embarcadero

Smart Cross-Platform App Development: Creating Native Apps with a Single Codebase

THE CASE FOR MOBILE APPS

The number of mobile phone users is forecast to reach 4.68 billion in 2019[1], and software developers are scrambling to meet user demands for mobile apps and to take advantage of this rapid-growth market. Mobile devices and their apps have found their way into almost all aspects of life, with adults in the U.S. now spending more than three hours per day on their mobile devices. It seems there is hardly a product or service in both consumer and enterprise worlds that can function without the direct or indirect use of mobile technology.

Enterprises are developing mobile apps for both internal use and customer use to improve user experience, increase brand awareness, remain competitive and improve productivity. Employees who use mobile versions of enterprise software are able to complete work tasks while traveling, working from hotels or coffee shops and after hours due to ease-of-use and increased accessibility. This equates to fewer delays in processes, and faster decision-making for your business.

Enterprises developing mobile applications, therefore, must choose the best development method to balance available time, limited budgets and high user demands for function and design.

TRADE-OFFS OF TRADITIONAL MOBILE APP DEVELOPMENT

There are three primary approaches to mobile app development: platform-specific native apps, hybrid apps and web apps. Each of these approaches provides advantages and disadvantages as shown in Figure 1.

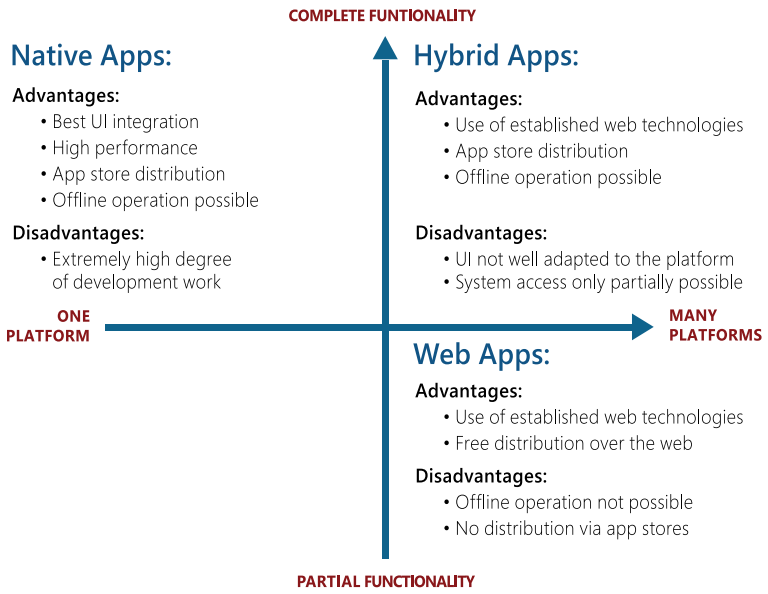


Figure 1: Mobile App Type Comparison.

[1] https://www.statista.com/statistics/274774/forecast-of-mobile-phone-users-worldwide/

Apple iOS and Google Android mobile devices currently account for 97.3% of the global smartphone market[2]. Native apps provide seamless user interfaces, fast performance and can qualify to be listed in platform-specific app stores including iOS App Store and Google Play. Native apps can be installed entirely in the device's memory and, unless the app requires it by design, do not require an internet connection.

Traditionally, to support both iOS and Android, developers would need to create two separate applications: one built for iOS using Swift (or Objective C), and one built for Android using Java or Kotlin.

Creating separate applications for each target platform means duplicating development efforts, either by hiring additional development teams to each focus on one platform (i.e. an iOS development team and Android development team) or releasing platform support asynchronously (i.e. creating one app first, then refocusing the development team's efforts on the second platform's version). Both options result in higher development costs and additional effort to keep two apps up-to-date with comparable features.

Some app manufacturers have tried to reduce development time by creating a single web app instead of multiple native apps. Unlike native apps, web apps run completely in the mobile device's browser and are based on modern Internet technologies such as HTML5, CSS3 and JavaScript. The initial benefit of this approach is that the app only needs to be created once since all devices can run web apps. The development team can use the same developers because there is only one language to learn and one app to code. Developers can use a variety of editors or IDEs to create web apps, and they can reduce development efforts by using available frameworks.

