



life.augmented

Mobile apps & Internet of Things



Agenda

1 IoT Components

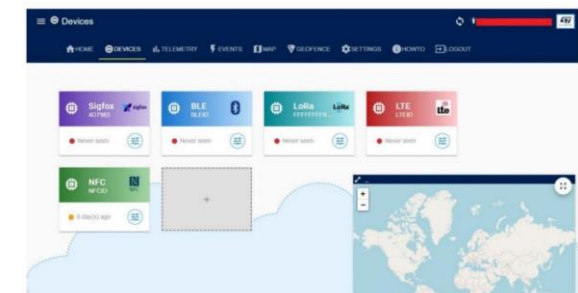
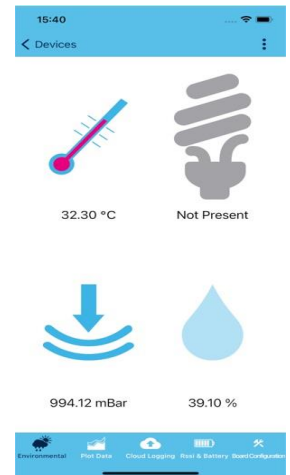
2 Mobile Platform

3 IoT App Architecture

4 Demos : Mobile App and
Web Application

IoT Components

- Hardware comprises of computation, sensing and connectivity at board level
- Mobile App on iOS / Android Platform for small sets of data / alarm
 - The purpose is consumption of data
- Web Application to monitor large sets of data and do some deep / complex computation of data
 - The purpose is both consumption and computation of data



