توسعه برنامه های موبایل

جلسه اول مجازی

بخش دوم

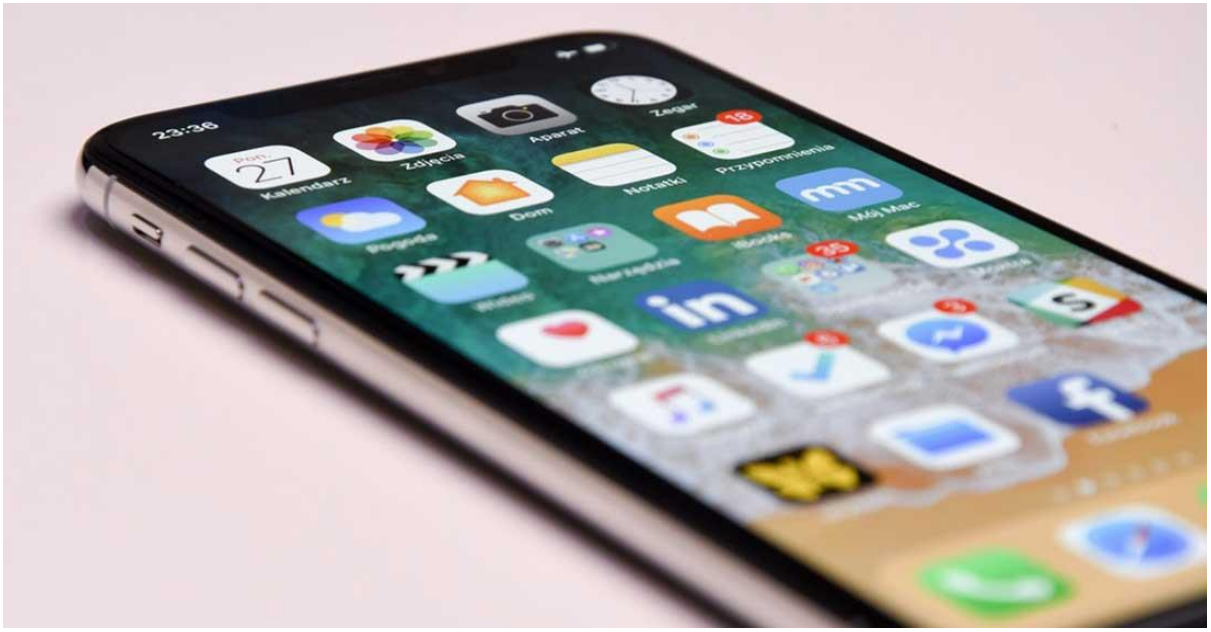
سحر صادقی

اصلی ترین پلتفرم های برنامه نویسی موبایل

از زمانی که کامپیوتر ها اختراع شده اند، تنها تعداد کمی پلتفرم مجزا برای برنامه نویسی موبایل وجود داشته است. اماتا همین اواخر، توسعه اپلیکیشن ها زیاد مورد توجه قرار نگرفته بود و هیچ پلتفرم جامعی برای آن وجود نداشت. این شرایط با معرفی آیفون در سال ۲۰۰۷ تغییر کرد .بعد از این اتفاق بود که میتوانیم بگوییم بازار برنامه نویسی موبایل رونق گرفت و توانست بازاری برای خودش بسازد.

در ابتدا Windows CE **معرفی شد که خوب هم به نظر میرسید اما نتوانستند مسیر درستی را پیش ببرند. سپس** Blackberry **طوری وارد بازار شد که انگار میخواهد دنیا را تصرف کند، شاید برای مدت کمی هم موفق به انجام این کار شد. اما امروزه، حداقل تا زمان نوشته شدن این مقاله، دو مدعی اصلی در این زمینه وجود دارند و بقیه پشت سر اینها قرار میگیرند.**