However, web apps create additional challenges for users and developers. Web apps have limited access to system functions and data on mobile devices, users cannot install web apps from the iOS App Store or the Google Play store and the app will not work in offline mode unless it is a Progressive Web App that requires additional programming to enable. Strictly speaking, web apps are essentially websites that are adapted to mimic mobile apps.

Because native apps are expensive and difficult to develop, and web apps provide an inferior user experience, hybrid apps have gained popularity. Hybrid apps mimic native apps by running in a "WebView container." At startup, the app opens a browser window to a predefined address. Then the app, created with HTML5, CSS3, and JavaScript, is displayed in this WebView. Developers can use several frameworks to create hybrid apps, including Cordova (PhoneGap) and Appcelerator Titanium Mobile. The framework allows the app to access some system functions such as the camera or address book, which provides additional functionality over pure web apps.

[2] http://gs.statcounter.com/os-market-share/mobile/worldwide

Using the right frameworks, hybrid app UIs adapt to supported platforms better than web apps. Hybrid apps can be distributed via app stores, which improves their visibility and availability when compared to web apps.

Although hybrid apps solve many of the challenges of web apps, performance can be slower than native mobile apps, resulting in a slower experience for users. According to a Google study, 29% of smartphone users will immediately abandon an app or mobile website if doesn't satisfy their needs, including if it is too slow[3].

It is possible to create native applications while reducing development efforts and improving time to market using cross-platform development with a shared programming language. Instead of duplicating design, development, and maintenance efforts, a single development team coding in C++ or Delphi can create a cross-platform native app that runs smoothly on iOS, Android and desktop platforms. Not only does this approach reduce development efforts, it reduces hiring and employment costs for enterprises which no longer need to maintain separate development teams because their developers have adequate skills to work on all platforms.

Figure 2: Overview of Common Mobile App Development Approaches

[3] https://www.thinkwithgoogle.com/marketing-resources/experience-design/speed-is-key-optimize-your-mobile-experience/

SMARTER CROSS-PLATFORM DEVELOPMENT

Embarcadero's RAD Studio empowers enterprises to target mobile platforms as part of their app development strategy from the start. RAD Studio creates "cross-platform" applications for the following target systems[4]:

- **Microsoft Windows:** Create Win32/Win64 applications for Windows desktop and server.
- **macOS:** Create apps for the macOS desktop.
- **Linux:** Server applications, i.e. applications with no graphical user interface. Linux applications for the desktop are also possible with a third-party library.
- **iOS:** Develop apps for iPhone and iPad.
- **Android:** Develop apps for mobile devices such as smartphones and tablets.

Visual design and a shared codebase reduce costs of app development. Developers can completely design the UI from within RAD Studio's visual designer. The user interface can be created once, and then easily customized for perfect per-device layout and behavior, including the use of native controls, all within the designer. The Live On-Device Preview feature offers developers the ability to broadcast the current design form to test devices and emulators to quickly test the UI without requiring a full compile or deployment.

When implementing business logic, RAD Studio offers a choice between both the Delphi programming language (i.e. Modern Object Pascal), and C++.