Mobile applications



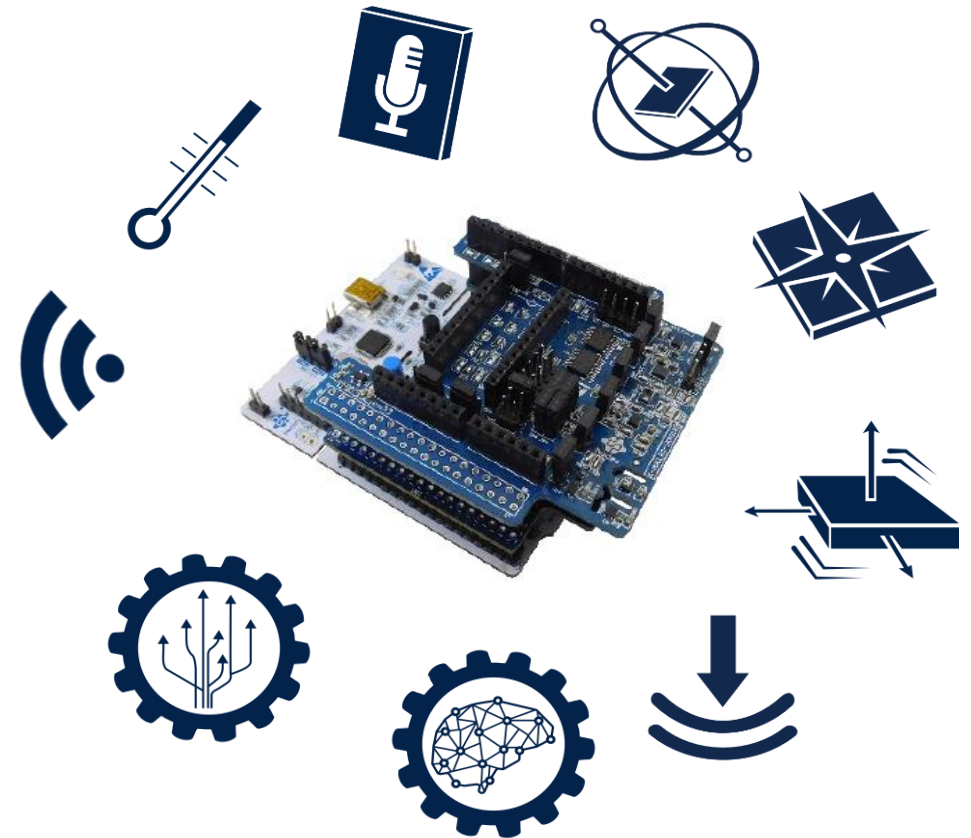
Wi-Fi

Bluetooth

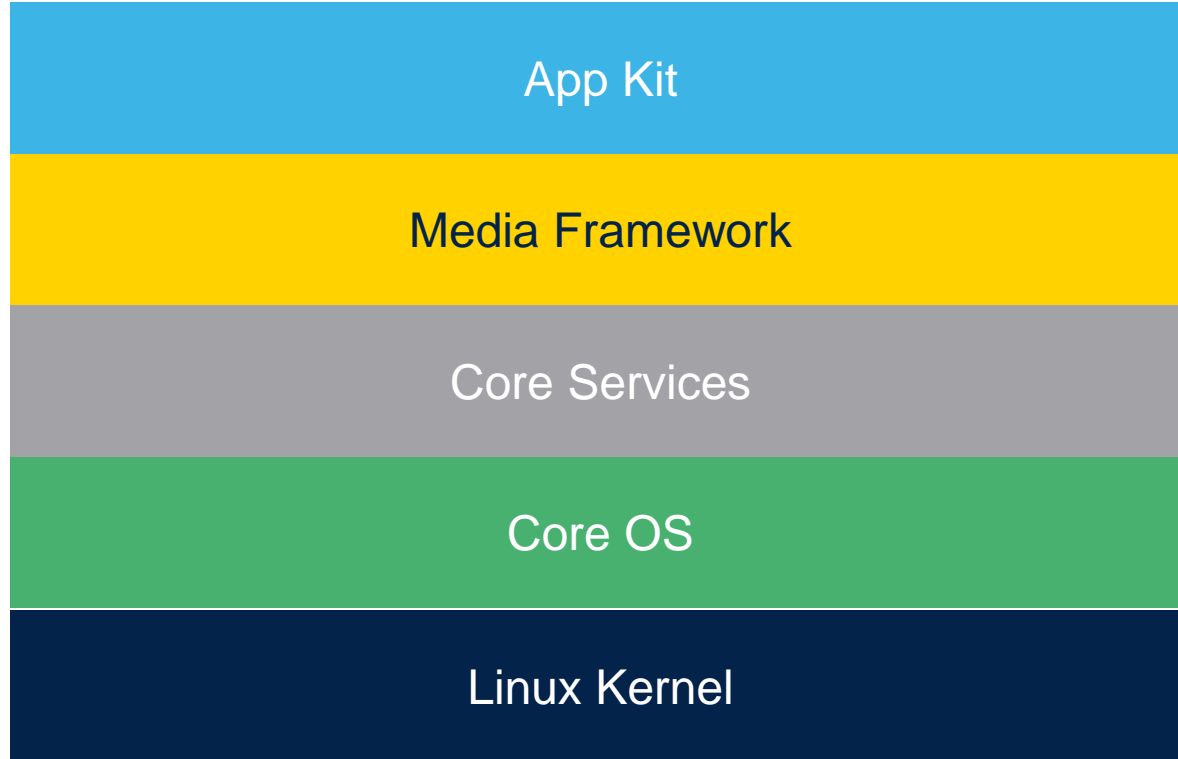
NFC

GPS

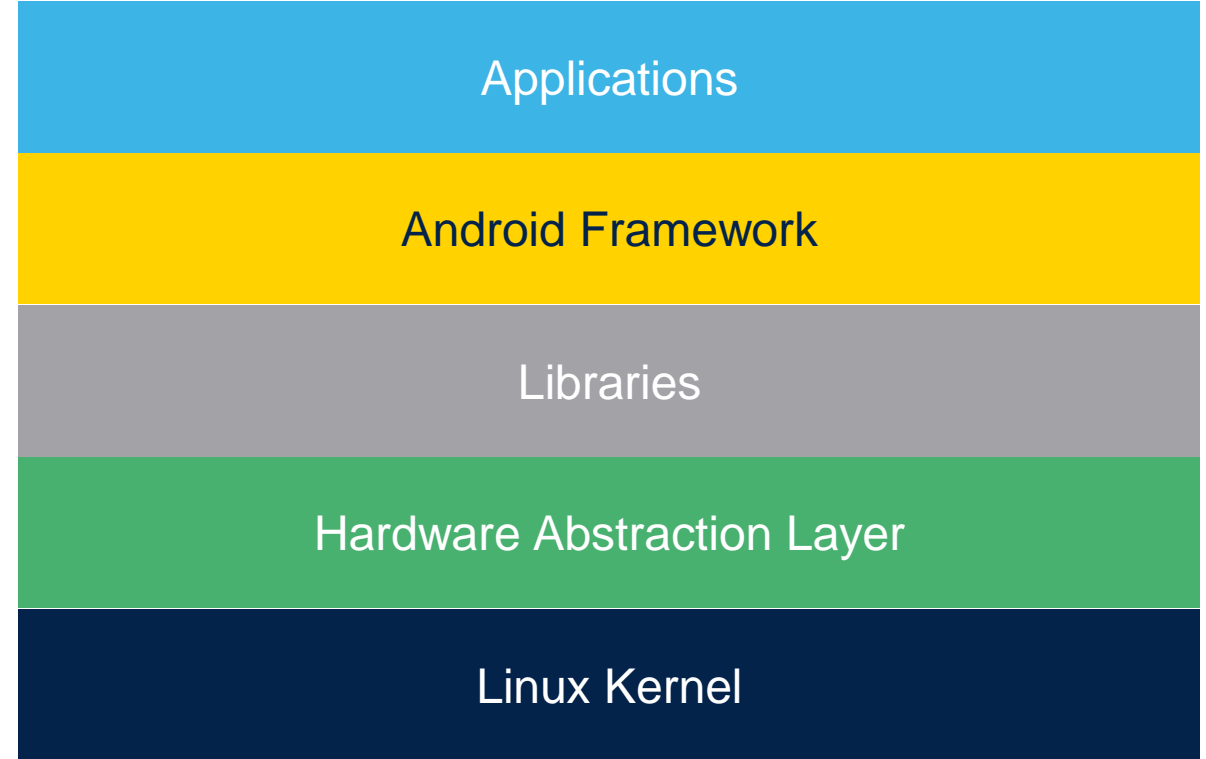
Sensors



Mobile OS architecture



iOS



Android

Hardware support in Mobile Phones

- Near Field Communication
- Bluetooth
- Motion Sensors
- Ambient Light Sensors
- Touchscreen
- GPS
- Fingerprint detection
-




Activity Recognition

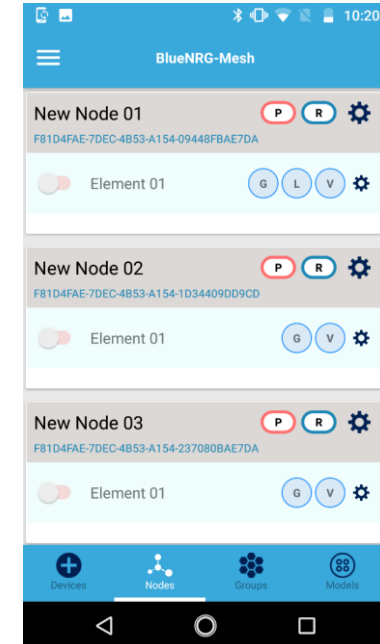



Vertical context detection
Stair count / Man fall down
Pose & Fitness monitoring

Vertical Detection



Altitude monitoring
Indoor V position for E911
Ambient pressure monitor



developers

PlatformAndroid StudioGoogle PlayJetpackKotlinDocsGames

Search

DOCUMENTATION

OverviewGuidesReferenceSamplesDesign & Quality

On-device search

Games

Tools

Customize or port game engines

Use prebuilt or turnkey game engines

Game best practices

Optimize

Distribute

Best practices

Testing

Performance

Accessibility

Privacy

Security

Android Developers > Docs > Guides

Rate and review

Near field communication overview

Near Field Communication (NFC) is a set of short-range wireless technologies, typically requiring a distance of less than 10 centimeters to initiate a connection. NFC allows you to share small payloads of data between an NFC tag and an Android-powered device, or between two Android-powered devices.

Tags can range in complexity. Simple tags offer just read and write semantics, sometimes with one-time-programmable areas to make the card read-only. More complex tags offer math operations, and have cryptographic hardware to authenticate access to a sector. The most sophisticated tags contain operating environments, allowing complex interactions with code executing on the tag. The data stored in the tag can also be written in a variety of formats; many of the Android framework APIs are based around a [NFC Forum](#) standard called NDEF (NFC Data Exchange Format).

Android-powered devices with NFC simultaneously support three main modes of operation:

1. **Reader/writer mode**, allowing the NFC device to read and/or write passive NFC tags and stickers.

DOCUMENTATION

OverviewGuidesReferenceSamplesDesign & Quality

App Actions

Slices

On-device search

Games

Tools

Customize or port game engines

Use prebuilt or turnkey game engines

Game best practices

Optimize

Distribute

Best practices

Testing

Android Developers > Docs > Guides

Rate and review

Bluetooth overview

The Android platform includes support for the Bluetooth network stack, which allows a device to wirelessly exchange data with other Bluetooth devices. The app framework provides access to the Bluetooth functionality through Bluetooth APIs. These APIs let apps connect to other Bluetooth devices, enabling point-to-point and multipoint wireless features.

Using the Bluetooth APIs, an app can perform the following:

- Scan for other Bluetooth devices.
- Query the local Bluetooth adapter for paired Bluetooth devices.
- Establish RFCOMM channels.

[Android Developers](#)

Software APIs support

Documentation > Bundle Resources > Entitlements > Near Field Communi... Language: Swift API Changes: Show

Property List Key

Near Field Communication Tag Reader Session Formats Entitlement

The Near Field Communication data formats an app can read.

Details

Key
com.apple.developer.nfc.readersession.formats

Type
Array of strings

Availability

iOS 11.0+

Framework

Core NFC

On This Page

[Details](#)
[PossibleValues](#)
[Discussion](#)
[See Also](#)

Possible Values

Framework

Core Bluetooth

Communicate with Bluetooth low energy and BR/EDR ("Classic") Devices.

Availability

iOS 5.0+

macOS 10.10+

Mac Catalyst 13.0+

tvOS 9.0+

watchOS 4.0+

On This Page

[Overview](#)
[Topics](#)
[See Also](#)

