Flutter: Eine Einführung

Cross-Platform leicht gemacht.

Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Dash (Vogel) Avatar von dashatar-dev.web.app

Kurz über mich

- Christian Krebel
- Full-Stack-Entwickler bei Smartsquare GmbH
- Vorliebe für Web- & Mobile-Entwicklung
- U.a. interessiert an PKM, Veganismus & Filmen

Inhalt

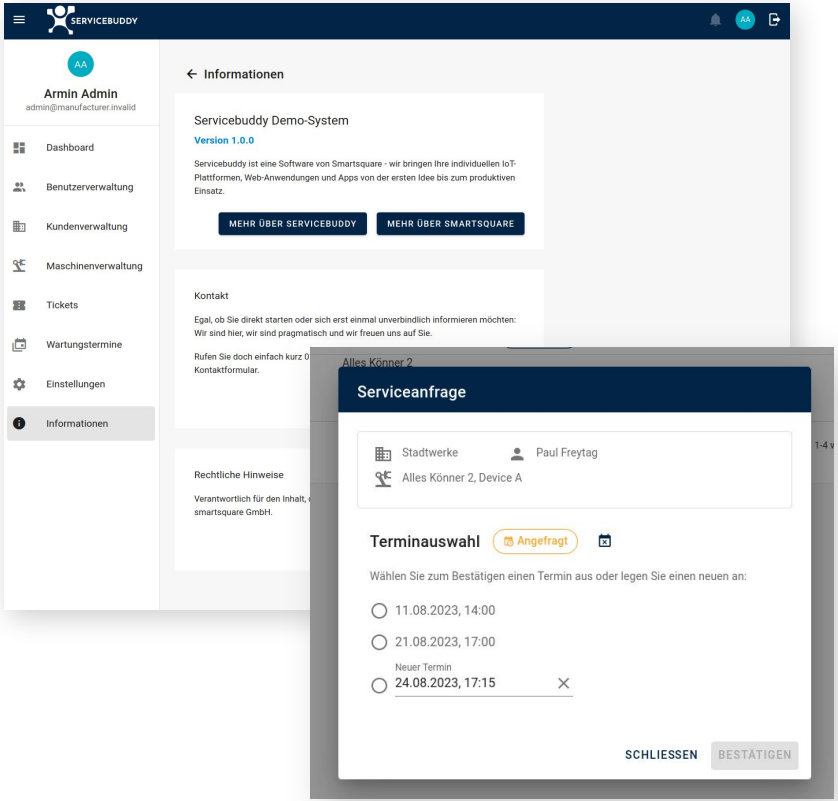
1. Story time
2. Welche App-Typen existieren?
3. Wann Cross-Platform?
4. Was ist Flutter?
5. Flutter im Detail
6. Dart im Detail
7. Beispiel im Live-Coding
8. Erkenntnisse

Story time

Companion App

Wir haben eine Vue.js Web App, jetzt brauchen wir eine Mobile App:

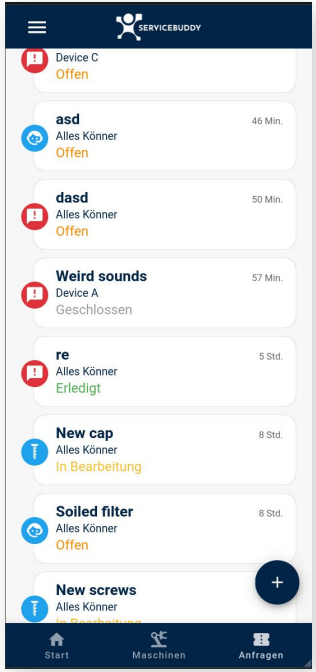
- Wenig neue Technologien
- Code wiederverwenden
- Gleiches UI



Hybrid App?

- Web App, die in Stores verfügbar ist
- Schien erst eine einfache Lösung zu sein
- Komisches Scrollverhalten
- Lange Ladezeiten
- Wirkt nicht nativ
- Fixt man einen Bug auf einer Plattform, entsteht ein neuer auf der anderen
- Wenig Code wiederverwendet
- DX verbesserungswürdig
- App schlecht testbar

Mit Capacitor implementiert



Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Quellen

Was gibt es noch?

Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

App-Typen

Native

- Apps entwickelt für eine Plattform mit den spezifischen Sprachen und APIs
- z.B. mit Swift und Objective-C für iOS und Java und Kotlin für Android
- Beste Performance und UX, da optimiert für Plattform
- Hoher Aufwand und Know-how notwendig

Cross-Platform

- Apps entwickelt mit Frameworks, die erlauben, eine Codebase auf mehreren Plattformen laufen zu lassen
- z.B. React Native, Flutter, Xamarin, KMP
- Nutzen meist native UI-Komponenten und kompilieren in nativen Code
- Schnelle Time-to-Market
- Mit wenig Aufwand nativer Look & Feel, trotzdem nicht abhängig von Plattformen

Hybrid

- Web-Apps, die in nativen Containern eingebettet sind (WebView) mithilfe von Frameworks
- z.B. Ionic mit Capacitor, Apache Cordova
- Werden entwickelt mit HTML, CSS, JavaScript
    - Meist auch mit Web-Frameworks wie React.js, Vue.js
- Bieten Schnittstellen (Plugins) und manchmal Komponenten an
- Lange Ladezeiten und wirken nicht nativ

Progressive Web App (PWA)

- Web Apps, die durch moderne Browser-APIs native Funktionen nutzen und auf Homescreens installiert werden können
- Abrufbar in Browsern auf allen Plattformen, aber nicht in Stores verfügbar
- Werden entwickelt mit HTML, CSS, JavaScript
- Offline-Funktionalität und Push-Benachrichtigungen generell möglich
- Je nach Plattform und Browser sind nur limitierte Funktionen verfügbar
- Können live aktualisiert werden
- Schnellste Time-to-Market, aber lange Ladezeiten und wirken nicht nativ

Vergleich

Kategorie	**Verhältnis**
Performance	Native > **Cross-Platform** > Hybrid > PWA
Development Time	PWA < Hybrid ≈ **Cross-Platform** < Native
User Experience	Native > **Cross-Platform** > Hybrid ≈ PWA
Platform Reach	PWA > Hybrid ≈ **Cross-Platform** > Native

⟹ Cross-Platform in der goldenen Mitte

Wann Cross-Platform?

Pro

- Nur eine Codebase
  - Weniger Entwicklungs- und Wartungskosten
  - Schnelle Time-to-Market
  - Gut für Prototypen, da schnelle Entwicklung
- Konsistente UX auf allen Plattformen
- Große Benutzerreichweite
- Future-proof: Anpassungen und Erweiterungen an neue Plattformen einfacher

Contra

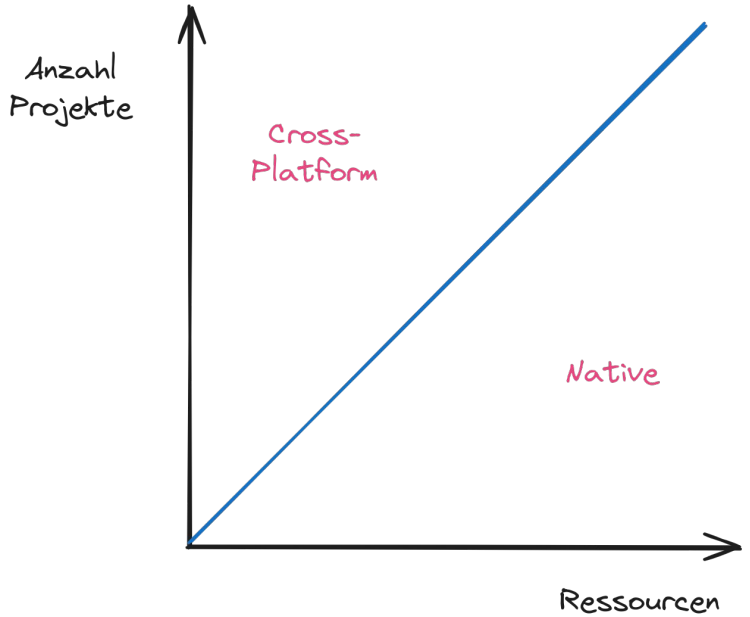
- Performance & UX: Native Apps immer noch an der Spitze
  - Wobei Flutter bei Animationen mithalten kann
- Gewisse plattformspezifische Funktionen schwierig zu implementieren
- Framework-Abhängigkeit
- Initiale Kosten: Neues Framework (und neue Sprache) lernen