iOS



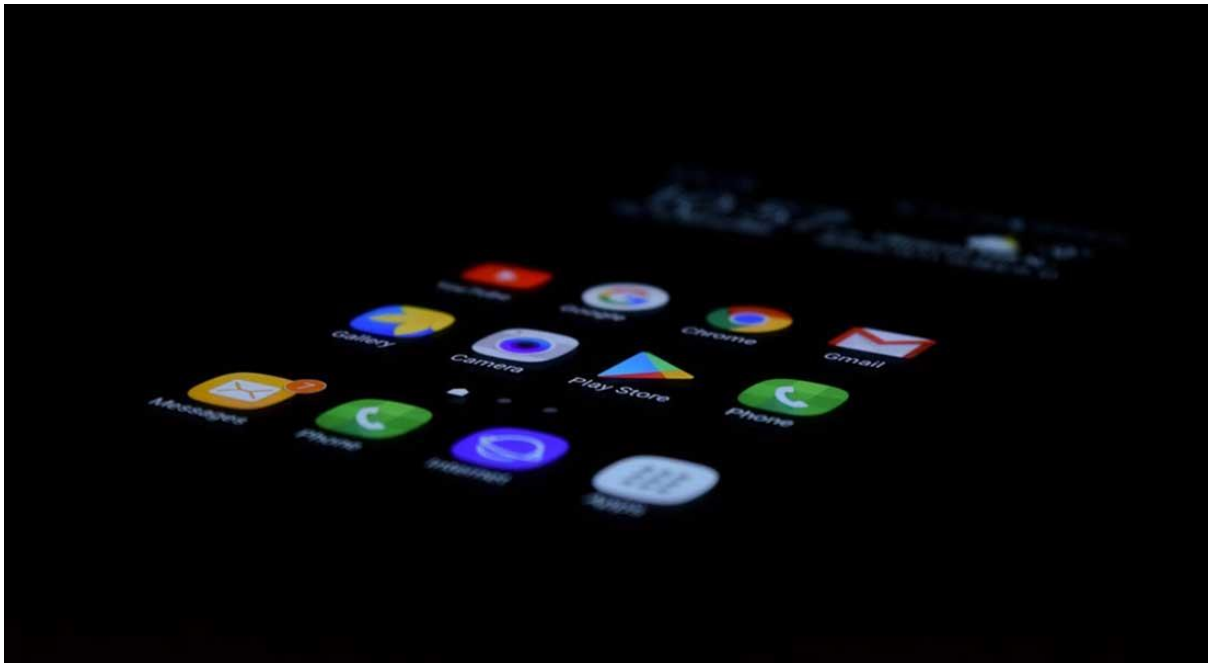
وقتی وارد پلتفرم های توسعه اپلیکیشن موبایل میشویم، تقریبا میتوانیم از iOS **به عنوان "غول مرحله آخر" یاد کنیم. به این دلیل که این پلتفرم بود که توانست برنامه نویسی موبایل را به عصر جدید بیاورد و مفهوم کاملی از این که دستگاه های موبایلی چگونه باید باشند را به ما انتقال داد.**

iOS**یک پلتفرم اختصاصی است که توسط کمپانی اپل** (Apple) **ساخته شده و بصورت اختصاصی روی دستگاه هایی که همین شرکت تولید میکند، اجرا میشود. در زمان نوشته شدن این مقاله،** iOS**روی** iPhone ، iPod، iPad، Apple Watch**و** Apple TV **اجرا میشود. اما من انتظار دارم در آینده دستگاه های بیشتری ساخته شوند که از** iOS **استفاده کنند** iOS **.بر پایه** Darwin **و** OS X **ساخته شده است و هسته ای بسیار شبیه به** Unix **دارد.**

این پلتفرم تعدادی از فرم ورک های مهم را با OS X **سهیم است، رابط کاربری آن هم بر اساس** Apple Cocoa UI **ساخته شده است که در اپلیکیشن های** OS X **از آن استفاده میشود. البته این رابط کاربری دوباره طراحی شده است که روی دستگاه های لمسی به راحتی کار کند که به آن** Cocoa Touch **گفته میشود.**

