

İÇİNDEKİLER

| | |
|--|-----------|
| Mobil Cihazlar ve Mobil Programlama | 3 |
| Mobil İşletim Sistemleri ve Platformlar | 4 |
| IOS | 4 |
| Android..... | 4 |
| Windows Phone..... | 5 |
| Amazon Fireos..... | 5 |
| Blackberry 10 | 5 |
| Firefox OS..... | 6 |
| Mobil Uygulama Geliştirilirken Dikkat Edilmesi Gerekenler | 6 |
| Hybrid ve Native..... | 6 |
| Hybrid | 6 |
| Native | 7 |
| Doğru Seçim Hangisi? | 7 |
| Hybrid ile Javascript Çatısı | 7 |
| PhoneGap Nedir? | 7 |
| Apache Cordova..... | 8 |
| Apache Cordova'nın Tarihi | 8 |
| Apache Cordova Nasıl Çalışır? | 8 |
| Apache Cordova'nın Avantajları ve Dezavantajları Nelerdir? | 8 |
| Apache Cordova'nın Çalışma Mantığı..... | 9 |
| İsmi PhoneGap mi? Apache Cordova mı? | 9 |
| Apache Cordova ve PhoneGap Hakkında Bilinmesi Gerekenler | 9 |
| Cordova Çalışma Ortamının Kurulumu..... | 9 |
| Git ve GitHub..... | 11 |
| Git | 11 |
| Github..... | 11 |
| Git Kurulumu ve Kullanımı..... | 12 |
| Command Line Interface (CLI) | 12 |
| Kullanılabilir CLI Komutlarını Öğrenmek..... | 12 |
| CLI Kullanımına Hazırlık | 12 |
| Bir Cordova Projesi Oluşturmak..... | 13 |
| Cordova Uygulamasına Platform Ekleme..... | 13 |
| Cordova Projesinden Platform Kaldırmak..... | 14 |
| Cordova'da Kurulu Platformlar | 14 |
| Cordova Versiyonu Öğrenmek | 15 |
| Cordova Sürümünü Güncellemek..... | 15 |
| Cordova Platform Versiyonunu Güncelleme | 15 |
| Cordova Plug-in'leri Nasıl Yüklenir? | 15 |
| CLI Üzerinden Plugin Kullanımı..... | 16 |
| Cordova Plug-in Nasıl Kaldırılır | 16 |
| Kurulu Plug-in Listelemek | 16 |
| Projeyi Derlemek | 16 |
| Uygulamayı Yerel Sunucu Üzerinde Çalıştırmak..... | 17 |
| Uygulamayı Emulator Üzerinde Çalıştırmak | 17 |
| Cordova Dizin Yapısı..... | 18 |
| 'www' Klasörü | 18 |

| | |
|--|-----------|
| “platforms” Klasörü..... | 18 |
| Plugins Klasörü..... | 18 |
| “Merges” Klasörü..... | 19 |
| Apache Cordova ile Proje Geliştirmek için Kullanılan Teknolojiler | 19 |
| Html | 19 |
| Css..... | 19 |
| Javascript | 19 |
| AngularJS | 19 |
| Node.JS | 20 |
| Mobil Projelerde Performans İpuçları | 20 |
| DTO Kullanımı..... | 20 |
| Uygulamaya Göre Yaklaşım | 20 |
| FastClick.js..... | 20 |
| Framework 7 | 20 |
| UI Components..... | 21 |
| Göze Çarpan Özellikler | 21 |
| Swipe Back | 21 |
| Swipe Actions | 21 |
| Dynamic NavBar | 21 |
| Pull To Refresh..... | 21 |
| Messages | 21 |
| BuildPhoneGap ile Proje Derleme | 21 |
| Emulator Ayarları ve Cordova Projesini Çalıştırmak..... | 23 |
| Ripple Mode | 24 |
| Device Mode..... | 24 |
| Android Studio Emulator | 24 |
| Android Device Manager ile Device Oluşturma..... | 25 |
| Cordova Plug-in Mantiğı..... | 26 |
| Core Plugins | 26 |
| Custom Plugin | 26 |
| Cordova Life Cycle Events | 26 |
| Cordova Local & Session Storage Kullanmak..... | 27 |
| Html5 Local Storage | 27 |
| Html5 Session Storage..... | 29 |
| LocalStorage ve Json Kullanarak Telefon Defteri | 30 |
| Cordova Projelerinde Türkçe Karakter Sorunu..... | 32 |
| Cordova Projesi Oluşturmak..... | 32 |
| Plugins | 34 |
| Plugin Yükleme | 34 |
| Plugin Kullanımında Dikkat Edilmesi Gerekenler | 35 |
| Core Plugins..... | 36 |
| Vibration Plugin | 36 |
| Device Plugin | 37 |
| Battery Plugin | 38 |
| Network Information Plugin | 39 |
| Kamera Plugin..... | 40 |
| WhiteList Plugin | 40 |

| | |
|--|-----------|
| Custom Plugin | 41 |
| Framework 7 Kullanımına Giriş | 41 |
| İlk Framework 7 Uygulaması | 42 |
| My-app.js | 42 |
| Apache Cordova ile WebAPI Kullanımı | 42 |
| Servisten Data Çekmek | 44 |
| Servise Data Göndermek | 45 |