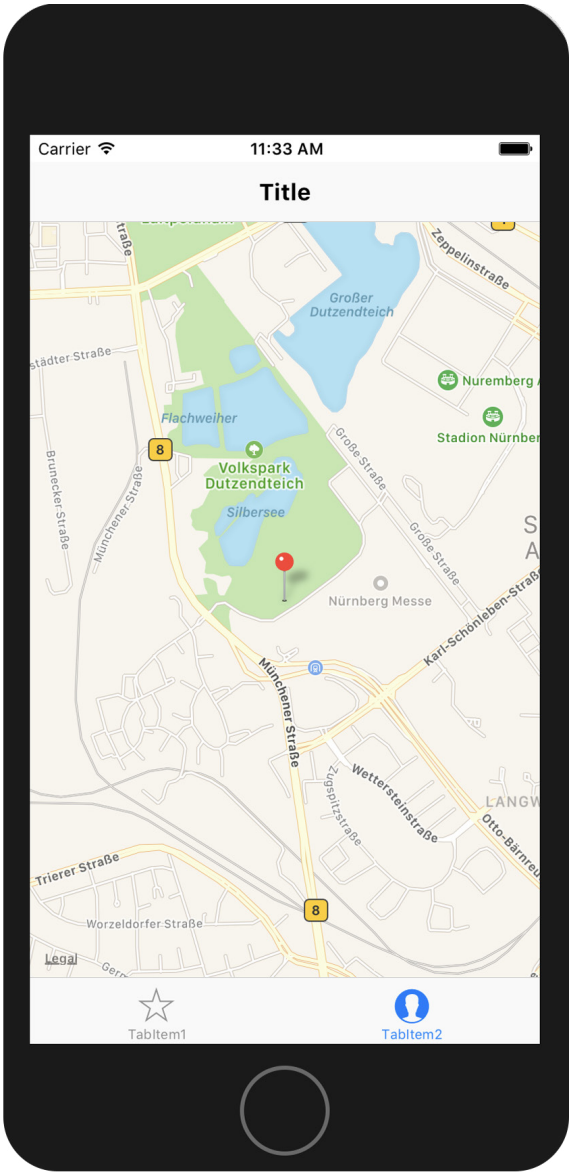
When organizations are ready to distribute their cross-platform application, RAD Studio offers integrated functionality to build and package up their application, ready for submission on the Google Play Store and Apple App Store. In addition, developers can also create AppX packages for distribution via the Microsoft Store.

[4] For a complete list of target platforms and supported OS versions, refer to http://docwiki.embarcadero.com/PlatformStatus/en/Main_Page

SMARTER DEVICE CAPABILITIES

In addition to an attractive user interface, mobile apps often need to access device-specific functions and sensors including GPS, accelerometer, cameras and more. Each platform implements these functions in a different manner, complicating development for shared applications. RAD Studio simplifies sensor and device-specific function access through an abstraction layer of visual and non-visual components.

With RAD Studio's easy-to-use components, developers can focus on delivering the application requirements instead of the differences in how iOS and Android implement a specific sensor.



As an example, Android devices use Google Maps and iOS devices use Apple Maps as the default map service. With RAD Studio, a map is encapsulated by the TMapView component to reference the correct platform-specific service to determine positional data and display it on a map. A map-based app that would otherwise require duplicated coding effort is created within a few minutes using TMapView (Figure 3).

It is as simple to add sensor functionality including motion sensors (TCustomMotionSensor), orientation sensors (TOrientationSensor), Bluetooth and Beacons to cross-platform apps as it is to use the TMapView component.

Figure 3: A map can be integrated into an app with a few mouse clicks.

SMARTER CONNECTIVITY

Mobile apps are increasingly using backend and cloud services to access data and expand app capabilities. RAD Studio provides a framework that allows you to build cloud services and easily connect to your back-end services and databases. RAD Studio Enterprise and Architect editions include RAD Server, a turn-key application foundation for rapidly building and deploying services-based applications. All C++ and Delphi code hosted on RAD Server is published as REST/JSON endpoints that are consumable by any type of client, making it simple for your mobile app to access the data and services it needs

RAD Studio provides comprehensive HTTP client and REST libraries, allowing developers to easily interact with any REST Server. Ready-to-use components for REST BaaS (backend-as-service) frameworks allow applications to connect with Kinvey and Parse API compatible BaaS providers.