Apple**برای توسعه دهندگان اپلیکیشن ها ابزار ها و کتابخانه های مختلفی را تولید کرده است. البته برای ساخت اپلیکیشن های اپل حتما نباید از این ابزار ها استفاده کنید، بلکه ابتدا لازم است یک کامپیوتر** Mac **داشته باشید که که** OS X **روی آن اجرا شود تا بتوانید نرم افزار های خودتان را بسازید.**

Android



اگر نخواهیم برای iOS **برنامه بسازیم، پس حتما برای اندروید (یا شاید هردوی آنها) برنامه نویسی میکنیم. اندروید دومین "غول" در زمینه پلتفرم های موجود برای برنامه نویسی موبایل است. اندروید یک مقدار دیرتر وارد بازی شد، یعنی اولین بار در سپتامبر ۲۰۰۸ معرفی شده است. یعنی تقریبا یک سال بعد از**iOS **، اما اکنون سهم بسیار زیادی از بازار موبایل را در اختیار دارد.**

در حقیقت اندروید یک سیستم عامل است که دارای بزرگترین سهم از بازار گوشی های موبایل است. بزرگی این سهم به اندازه ۸۰ درصد از بازار میباشد (در مقایسه با سهم ۱۸ درصدی اپل). این عددها شاید کمی گول زننده باشند، چون بازار اندروید بخش بندی شده است و شامل گوشی ها و دستگاه های بسیار زیادی میشود که توسط سازندگان مختلف ساخته میشوند و نسخه های مختلفی از سیستم عامل اندروید را اجرا میکنند.

این تفاوت اصلی میان اندروید و iOS **است:**

اندروید، که توسط گوگل پشتیبانی میشود، یک سیستم عامل باز است .اما iOS **که توسط اپل پشتیبانی میشود، اینگونه نیست.**

هر کسی میتواند یک دستگاه اندرویدی بسازد و این سیستم عامل طوری طراحی شده است که بتواند روی انواع مختلفی از پلتفرم های سخت افزاری و دستگاه های مختلف با اجزا و توانایی های بسیار متفاوت اجرا بشود.

iOSطوری طراحی شده است که فقط و فقط روی تعدادی از دستگاه های ساخت Apple اجرا بشود.

اندروید روی هسته لینوکس ساخته شده است و سورس کد اندروید هم بصورت متن باز توسط گوگل معرفی شده است. همانند اپل، اندروید هم تعدادی ابزار تخصصی برای توسعه اپلیکیشن های اندرویدی و برنامه نویسی موبایل معرفی کرده است. اما باز هم شما نیاز ندارید که فقط از آنها استفاده کنید. اما همانطور که در بقیه مقاله های سایت هم اشاره کردیم، زبان مورد نیاز برای ساخت اپلیکیشن های اندروید، جاوا است.

بقیه پلتفرم ها

هر کس دیگری در بازار پلتفرم های موبایل باقی مانده است، باید یک سهم بسیار کوچک ۲ درصدی را میان خود تقسیم کنند. مثلا برای یادآوری میتوانیم بگوییم ویندوز و بلک بری در این قسمت وجود دارند.

میتوانیم بگوییم تنها چیزی که بین این پلتفرم ها و نابودی قرار گرفته است، گذر زمان است.

بخاطر سهم بسیار کمی که این پلتفرم ها از بازار دارند، من حتی نمیخواهم درباره آنها صحبت کنم. زیرا دوست ندارم باعث بشوم که شما وقت و سرمایه و انرژی خودتان را برای پلتفرم هایی که امکان موفقیتی ندارند هدر بدهید. اما بازهم میگویم که چندتایی از آنها میتوانند برای ساختن اپلیکیشن های کراس پلتفرم (که روی چند پلتفرم اجرا میشوند و جلوتر درباره آنها صحبت میکنیم)، استفاده کنید. یعنی شما میتوانید این پلتفرم های حاشیه ای را بدون هیچ هزینه ای پوشش بدهید.