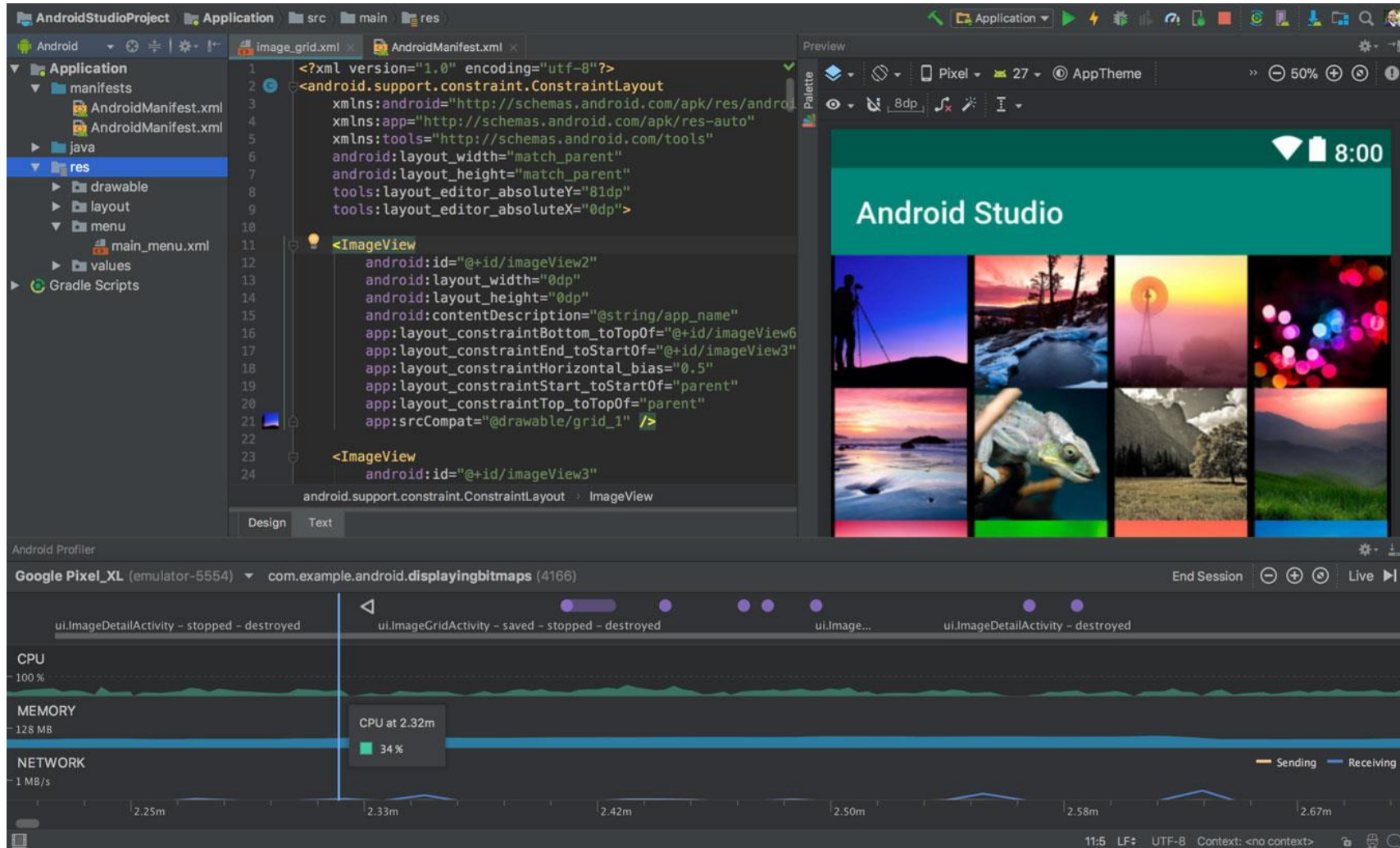
Overview

The Core Bluetooth framework provides the classes needed for your apps to communicate with Bluetooth-equipped low energy (LE) and Basic Rate / Enhanced Data Rate (BR/EDR) wireless technology.

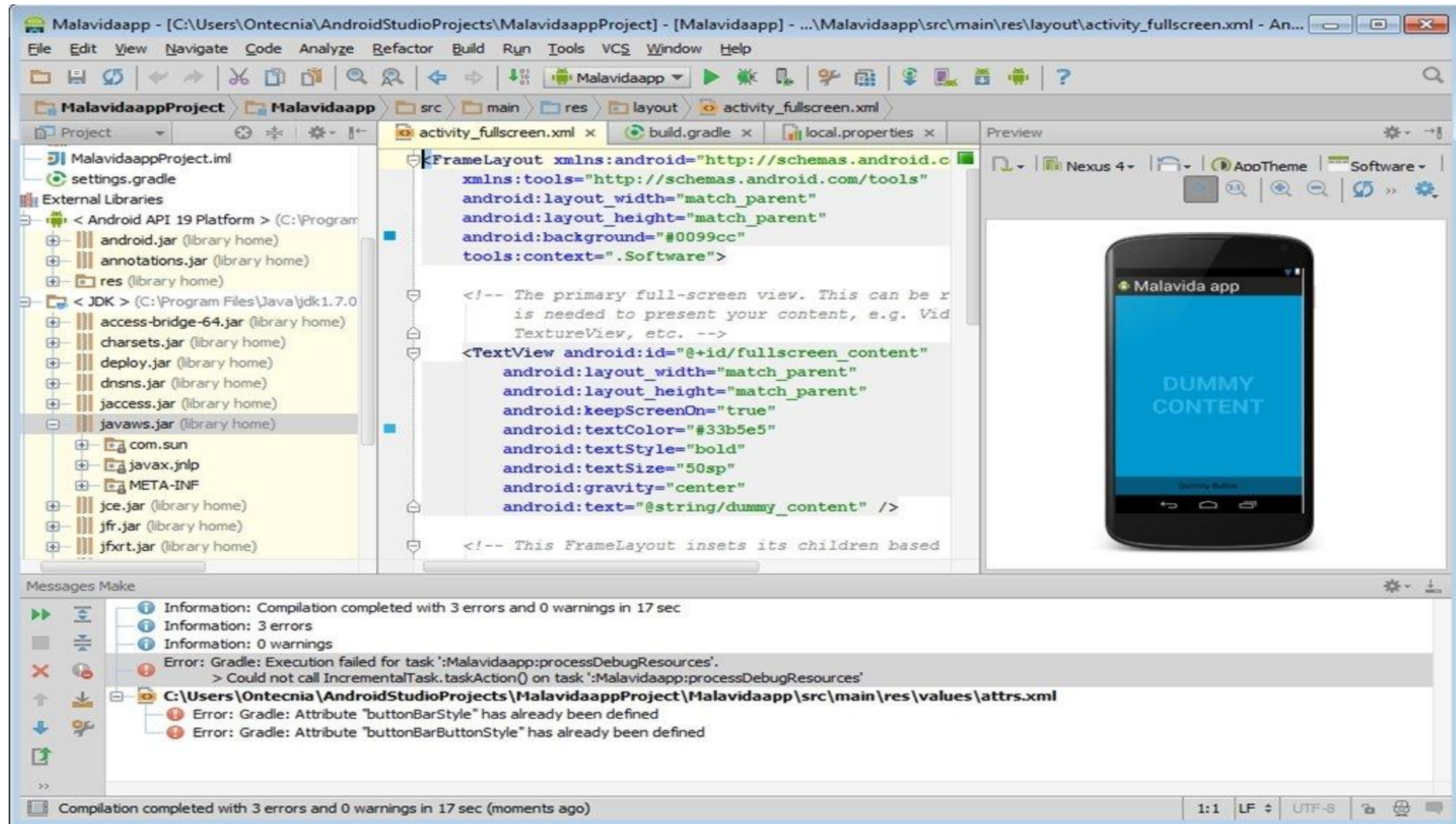
Don't subclass any of the classes of the Core Bluetooth framework. Overriding these classes isn't supported and results in undefined behavior.

Core Bluetooth background execution modes aren't supported in iPad apps running on macOS.