SMARTER SETUP AND CONFIGURATION

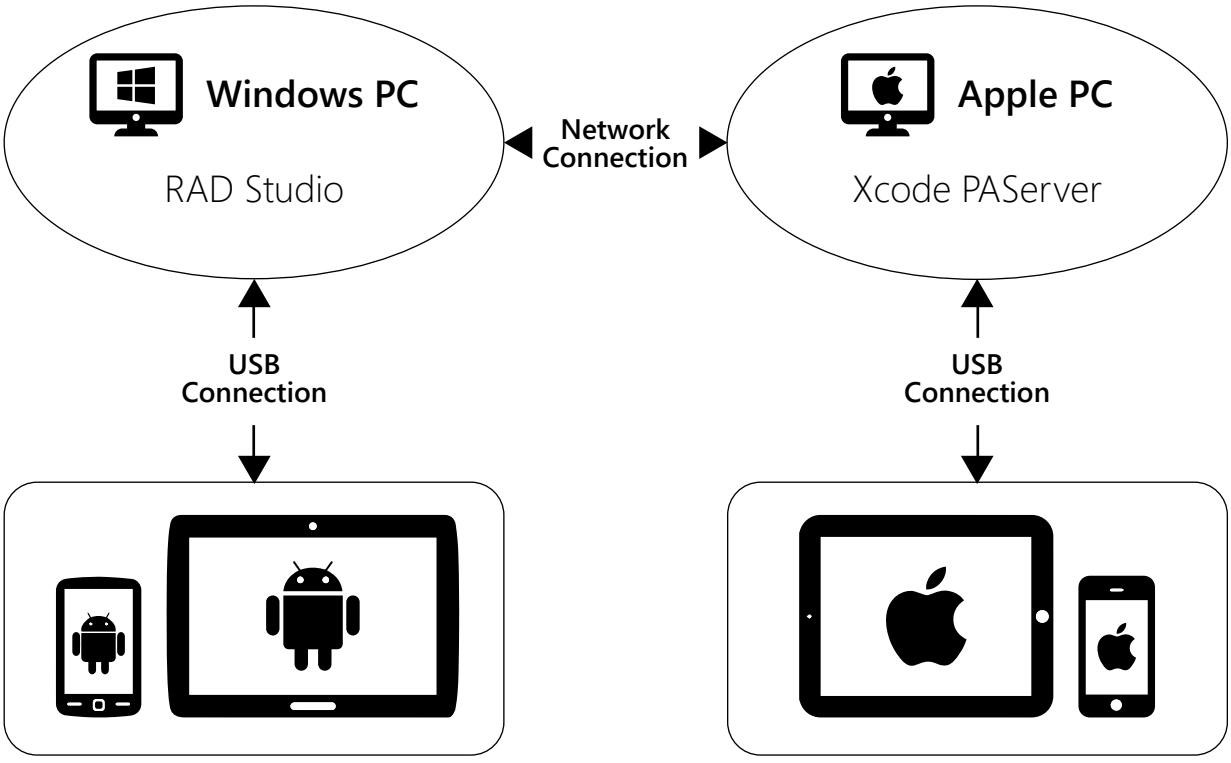


Figure 4: System environment for creating mobile apps.

Developers can design, develop and test cross-platform apps from the RAD Studio IDE, which runs on Microsoft Windows. Mobile apps can be developed and tested on either physical devices or by using the Android emulator and iOS simulator.

Access to a macOS computer is required to generate an app package for iOS. The Platform Assistant Server ('PAServer') is used to provide a connection between the development computer and Mac. The iOS simulator runs on the Mac to test the apps. An iPhone or iPad can also be connected to a Mac PC when deploying to the device.

GET STARTED TODAY

Download a full-functioned, FREE 30-Day Trial of RAD Studio (no credit card required) to start developing smart cross-platform apps today.

The RAD Studio 30 Day Trial provides you with:

- Windows, macOS, Android, and iOS application development with a single codebase
- Visual design with VCL (for Windows) and FireMonkey (for cross-platform) frameworks
- Integrated debugging tools that can be used for debugging on any device
- Create database applications with local or embedded databases
- Use of hundreds of prebuilt components to shorten your development cycle

After installing the Free Trial, create a new cross-platform app in RAD Studio via the following menu path:
FILE | NEW | CROSS-PLATFORM APPLICATION. RAD Studio provides a choice of templates, such as a tabbed navigation app (Figure 4), or start with a blank application.

Templates, styles, and components work with the visual designer to reduce the complexity and effort of creating the cross-platform interface of the app. During development, you can already see the components that are available for the respective system platform (Figure 6).