هیچوقت توسعه این پلتفرم ها را بصورت اختصاصی یاد نگیرید. اگر میخواهید برنامه نویسی موبایل را یاد بگیرید و یک توسعه دهنده بشوید، یکی از پلتفرم های اندروید یا iOS را انتخاب کنید.

برنامه نویسی موبایل چگونه انجام میشود؟

وقتی که iOS و اندروید منتشر شدند، برای اینکه بتوانید توسعه اپلیکیشن برای آنها را یاد بگیرید، باید از ابزار های بومی که خود این کمپانی ها تولید کرده بودند استفاده میکردید. برای iOS محیط برنامه نویسی XCode و زبان Objective-C استفاده میشد. برای اندروید هم افزونه های Android SDk که روی محیط های برنامه نویسی Eclipse و NetBeans نصب میشد به همراه زبان جاوا باید استفاده میشد.

اما امروز خیلی چیزها عوض شده اند. و گزینه های بسیار بیشتری در اختیار شما قرار دارند. یعنی تعداد بیشماری از ابزار، فرم ورک ها و پلتفرم ها و اکوسیستم های کاملی برای برنامه نویسی موبایل وجود دارند. اما با اینکه روش های بسیار زیادی وجود دارد که میتوانید از آنها استفاده کنید، میتوانیم آنها را در چند دسته کلی قرار بدهیم.

توسعه بومی(Native Development)

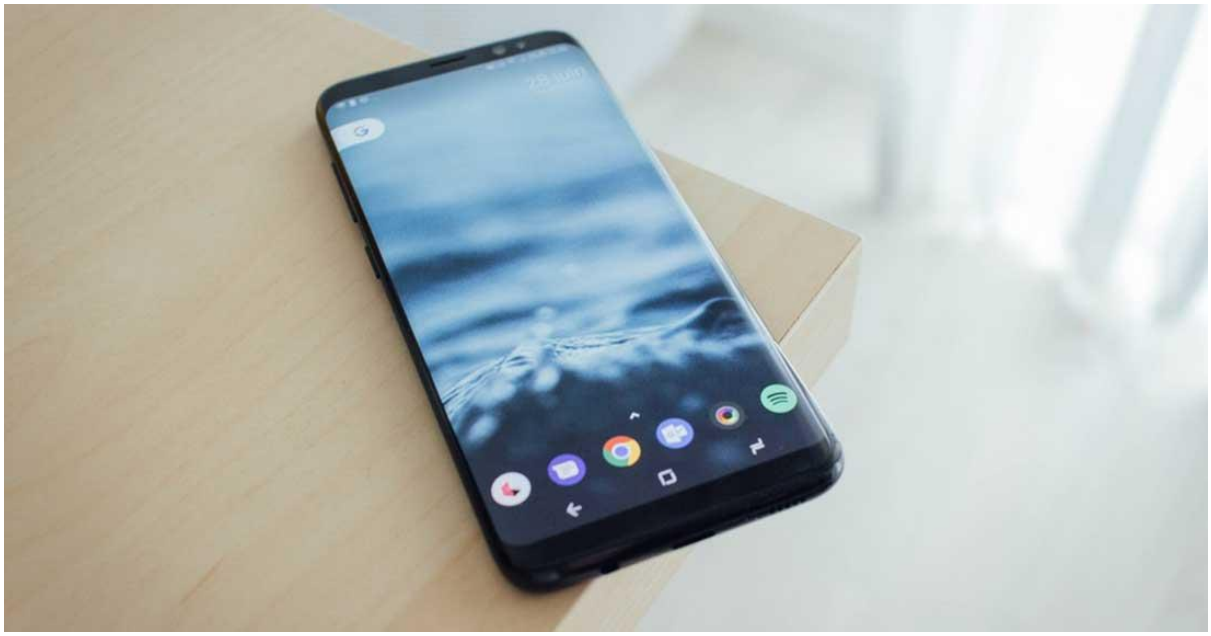
این واضح است که ما میتوانیم برنامه نویسی موبایل را با استفاده از ابزار ها و سیستم عامل هایی که توسط کمپانی های سازنده اعلام میشوند، انجام بدهیم. همانطور که اشاره کردم برای iOS میتوانید از XCode و زبان Objective-C استفاده کنید. اما اپل یک زبان جدید به نام Swift تولید کرده است که الان زبان انتخابی برای توسعه اپ های iOS شناخته میشود.

