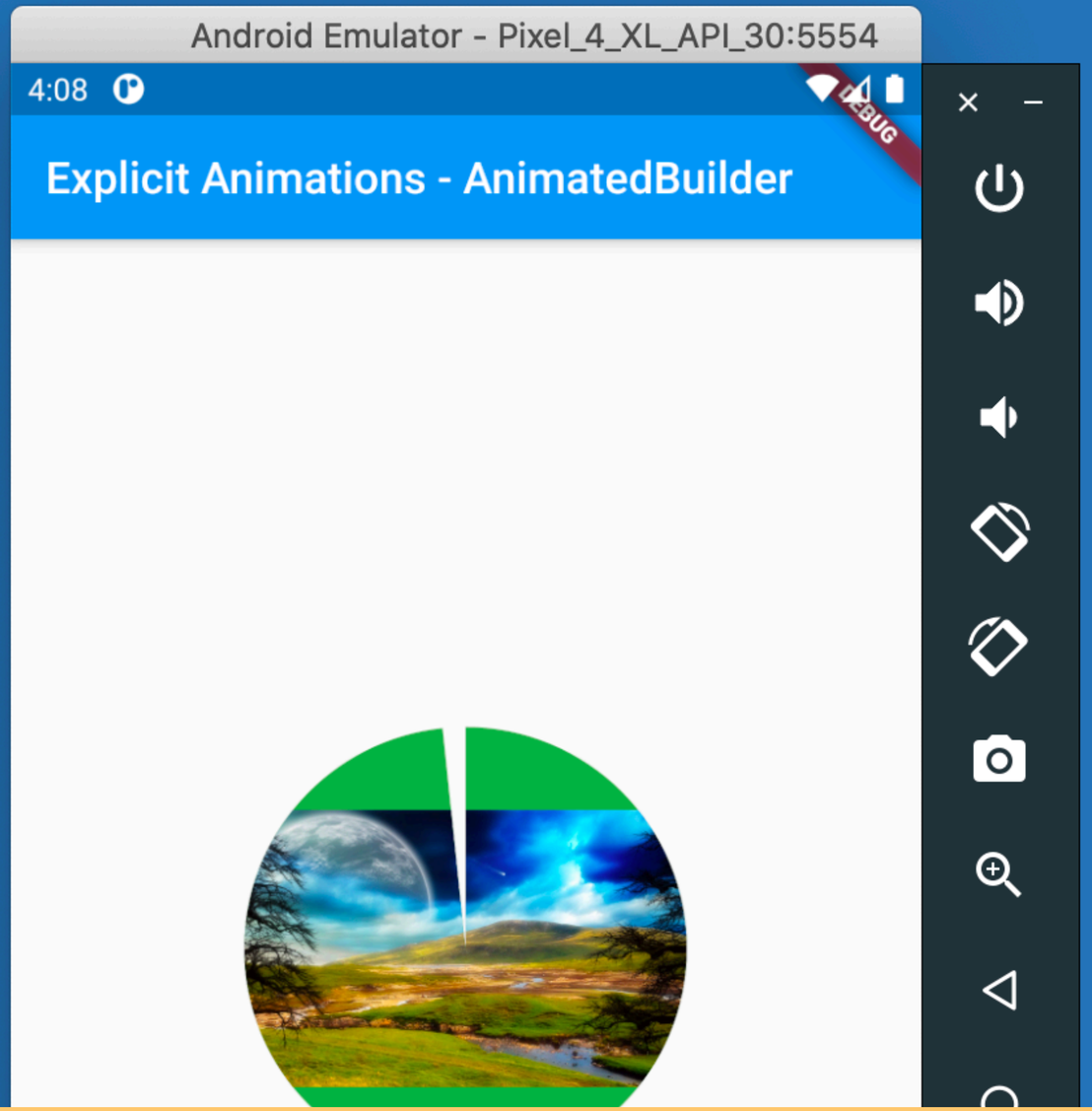
Jank


Android Emulator - Pixel_4_XL_API_30:5554
7:48
DEBUG
Hello Stream!
Весь ваш код выполняется в одном потоке

Jank


Android Emulator - Pixel_4_XL_API_30:5554
7:55
DEBUG
Explicit Animations - AnimatedBuilder

Анализ производительности на эмуляторе



Debug, Profile, Release

- https://flutter.dev/docs/testing/build-modes

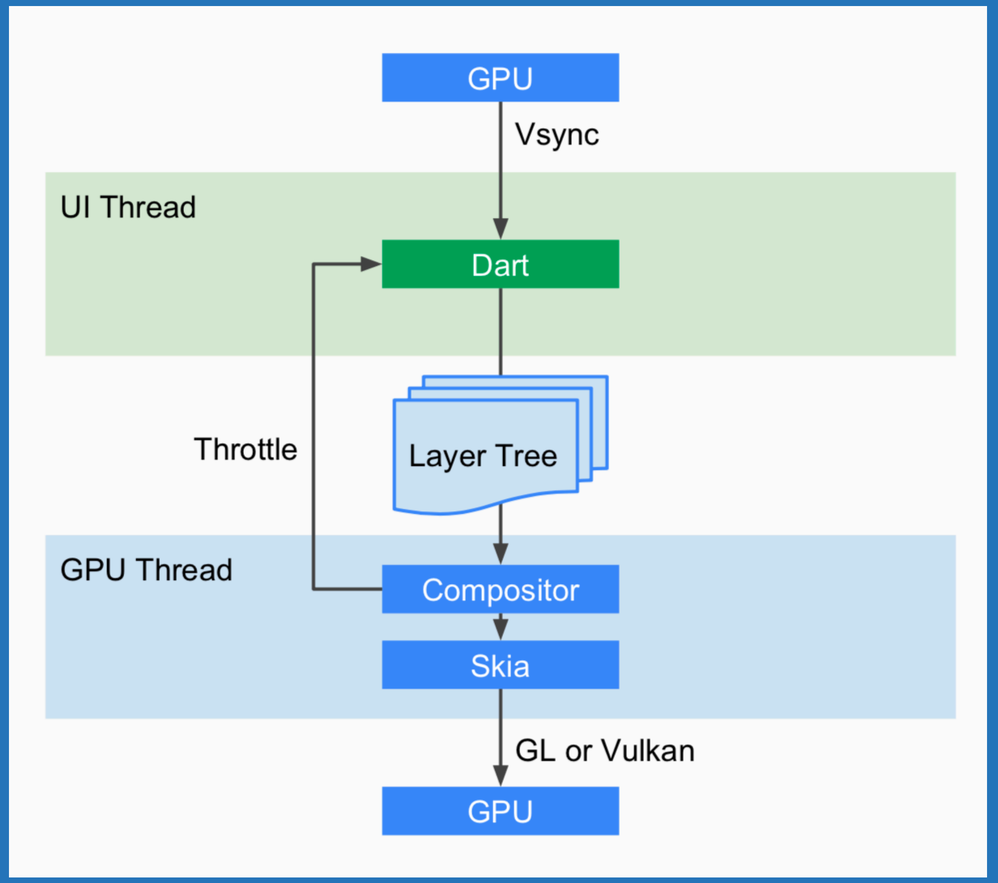
- **Debug**

- **Profile** - Этот режим недоступен в эмуляторе (симуляторе) по причине, что поведение в эмуляторе не дает реальной картины производительности приложения. Этот режим очень близок к режиму Release, за исключением того, что есть логгинг, трассировка событий и VM Service, который позволяет вам подключиться при помощи DevTools или Observatory. Flutter умышленно не разрешает запуск приложения в Profile редеем на эмуляторе, так как это не имеет смысла.

- **Release -** В этом режиме мы получаем AoT сборку, нет никакой отладочной информации, нет трассировки событий. Tree Shaking выбросил весь неиспользуемый код.