[Apple Developer](#)

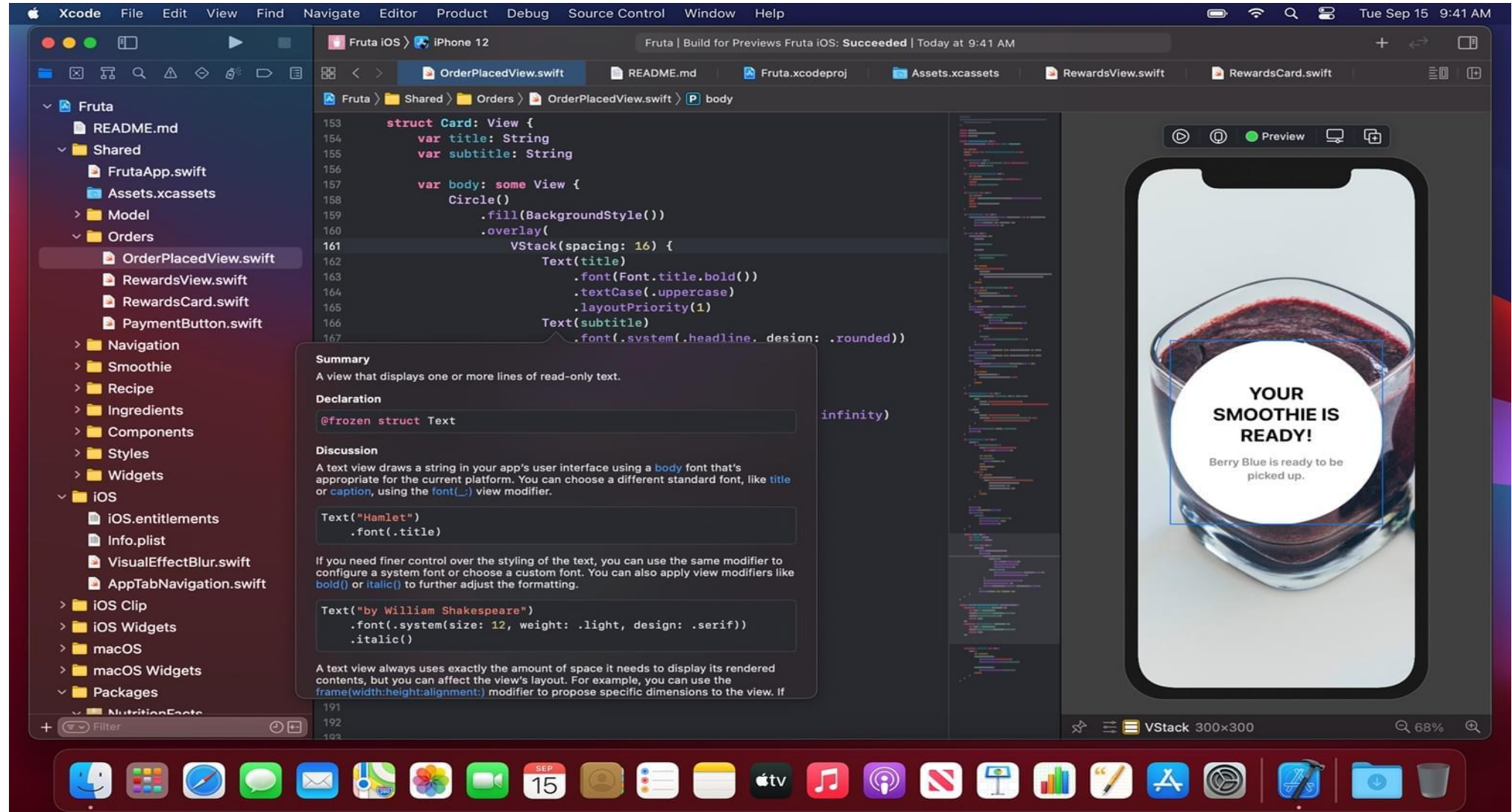


Android Studio (2/2)

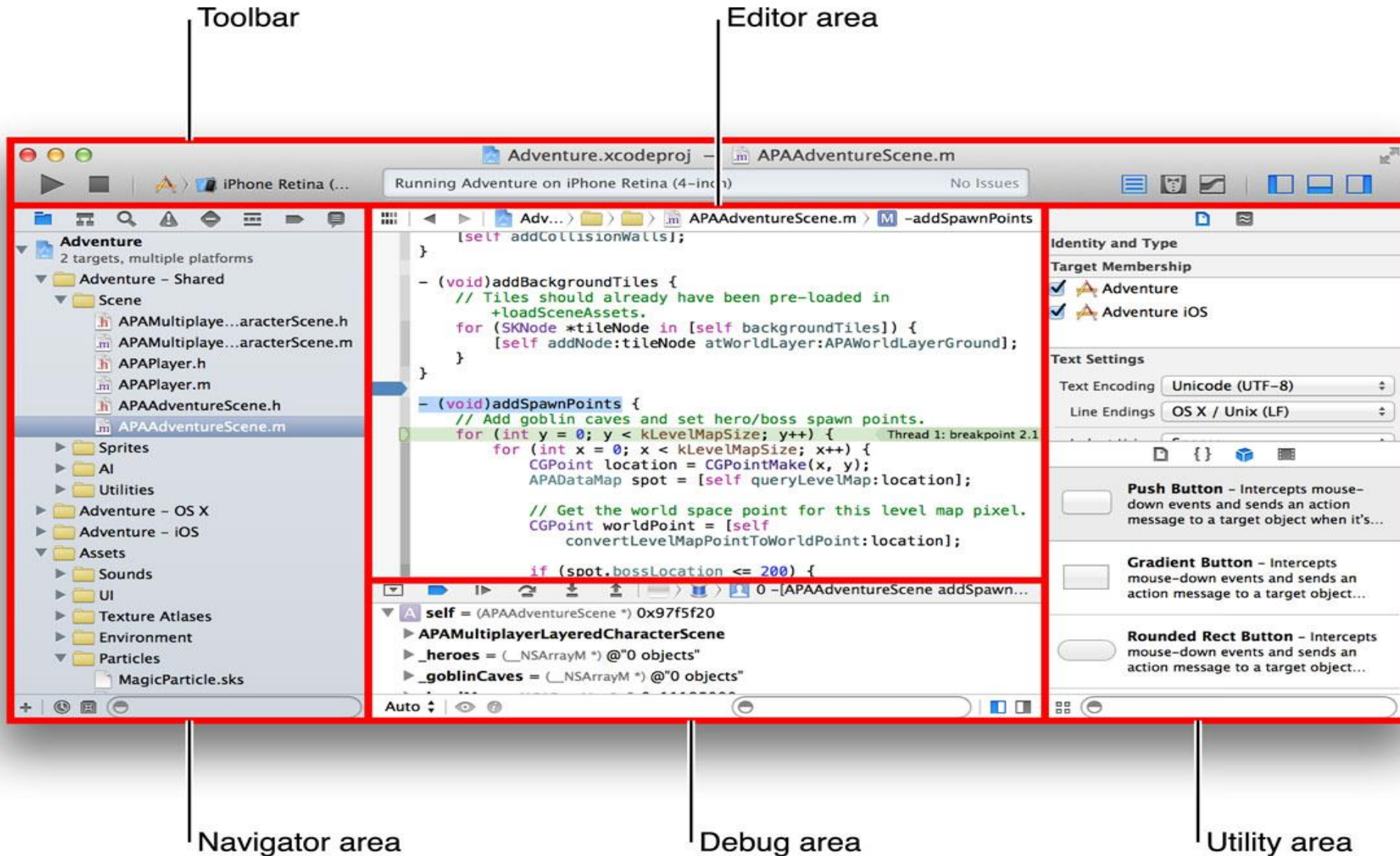




Xcode IDE (1/2)

