# ****Enterprise Applications to Mobile Devices****

**Did you know?**

* 91% of mobile users keep their device within arm's reach 100 percent of the time
* 75% of mobile shoppers take action after receiving a location-based message
* 96% year to year increase in mobile cyber Monday sales from 2012 through 2013

Extending enterprise applications to mobile devices is increasingly becoming a priority for organizations optimizing their workforce. To achieve the desired result of a robust, scalable, secure, and responsive mobile solution with multiple device platform support, many components need to work together. The challenge is to seamlessly extend various flavors of enterprise applications, many based on a variety of technologies and platforms, on to mobile devices.

## Mobile Solution Overview

When extending enterprise applications to mobile devices, many solutions require a three-tier approach: the enterprise application itself, mobile middleware, and the mobile client application.

**Enterprise Application.**There are, of course, many flavors of enterprise applications that can be extended on to mobile devices, such as Customer Relation Management (CRM), Enterprise Resource Planning (ERP), and Business Intelligence (BI).

**Mobile Middleware.**As most enterprise applications don’t have a direct way of working with devices, mobile middleware (as it will be called in this article) plays a crucial role. Some of the important features of this tier include security, data synchronization, device management, and the necessary support for multiple devices.

**Mobile Client Application.**The mobile client application is, of course, the software that will run on the device. There are many considerations at this tier, including data availability, communication with middleware, local resource utilization, and local data storage. In addition, many business factors need to be considered. For example, who are the target users? How critical is it to have the latest data? Are there restrictions for storing data on the device? What provisions are there in case of no network connectivity?

## Mobile Middleware

Mobile middleware is software that connects disparate mobile applications, programs and systems.

Mobile middleware essentially hides the complexities of working in mobile environments, allowing for smoother device-to-device interaction, mobile-computing integration and mobile application development. Like other kinds of middleware, mobile middleware typically provides messaging services to enable communication between different applications.

In both native and HTML 5 mobile development strategies, mobile middleware serves to connect devices to native hardware. It also enables enterprise architects and developers to reuse the same back-end logic for multiple mobile devices without having to rewrite back ends to work with each separate device.

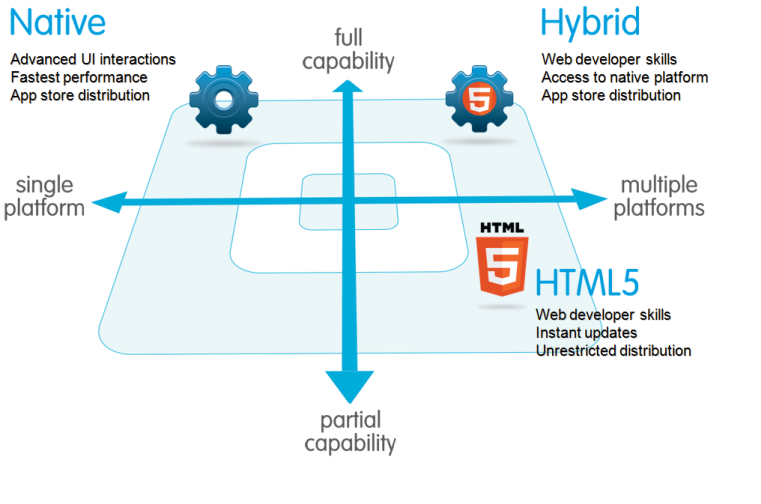
The ideal mobile middleware product should work with every type of wireless handheld device, run on any server platform in your shop, reliably deploy the application, automatically handle synchronization issues during periods of disconnection, be responsive and intuitive to use, offer the highest level of security and adhere to industry standards

## Mobile Client Application

Developers can choose from different mobile application approaches,

* **Native apps** are specific to a given mobile platform (iOS or Android) using the development tools and language that the respective platform supports (e.g., Xcode and Objective-C with iOS, Eclipse and Java with Android). Native apps look and perform the best.
* **HTML5 apps** use standard web technologies—typically HTML5, JavaScript and CSS. This write-once-run-anywhere approach to mobile development creates cross-platform mobile applications that work on multiple devices. While developers can create sophisticated apps with HTML5 and JavaScript alone.
* **Hybrid apps** make it possible to embed HTML5 apps inside a thin native container, combining the best (and worst) elements of native and HTML5 apps

each with its own set of pros and cons.



### Native Mobile Applications

Due to native apps having to be built with the specific development tools and languages for each respective platform, developers that are well versed in either iOS or Android development are the best for the job. Developers are able to take full advantage of all the unique device features, such as the camera, accelerometer, compass, GPS or even the iPhone 5S’ fingerprint sensor.

Native apps have the benefit of familiarity as developers already have a degree of familiarity with the respective software development kits. iOS developers and Android developers know how the code will function and run efficiently on the targeted platform.