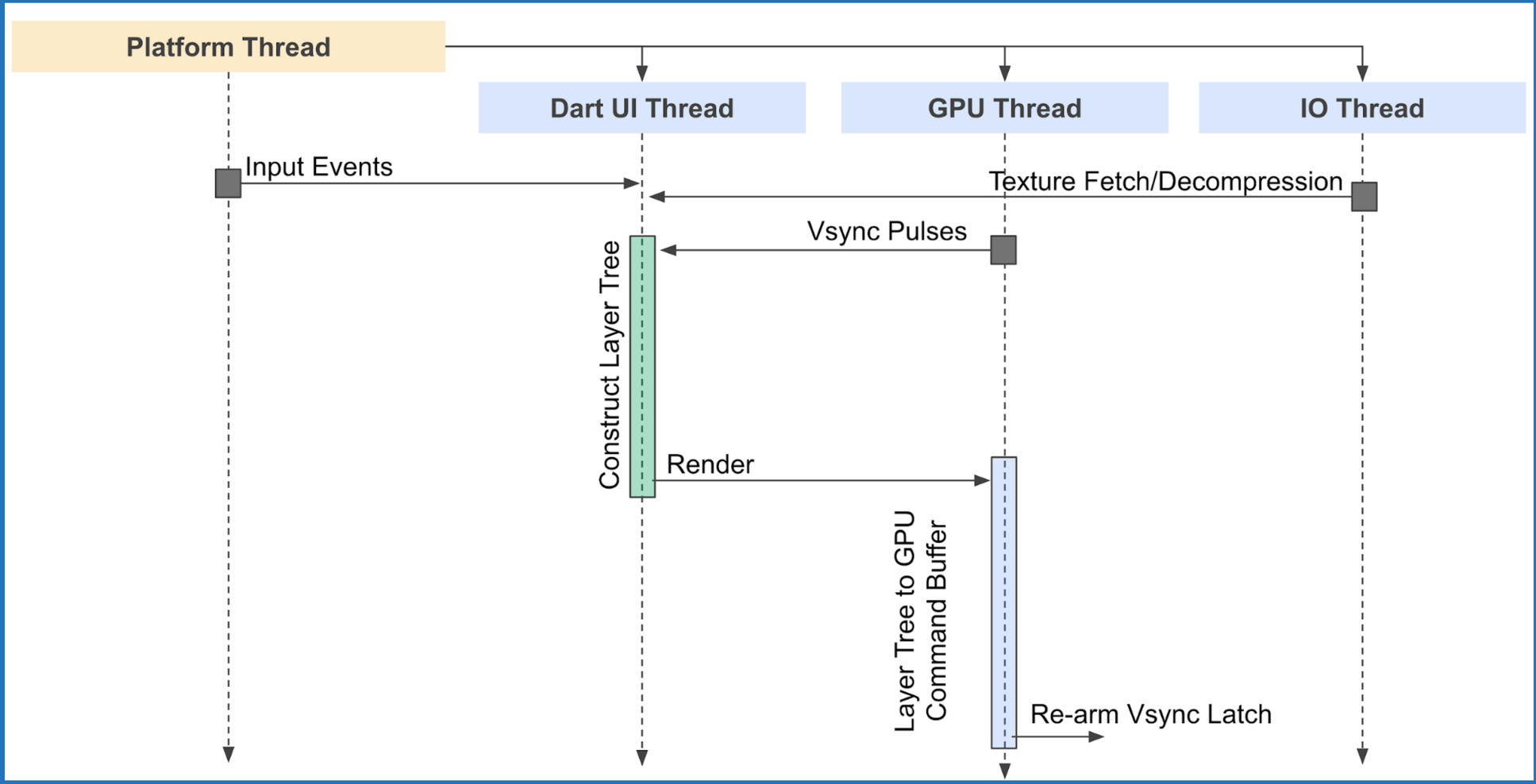
Немного вспомним про устройство Flutter

https://flutter.dev/docs/perf/rendering/ui-performance#flutters-threads



Немного вспомним про устройство Flutter

https://flutter.dev/docs/perf/rendering/ui-performance#flutters-threads



Потоки

https://flutter.dev/docs/perf/rendering/ui-performance#flutters-threads

- **UI** - В этом потоке выполняется код вашего приложения, в том числе построение виджета Build, формирование дерева элементов, дерева тендера, композиция, построение Render View. В результате мы получаем некое описание того, что должно быть на экране (Scene). Это описание не зависит от того, чем будет рендериться (OpenGL, Vulkan) и в конечном итоге оно передается в потоке Raster (GPU).

- **Raster (GPU)** - The raster thread takes the layer tree and displays it by talking to the GPU (graphic processing unit). You cannot directly access the raster thread or its data but, if this thread is slow, it's a result of something you've done in the Dart code. Skia, the graphics library, runs on this thread. Shown in the top row of the performance overlay. This thread was previously known as the "GPU thread" because it rasterizes for the GPU. But it is running on the CPU. We renamed it to "raster thread" because many developers wrongly (but understandably) assumed the thread runs on the GPU unit.

Потоки

https://flutter.dev/docs/perf/rendering/ui-performance#flutters-threads

- **Platform -** Здесь выполняется код плагинов (platform channel).
- **I/O -** Здесь выполняются все операции IO, такие как обращения к сети или к файловой системе.

О чем будем говорить

Немного теории

Tooling и поиск проблем - Observatory

Tooling - DevTools

Tooling и поиск проблем

```
flutter run --debug
flutter run --profile
flutter run --release
```

Observatory

Observatory > vm@ws://127.0.0.1:58080/eqaEwSB5pfk=/ws

Refresh

Process

pid	13809
current memory	264.5MB
peak memory	269.2MB
malloc memory	unavailable
malloc allocation count	unavailable
view malloc profile	view process memory

VM

name	vm@ws://127.0.0.1:58080/eqaEwSB5pfk=/ws
version	2.10.4 (stable) (Wed Nov 11 13:35:58 2020 +0100) on "android_ia32"
embedder	Flutter
current memory	81.8MB
started at	2020-12-24 19:41:34.757
uptime	0:04:36.095000
refreshed at	2020-12-24 19:46:10.853
see flags	view timeline

Isolate Group 17964230205068528000 (main.dart)

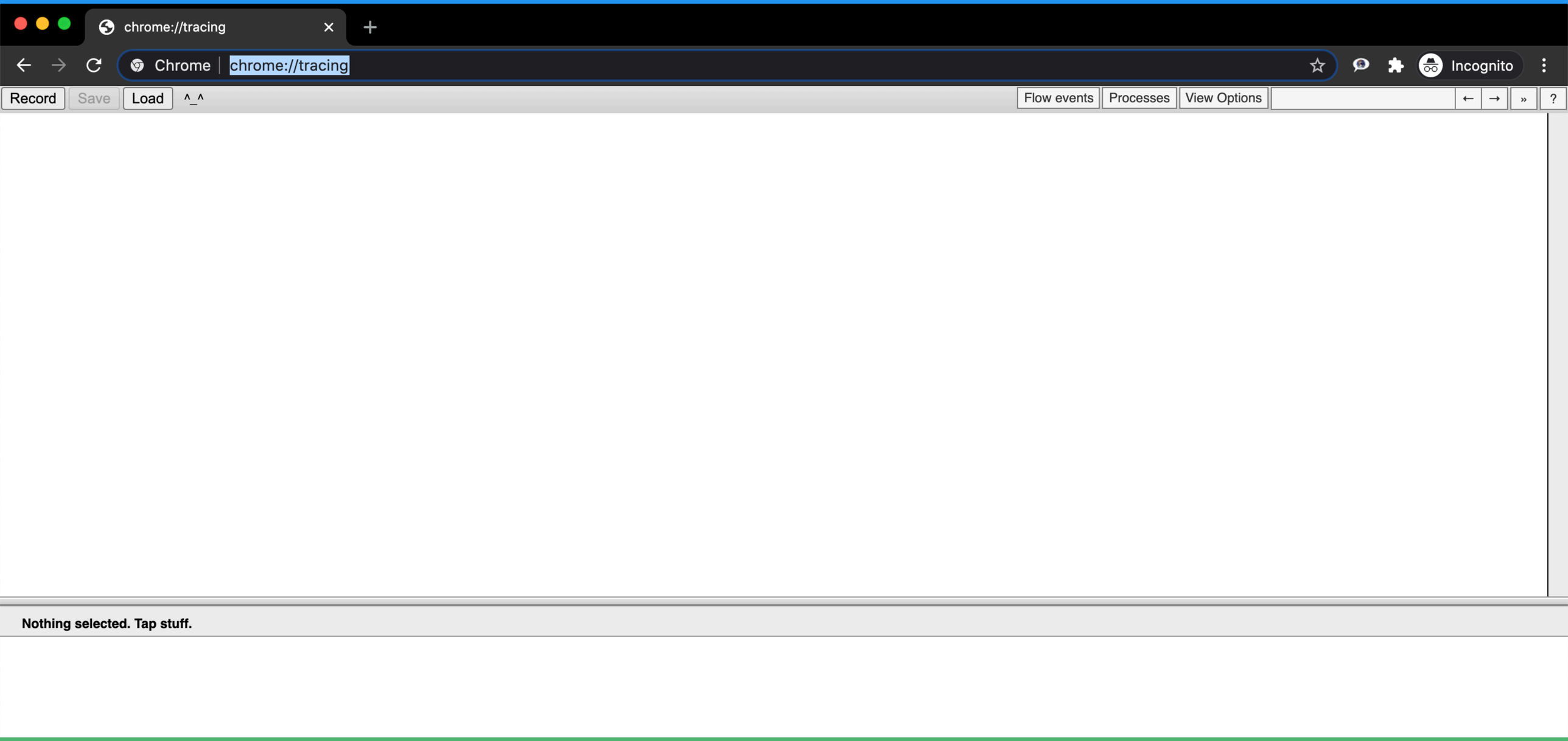
Isolate 3480225669762223 (main)

idle [debug]

new-space 512.0KB of 8.0MB / old-space 47.9MB of 50.4MB / heap 48.4MB of 58.4MB

debugger | class hierarchy | cpu profile | cpu profile (table) | allocation profile | heap snapshot | heap map | metrics | persistent handles | ports | logging