Mobil Cihazlar ve Mobil Programlama

Günlük hayatınızı bir düşünün, okullarımızda, iş yerlerimizde, seyahat ederken her zaman yanımızda olan bir cihaz.. Cep Telefonlarımız.. yani Mobil cihazlarımız..

Kısa sürede hayatımızın olmazsa olmazı haline gelen cep telefonları, teknoloji geliştikçe dahada vazgeçilmez oluyor. Her gün kullandığımız cep telefonlarımız, hayatımızı planladığımız ve sosyal hayatımızı organize ettiğimiz ajandalarımız, market listemizi hatırlatan ve hatta özel günlerimizi bize hatırlatan kişisel asistanlarımız haline geldi. Peki bu süreç nasıl işledi? Ne zaman hayatımızın bir parçası haline geldiler? Hayatımıza bu kadar çabuk yerleşen yeni arkadaşlarımızın gelişim sürecine bir göz atalım..

İlk cep telefonu için çalışmalar 1940'lı yıllara dayanmaktadır. İlk cep telefonu "Motorola DynaTAC 8000x" isimli telefondur, 1973 yılında üretilmiştir ve ağırlığı 1kg'dır. Bu telefon tam dolu bir batarya ile 20 dakikalık bir görüşme sağlamaktadır. Satışa sunulan ilk cep telefonunu tasarlayan kişi "Martin Cooper"dır.

İlk telefondan itibaren üretilen tüm cep telefonları küçülme ve incelme yarışına girdi, bunun sebebi onları daha kolay yanımızda taşımamız ve hayatımızın her alanında kolaylıklar sağlaması için kullanmamızdır. Mobil programlama gün geçtikçe popüler oluyor. Hergün çeşitli uygulamalar marketlere kazandırılıyor. Telefonların küçük ekranlarının kullanışlı olmamasından dolayı daha rahat kullanabilmemiz için tablet adı verilen daha büyük boyutlarda mobil cihazlar üretildi. Artık evlerimizde kullanılan akıllı televizyonlarda çeşitli uygulamaları çalıştırabilmektedir.

Tabi bu gelişim süresince yazılımsal açıdan bakıldığında platformları tanımalıyız ve hızla büyüyen mobil pazarda doğru uygulamayı hazırlamak için doğru teknik ve araçları seçerek işe başlamalıyız.

Mobil İşletim Sistemleri ve Platformlar

Günümüz mobil cihazlarında kullanılan çeşitli platformlar bulunmakta. Bilindiği gibi çalışan her elektronik cihazın onu çalıştıracak bir yazılıma (işletim sistemi) ihtiyacı vardır. En popüler bazı işletim sistemleri aşağıda verilmiştir. Aşağıdaki liste bir sıralama değildir.

IOS

IOS Apple'ın başlangıçta iPhone için geliştirdiği ancak daha sonra iPod touch ve iPad'de de kullanılan, Mac OS x'den türetilmiş olan bir mobil işletim sistemidir. IOS içinde 4 katman bulundurmaktadır: Core OS tabakası, Core Servisleri tabakası, Medya tabakası ve Cocoa Touch tabakası. Yazılım cihazın içinde 2500 MB'lık bir alan kaplamaktadır. IOS işletim sistemi yapısı nedeniyle AppleApp Store ve iTunes dışında hiçbir yerden uygulama yüklenemez.

Başlangıçta, üçüncü tarafların geliştirdiği uygulamalar desteklenmiyordu. Steve Wozniak daha sonra geliştiricilerin uygulamalar geliştirmesini savundu. 7 Haziran 2010 WWDC 2010'da Apple, iPhone OS'in adını iOS olarak değiştirdiğini açıkladı.

Android

Android; Google ve Open Handset Alliance tarafından, mobil cihazlar için geliştirilmekte olan, Linux tabanlı özgür ve ücretsiz bir işletim sistemidir. Sistem açık kaynak kodlu olsa da, kodlarının ufak ama çok önemli bir kısmı Google tarafından kapalı tutulmaktadır. Google tarafından ücretsiz olmasının sebebi, sistemin daha hızlı ve çabuk gelişmesi, birçok popüler marka tarafından kullanılması ve bu sayede reklamlarını daha fazla kişiye ulaşmasını sağlamaktır. Google, Android sistemi üzerinde çalışan Google Play marketteki oyun ve uygulamalar üzerinde aldığı reklamları yayınlayarak para kazanmaktadır. Android'in desteklenen uygulama uzantısı ".apk"dır.