در دنیای اندروید البته زیاد قضیه فرق نکرده است. به غیر از اینکه گوگل محیط برنامه نویسی Android Studio که اختصاصی خودش میباشد را معرفی کرد ولی زبان برنامه نویسی همان جاوا باقی مانده است. (البته اگر خیلی دل و جرأت دارین میتونین از C و C++ هم استفاده کنین :دی.)

یکی از ایراد هایی که توسعه بومی (Native Development) دارد این است که باید همه کدهای اپلیکیشن خودتان را هم در اندروید و هم در iOS دوباره از نو بنویسید. (مثلا اگر بخواین هم ویندوز فون یا یکی از پلتفرم های دیگه رو پشتیبانی کنین، باید همه اپلیکیشن رو دوباره از نو بنویسین.)

البته شاید در بسیار از موارد، این قضیه زیاد بزرگ نباشد. اما اگر اپلیکیشن شما نیاز دارد که در چند پلتفرم حضور داشته باشد، باید به این فکر کنید که برای ارائه آپدیت های بعدی، مجبور هستید چندین بار نسخه جدید را برای پلتفرم های مختلف تولید کنید که ممکن است مقداری هزینه اضافی روی دست شما بگذارد.

همچنین برنامه نویسی موبایل در پلتفرم های iOS و اندروید کاملا متفاوت است. ابزار ها فرق میکنند، زبان ها متفاوت هستند، فرم ورک ها یکسان نیستند و حتی مراحل برنامه نویسی آنها مانند هم نیست. یعنی اگر بخواهید یک اپلیکیشن را برای اندروید و iOS منتشر کنید، باید دو پلتفرم کاملا مجزا و جداگانه را یاد بگیرید.



اما با این حال توسعه بومی مزیت هایی هم دارد:

بزرگترین مزیت آن سرعت بالای برنامه نویسی است. هرچند فرم ورک های کلاس پلتفرم مانند) Xamarin زامارین) میتوانند از این لحاظ با توسعه بومی رقابت کنند، چون کدها را به زبان بومی ترجمه میکنند. کمی جلوتر به این مورد

میپردازیم. اگر استفاده از فرم ورک هایی که کدها را به زبان بومی ترجمه میکنند را کنار بگذاریم، کدهای بومی (Native Code)از همه راه حل های دیگر سریعتر میباشند.

اگر از توسعه بومی استفاده کنید، ابزار های بهتری برای اشکال یابی در اختیار شما قرار دارند. زیرا قرار نیست با چندین لایه که به شدت انتزاعی هستند سر و کله بزنید .همچنین میتوانید از تعدادی ویژگی های بومی که برای آن پلتفرم عرضه شده است هم استفاده کنید و به سطح سخت افزار نزدیک تر بشوید و بهتر آن را کنترل کنید. (البته شاید بعضی از ابزارهای Cross-Platformبتوانند این مورد را هم پوشش بدهند.)

البته به نظر من اگر میخواهید مثلا برنامه نویسی موبایل برای اندروید را بصورت اختصاصی ادامه بدهید، بهترین راه حل استفاده از توسعه بومی با ابزارهای رسمی اندروید است .زیرا خروجی Android Studio را نمیتوانید با هیچ محیط برنامه نویسی دیگری مقایسه کنید. اما اگر بخواهید اپلیکیشن خودتان را روی چندین پلتفرم منتشر کنید، این نوع از برنامه نویسی موبایل شاید بهترین راه حل نباشد.