Observatory



Observatory - Timeline

Observatory > vm@ws://127.0.0.1:8081/_J81YR10aLk=/ws > timeline

load | save | clear | Refresh

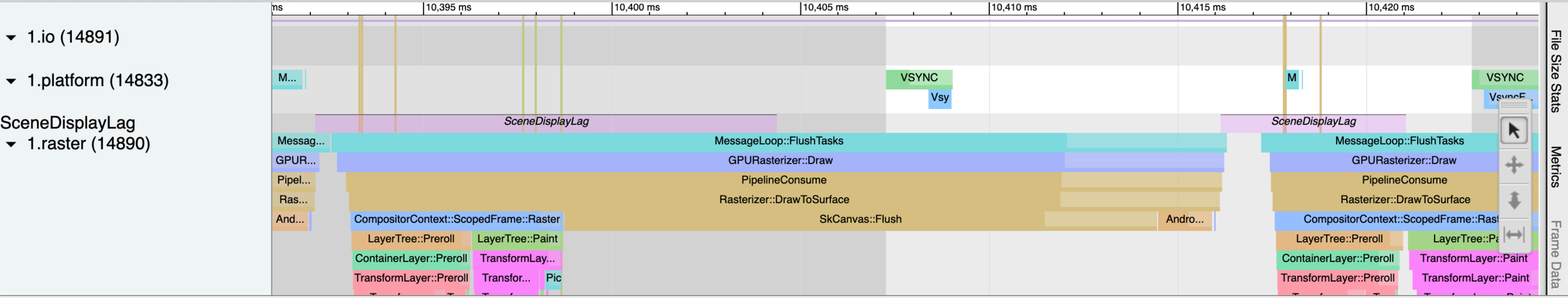
Timeline settings

Recorder: Ring

Recorded Streams Profile: None - Dart Developer - Flutter Developer - VM Developer - All

Recorded Streams: ☐API ☐Compiler ☐CompilerVerbose ☑Dart ☐Debugger ☑Embedder ☑GC ☐Isolate ☐VM

^_^ | Flow events | Processes | View Options | ← | → | » | ?

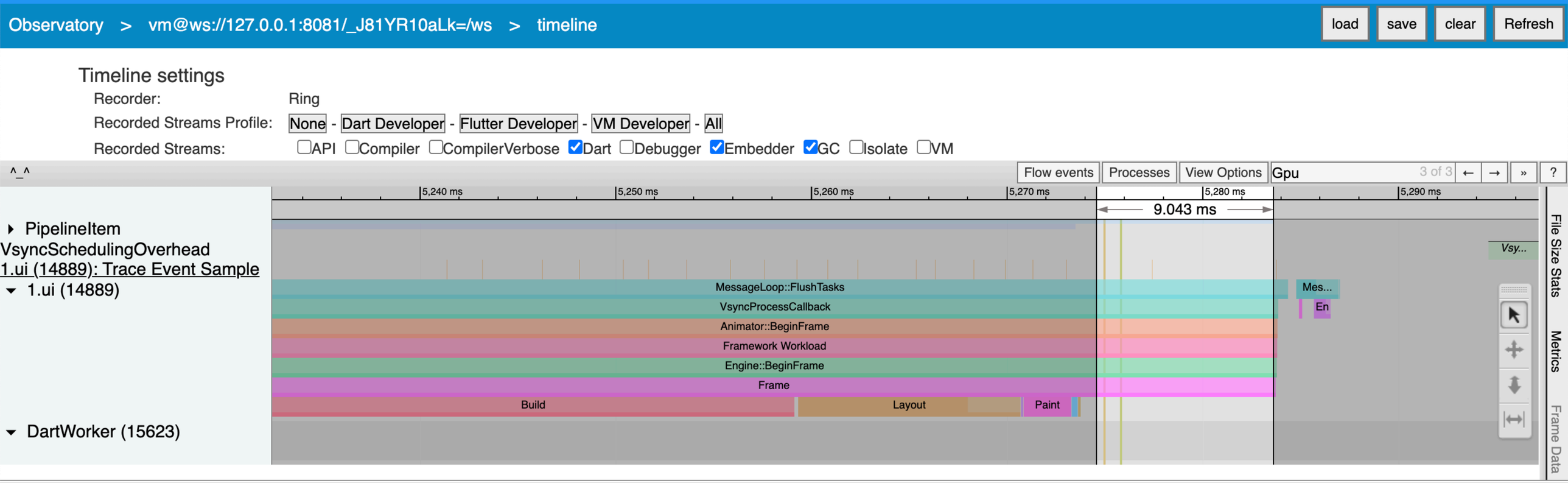


1 item selected. Slice (1)

Title	Build
Category	Dart
User Friendly Category	other
Start	10,411.593 ms
Wall Duration	1.162 ms
CPU Duration	0.808 ms
▼Args	
mode	"basic"
isolateId	"isolates/2367737761583979"
isolateGroupId	"isolateGroups/5230993715070334665"

Event(s)	Link
Overlapping samples	Sample at 10,411.969 ms
Preceding events	10 events of various types
All connected events	128 events of various types

Observatory - Timeline

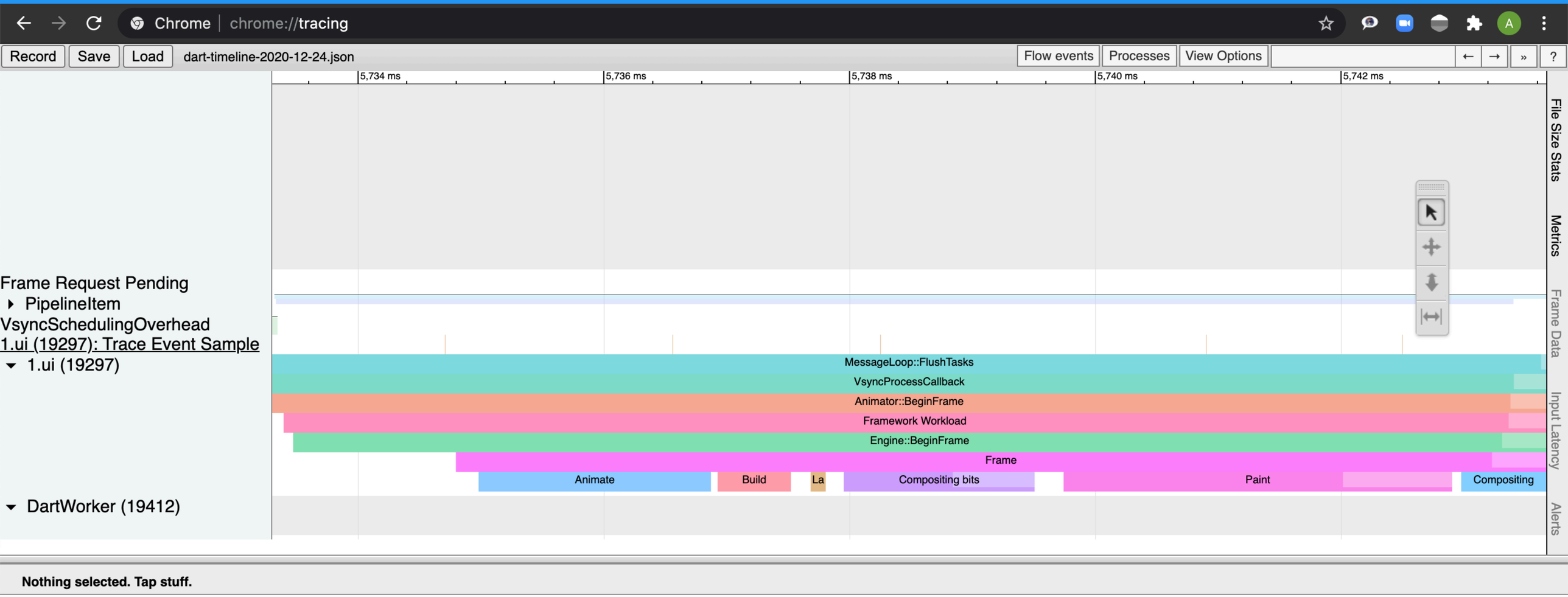


1 item selected. Slice (1)

Title	Build
Category	Dart
User Friendly Category	other
Start	5,167.215 ms
Wall Duration	91.912 ms
CPU Duration	19.817 ms
▼Args	
mode	"basic"
isolateId	"isolates/2367737761583979"
isolateGroupId	"isolateGroups/5230993715070334665"