Android, aygıtların fonksiyonelliğini genişleten uygulamalar yazan geniş bir geliştirici grubuna sahiptir. Android için halihazırda 1 milyondan fazla uygulama bulunmaktadır. Google Play Store ise, Android işletim sistemi uygulamalarının çeşitli sitelerden indirilebilmesinin yanı sıra, Google tarafından işletilen kurumsal uygulama mağazasıdır. Geliştiriciler, ilk olarak aygıtı, Google'ın Java kütüphanesi aracılığıyla kontrol ederek Java dilinde yazmışlardır.

Open Handset Alliance, 5 Kasım 2007'de Android'i kurduğunu duyurmuştur ve ardından 34 adet donanım, yazılım ve telekom şirketi, mobil cihazlar için telif hakkı olmayan bir işletim sisteminin teknolojinin gelişimi için yararlı olduğu konusunda hemfikir olmuşlardır.

Android, Linux çekirdeği üzerine inşa edilmiş bir mobil işletim sistemidir. Bu sistem ara katman yazılımı, kütüphaneler ve API C diliyle yazılmıştır. Uygulama yazılımları ise, Apache harmony üzerine kurulu Java-uyumlu kütüphaneleri içine alan uygulama iskeleti üzerinden çalışmaktadır. Android, derlenmiş Java kodunu çalıştırmak için dinamik çevirmeli (ART) Android RunTime kullanır ve cihazların fonksiyonelliğini artıran uygulamaların geliştirilmesi için çalışan geniş bir programcı-geliştirici çevresine sahiptir. Google aynı zamanda işletim sistemindeki hataları bulan kullanıcıları para ödülü ile ödüllendirmektedir.

Windows Phone

Windows Phone, Microsoft tarafından geliştirilmiş ve Windows Mobile serilerinin devamı olan bir mobil işletim sistemidir. 2010 Şubat ayında Mobil Dünya Kongresi'nde ilk kez duyurulan sistem; Windows Mobile'daki kurumsal kullanıcı kitlesine nazaran Windows Phone'da hedef kitlesi Microsoft tarafından sadece son kullanıcılar olarak belirlenmiştir.

2010 Nisan ayında geliştirici araçları Microsoft tarafından indirilebilir olarak uygulama geliştiricilerine sunulmuştur. 21 Ekim 2010'da Avrupa, Singapur, Avustralya ve Yeni Zelanda'da; 8 Kasım 2010'da Birleşik Devletler ve Kanada'da; 24 Kasım 2010'da Meksika'da; 3 Aralık 2010'da PAL bölgelerinde ve 2011'in ilk tarihlerinde Asya'da kullanıma sunuldu.

Windows Phone, Microsoft tarafından geliştirilen ve 3. parti yazılımları destekleyen Metro adından bir kullanıcı arayüzü kullanmaktadır. Mango güncellemesi ile hareketli kutucuklar halini alan arayüz ayrıca tabletler için geliştirilen Windows 8'de de kullanılmıştır.

Amazon Fireos

Fire OS, Amazon'un teknolojik cihazları içerisinde kullanılan Android tabanlı bir işletim sistemidir. Fire OS, Android'in özelleştirilmesi ve Amazon tarafından kişiselleştirilmesiyle oluşturulmuş ve Kindle Fire tabletler, Fire Phone akıllı cihazlar başta olmak üzere birçok ürününde kullanılmıştır.

Taban olarak baktığımızda Fire OS'de kısmen bir Android işletim sistemidir ve Android'in sunduğu birçok özelliği içerisinde barındırmaktadır. Standart Android sürümlerinden farklı olarak Fire OS cihazların içerisinde Google Chrome yerine Silk Browser, Google Play'in yerine Amazon Appstore, Google Drive'in yerine Cloud Player ve Cloud Driver, Gmail yerineyse Amazon'un geliştirdiği e-mail client'ı yer almaktadır.

Blackberry 10

BlackBerry 10 ya da yaygın kullanılan şekliyle BB10, 30 Ocak 2013 tarihinden itibaren piyasaya sürülmekte olan yeni nesil BlackBerrycihazlarda kullanılmak üzere geliştirilen mobil platformun adıdır.

BlackBerry 10 mobil işletim sistemi, ekran üzerinden kaydırmalı (gesture) bir kullanıcı arayüzü, yeni akıllı sanal klavye ve TimeShift (Zaman Kaydırma) özelliğine sahip kamera gibi özelliklerle kullanıcıların karşısına çıktı.