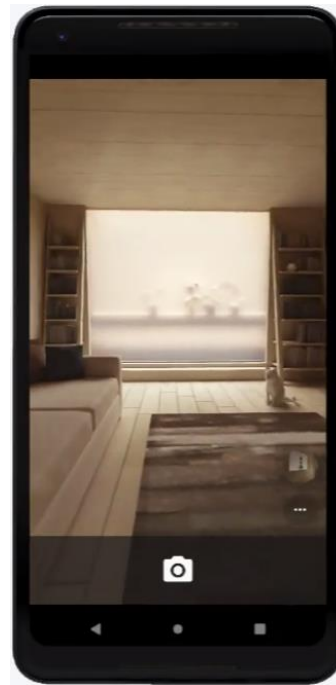
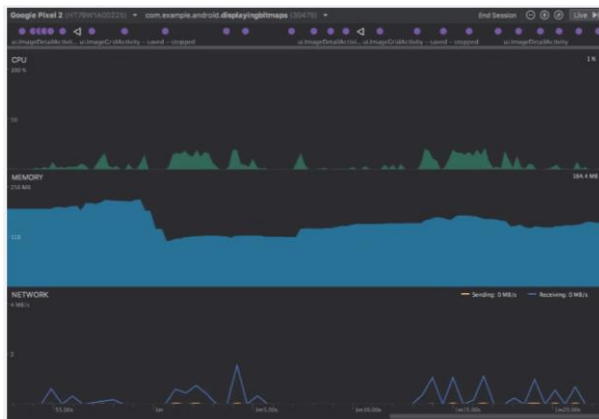


Xcode IDE (2/2)

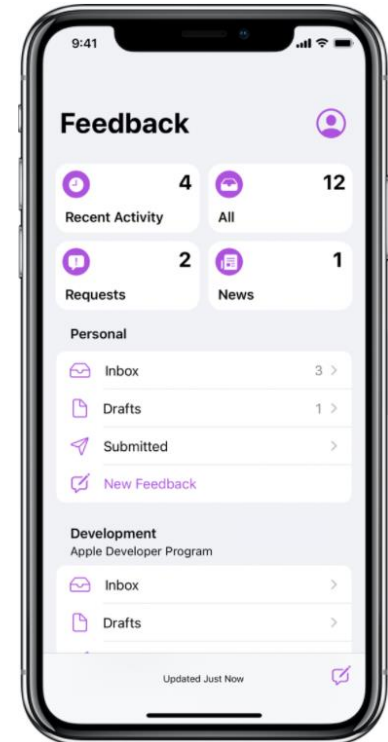
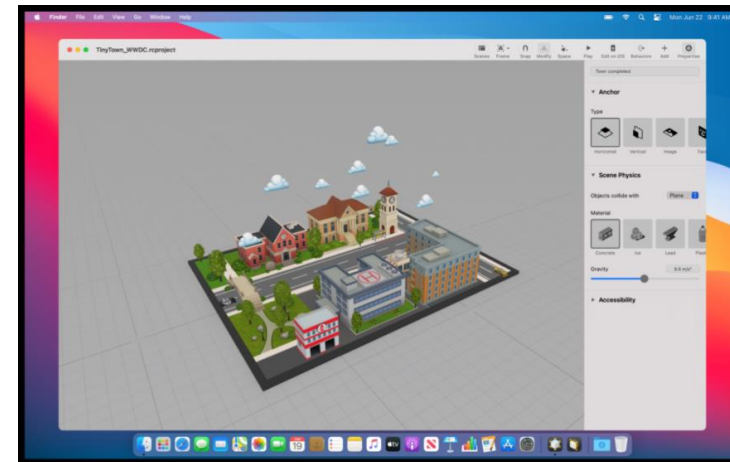


IDE features

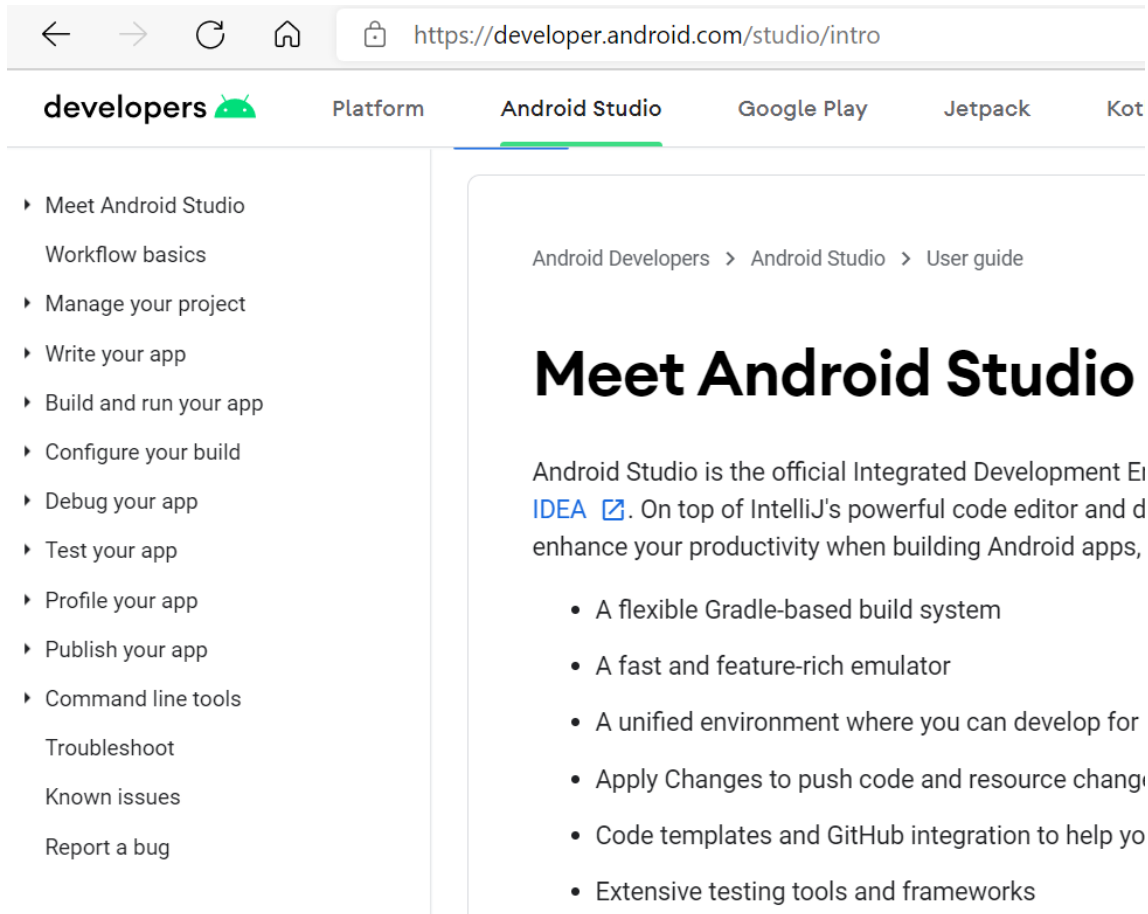
- Android Studio
 - APK Analyzer
 - Fast Emulator
 - Intelligent Code Builder
 - Flexible Build Systems
 - Visual Layout Editor
 - Realtime Profilers



- iOS
 - CloudKit Console
 - Command Line Tools
 - Dashboard
 - Reality Composer
 - Reality Builder
 - Swift UI



Documentation for each platform



A screenshot of the Android Studio documentation page on the developer.android.com website. The browser address bar shows the URL https://developer.android.com/studio/intro. The page has a navigation bar with links for Platform, Android Studio (which is highlighted), Google Play, Jetpack, and Kotlin. On the left side, there is a sidebar menu with various topics like 'Meet Android Studio', 'Workflow basics', 'Manage your project', etc. The main content area is titled 'Meet Android Studio' and includes a sub-header 'Android Developers > Android Studio > User guide'. The text describes Android Studio as the official Integrated Development Environment (IDE) on top of IntelliJ's powerful code editor. It lists several features: a flexible Gradle-based build system, a fast and feature-rich emulator, a unified environment for developing for multiple platforms, the ability to apply changes to push code and resource changes, code templates and GitHub integration, and extensive testing tools and frameworks.