فرم ورک ها و ابزار هایCross-Platform

گزینه بعدی که برای برنامه نویسی موبایل وجود دارد این است که از ابزارها یا فرم ورک هایی استفاده کنید که میتوانند یک اپلیکیشن را روی چندین پلتفرم بسازند. در اصطلاح به این ابزارها Cross-Platform گفته میشود. بستگی به نیازی که شما دارید، تعداد زیادی از این راه حل ها تولید شده اند و میتوانید از بین آنها انتخاب کنید.

تعدادی از این موارد میتوانند کدهای بومی برای شما تولید کنند و از کتابخانه های Native واقعی استفاده کنند.پس در حقیقت فقط یک لایه روی زبان و ابزارهای بومی کشیده اند، اما هنوز برای کار کردن با آنها نیاز دارید که استفاده از فرم ورک ها و کتابخانه های Native را بلد باشید.

بقیه گزینه ها یک اپلیکیشن هیبرید (ترکیبی) میسازند که تعدادی از کامپونت های بومی و تعداد از کامپونت های HTML را درون خودشان دارند. این اپلیکیشن ها معمولا بسیاری از کارکرد ها و همچنین رابط کاربری خودشان، به مرورگر اینترنتی خود گوشی نیاز دارند. لیست گزینه های شما برای این مدل از برنامه نویسی موبایل هر روز در حال بزرگتر شدن است. پس انتخاب از میان آنها میتوانید سخت باشد. موارد اصلی که باید در هنگام انتخاب فرم ورک های-Cross Platformبه آنها توجه داشته باشید اینها هستند:

- از چه زبان برنامه نویسی باید استفاده کنید؟
- میخواهید اپلیکیشن هیبرید بسازید یا بومی؟
- میخواهید کدهای شما روی چند پلتفرم اجرا بشوند؟
- میخواهید بتوانید دوباره از کدهایتان استفاده کنید یا نه؟

زبان برنامه نویسی

میخواهید برای اپلیکیشنی که قرار است بسازید، از کدام یک از زبان های برنامه نویسی موبایل استفاده کنید؟

بیشتر راه حل هایی که برای Cross-Platform **شدن روبروی شما قرار دارد، از یک زبان برنامه نویسی استفاده میکنند. شاید شما نخواهید فشار یاد گرفتن یک فرم ورک، برنامه نویسی موبایل و یک زبان جدید را یکجا تحمل کنید. پس احتمالا باید یک فرم ورک** Cross-Platform **استفاده کنید که زبانی که شما از قبل بلد بودید را پشتیبانی کند.**

بومی یا هیبرید؟

تعداد زیادی راه حل Cross-Platform **وجود دارد که کدهای شما را به زبان بومی همان سیستم عامل که قرار است روی آن اجرا بشود تبدیل میکند و مستقیما از** API **ها و فرم ورک های بومی همان زبان هم استفاده میکند.**

یکی از آنها (Xamarin **زامارین) است که به شما اجازه میدهد با زبان سی شارپ کدنویسی کنید و از همه امکانات برنامه نویسی موبایل به شیوه بومی، آن هم بصورت کامل استفاده کنید. البته گزینه های زیاد دیگری هم هست که از بین آنها انتخاب کنید. بقیه راه حل های** Cross-Platform **مثلا مانند**Cordova **، از تکنیک هیبرید استفاده میکنند که در حقیقت اپلیکیشن ها را کاملا بومی تولید نمکنند، اما مانند بومی ها بنظر میرسند.**

از لحاظ تکنیکی و فنی، اپلیکیشن های بومی سرعت بالاتر و ظاهر و احساس متناسب تری با سیستم عاملی که آنها را اجرا میکند، دارند **. اما با اینحال، تعدادی از راه حل های هیبرید، به قدری به کدهای** Native **نزدیک شده اند که تشخیص تفاوت آنها بسیار دشوار شده است.**