Kullanıcının aynı anda 8 farklı uygulamanın küçültülerek ekrana sığmasını sağlayan "Active Frames" özelliği, uygulamaların arka tarafta çalışarak kullanıcıya gerçek zamanlı olarak bilgi akışını ekrana taşıyor. "Hub" adı verilen özellikse, işletim sisteminin tüm telefon çağrısı, e-posta, SMS, Twitter, Facebook, LinkedIn ve diğer sosyal ağlardan gelen iletileri tek bir ekranda sunmasını sağlıyor.

BlackBerry, yazılım geliştiricilerin mevcut uygulamalarını BB10 platformuna taşımalarını kolaylaştırmak adına, 28 Nisan-1 Mayıs 2012 tarihleri arasında Florida'nın Orlando kentinde düzenlenen BlackBerry 10 JAM Konferansı sırasında geliştiricilere prototip bir cihaz dağıttı. "BlackBerry 10 Alpha Device" adı verilen bu cihaz, geliştiricilere üzerinde uygulama testlerini yapabilmeleri için fiziki bir ortam sağlamaktaydı.

Java tabanlı BlackBerry OS 6.0 ve 7.x sürümlerine kıyasla, BlackBerry 10 platformu uygulama geliştiricilere daha geniş bir yazılım geliştirme dil seçeneği ve araç zenginliği sunmaktadır. BlackBerry 10 platformu için duyurulan, başlıca geliştirme dil ve araçları aşağıdaki gibidir:

- BlackBerry NDK (C/C++)
- BlackBerry WebWorks (HTML5)
- Adobe Air/Action Script (OS 10.3.1 sonrası terkedildi)
- Cascades

Firefox OS

Mozilla tarafından özellikle telefonlar için geliştirilen Linux tabanlı açık kaynak bir işletim sistemidir. İşletim sistemi ile beraber uygulamaların yer aldığı Firefox Marketplace de kullanıcılarla buluştu. İşletim sistemini barındıran ilk cihazlar Latin Amerika ve Deutsche Telekom'un bulunduğu AB ülkelerinde satışa çıkmıştır. Birçok üretici ve operatör de işletim sistemi ile ilgili çalışmalarını sürdürmektedir.

Mobil Uygulama Geliştirilirken Dikkat Edilmesi Gerekenler

Mobil cihazlar geliştikçe, bu cihazlar üzerinde çalışan yazılımlarda gelişmektedir. Peki bu yazılımlar nasıl geliştiriliyor? Mobil uygulama geliştirme tekniklerimiz nelerdir? Geliştireceğiniz mobil uygulama için nelere dikkat etmeliyiz?

- Geliştirilen uygulamanın stabil olarak çalışması
- Uygulamanın performansı
- Uygulama arayüzünün geliştirilmesi
- Kolay kullanılabilirlik ve etkileşim
- Geliştirilebilirlik ve kolay yönetilebilirlik
- Kullanılacak platformun özellikleri, araçları ve desteği

Yukarıdaki kriterler göz önüne alındığında ilk mobil uygulamamıza başlayabiliriz. Karar vermemiz gereken son bir durum kaldı. Uygulamanın geliştirme tekniği : Native mi? Hybrid mi?

Hybrid ve Native

Hybrid ve Native mobil programlamaya başlamadan önce bilmemiz gereken iki kavramdır. Mobil programlamaya başlamadan önce bu iki tekniğin ne olduğunu ve aralarındaki farkları inceleyerek başlayalım.

Hybrid

Hybrid programlama, tek bir programlama dili ile bir çok mobil platforma çıktı verme yöntemidir. Hybrid tekniği “write once, run everywhere” yani tek bir kod çıktısı ile birden fazla platformda çalıştırılabilmek üzerine geliştirilmiştir. Bu yöntem ile daha kısa sürede daha çok platforma proje oluşturabiliyoruz. Hybrid uygulamaların temellerinde web standartları vardır. Html5, Css3, Javascript, JQuery Mobile, AngularJS, Node.js gibi teknolojiler kullanılabilir. Üstelik geliştirmesi zaman ve bütçe açısından az maliyetlidir. Ayrıca Hybrid uygulama mimarisinin arkasında MIT bulunmaktadır. Bu teknolojileri kullanarak hızlı prototipleme özelliği ile bir çok platforma kolaylıkla uygulama üretebiliriz. Hybrid uygulamaların avantajları;

- Tek bir dil ve web standartlarını kullanarak birden fazla platforma uygulama yazılabilir.
- Web Standartlarına hakim bir geliştirici kolaylıkla öğrenebilir.
- Web Standartları kullanıldığından kaynak döküman ve örnek çok fazladır.
- Birden çok platforma uygulama yazıldığında zamandan tasarruf sağlar.
- Kullanılan platformun native özelliklerine erişmek için cihaz geliştiricisi tarafından yayınlanmış javascript kütüphaneleri kullanabilirsiniz.