Event(s)	Link
Overlapping samples	10 Samples
Preceding events	10 events of various types
All connected events	54 events of various types

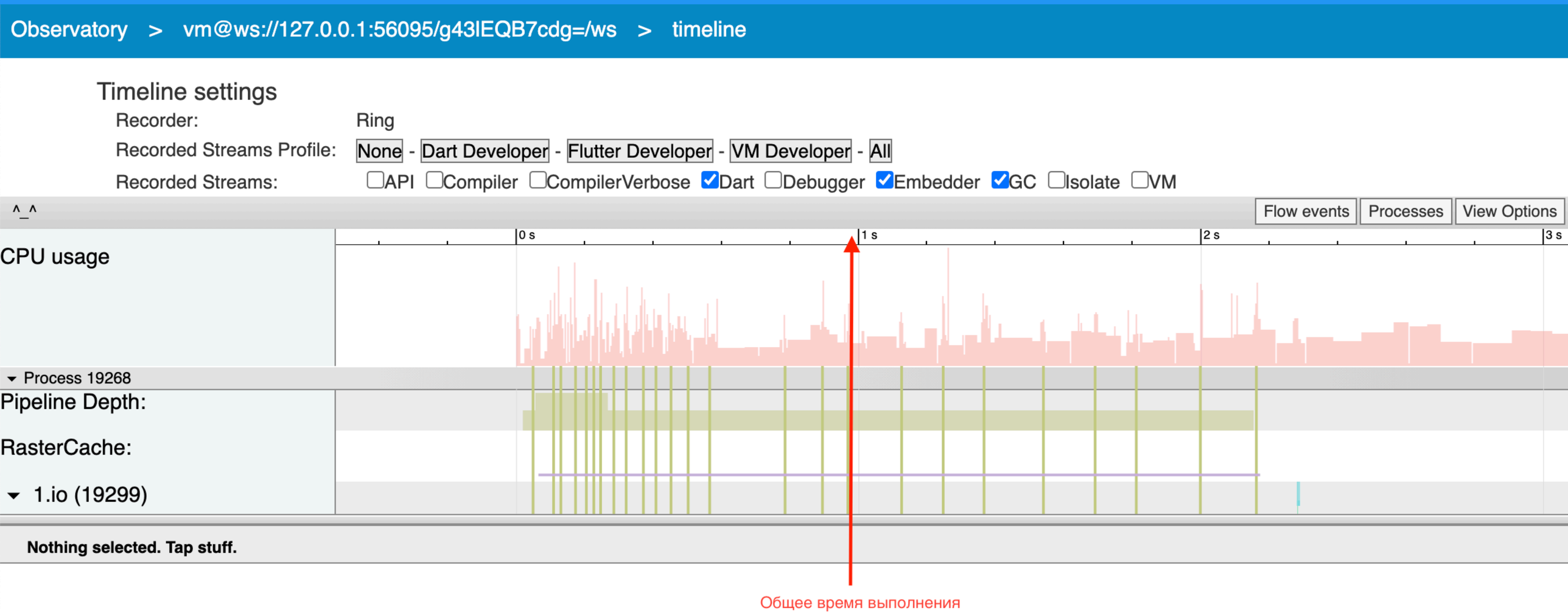
Observatory - Save and Load



Observatory - Timeline

DEMO

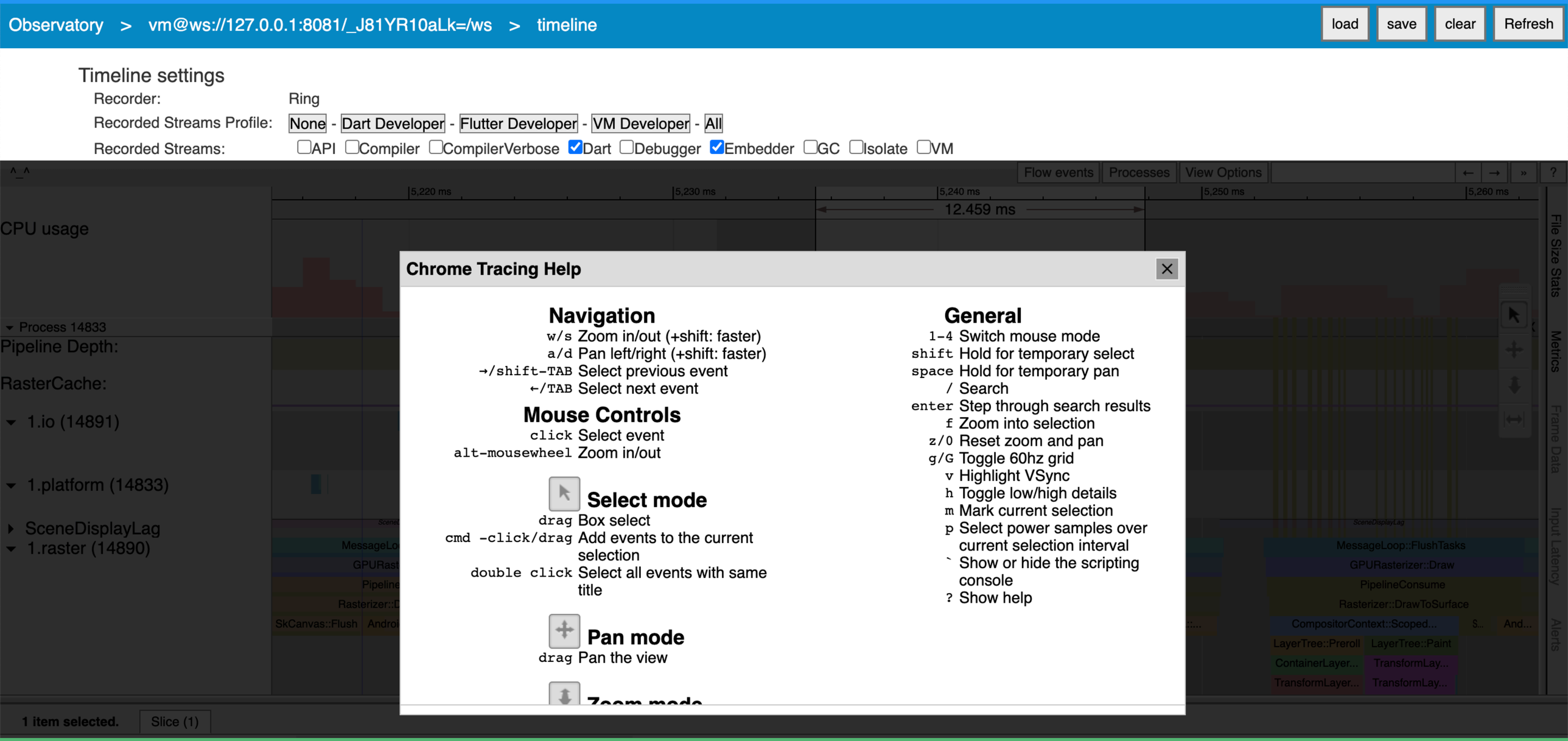
Observatory - Пробуем



Observatory - Timeline - WASD


W
S
D
A
Z
X
C
Q
E
R
F
T

Observatory - Timeline - WASD



Observatory - Tracing (Sync)

```
developer.Timeline.startSync('some event name');

/// Здесь любой синхронный код

developer.Timeline.finishSync();
```

Observatory - Tracing (Async)

```
final timelineTask = TimelineTask()
  ..start(
    ‘Some event name',
    arguments: <String, dynamic>{},
  );

/// Здесь любой код, в том числе в другом изоляте

timelineTask.finish();
```

Observatory - Tracing

DEMO

Observatory - Tracing - Пробуем

Observatory > vm@ws://127.0.0.1:56095/g43lEQB7cdg=/ws > timeline

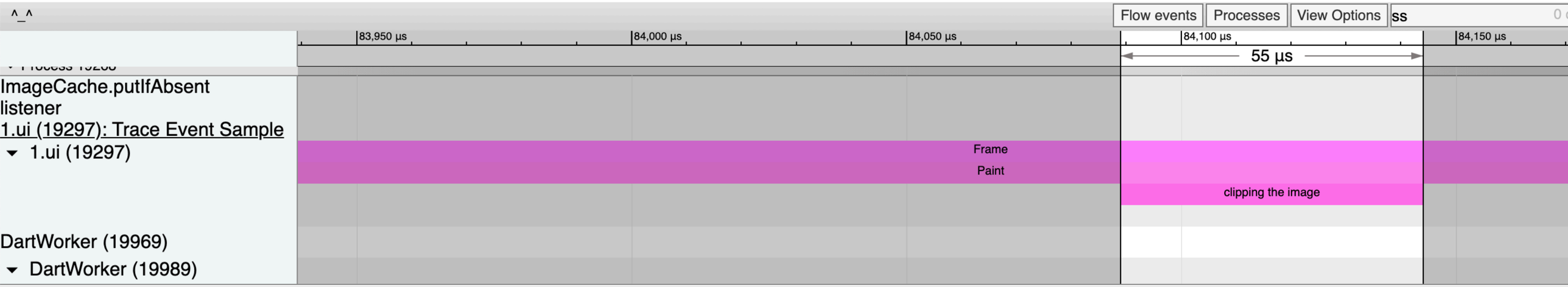
load save

Timeline settings

Recorder: Ring

Recorded Streams Profile: None - Dart Developer - Flutter Developer - VM Developer - All

Recorded Streams: API Compiler CompilerVerbose Dart Debugger Embedder GC Isolate VM



1 item selected. Slice (1)

Title	clipping the image
Category	Dart
User Friendly Category	other
Start	84.089 ms
Wall Duration	0.055 ms
CPU Duration	0.055 ms
▼Args	
isolateId	"isolates/3133821675280135"
isolateGroupId	"isolateGroups/16293999248976435424"

Observatory - Tracing - Пробуем

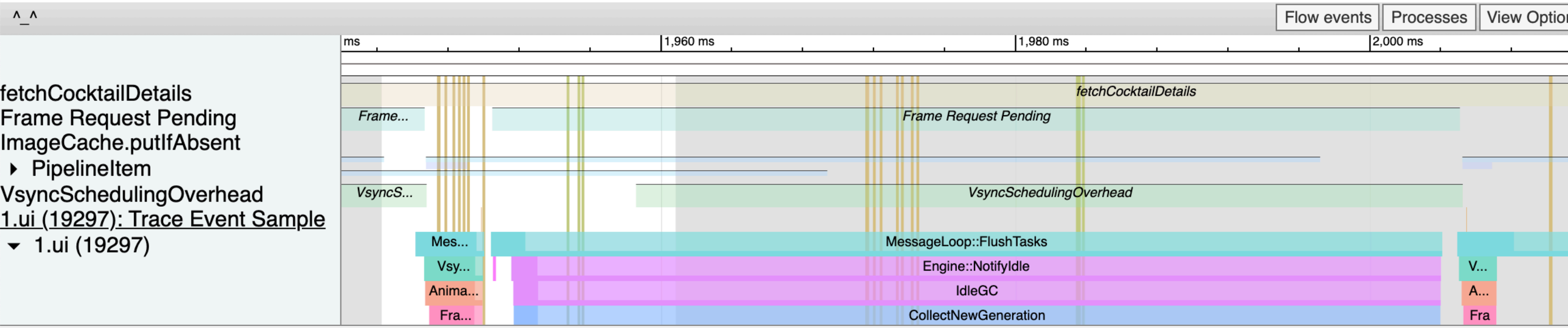
Observatory > vm@ws://127.0.0.1:56095/g43lEQB7cdg=/ws > timeline

Timeline settings

Recorder: Ring

Recorded Streams Profile: None - Dart Developer - Flutter Developer - VM Developer - All

Recorded Streams: ☐API ☐Compiler ☐CompilerVerbose ☑Dart ☐Debugger ☑Embedder ☑GC ☐Isolate ☐VM



1 item selected. Async Slice (1)

Title	fetchCocktailDetails
Category	Dart
Start	1,750.620 ms
Wall Duration	572.674 ms
▼Args	
id	"16943"

Event(s)	Link
Overlapping samples	78 Samples

Observatory - CPU - Профилирование кода

Flame chart
This tab of the profiler shows CPU samples for the selected frame event (such as Layout in the following example). This chart should be viewed as a top-down stack trace, where the top-most stack frame calls the one below it. The width of each stack frame represents the amount of time it consumed the CPU. Stack frames that consume a lot of CPU time may be a good place to look for possible performance improvements.

Call tree
The call tree view shows the method trace for the CPU profile. This table is a top-down representation of the profile, meaning that a method can be expanded to show its callees.

Bottom up
The bottom up view shows the method trace for the CPU profile but, as the name suggests, it’s a bottom-up representation of the profile. This means that each top-level method in the table is actually the last method in the call stack for a given CPU sample (in other words, it’s the leaf node for the sample).
In this table, a method can be expanded to show its callers.

Observatory - CPU - todo: замени картинку

Observatory > vm@ws://127.0.0.1:56095/g43lEQB7cdg=/ws > main > package:cocktail_app/profiling/animated_builder.dart

Refresh

dependencies (4) { ⊞ }

scripts (2) { ⊞ }

classes (6) { ⊞ }

variables (1) { ⊞ }

```
                  import 'dart:developer';
                  import 'dart:io';
                  import 'dart:math';

                  import 'package:flutter/material.dart';

                  const double _logoSize = 200.0;

                  class ArcClipper extends CustomClipper<Path> {
                    final double _clipFactor;

B                   ArcClipper(this._clipFactor);

                    @override
B                   Path getClip(Size size) {
B     0.35%  0.00%    Timeline.startSync('clipping the image');
B                     final radius = size.width / 2;
B                     final center = Offset(size.width / 2, size.height / 2);
B                     final angle = pi * 2 * _clipFactor;
B     0.03%  0.00%    final path = Path()
B                       ..moveTo(center.dx, center.dy)
B                       ..lineTo(center.dx, center.dy - radius)
B     3.59%  0.00%      ..arcTo(calculateRect(center, size), -pi / 2, angle, true)
B                       ..lineTo(center.dx, center.dy)
B                       ..close();
B     0.93%  0.00%    Timeline.finishSync();
```

Observatory - CPU - Профилирование кода

Observatory > vm@ws://127.0.0.1:56095/g43lEQB7cdg=/ws > main > package:cocktail_app/profiling/animated_builder.dart

Refresh

dependencies (4) { ⊞ }

scripts (2) { ⊞ }

classes (6) { ⊞ }

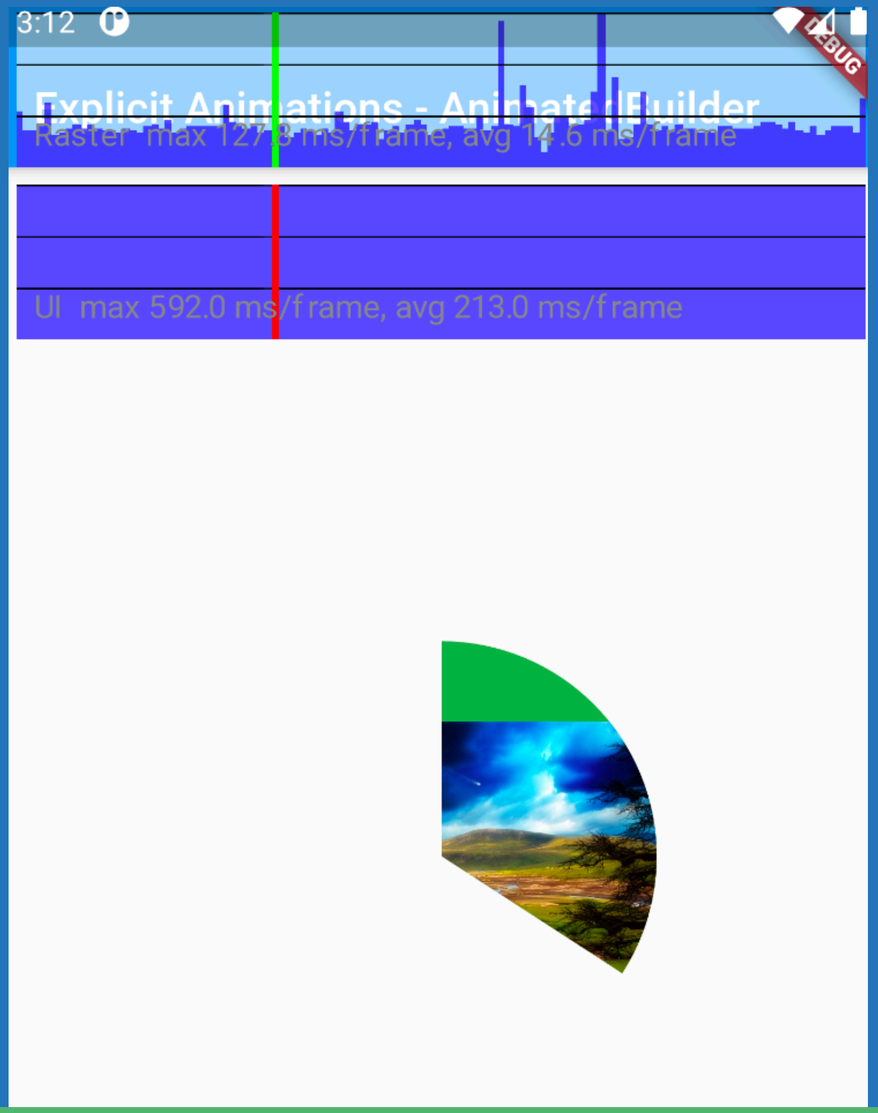
variables (1) { ⊞ }

```
                  import 'dart:developer';
                  import 'dart:io';
                  import 'dart:math';

                  import 'package:flutter/material.dart';

                  const double _logoSize = 200.0;

                  class ArcClipper extends CustomClipper<Path> {
                    final double _clipFactor;

B                   ArcClipper(this._clipFactor);

                    @override
B                   Path getClip(Size size) {
B      0.35%  0.00%   Timeline.startSync('clipping the image');
B                     final radius = size.width / 2;
B                     final center = Offset(size.width / 2, size.height / 2);
B                     final angle = pi * 2 * _clipFactor;
B      0.03%  0.00%   final path = Path()
B                       ..moveTo(center.dx, center.dy)
B                       ..lineTo(center.dx, center.dy - radius)
B      3.59%  0.00%     ..arcTo(calculateRect(center, size), -pi / 2, angle, true)
B                       ..lineTo(center.dx, center.dy)
B                       ..close();
B      0.93%  0.00%   Timeline.finishSync();
```