← → ↻ 🏠 🔒 https://developer.android.com/studio/intro

developers 🤖 Platform **Android Studio** Google Play Jetpack Kotlin

▶ Meet Android Studio
Workflow basics

▶ Manage your project

▶ Write your app

▶ Build and run your app

▶ Configure your build

▶ Debug your app

▶ Test your app

▶ Profile your app

▶ Publish your app

▶ Command line tools

Troubleshoot

Known issues

Report a bug

Android Developers > Android Studio > User guide

Meet Android Studio

Android Studio is the official Integrated Development Environment (IDE) on top of IntelliJ's powerful code editor and d enhance your productivity when building Android apps,

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for
- Apply Changes to push code and resource change
- Code templates and GitHub integration to help yo
- Extensive testing tools and frameworks

Develop

Xcode

Swift

Swift Playgrounds

TestFlight

Documentation

Videos

Downloads

Distribute

Developer Program

App Store

App Review

Mac Software

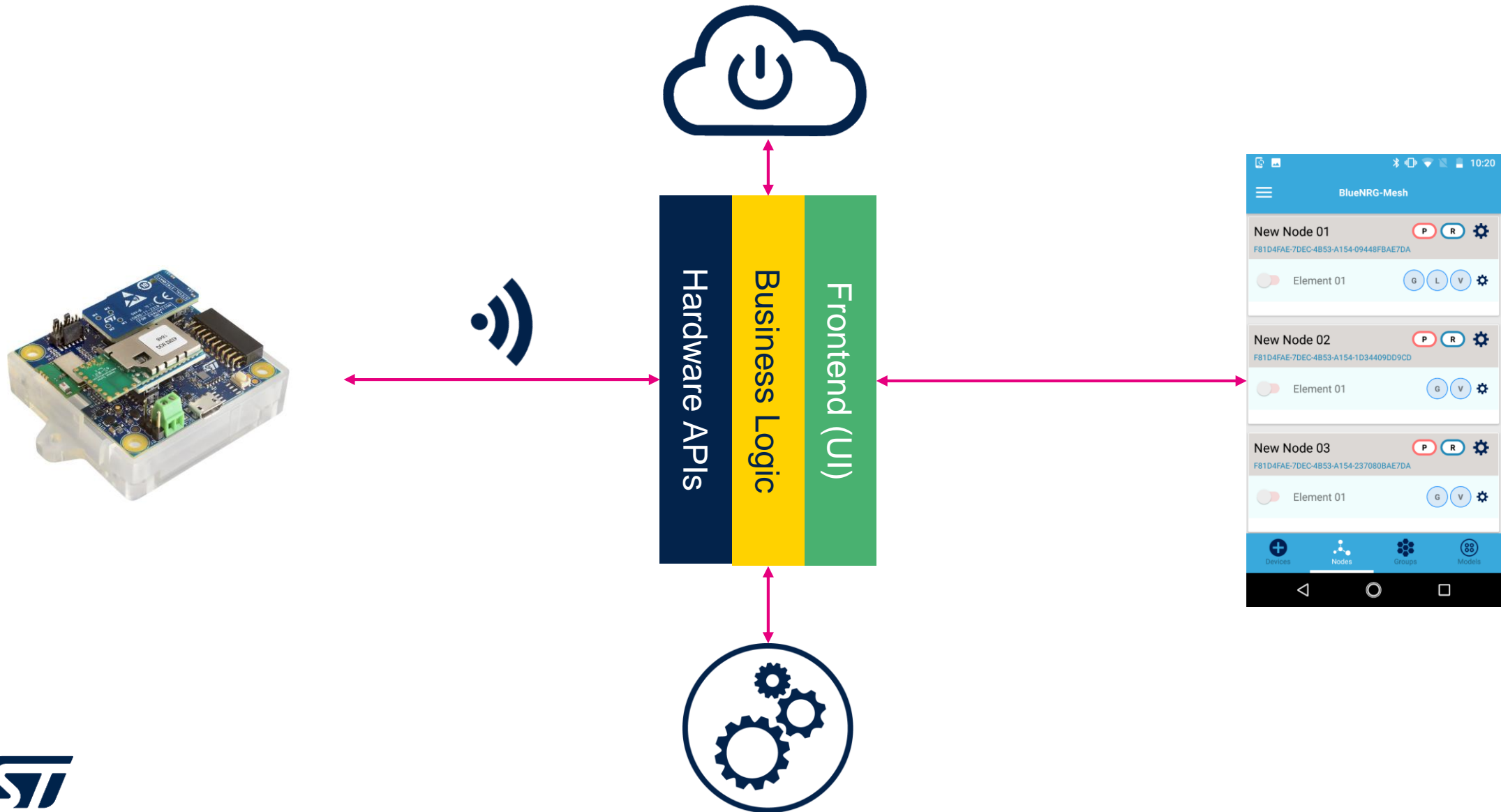
Apps for Business