Hybrid uygulamaların dezavantajları

- Performans bakımından, native uygulamalara göre daha yavaştır.
- Cihaz üzerindeki native özelliklere erişmek için extra kütüphaneler kullanılır ve bu kütüphaneler cihaz geliştiricisi tarafından yayınlanır. Erişmek istediğiniz native özelliğe ait bir kütüphane yayınlanmadıysa özelliği kullanamayabilirsiniz.

Native

Her işletim sistemi için, platformun sağladığı programlama dilleri kullanılarak geliştirilen uygulamalara native uygulamalara verilen isimdir. Native uygulamalar işle cihazın tüm özelliklerine kolaylıkla erişilebilir. Cihazın üzerindeki her özelliğe erişebilmesi için platformun desteklediği dilin yanı sıra yine platformun desteklediği araçlar kullanılabilir. Her platform için kendi programlama diliyle uygulama yazılması gerekmektedir.

- Platformun desteklediği programlama dili ile yazıldığından bellek yönetici daha kolay yapılmaktadır.
- Cihaz üzerindeki native (yerel) özelliklere erişmek daha kolaydır.
- Kullanılan platformun tüm özelliklerine ek bileşenler ve kütüphaneler kullanmadan erişebilirsiniz.

Native uygulamaların dezavantajları;

- Birden fazla platforma uygulama hazırlarken, kullanılacak platformlar tarafından desteklenen programlama dilleriyle geliştirme yapılır.
- Her platform için geliştirilen uygulama için extra zaman harcanır.

Doğru Seçim Hangisi?

Kullanılacak araç ve teknikleri tanıdık, bu teknikler göz önünde bulundurularak geliştirilecek uygulamanın özelliklerine göre seçim yapmalıyız. Bu durumda eğer çok fazla donanım ve etkileşime ihtiyacımız varsa native uygulama geliştirmek daha doğru bir seçimdir.

Temel olarak daha az donanım gerektiren, veri yükü daha az ve performans bakımından çok üstün performanslar beklenmeyen uygulamalar oluştururken hybrid uygulama geliştirmeyi seçmemiz daha mantıklı olacaktır.

Kısacası; doğru seçim ne native, ne de hybrid uygulama geliştirmektedir. İhtiyaç duyulan çözüm kullanıcı istek ve deneyimine göre uygulama gereksinimini karşılayacak şekilde geliştirme yapmaktır.

Hybrid ile Javascript Çatısı

Hybrid uygulama geliştirirken javascript kullanılacağını belirtmiştik. Peki geliştireceğimiz uygulamada hangi javascript çatısını tercih etmeliyiz? Belirlediğimiz kriterlerin amaçlarından bir tanesinde doğru aracı seçmektir. Hybrid tekniğinde hiç bir yan araç kullanmadan geliştirme yapılabilir. Ama bu zaman kaybı demektir.

Uygulama gereksinimine bağlı olarak hızlı, geliştirilebilir ve esnek yapıya sahip olmak için farklı ekran çözünürlük ve platformlar ile benzer uyumu yakalamayı hedefleyen javascript kütüphaneleri seçmeliyiz. Hybrid uygulamalarda kullanılabileceğiniz araçlar aşağıda listelenmektedir.

- Cordova
- jQuery Mobile
- jQTouch
- Sencha Mobile
- Titanium
- Worklight
- Kony
- Corona
- PhoneGap

PhoneGap Nedir?

Phonegap; Nitobi şirketi tarafından 2008 yılında duyurulmuş açık kodlu bir mobil uygulama geliştirme (framework) çatısıdır. PhoneGap geliştirme çatısını 2011 yılında Adobe System tarafından satın alınmıştır ve Apache geliştirme vakfına bağışlanarak Apache 2.0 Lisance ile açık kaynak olarak dağıtılmaya başlanmıştır.

PhoneGap cross platform mobil uygulama geliřtirmek için kullanılmaktadır. PhoneGap Html5, Css ve Javascript'in gücünü kullanarak Web to Native Wrappers (Web'den Native uygulamaya) mantığı ile çalışmaktadır.

PhoneGap řu anda IOS,Android, Blackberry, WebOS, Windows Phone, Symbian, Bada vb. Platformlara uygulama geliřtirmeyi destekler.

HTML5'in gücünü kullanarak sistem kaynaklarına / sensörlere erişmek teorik olarak bir dereceye kadar mümkündür ancak burada dikkat edilmesi gereken nokta Phonegap'in bu işlemi yaparken native uygulama içerisinden erişim yetkilerini alarak HTML5'e aktarmasıdır.Hybrid uygulamalar; HTML sayfaları, JavaScript kodları ve CSS dosyaları bulunan native (yerel, yüklenen) uygulamalardır. PhoneGap ile yapılan uygulamalar her bir platforma çıktı verebilir.