If one wants to cover a larger audience across all platforms, separate native apps for each device will be required, but this approach is much more expensive. Going native involves a larger upfront investment in infrastructure, developers and technology as compared to hybrid apps.

With hybrid development, one cannot take advantage of the device specific features and themes. Reproducing those features with a hybrid approach becomes much trickier.



### HTML5 Mobile Applications

HTML5 mobile app is basically a web page, or series of web pages, that are designed to work on a tiny screen. As such, HTML5 apps are device agnostic and can be opened with any modern mobile browser. And because your content is on the web, it's searchable, which can be a huge benefit depending on the app (shopping, for example).

If you have experience developing Web apps, you'll take to HTML5 like a duck to water. If you're new to Web development, the technological bar is lower; it's easier to get started here than in native or hybrid development. Unfortunately, every mobile device seems to have their own idea of what constitutes usable screen size and resolution, and so there's an additional burden of testing on different devices. Browser incompatibility is especially rife on Android devices, so browser beware.

In the last year, HTML5 has emerged as a very popular way for building mobile applications. Multiple UI frameworks are available for solving some of the most complex problems that no developer wants to reinvent. iScroll does a phenomenal job of emulating momentum style scrolling. JQuery Mobile and Sencha Touch provide elegant mobile components, with hundreds if not thousands of plugins that offer everything from carousels to super elaborate controls.

So if HTML5 apps are easier to develop, easier to support, and can reach the widest range of devices, where do these apps lose out? We already reviewed the major benefits of native development, so we'll just reiterate that you can't access native features on the device. Users won’t have the familiarity of the native look and feel, or be able to use compound gestures they are familiar with. But strides are being made on all fronts, and more and more functionality is supported by browsers all the time.



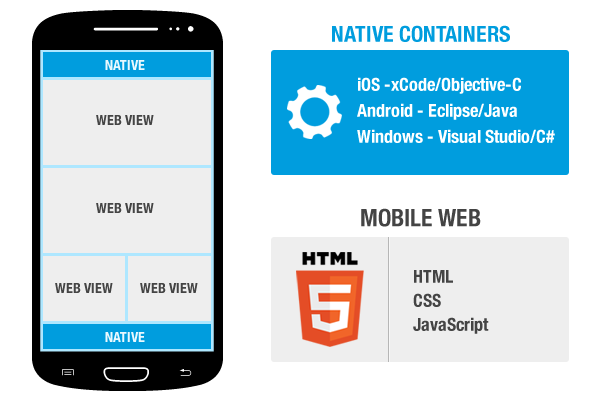
**Hybrid Mobile Applications**

Hybrid apps are part native and part HTML5. Like native apps, they are installed on a device and live in an app store. But unlike native apps, they are built using HTML5 approaches and are subsequently placed in a ‘wrapper’ that allows for distribution and use on iOS or Android.

It is a ‘Write Once, Run Anywhere’ strategy, similar to what made Java such a dominant force many years back. This results in a cross-platform, consistent user interface that works well on most devices. The Netflix app is one example of a hybrid app which runs the same code base on all platforms.

Enterprises now are increasingly finding the need to support multiple platforms. For hybrid apps only a small portion of code needs to be re-implemented for different platforms. By a prediction last year that by 2016 more than 50 % of the apps deployed will be developed with a hybrid approach.

A native app, on the downside, can be used only for its specific platform. This thereby restricts their potential audience.



## Which Is More Effective… Native or Hybrid?



|  |  |  |  |
| --- | --- | --- | --- |
|  | Native | HTML5 | Hybrid |
| Graphics API | Native | Complicated | Complicated |
| Look, Feel, Sense, UX | Native xml | Web or HTML 5 | xml Interpreter |
| Performance | Fastest | Kindly slow | Fast enough |
| Device Sensors | Yes | Depends on tech very limited | Depends on tech |
| Offline Storage | Device storage | Shared web storage | Limited device storage and shared web storage |
| Connectivity | Online and offline usage | Mostly online usage | Online and offline usage |
| Development Environment | Objective-C, Java, C#, JS | HTML5/JS/CSS | HTML5/JS/CSS |
| Security | Most secure | Opens for attacks | Depends on tech. Mostly secure |
| Device Specific Features | Yes | Depends on tech | Depends on tech |

It is evident that both approaches have certain drawbacks and respective benefits. Deciding between the two approaches depends on many factors such as the type of application being built, development talent and resources, allotted budget, and so on.

If it is a productivity, utility or enterprise app, hybrid apps are usually the way to go. If the app in question involves gaming, photos, or videos then building a native app makes sense. Native development optimizes the performance of media and graphic capabilities since developers can access device specific features.

The bottom line is that there are many factors that comprise an enterprise’s mobility strategy and there is no “one-size fits all” approach

Checkout this infographic from CMS WIRE:

