Dart is Boring

Adrian Kosmaczewski

2022-06-16

Lately, I've been playing with Dart, the programming language powering the cross-platform Flutter UI framework. I've added a Dart implementation to my collection of (now 21) programming languages in the Conway project, and another to the collection of sample Fortune web APIs.

As a language, Dart certainly fulfills the promise of the "no surprises" mantra. I could go as far as to say that Dart is essentially boring. Don't get me wrong; I think that's a good thing. The language does what is expected to do, without frills or surprises, but sadly without much fun, either.

Conway

If you look at the program's code, there's hardly anything you can't understand immediately.

```
import 'dart:io';
import 'coord.dart';
import 'world.dart';

void clrscr() {
  print("\x1B[2J\x1B[0;0H");
}

void main() {
  var alive_cells = World.blinker(new Coord(0, 1));
  alive_cells.addAll(World.beacon(new Coord(10, 10)));
  alive_cells.addAll(World.glider(new Coord(4, 5)));
  alive_cells.addAll(World.block(new Coord(1, 10)));
  alive_cells.addAll(World.block(new Coord(18, 3)));
  alive_cells.addAll(World.tub(new Coord(6, 1)));
  var world = new World(30, alive_cells);
  var generation = 0;
  clrscr();
  while (true) {
    generation += 1;
```

```
    print("$world");
    print("Generation $generation");
    world = world.evolve();
    sleep(Duration(milliseconds: 500));
    clrscr();
  }
}
```

Dart seems to be exclusively designed for one and only one thing: to power the Flutter framework. Flutter allows developers to build applications for iOS, Android, Linux, Windows, Mac, and the web with just one codebase, and I know a few passionate developers who are excited to use it.

As expected of any Turing-complete language, people use Dart for many other things. For example, there are a few web frameworks like Conduit and Jaguar. I will talk about Conduit later in this article.

I suppose Flutter is much more interesting than Dart, to be honest. This experiment had more to do with understanding Dart as a language, to get a quick overview of its capabilities and syntax, and nothing else. In terms of verbosity, for example, Dart is more or less in the same league as Java and PHP.

The Dart developer experience was good; I used the Dart Visual Studio Code extension, which provided excellent feedback during coding.

Dart also includes lots of "modern" programming language features:

- It's open-source;
- Generics;
- Ruby-style mixins;
- Garbage collection;
- `async` and `await`;
- Functional constructs;
- Null safety;
- It has integrated unit testing;
- There's an online playground to test the language;
- It has type inference;
- Features a "batteries included" library of data structures and algorithms;
- It provides an automatic source code formatting tool;
- And it compiles code to native binaries.

All very modern indeed, as expected by a language designed by Lars Bak.

On the other hand, it does have class inheritance, variables are not immutable by default, and it requires semicolons. Boohoo. I'm kidding of course. Sense the irony.

Fortune

I also implemented the Fortune application using Dart, and I have a few observations:

- All those web API frameworks I found to create such a project lack traction and community involvement; low number of releases, not a lot of open issues, not a lot of PRs, etc. Some of them are simply discontinued. I guess Dart is not the most popular language to use in this context at the moment.
- I used Conduit, itself a fork of Aqueduct which had been archived by its owners.
- Conduit does not include a template library, so I just used the mustache_template package instead, and it worked perfectly well.

The resulting router is quite straightforward:

```
@override
Controller get entryPoint {
  final router = Router();
  final source = File("templates/fortune.html").readAsStringSync();
  final template = Template(source);
  final rng = Random();
  final hostname = Platform.localHostname;
  const version = "1.0-dart";

  router.route("/").linkFunction((request) async {
    var message = "";
    await Process.run('fortune', []).then((ProcessResult pr) {
      message = pr.stdout.toString();
    });
    final random = rng.nextInt(1000);
    final json = {
      'number': random,
      'message': message,
      'version': version,
      'hostname': hostname
    };
    final acceptHeader = request.acceptableContentTypes.first;
    if (acceptHeader.toString() == "text/plain") {
      final text = "Fortune $version cookie of the day #$random:\n\n$message";
      return Response.ok(text)..contentType = ContentType.text;
    } else if (acceptHeader.toString() == "application/json") {
      return Response.ok(json)..contentType = ContentType.json;
    }
    final html = template.renderString(json);
    return Response.ok(html)..contentType = ContentType.html;
```

```
  });

  return router;
}
```

Building the Docker container was a bit more delicate, as always; the thing is that even though Dart can compile to native binaries using the `dart compile exe bin/main.dart -o bin/main` command, in the case of Conduit this doesn't work; one must use the `conduit` CLI instead, as explained in this issue by the author.

We then use that binary directly in an Alpine-based image, with the required `fortune` package, and the final product is really small and fast to boot.

All things considered, the final score of the Dart-based container image is impressive: very small final container image size; very low memory consumption; and very fast build times. Check the results in the table on this page for details, where Dart appears now in the 4th place, behind Rust, C, and Go, and before C++.

Unfortunately, the fragmented and fragile perception I got of the Dart ecosystem would make me wary of choosing it for a production project, and for this reason, I won't change my final recommendation:

> Taking everything into account, I'd consider using **Go, C#, or TypeScript** (in that order) for such an API anytime.

Conclusion

All in all, Dart is boring, and that's its greatest feature. It does what it says it will do. It has an excellent library and the support of a major vendor like Google behind. I would love to see more activity on other projects than Flutter around Dart, I think the language has lots of potential, and I can now understand why some people are so bullish about it. But the ecosystem, compared to those of other languages like Go, C#, or TypeScript, is still quite weak.