Apache Cordova

Apache Cordova, mobil platformlar için hybrid uygulama geliřtirmek için kullanılan bir platformdur ve yerel cihaz fonksiyonlarına ulaşmak için Javascript ile kullanılan bir API Kümesidir. Html, Css, JS kullanarak hybrid uygulamalar geliřtirmemizi sağlar.

Bir geliřtirici, Cordova API kullanarak yerel kodlama dillerini kullanmadan (Java, Objective C, Swift, C vb.) projesini oluşturabilir. Hybrid uygulamalarda web teknolojileri kullanılmaktadır ve yerel uygulama için herhangi bir host işlemi gerekmez. Hybrid uygulamalar web standartlarına göre yazıldığından dolayı birden fazla platformda çalışabilir. Web platformu standartlarına göre yazıldığı için uygulamada hiçbir deęişiklik yapmadan yada çok ufak deęişikliklerle tüm platformlarda çalışmasını kolayca sağlayabilirsiniz.

Cordova kullanarak oluşan uygulamalar SDK kullanılarak uygulama olarak paketlenebilir ve her cihaza yüklenmek için cihazın uygulama mağazasından ulaşılıp yüklenebilir. Cordova cihaza özgü özelliklere erişebilmek için cihaza özgü JavaScript kütüphaneleri sağlar ve uygulama geliřtirmeye yardımcı olur. Cordova IOS, Android, Blackberry, WP, Palm, Bada ve Symbian platformlarında kullanılabilir.

En büyük avantajlarından bir tanesi Andorid geliřtirmek için Eclipse veya Android Studio, IOS geliřtirmek içinse xCode kullanma zorunluluęumuzun ortadan kalkmasıdır. Cordova ile geliřtirme yaparken platform bağımsızlığından dolayı kolayca istedięimiz IDE yi kullanabiliriz yani Eclipse ve xCode bir zorunluluk olmaktan çıkıyor.

Apache Cordova'nın Tarihi

Apache Cordova (eski adıyla phoneGap) popüler mobil uygulama geliřtirme kütüphanesidir ve 20 geliřtiriciyle beraber Kanada, Vancouver merkezli bir web ve uygulama geliřtirme firması olan Nitobi firması tarafından geliřtirilmiştir.

Native uygulamalar geliřtirmek için web standartlarını kullanmak amacıyla iPhoneDevCamp etkinliğinde ortaya çıktı. 2009 yılında Web 2.0 Expo LaunchPad yarışmasını kazandı ve open source (MIT) lisansı altında yayınlandı.

Adobe System, Ekim 2011' de Nitobi firmasının Adobe System tarafından satın alınmasından sonra Apache Yazılım Vakfına (Apache Software Foundation) bağışlandı ve Apache Cordova ismini aldı.

Apache Cordova projesine katkıda bulunan firmalar arasında Adobe, BlackBerry, Google, IBM, Intel, Palm, RIM, Sencha Microsoft ve Mozilla gibi firmalar bulunmaktadır.

Apache Cordova Nasıl Çalışır?

Apache Cordova, günümüz web standartlarını kullanarak farklı mobil platformlar için uygulama geliřtirmenizi sağlayacak bir geliřtirme platformudur. Web Standartlarına uygun olarak geliřtirme yapıldığından iOS, Android, Blackberry, Symbian, Bada, Windows Phone 7 gibi pek çok mobil platform için geliřtirme yapılabilir. Uygulama geliřtirmek için web uygulaması geliřtirebileceğimiz herhangi bir IDE'yi kullanabiliriz. Html5, CSS3, Javascript kullanarak uygulamalarımızı geliřtirebiliriz. Geliřtirilen bu kodlar Cordova kütüphaneleri ile paketlenerek native uygulama haline getirilip, hybrid web uygulaması olarak ortaya çıkar ve uygulama User Interface'i bir WebViews olarak kullanılır.

Apache Cordova'nın Avantajları ve Dezavantajları Nelerdir?

- Cordova ile yazılan uygulamaları, tüm platformlarda çalıştırabiliriz. Ancak bunun için her platforma özel uyarlama ve ayrı derlemeler yapılması gerekmektedir.

- Cordova, Native ile Web uygulamaları arasında bir köprü durumunda olmasında dolayı hem uygulama marketlerinde yer alabilir, hem de web uygulamalarının yeteneklerini kullanabilir.
- HTML5'in gücünü kullanarak sistem kaynaklarına erişmek teorik olarak bir dereceye kadar mümkün. Ancak burada dikkat edilmesi gereken nokta, Cordova'nın bu işlemi yaparken native uygulama içerisinde erişim yetkilerini alarak HTML5'e aktarmasıdır.
- Cordova ile uygulama yapmak bir web uygulaması yapmak DEĞİLDİR. Uygulama HTML5, JavaScript ve CSS3 kullanılarak yazılıyor olsa da oluşan yükleme dosyası native'dir. Örneğin; Android sürümü için bir APK dosyası oluşur.
- Ücretsiz ve açık kaynak kodlu.
- Geliştiriciler için resmi sitesinde yeterli ve zengin doküasyon desteği veriliyor.
- Cross Platform Developer Tools 2012 raporuna göre %61 ile "Hedef Platformlara Erişim" Cordova'nın en önemli seçim nedeni.
- Cross Platform Developer Tools 2012 raporuna göre native uygulamalar, cross platform uygulamalardan hız ve performansa göre her zaman bir adım önde. Araştırma katılımcılarının %29'u ilk sırada bu durumu belirtiyor.

Apache Cordova'nın Çalışma Mantığı

Öncelikle Apache Cordova'nın nasıl çalıştığını anlamak için Chromeless Web Browser hakkında bilgi sahibi olmamız gerekmektedir. Chromeless Web Browser, Mozilla Labs'ın geliştirmiş olduğu bir projedir. Bu proje kendi tarayıcısını kendin yap mantığına dayanır. Bu sayede kişiye özel tarayıcı yapmak mümkündür.

Chromeless Web Browser içinde geliştiricinin hazırladığı web sayfalarını servis eder. Uygulamayı çalıştıran browser cihaz üzerinde tam ekran çalıştığından kullanıcı, uygulamanın bir web sayfası olduğunun farkına varmaz. Tüm mobil platformlarda web browser bulunmaktadır. Web browserlar, mobil platformlarda webView, UIWebView gibi isimlerle geliştiriciye sunulmuştur.

İsmi PhoneGap mi? Apache Cordova mı?

PhoneGap projenin başlangıçta kullandığı isimdir. Adobe System'in projeyi satın almasından ve Apache yazılım vakfına devrinden sonra Apache Lisans 2.0 ile dağıtılmaya başlanan platformun adı ise Apache Cordova'dır. Cordova, PhoneGap'in açık kaynak dağıtımıdır.

Apache Cordova ve PhoneGap Hakkında Bilinmesi Gerekenler

- Cordova, PhoneGap temelli fakat Apache geliştirme vakfı tarafından desteklenen açık kaynak kodlu bir platformdur.
- Geliştiriciler için apache cordova sayfasında yeterli seviyede doküman ve örnek bulunmaktadır.
- HTML5, CSS ve Javascript üzerinde hakimiyet sahibiyseniz kolaylıkla apache cordovayı kullanarak proje geliştirebilirsiniz.
- Farklı işletim sistemleri ve platformlar için çok daha kısa sürede ve az maliyet ile uygulama geliştirebilirsiniz.
- Chromeless web browser mantığı ile çalıştığından, web uygulaması geliştirirken kullandığınız javascript ve CSS kütüphanelerini kullanabilirsiniz.
- UI tarafı javascript ve CSS ile yapıldığından tamamen özgür tasarımlar ve kontroller kullanabilirsiniz.
- Yüksek grafik desteği veren uygulamalar, arkaplanda çalışacak bir uygulamalarda ve çok büyük verinin gerektiği uygulamalarda, native uygulamalara göre dezavantajlıdır.

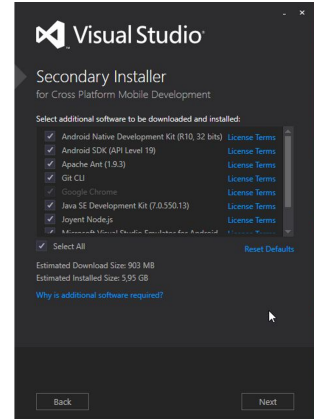
Cordova Çalışma Ortamının Kurulumu

Apache Cordova ile çalışabilmek için aşağıda verilen listede bulunan kurulumları yapmamız gerekmektedir. Öncelikle kullanacağımız işletim sistemi ve IDE'yi yüklemeliyiz.

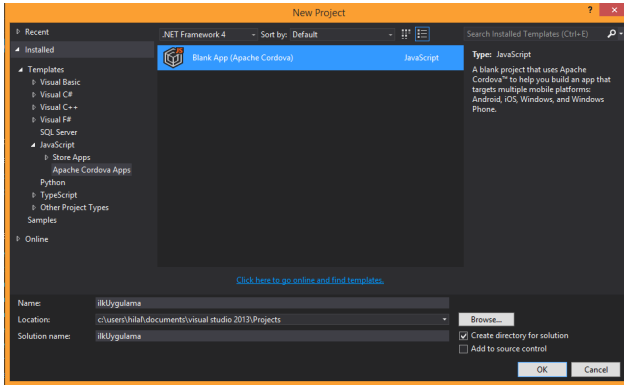
- Windows 7 ve üzeri
- Visual Studio 2013 Enterprise üzerine kurulan Apache Cordova Paketi veya Visual Studio 2015
- Android SDK (Emulator için)
- Java Runtime
- iTunes

IDE yüklemesi ardından gerekli kurulumlara başlayabiliriz. Aşağıdaki listedeki kurulumların yapılması yeterlidir. Kurulumları yapmak için <https://www.microsoft.com/en-us/download/details.aspx?id=42675> bağlantısını ziyaret ederek Visual Studio For Apache Cordova CTP3.2 kurulum dosyasını indirerek kolayca kurulum yapabilirsiniz.