Je nach Fall

Wie wichtig sind …

- Performance
- UX
- Time-to-Market
- Kosten
- Wartbarkeit
- Reichweite (Stores vs. Browser)

… ?



Was ist Flutter?

Flutter

- UI-SDK für Cross-Platform-Apps
- 2017 von Google veröffentlicht
- Open-source (New BSD Lizenz)
- In C, C++ und Dart implementiert
- Zielplattformen: Web, Android, iOS, macOS, Linux, Windows, Fuchsia

- Flutter Apps werden mit Dart entwickelt
- Anders als Native oder React Native: eigene Render-Engine
  - Gibt Pixeldaten zurück statt Plattform-Komponenten zu nutzen

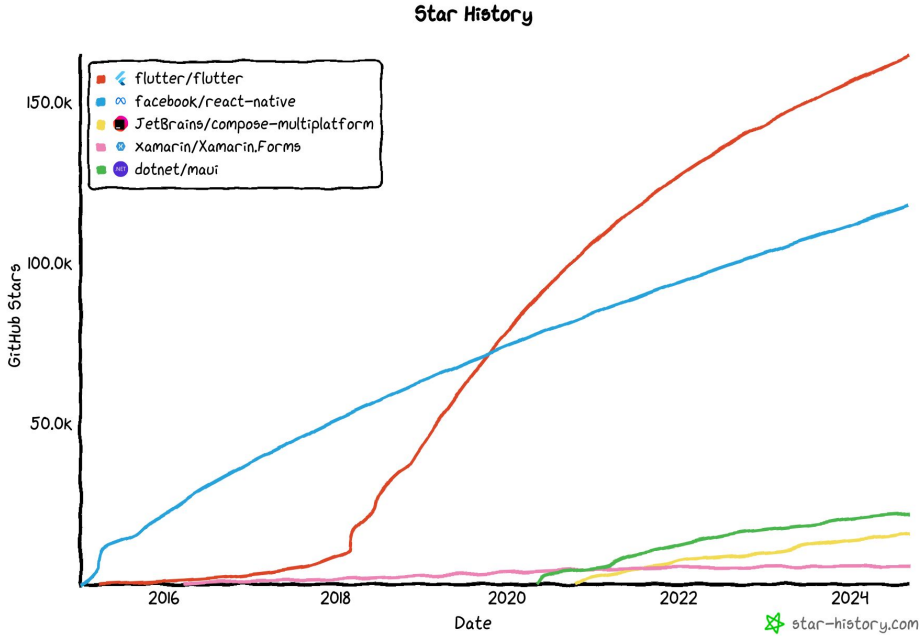
Noch mehr Flutter

- Google Material Design & Apple Cupertino direkt verfügbar
- Viele Kompilierungs- und Deployment-Optionen
  - Für Releases immer Ahead-of-time
  - Außer für Web: transpiliert zu JS oder WebAssembly
- Kann in bestehende Apps und Systeme eingebettet werden (Embedder API)
- Optimiert um auf alten Geräten laufen zu können: *“powered by hardware-accelerated 2D graphics libraries”*
- Stateful Hot-reload für schnelles Entwickeln und DX

Out of the box

- Paketmanager (pub.dev)
- Konfigurierbarer Linter & Code Analyzer
- Testframework
  - Unittest
  - Komponententests (Widget)
  - Integrationstests