پشتیبانی پلتفرم

یکی از مواردی که باید حتما در نظر بگیرید، پشتیبانی پلتفرم است. همه راه حل های Cross-Platform **برنامه نویسی موبایل، از اندروید و** iOS **پشتیبانی میکنند (خیالتون تخت از این نظر). اما بعضی از آنها میتوانند حتی نسخه های دسکتاپ مانند** Windows **و** Mas OS X **را هم تولید کنند. تعدادی از آنها نیز سیستم عامل های کوچک تر یا حتی** Raspberry Pi **را ساپورت میکنند.**

مثلا اگر مشتری های شما از Blackberry **استفاده میکنند و میخواهید آنها را از دست ندهید، باید دنبال ابزاری بگردید که توانایی ساخت اپلیکیشن برای این سیستم عامل را داشته باشد.**

به هرحال، اگر نیاز جدی به پلتفرم هایی غیر از اندروید و iOS **ندارید، زیاد نگران پلتفرم های دیگری که ممکن است پشتیبانی نشوند، نداشته باشید و بدون در نظر گرفتن آنها ابزار مناسب را انتخاب کنید.**

البته بازی سازی یک بخش جدا در میان دنیای برنامه نویسی موبایل است که باید درنظر گرفته بشود.

یعنی اگر در حل توسعه و ساخت یک بازی هستید، باید یک راه حل Cross-Platform **پیدا کنید که بتواند وسیع ترین بازه ممکن از پلتفرم ها را پوشش بدهد. ابزارهایی مانند** Unity 3D **شما را قادر میکنند که محصولتان روی هر پلتفرمی که فکرش را بکنید اجرا بشود، حتی روی وب.**

قابلیت استفاده دوباره از کد

در آخر، شما باید قابلیت استفاده دوباره از کدها را هم در نظر بگیرید. نباید اینطور فکر کنید که وقتی از ابزارهای Cross-Platform**استفاده میکنید، میتوانید فقط یک بار کدهای خودتان را بنویسید و آنرا روی همه پلتفرم هایی که ساپورت میشوند اجرا کنید.**

البته باید بدانید معمولا ابزارهایی که از کدهای Native **پشتیبانی میکنند، کمترین میزان استفاده دوباره از کدها را دارند. زیرا آنها بیشترین نزدیکی و استفاده را از فرم ورک ها، کتابخانه ها و رابط های کاربری بومی** (Native) **دارند.**

بنابراین ممکن است شما روبروی ای دوراهی قرار بگیرید که بیشتر بومی باشید و خودتان را به ظاهر و حس سیستم عاملی که برنامه روی آن اجرا میشود نزدیک تر کنید، یا اینکه کدهای بیشتری را تقسیم کنید. اما اخیرا راه حل های-Cross Platform **مانند زامارین ارائه شده اند که به شما اجازه میدهند از هر دو روش سود ببرید.**

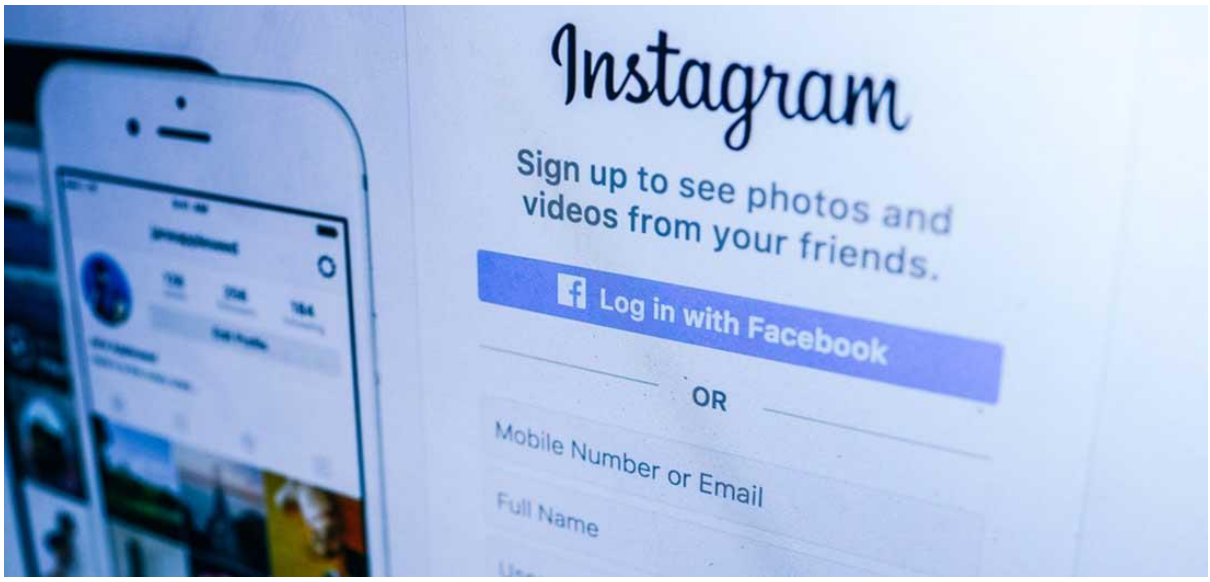
برای مثال، Xamarin**از یک کتابخانه رابط کاربری به نام** Xamarin Forms **استفاده میکند که به شما اجازه میدهد درصد بیشتری از استفاده دوباره از کدها را بین پلتفرم های مختلف با ساختن یک لایه انتزاعی روی زبان و فرم ورک های بومی سیستم عامل زیرین داشته باشید.**

در نهایت، قابلیت استفاده دوباره از کد به این بستگی دارد که چه نوع اپلیکیشنی میسازید و میخواهید چه مقدار برنامه شما مانند نرم افزارهای بومی آن سیستم عامل باشد.

Web App**های موبایل (وب اپ یا وب اپلیکیشن)**

وب اپ به اپلیکیشن هایی گفته میشود که روی سرور های اینترنتی اجرا میشود و برای دسترسی به آنها باید به وبسایت آن اپلیکیشن مراجعه کنید. یعنی اپلکیشن روی سرور های یک سایت نصب شده و همانجا اجرا میشود. شما فقط میتوانید از طریق اینترنت آنرا کنترل کنید و کاری که میخواهید را انجام بدهید. به عنوان آخرین روش، شما میتوانید تصمیم بگیرید که اپلیکیشن شما بر پایه وب باشد یا نه.

در طی سالهایی که برنامه نویس ها در حال ساخت اپلیکیشن های مختلف بوده اند، این گزینه آرام آرام توانسته خودش را بیشتر نشان بدهد. دلیل پر استفاده تر شدن این روش به قدرتمند تر شدن مرورگر های اینترنتی برمیگردد .امروزه مرورگر هایی که روی دستگاه های موبایل نصب شده است که تکنولوژی های زیادی دارند و میتوانند همه صفحه های اینترنتی را به راحتی نمایش بدهند.



با در دسترس بودن این امکان در مرورگرها، میتوانید یک وب اپ، مانند بقیه وب اپ هایی که وجود دارند بسازید، اما باید آن را مخصوص کار کردن روی دستگاه های موبایل طراحی بکنید.

بسیاری از مرورگرهایی که روی سیستم عامل های موبایل وجود دارند، میتوانند به شما این اجازه را بدهند که از امکانات و قابلیت های بومی سیستم، از درون خود مرورگر دسترسی داشته باشید. یعنی میتوانید مثلا به اطلاعات لوکیشن کاربر یا دوربین آن را هم از طریق وب اپ دسترسی داشته باشید.

همچنین تعدادی فرم ورک وجود دارند که به شما کمک میکنند وب اپ موبایل خودتان را بسازید. این وب اپ ها وقتی روی سیستم عامل اجرا میشوند، مانند اپلیکیشن های بومی به نظر میرسند.

یک حقیقت را به شما بگویم: آینده به وب اپ ها تعلق دارد! فقط هنوز به آنجا نرسیده ایم.