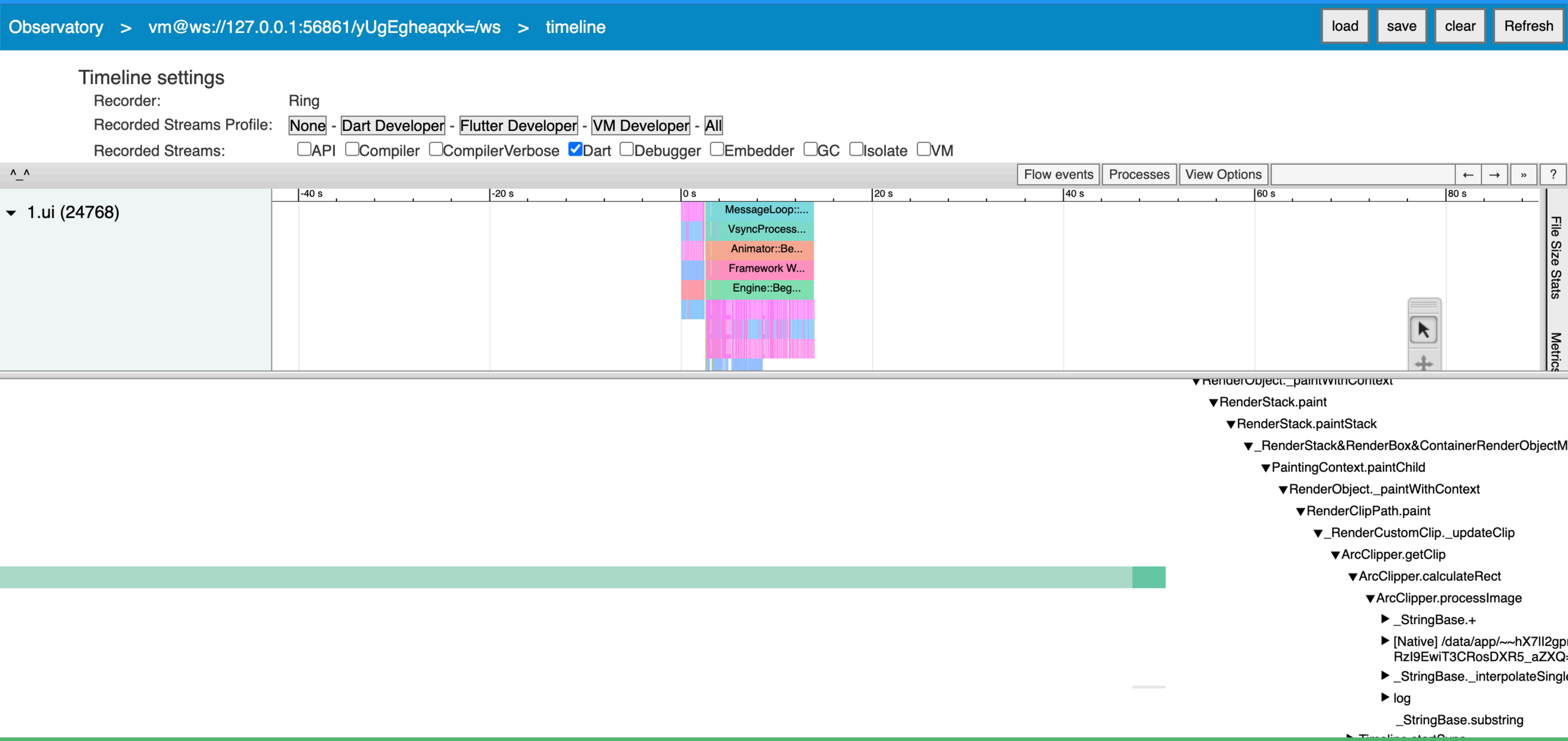
Observatory - CPU - Проблема в UI потоке

DEMO

Observatory - CPU - Проблема в UI потоке - Demo



Observatory - CPU - Профилирование кода



Observatory - CPU - Профилирование кода

Observatory > vm@ws://127.0.0.1:56095/g43lEQB7cdg=/ws > main > package:cocktail_app/profiling/animated_builder.dart

Refresh

dependencies (4) { ⊞ }

scripts (2) { ⊞ }

classes (6) { ⊞ }

variables (1) { ⊞ }

```
                  import 'dart:developer';
                  import 'dart:io';
                  import 'dart:math';

                  import 'package:flutter/material.dart';

                  const double _logoSize = 200.0;

                  class ArcClipper extends CustomClipper<Path> {
                    final double _clipFactor;

B                   ArcClipper(this._clipFactor);

                    @override
B                   Path getClip(Size size) {
B      0.35%  0.00%   Timeline.startSync('clipping the image');
B                     final radius = size.width / 2;
B                     final center = Offset(size.width / 2, size.height / 2);
B                     final angle = pi * 2 * _clipFactor;
B      0.03%  0.00%   final path = Path()
B                       ..moveTo(center.dx, center.dy)
B                       ..lineTo(center.dx, center.dy - radius)
B      3.59%  0.00%     ..arcTo(calculateRect(center, size), -pi / 2, angle, true)
B                       ..lineTo(center.dx, center.dy)
B                       ..close();
B      0.93%  0.00%   Timeline.finishSync();
```

Observatory - CPU - Профилирование кода

```
                                class AnimatedBuilderApp extends StatelessWidget {
   10                             @override
B  11                             Widget build(BuildContext context) {
B  12                               return MaterialApp(
B  13                                 home: MyHomePage(),
   14                               );
   15                             }
   16                           }
   17
   18                           class ArcClipper extends CustomClipper<Path> {
   19                             final double _clipFactor;
   20
B  21                             ArcClipper(this._clipFactor);
   22
B  23                             Rect calculateRect(Offset center, Size size, int i) {
B  24   91.77%   0.00%             processImage();
B  25                             return Rect.fromCenter(center: center, height: size.height, width: size.width);
   26                             }
   27
   28                             @override
B  29                             Path getClip(Size size) {
B  30    0.06%   0.00%             developer.Timeline.startSync('clipping the image');
B  31                             final radius = size.width / 2;
B  32                             final center = Offset(size.width / 2, size.height / 2);
B  33                             final angle = pi * 2 * _clipFactor;
B  34                             final path = Path()
B  35                               ..moveTo(center.dx, center.dy)
B  36                               ..lineTo(center.dx, center.dy - radius)
B  37   91.79%   0.00%               ..arcTo(calculateRect(center, size, iterations), -pi / 2, angle, true)
B  38                               ..lineTo(center.dx, center.dy)
B  39                               ..close();
B  40    0.08%   0.00%             developer.Timeline.finishSync();
B  41                             return path;
   42                             }
   43
B  44                             void processImage() {
B  45                               String message = '';
B  46    0.66%   0.02%               for (int i = 0; i < iterations; i++) {
B  47   90.96%   0.02%                 message += '$i';
   48                               }
```

Observatory - CPU - Профилирование кода

Observatory > vm@ws://127.0.0.1:56861/yUgEgheaqxk=/ws > main > package:cocktail_app/profiling/animated_builder.dart > calculateRect

Refresh

Class	Function
Shallow size	96B
Reachable size	... ↺
Retained size	... ↺
Retaining path	{ ⊞ }
Inbound references	{ ⊞ }

kind	regular
owner	ArcClipper
script	animated_builder.dart:23:3
current code	[Unoptimized] ArcClipper.calculateRect
unoptimized code	[Unoptimized] ArcClipper.calculateRect (usage count: 1503)
ic data array	*_List* (5) { ⊞ }
deoptimizations	0
optimizable	yes
inlinable	yes
intrinsic	no
recognized	no
native	no
vm name	calculateRect

B	23			Rect calculateRect(Offset center, Size size, int i) {
B	24	92.54%	0.00%	**processImage**();
B	25			return Rect.**fromCenter**(center: center, height: size.**height**, width: size.**width**);
	26			}

Observatory - Logging

Observatory > vm@ws://127.0.0.1:54372/XKADHP60Q3o=/ws > main > logging

clear

Logging

Show messages with severity WARNING

```
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:21 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
WARNING 2020-12-24 19: 3:22 01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768
```

Observatory - Memory

Observatory > vm@ws://127.0.0.1:54372/XKADHP60Q3o=/ws > main > allocation profile

Auto-refresh on GC | Refresh | GC | Download

Allocation Profile

last forced GC at ---

New Generation

used	512.0KB of 8.0MB
external	0B
collections	221
average collection time	9.53 ms

Old Generation

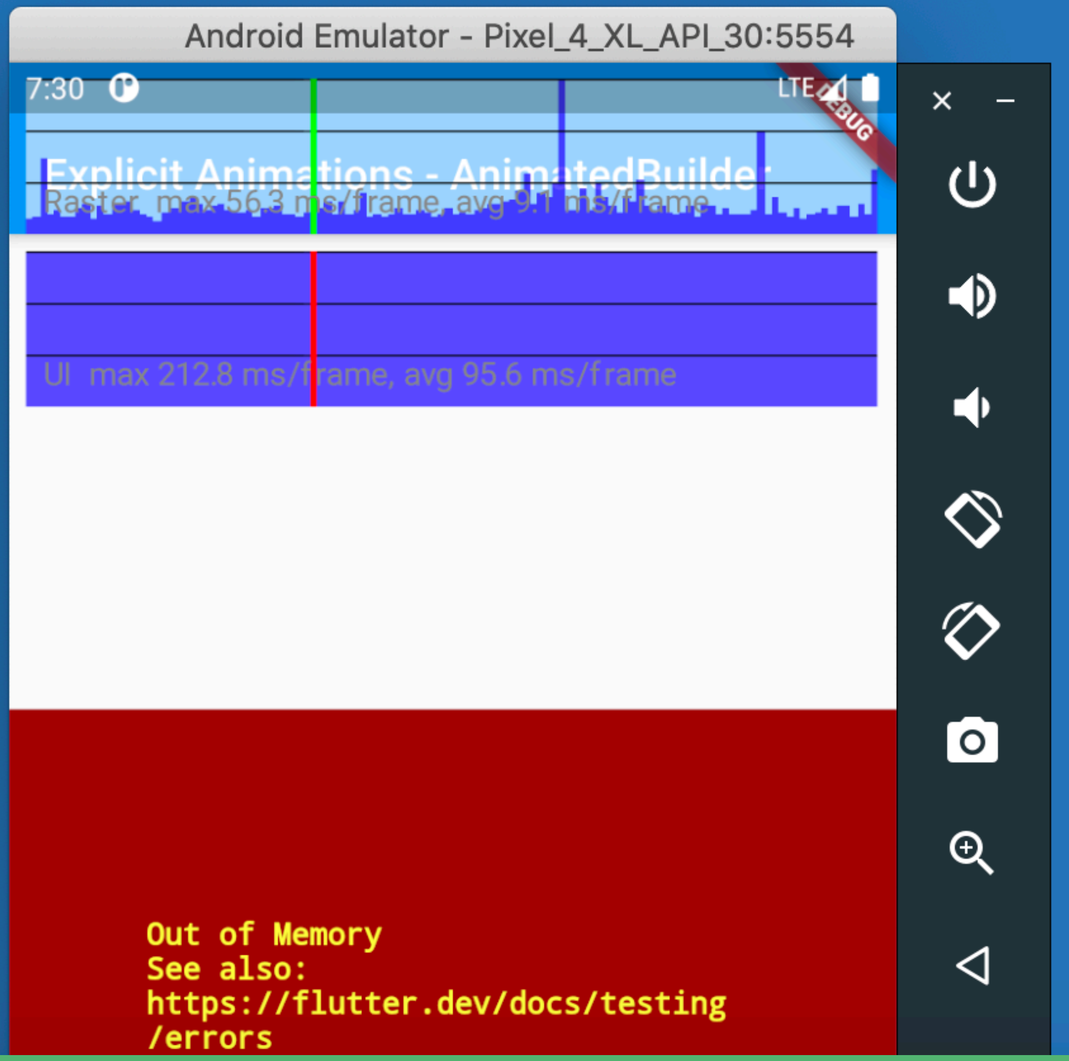
used	1.5GB of 1.5GB
external	31.5KB
collections	197
average collection time	compact ▲

Total

used	1.5GB of 1.5GB
external	31.5KB
collections	418
average collection time	22.00 ms

New Generation				Old Generation				Total				
Internal	External	Size	Instances	Internal	External	Size	Instances	Internal	External	Size▲	Instances	Class
816B	0B	816B	36	1.5GB	0B	1.5GB	42511	1.5GB	0B	1.5GB	42547	_OneByteString
352B	0B	352B	5	5.2MB	0B	5.2MB	90	5.2MB	0B	5.2MB	95	_TwoByteString
1.6KB	0B	1.6KB	38	4.6MB	0B	4.6MB	122254	4.6MB	0B	4.6MB	122292	_List
0B	0B	0B	0	4.0MB	0B	4.0MB	8728	4.0MB	0B	4.0MB	8728	Instructions
0B	0B	0B	0	2.4MB	0B	2.4MB	26207	2.4MB	0B	2.4MB	26207	Function
0B	0B	0B	0	1.3MB	0B	1.3MB	8728	1.3MB	0B	1.3MB	8728	Code
0B	0B	0B	0	1.3MB	0B	1.3MB	8081	1.3MB	0B	1.3MB	8081	PcDescriptors
0B	0B	0B	0	1.1MB	0B	1.1MB	204	1.1MB	0B	1.1MB	204	_Uint32List
0B	0B	0B	0	1.0MB	0B	1.0MB	33982	1.0MB	0B	1.0MB	33982	ICData
0B	0B	0B	0	985.0KB	0B	985.0KB	8140	985.0KB	0B	985.0KB	8140	CodeSourceMap
0B	0B	0B	0	704.3KB	0B	704.3KB	11269	704.3KB	0B	704.3KB	11269	Field
0B	0B	0B	0	649.1KB	0B	649.1KB	5193	649.1KB	0B	649.1KB	5193	Class
0B	0B	0B	0	476.7KB	0B	476.7KB	211	476.7KB	0B	476.7KB	211	_Int16List
0B	0B	0B	0	311.5KB	0B	311.5KB	7975	311.5KB	0B	311.5KB	7975	_Type
0B	0B	0B	0	287.3KB	0B	287.3KB	946	287.3KB	0B	287.3KB	946	_Int8List