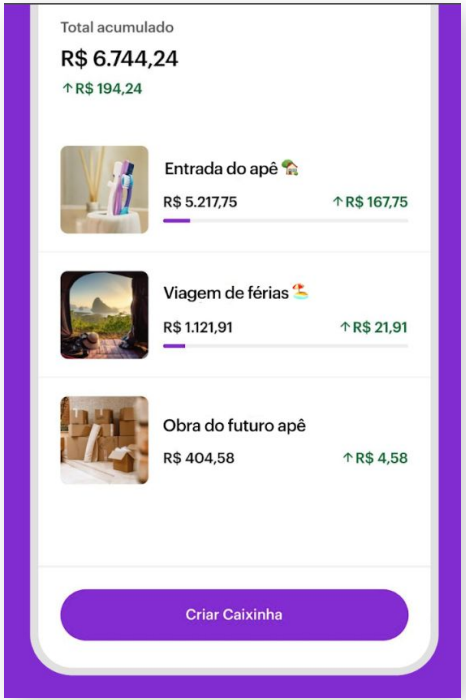
Beliebt


Star History
flutter/flutter
facebook/react-native
JetBrains/compose-multiplatform
xamarin/Xamarin.Forms
dotnet/maui
150.0k
100.0k
50.0k
GitHub Stars
2016
2018
2020
2022
2024
Date
star-history.com

Flutter in freier Wildbahn

Nubank

- Neobank aus Südamerika
- Eigenes Server-Driven-UI-Framework basierend auf Flutter und Clojure



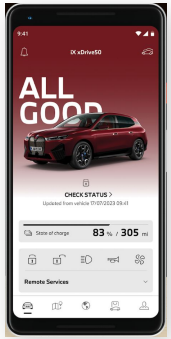
Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

[13], [14], [15] **Quellen**

Die Automobilindustrie setzt auf Flutter

BMW

- Companion App

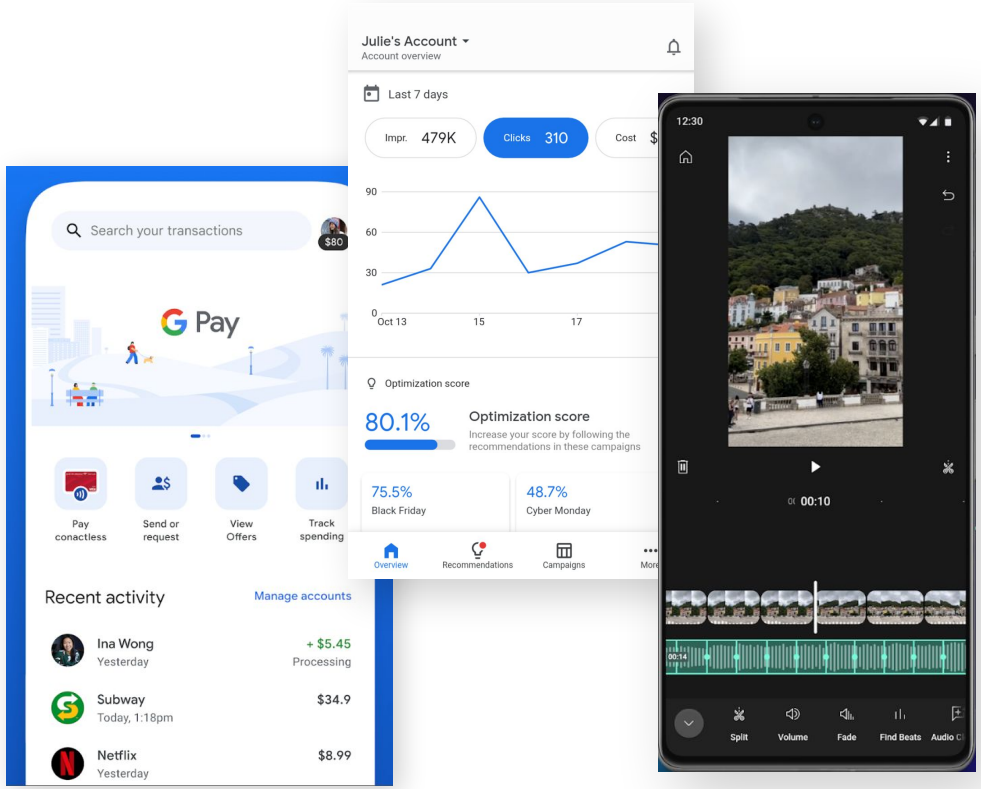


Toyota

- Infotainmentsystem

Google natürlich auch

- z.B. Google Pay (Indien)
- Google Ads
- YouTube Create



Und viele weitere …

Alibaba Group - Bauhaus - Burger King Finland - ByteDance

Delivery Hero - eBay - MediaMarktSaturn - MOIA (VW Group)