Safari Extensions

Marketing Resources

Trademark Licensing

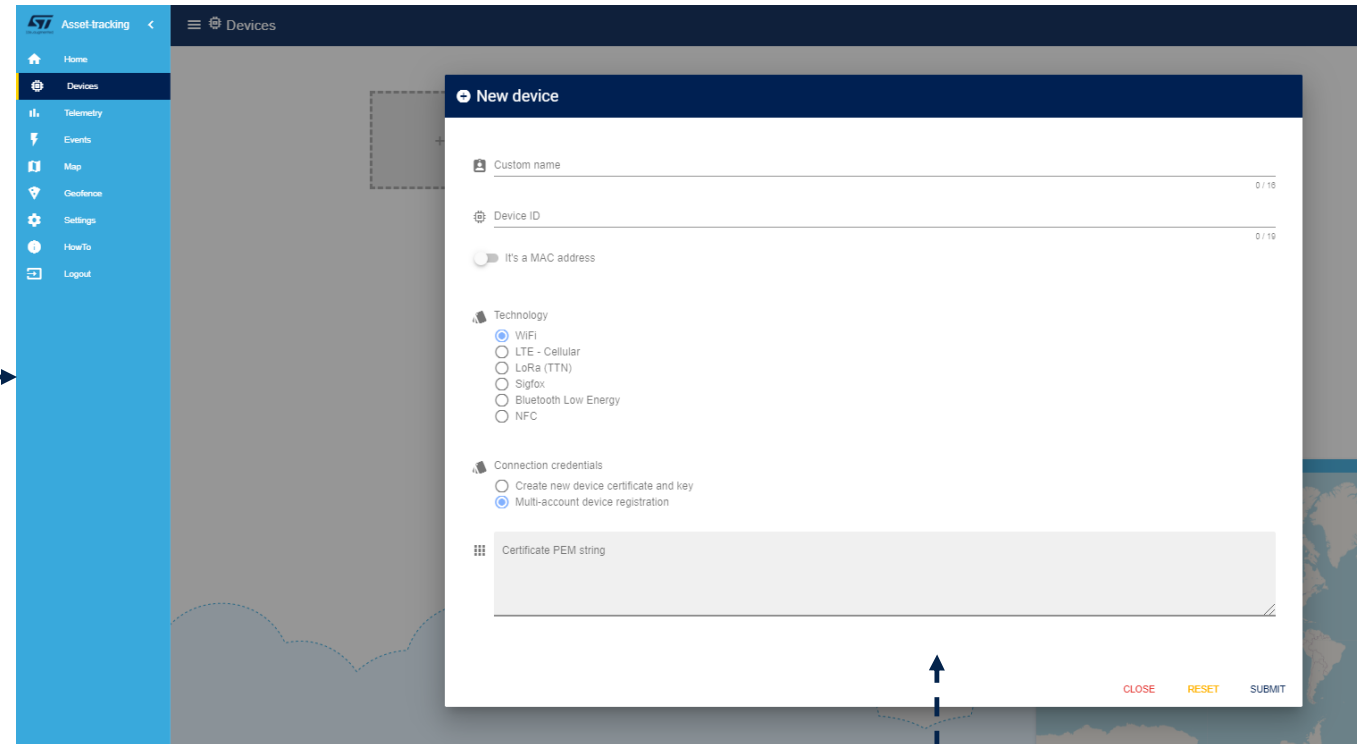
Typical mobile app architecture



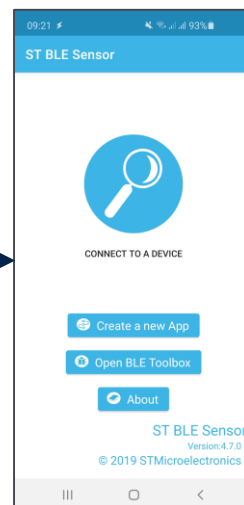
IoT data to cloud



Wi-Fi / Internet








BLE



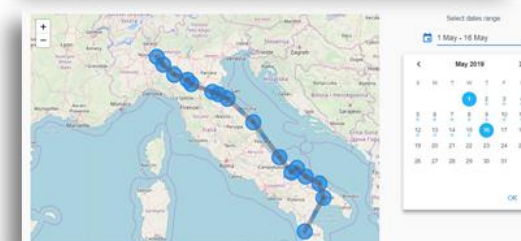
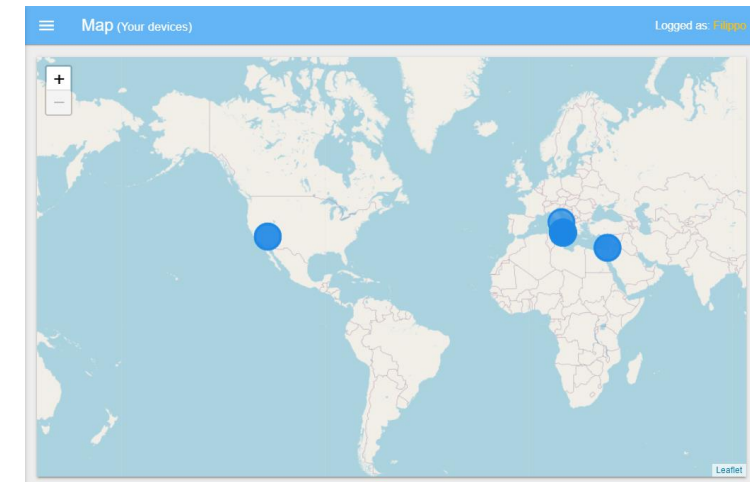
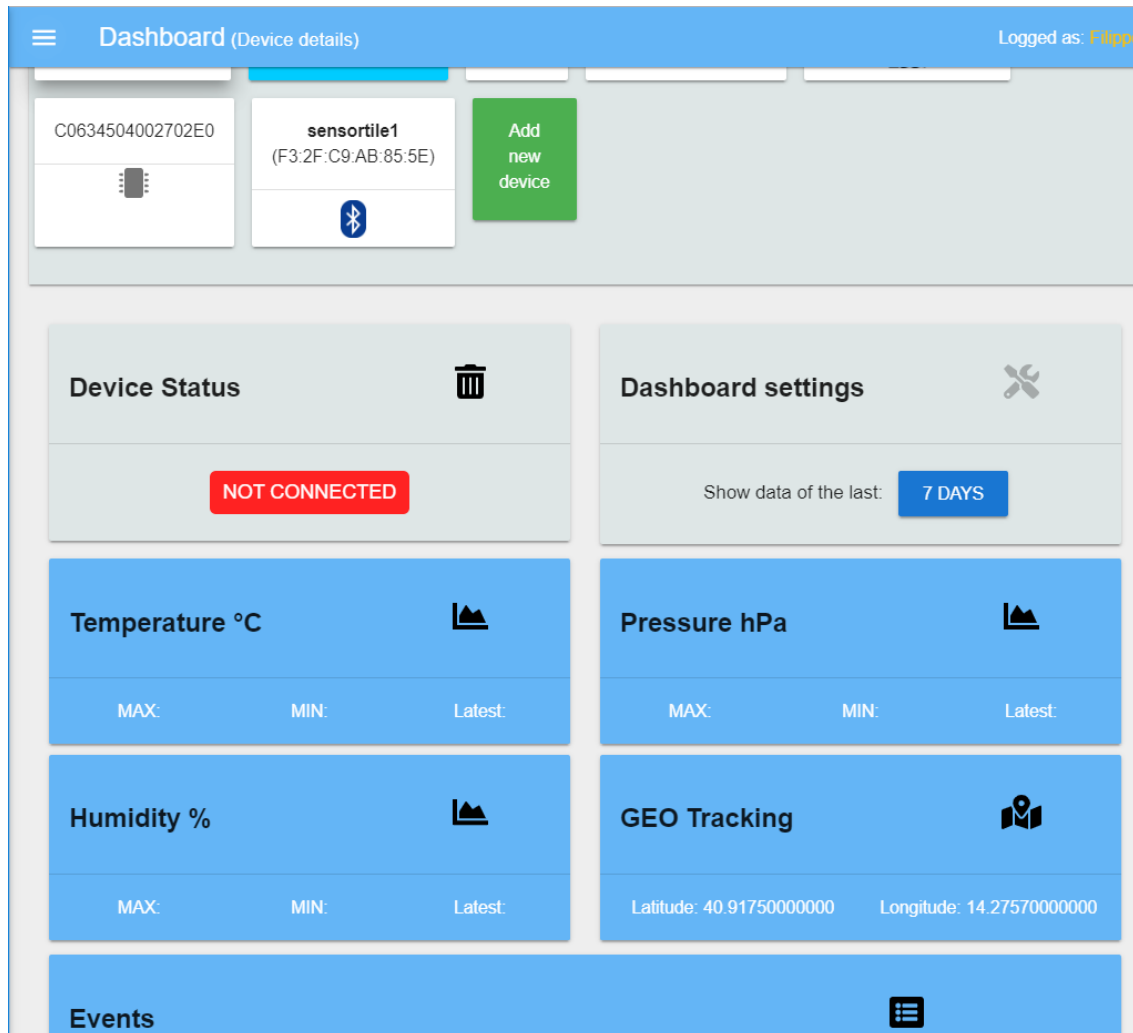
Mobile app to Cloud

ST asset tracking dashboard

The screenshot displays the 'ASSET-TRACKING PLATFORM' dashboard with the subtitle 'Dashboard and Device Management'. The interface features a dark blue header with a 'Home' link and an ST logo. The main content area is divided into five numbered steps, each with a colored header bar, an icon, a description, and a 'GO' button. The background of the dashboard is a satellite image of a port area with shipping containers.

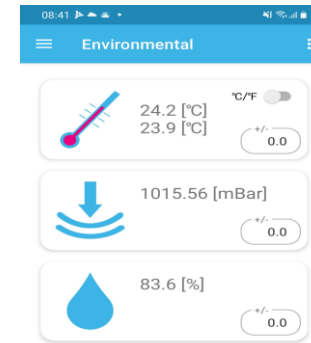
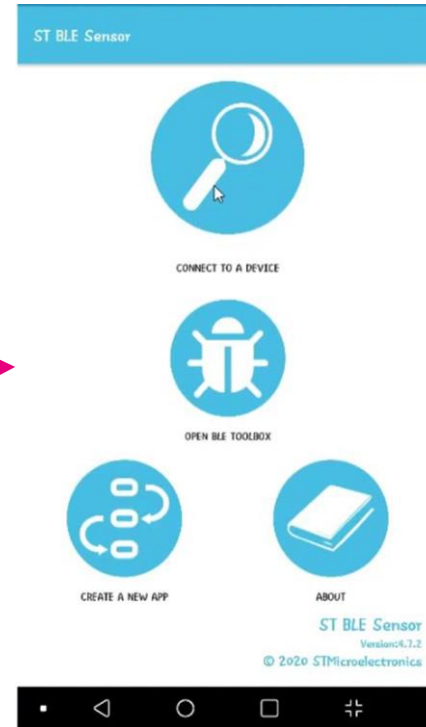
- 1. Register and configure your devices**
Follow this [guide](#) to register a new device based on its connectivity capability.