Observatory - Memory



Observatory

Practice

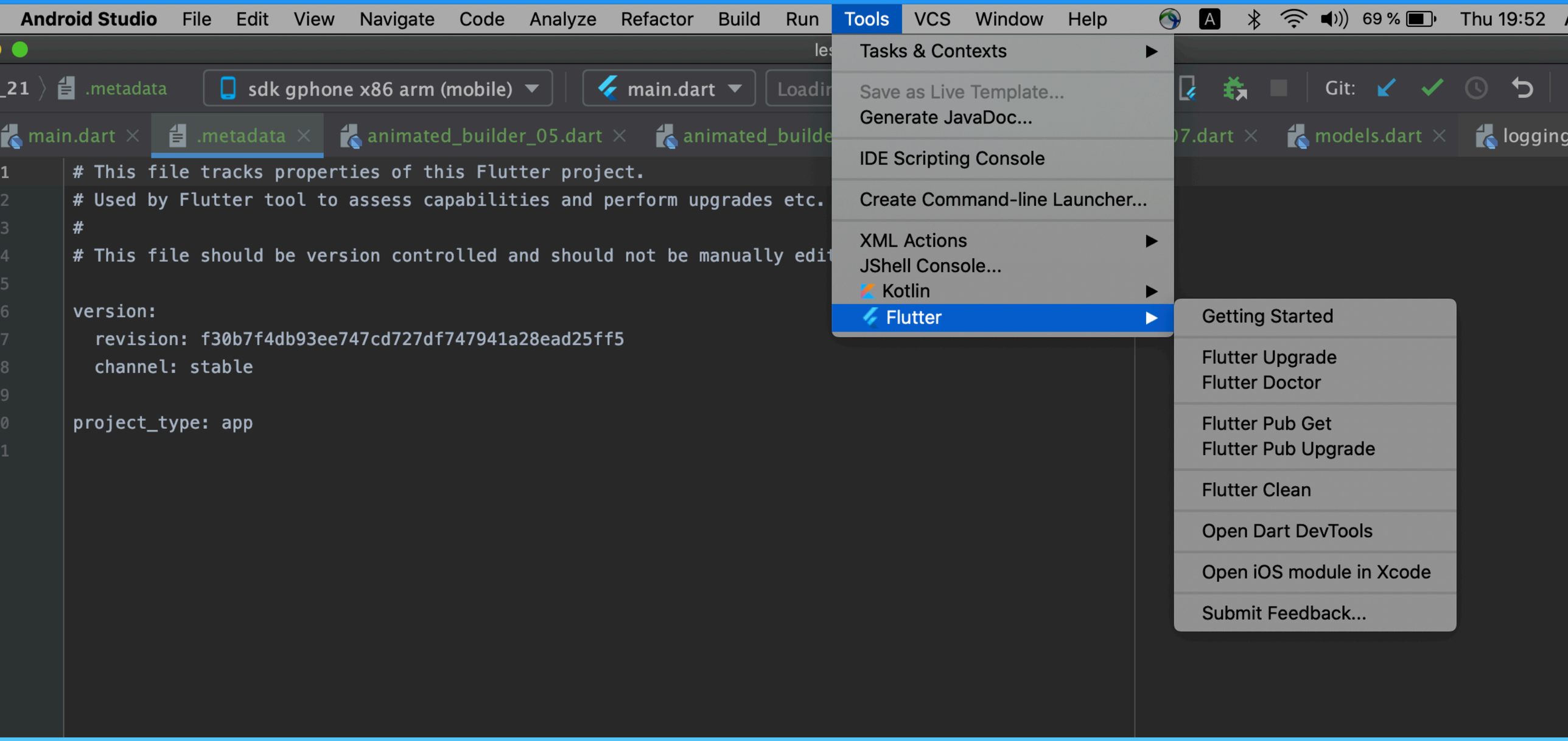
О чем будем говорить

Немного теории

Tooling и поиск проблем - Observatory

Tooling - DevTools

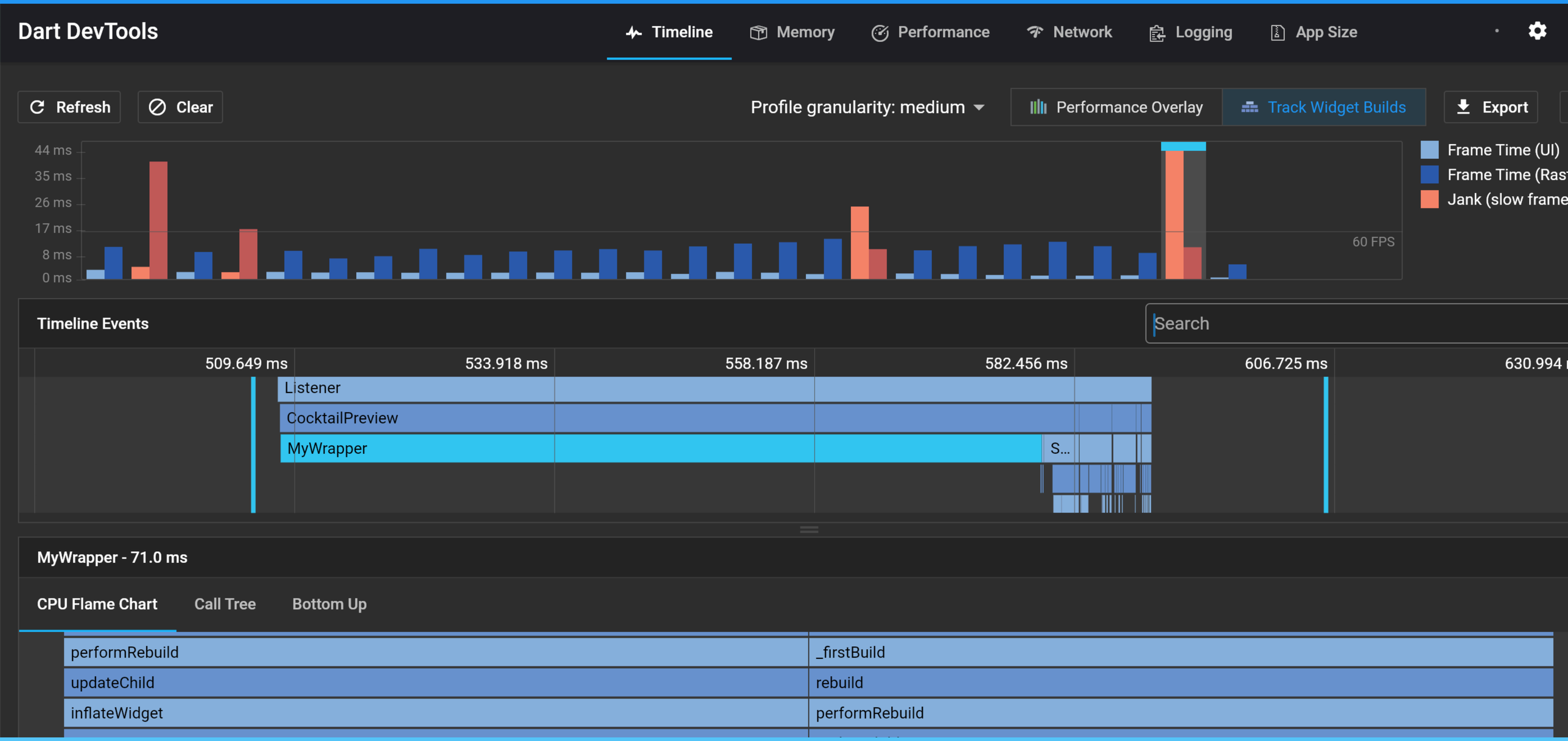
DevTools


Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
69 % Thu 19:52
sdk gphone x86 arm (mobile)
main.dart
Git:
main.dart
.metadata
animated_builder_05.dart
models.dart
This file tracks properties of this Flutter project.
Used by Flutter tool to assess capabilities and perform upgrades etc.
#
This file should be version controlled and should not be manually edit
version:
revision: f30b7f4db93ee747cd727df747941a28ead25ff5
channel: stable
project_type: app
Tasks & Contexts
Save as Live Template...
Generate JavaDoc...
IDE Scripting Console
Create Command-line Launcher...
XML Actions
JShell Console...
Kotlin
Flutter
Getting Started
Flutter Upgrade
Flutter Doctor
Flutter Pub Get
Flutter Pub Upgrade
Flutter Clean
Open Dart DevTools
Open iOS module in Xcode
Submit Feedback...

DevTools

```
❯ flutter pub global run devtools
Serving DevTools at http://127.0.0.1:9100.

Hit ctrl-c to terminate the server.
```

DevTools


Dart DevTools
Timeline
Memory
Performance
Network
Logging
App Size
Refresh
Clear
Profile granularity: medium
Performance Overlay
Track Widget Builds
Export
44 ms
35 ms
26 ms
17 ms
8 ms
0 ms
60 FPS
Frame Time (UI)
Frame Time (Ras
Jank (slow frame
Timeline Events
Search
509.649 ms
533.918 ms
558.187 ms
582.456 ms
606.725 ms
630.994
Listener
CocktailPreview
MyWrapper
S...
MyWrapper - 71.0 ms
CPU Flame Chart
Call Tree
Bottom Up
performRebuild
_firstBuild
updateChild
rebuild
inflateWidget
performRebuild

Сегодня мы рассмотрели:

- Немного теории
- Tooling и поиск проблем
- Tooling - DevTools

Рекомендации

- Старайтесь обновлять состояния виджетов небольшими порциями
- Обновляйте состояние только тогда, когда это действительно необходимо
- Не держите большие вычисления в build методе. Если есть тяжелые операции, выносите их в отдельный изолят
- Минимизируйте использование дорогих виджетов, использующих тяжелые операции с битмапами, эффектами типа Blur, Opacity, и прочие

Преждевременная оптимизация?

- Действительность такова, что на многие вопросы об оптимизации производительности есть такой ответ - «зависит от обстоятельств». Стоит ли эта конкретная оптимизация для этого конкретного виджета усилий и затрат на обслуживание? Имеет ли смысл такой подход в данной конкретной ситуации?

- Чтобы ответить на эти вопросы - необходимо тестирование и измерения. Определите количественно, какое влияние каждый выбор имеет на производительность, и примите решение на основе этих данных.

Заполните, пожалуйста,
опрос о занятии по ссылке в чате

Спасибо за внимание!
Приходите на следующие вебинары

Смирнов Андрей wrike

Курс **Мобильная разработка на Flutter**