Philips hue - PUBG Mobile - Sonos - Tencent - ubuntu

Universal Studios - Wolt - Xiaomi

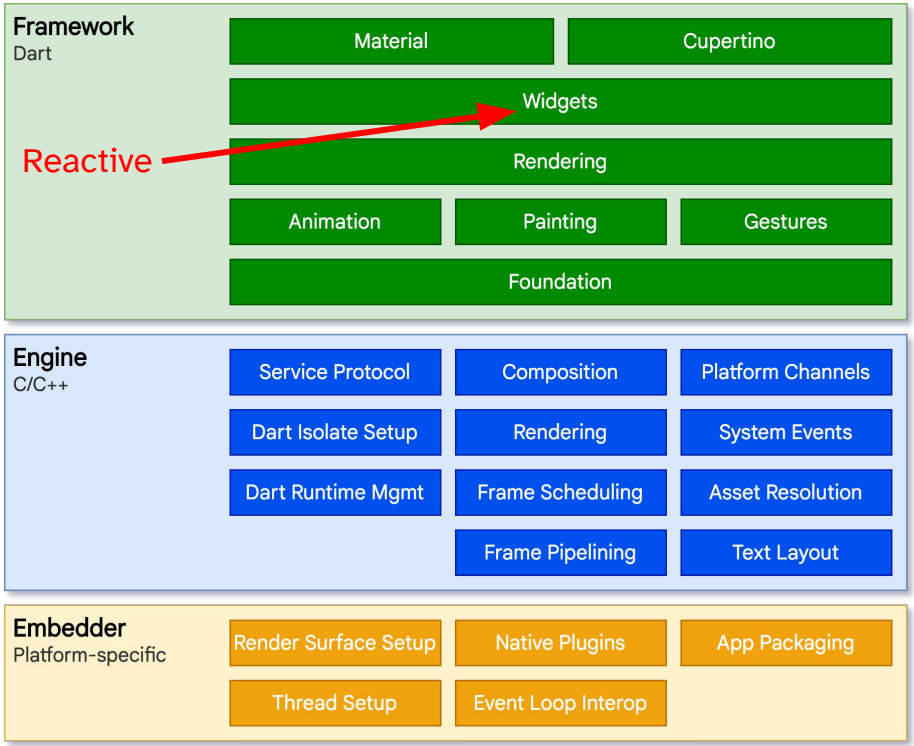
Flutter im Detail

Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Layers, Layers, Layers

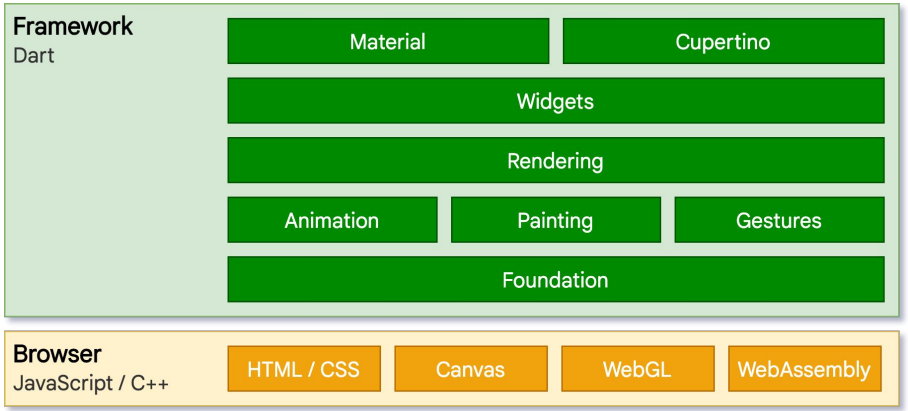
3 Layers, die austauschbar sind

- Embedder: Koordiniert Plattform-Services
- Engine: Wrappt Basisfunktionen wie Rendering-Engines & Input als Dart-Package
- Framework: Schnittstelle zum Entwickler (Layouting, Gesten, Widget-tree usw.)