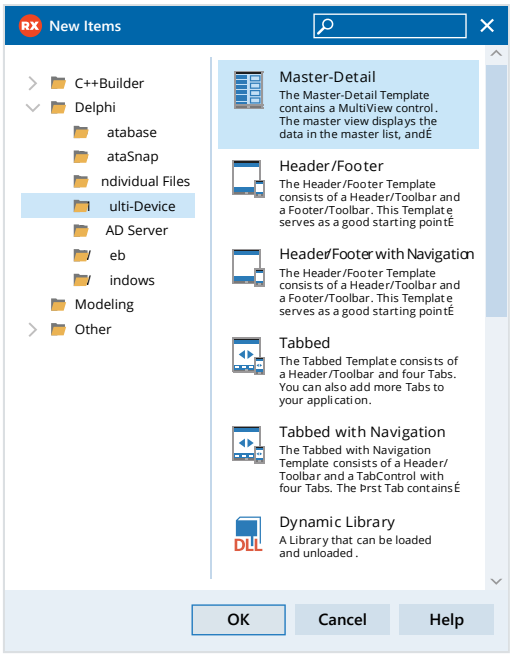


Figure 5: Create a cross-platform app in RAD Studio.

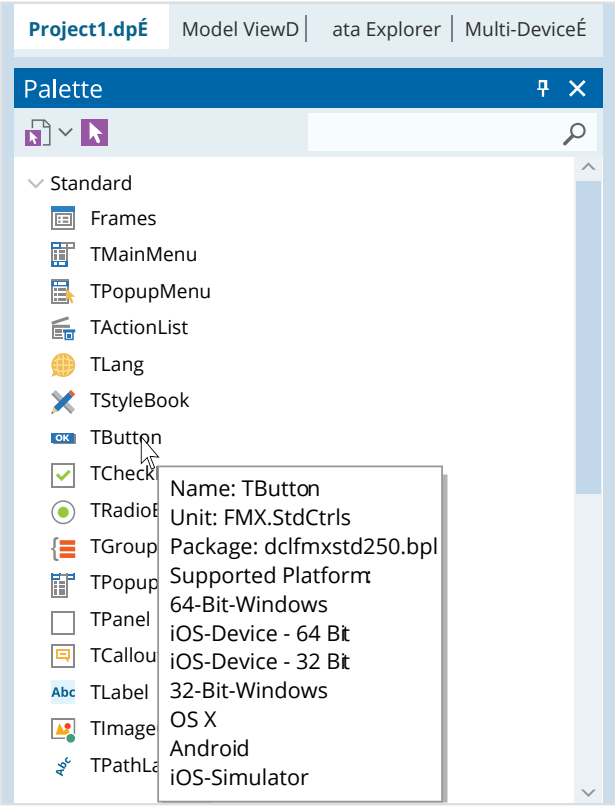


Figure 6: RAD Studio simplifies development with components.

The FireMonkey cross-platform UI framework converts the universal design to adapt to the respective target platform. This way, tabs appear at the bottom of the screen in iOS and at the top of the screen in Android.

The developer doesn't have to worry about these details. This reduces the work involved in the development and ensures excellent user acceptance.

The user interface can be adapted automatically or tailored by the developer to different mobile device hardware. The integrated preview allows developers to quickly view how the user interface will look in a wide variety of device types at design time.

Connecting between the components and the program logic, the data layer, or between several components in the user interface can be greatly simplified by Live Bindings. At a purely visual level, the programmer can couple the program parts without having to write extra code.

The result is a fast native application that can run on the target systems, including iOS and Android, and can be distributed via app stores.

DEVELOPERS BENEFIT FROM USING RAD STUDIO

Whether you are creating a new mobile app or adding mobile platform support to an existing desktop application, RAD Studio provides the performance and device-specific functionality of native applications with the ease of designing, developing and maintaining a single codebase.

RAD Studio's visual designer and visual development frameworks ensure your next cross-platform app looks flawless on every platform. Comprehensive libraries of prebuilt visual and non-visual components accelerate the development process to quickly deliver apps that meet app store acceptance requirements and impress your users.

Enterprises save development effort and costs by hiring a C++ or Delphi development team to create a cross-platform application using RAD Studio. Shorter development cycles means getting to market before your competitors, which in turn increases adoption rates, and ultimately, revenue.

See how easy it is to develop cross-platform, natively compiled apps today with a free, 30-Day Trial.

About Embarcadero: Embarcadero empowers application developers with tools that solve productivity and go-to-market problems. With Embarcadero's technologies, developers can design, create and deploy applications for all platforms from a single codebase, saving considerable time, money and frustrations.

More than three million C++ and Delphi developers worldwide trust Embarcadero's award-winning products to deliver mission-critical applications. An overview of Embarcadero products is available at https://www.embarcadero.com/products.