- 2. Look at your devices telemetry**
Select one or more device and the window time interested in, and then visualize telemetry data received by devices.

- 3. Analyze the events detected by your devices**
Select one or more device and the window time interested in, and then visualize events received by devices.

- 4. Monitor your device on their geo localization**
Select one or more device and the window time interested in, and then visualize geo-position data received by devices.

- 5. Set and detect geofencing events**
Select one or more device and draw an area on map in order to enable tracking of geofence events raised by devices.


ST asset tracking dashboard



Demos

ST BLE Sensor App



Gateway Demo



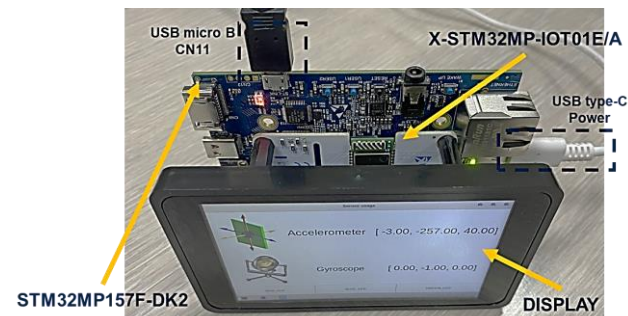
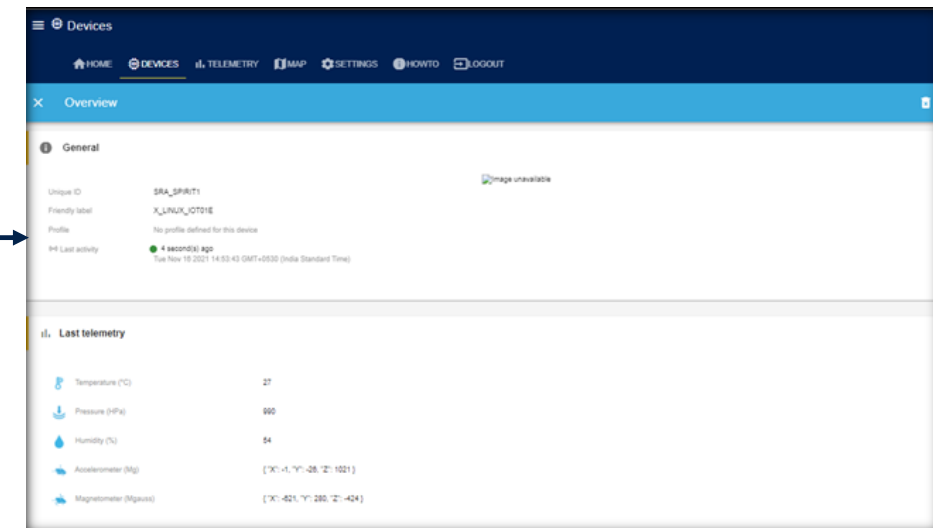
Sensor Node

Sub1-GHz

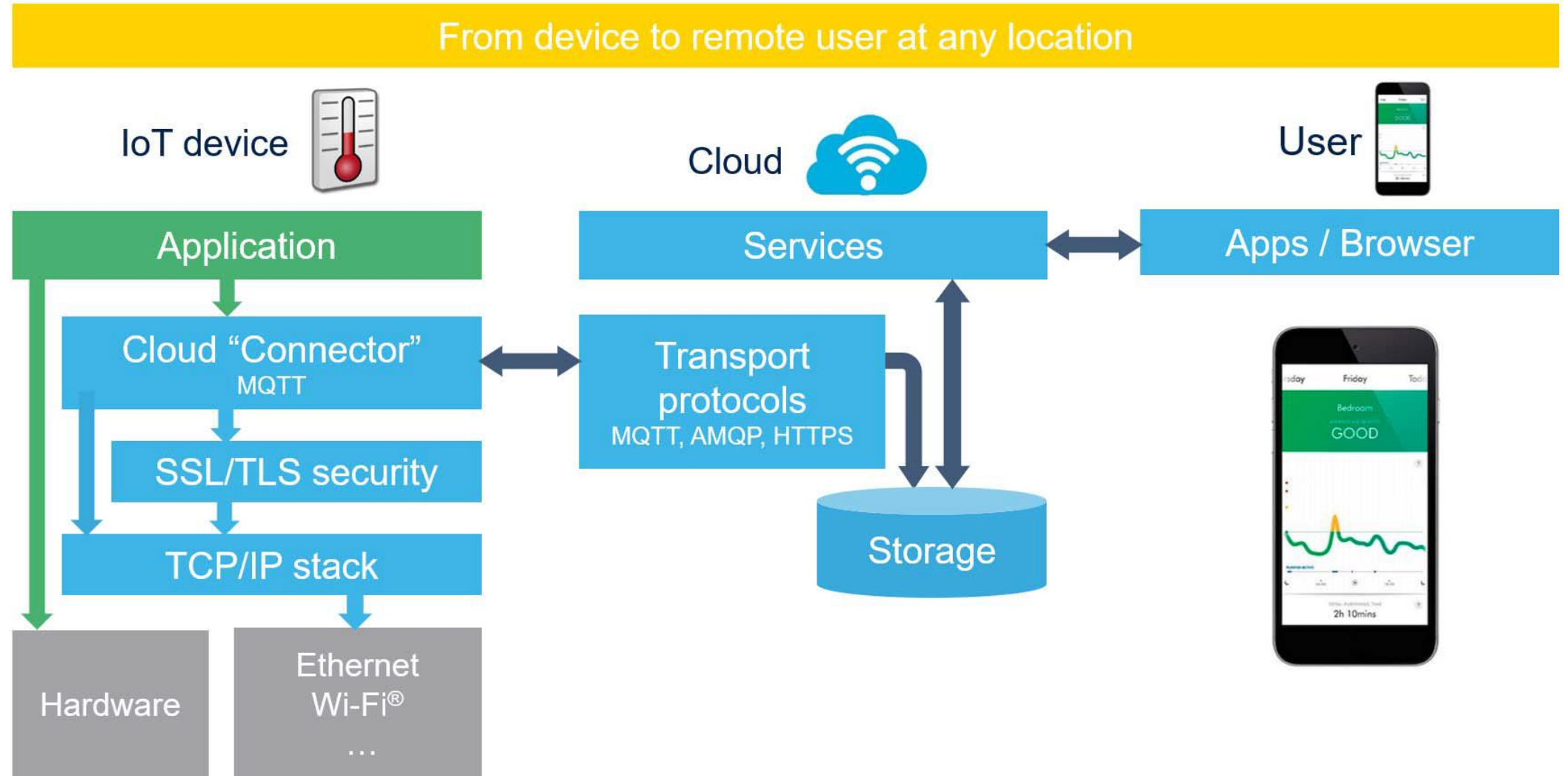


MPU based Gateway

Wi-Fi



Azure IoT Ecosystem



Our technology starts with You



Find out more at www.st.com

© STMicroelectronics - All rights reserved.

ST logo is a trademark or a registered trademark of STMicroelectronics International NV or its affiliates in the EU and/or other countries.

For additional information about ST trademarks, please refer to www.st.com/trademarks.

All other product or service names are the property of their respective owners.



life.augmented