Web

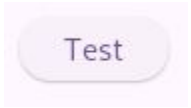
- Keine Dart runtime, Dart kompiliert zu JS
- Engine reimplementiert mit Browser-API
- 2 Modes
  - HTML, CSS, Canvas, SVG
  - WebGL, WebAssembly
- Flutter im Web ist nur bedingt zu empfehlen
  - Wirkt nicht nativ, unperformant, kein SEO



Alles ist ein Widget

- Widget-Baum ähnlich zum DOM-Baum im Web
- Je nach Plattform kann ein anderes angezeigt werden
  - Kann aufwendig sein
  - Das Package `Flutter Platform Widgets` verpackt beide automatisch

```
if (Platform.isAndroid) {
 return ElevatedButton(onPressed: onPressed, child: child);
} else if (Platform.isIOS) {
 return CupertinoButton.filled(onPressed: onPressed, child: child);
}
```



Material

Cupertino

Die 2 Widget-Typen

Stateless

Stateful

```
class StatelessBtn extends StatelessWidget {
 const StatelessBtn({super.key, required this.onPressed,
this.text = 'Test'});

 final VoidCallback onPressed;
 final String text;

 @override
 Widget build(BuildContext context) {
   return TextButton(
     onPressed: onPressed,
     child: Text(text),
   );
 }
}
```

Die 2 Widget-Typen

Stateless

Stateful

```dart
class StatefulBtn extends StatefulWidget {
 const StatefulBtn({super.key, required
this.text});
 final String text;

 @override
 State<StatefulBtn> createState() ⇒
_StatefulBtnState();
}
```

```dart
class _StatefulBtnState extends State<StatefulBtn> {
 int _counter = 0;

 void _increment() {
   setState(() {
     _counter++;
   });
 }

 @override
 Widget build(BuildContext context) {
   return TextButton(
     onPressed: _increment,
     child: Text('${widget.text}: $_counter'),
   );
 }
}
```

Die 2 Widget-Typen

- Achtung: Kann schnell zu Wrapper-Hell werden
    - Viele kleine Komponentenklassen statt weitere `build`-Funktionen
- Nur was in `setState()` gesetzt wird, triggert einen Repaint

Dart im Detail

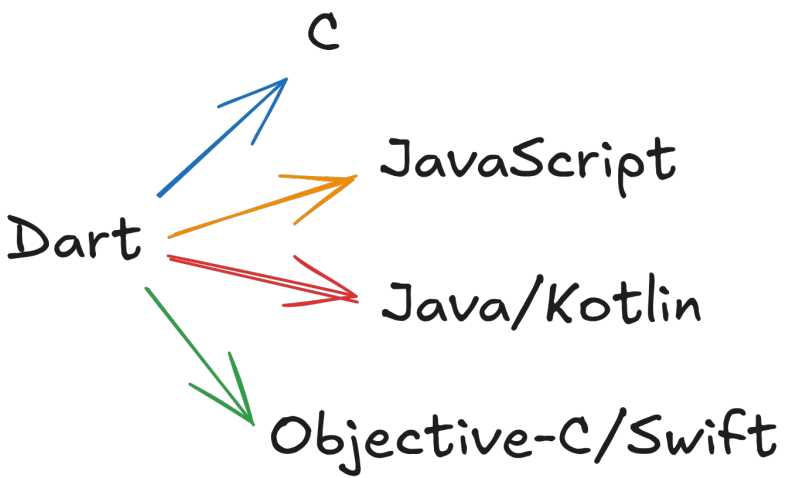
Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Dart

- Programmiersprache von Google
- 2013 veröffentlicht
- Sollte erst JS “ersetzen”
- C-Style Syntax, ähnlich zu Kotlin, Swift, TS
- Klassenbasiert & objektorientiert
- Optionale Type Safety (seit Dart 2)
- Sound Null Safety (seit Dart 3)

Noch mehr Dart

- Interoperabilität
- Rufe in Dart Code aus anderen Sprachen auf
- z.B. in einem Flutter Package, das Funktionen nativ implementiert



Coole Syntax

Iteriere über Streams

```
var server = await HttpServer.bind(....);

// HttpServer implements Stream<HttpRequest>
await server.forEach((HttpRequest request) {
 request.response.write('Hello, world!');
 request.response.close();
});
```

Cascade Notation

```
final element = Element.div();
final style = element.style;
style.width = '50%';
style.height = '4em';
append(element);
```

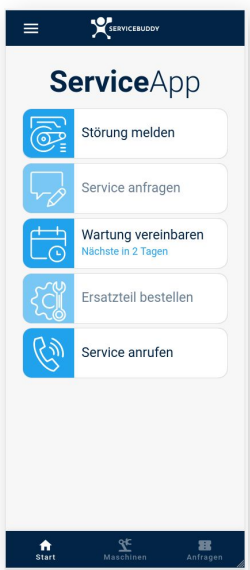


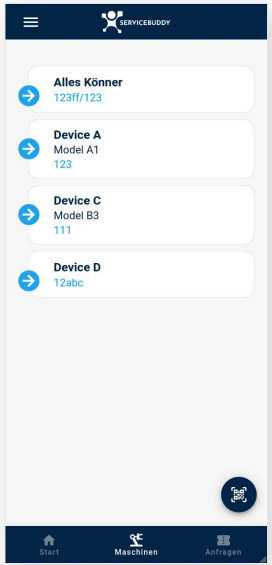
```
append(Element.div()
..style.width = '50%'
..style.height = '4em');
```

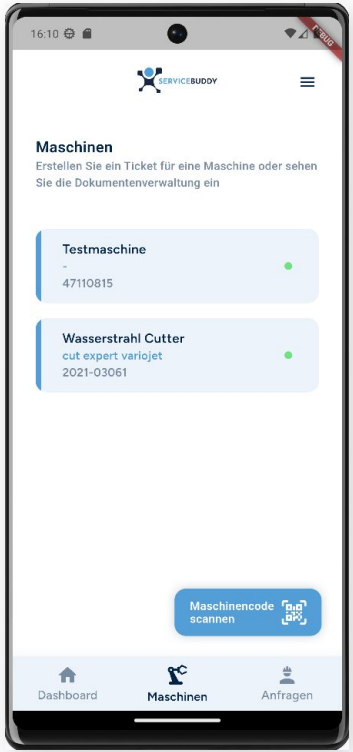
Migration time

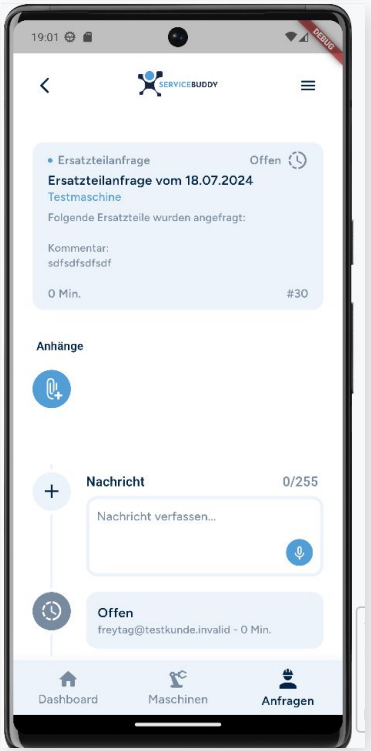
Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Migration zu Flutter



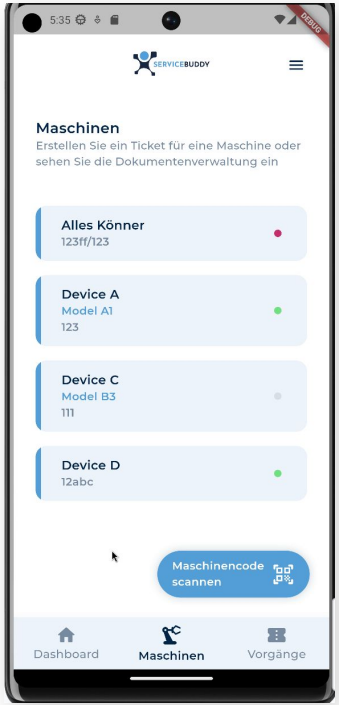






Neue App

- ✅ In Stores verfügbar
- ✅ Gutes Scrollverhalten
- ✅ Kurze Ladezeiten
- ✅ Wirkt nativ
- 🆗 Plattformspezifische Bugs reduziert
- 🆗 Kein Code aus Web App wiederverwendet
- ✅ Gute DX
- ✅ App einfach testbar



Live-Coding

Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Erkenntnisse

Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Erkenntnisse

- Cross-Platform als “goldene Mitte”, Hybrid Apps nicht optimal
- Flutter ist beliebt und wird breit eingesetzt
- Flutter Apps sind leicht zu entwickeln, performant und können universal deployed werden
- Flutter im Web ist noch nicht ausgereift
- Flutter Komponenten müssen klein gehalten werden → Wrapper-Hell

Dankeschön!

Kontakt @ChristianKrebel / ChristianKrebel.de / krebel@smartsquare.de

Dash (Vogel) Avatar von dashatar-dev.web.app

Quellen

1. https://blog.emb.global/native-app-vs-hybrid-app-vs-pwa/
2. https://reactnative.dev
3. https://flutter.dev
4. https://ionic.io/build
5. https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps
6. https://daily.dev/blog/pwa-vs-native-apps-vs-hybrid-comparison
7. https://eleks.com/types-of-software-development/the-advantages-of-cross-platform-mobile-app-development/#:~:text=Cross-platform%20mobile,multiple%20platforms.
8. https://blog.felgo.com/cross-platform-app-development/advantages#:~:text=With%20cross-platform,proof%20option.
9. https://en.wikipedia.org/wiki/Flutter_(software)
10. https://star-history.com/#flutter/flutter&facebook/react-native&JetBrains/compose-multiplatform&xamarin/Xamarin.Forms&dotnet/maui&Date
11. https://github.com/flutter/engine/blob/main/docs/Custom-Flutter-Engine-Embedders.md

Quellen

12. https://github.com/flutter/flutter
13. https://leancode.co/blog/list-of-enterprise-companies-using-flutter
14. https://building.nubank.com.br/server-driven-ui-framework-at-nubank/
15. https://play.google.com/store/apps/details?id=com.nu.production
16. https://kodytechnolab.com/blog/build-flutter-app-for-automotive-industry-like-bmw-toyota/
17. https://abaltatech.com/blog/2023/09/flutter-for-automotive/
18. https://play.google.com/store/apps/details?id=de.bmw.connected.mobile20.row&hl=en
19. https://pressroom.toyota.com/toyotas-all-new-audio-multimedia-system-is-here-and-it-is-a-game-changer/
20. https://flutter.dev/showcase/google-pay
21. https://play.google.com/store/apps/details?id=com.google.android.apps.nbu.paisa.user
22. https://play.google.com/store/apps/details?id=com.google.android.apps.youtube.producer&hl=en
23. https://play.google.com/store/apps/details?id=com.google.android.apps.adwords&hl=en

Quellen

24. https://docs.flutter.dev/resources/architectural-overview
25. https://docs.flutter.dev/cookbook/testing
26. https://dart.dev
27. https://en.wikipedia.org/wiki/Dart_(programming_language)
28. https://pub.dev/packages/flutter_platform_widgets
29. https://www.smartsquare.de/servicebuddy-app/
30. https://renato.athaydes.com/posts/interesting-dart-features.html
31. https://api.dart.dev/stable/3.5.3/dart-io/HttpServer-class.html

Icons from https://icones.js.org (MaterialSymbols, SimpleIcons)