- Joyent Node.js
- Google Chrome
- Git Command Line Tools
- Apache Ant
- Oracle Java 7 (JDK)
- Android SDK
- Apple iTunes
- SQLite
- WebSocket4Net



Kurulumları tamamlamak için indirdiğiniz dosyayı çalıştırın yandaki resimde görünen pencere ile karşılaşacaksınız. Install butonuna tıklayarak kurulumu gerçekleştirebilirsiniz.

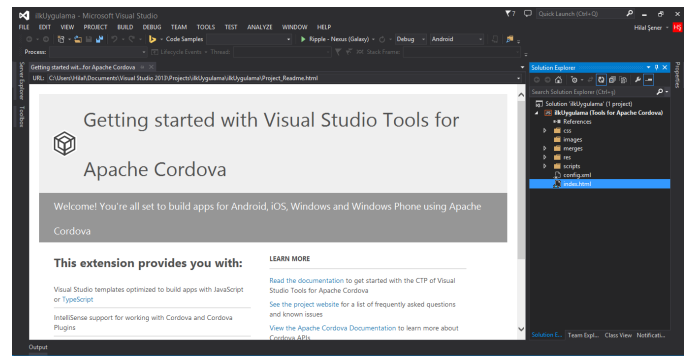


Kurulumlar tamamlandıktan sonra artık yeni bir proje oluşturabiliriz. Yeni bir proje oluşturmak için öncelikle Visual Studio IDE'yi açalım. File > New > Project yolunu izleyelim ve yeni bir proje oluşturma penceresi açın. Açılan pencerede Templates > Javascript > Apache Cordova Apps altında Blank App (Apache Cordova) projesini seçin ve bir isim verin, ardından projenin kayıt edileceği konumu seçin ve bir solution adı vererek OK butonuna tıklayarak projeyi oluşturun.

Not : Eğer yukarıdaki kurulumu yapmadıysanız Apache Cordova projesi oluşturamazsınız.

Projemizi ilk açtığımızda karşımıza sağdaki gibi bir sayfa gelecek. Solution Explorer' dan index.html dosyasını açalım ve biraz inceleyelim.

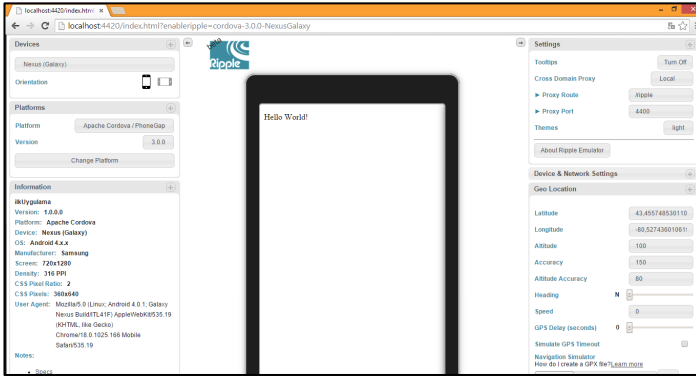
index.html' i açtığımızda aşına olduğumuz html kodları ile karşılaşacaksınız. Emulatorleri incelediğimizde Android, iOS, Windows Phone 8 vs. seçeneklerimiz var ve hatta bir yanındaki açılır menüden cihazımızın modelini dahi seçebiliyoruz. Device seçeneği ile bilgisayara bağladığımız mobil cihaz üzerinde uygulamayı deploy etmemizde mümkün olmaktadır.



Not : *Emulatör üzerinde projeyi görüntüleyebilmeniz için gerekli SDK'ların yüklü olması gerekmektedir. Örneğin Android için test edeceksek Android SDK'yı yüklemelisiniz.*

Şimdi index.html sayfasını açarak uygulamamız üzerinde html kullanarak ufak değişiklikler yapalım.

<p>HelloWorld</p> etiketlerini projemize ekledikten sonra çalıştıralım. Uygulamayı Ripple(Nexus - Galaxy) üzerine deploy ederek çalıştırın. Uygulama çalıştığında Chrome üzerinde aşağıdaki resimde gördüğünüz sayfa ile karşılaşacaksınız.



Farkettiğiniz üzere emulatör Chrome üzerinde çalışmaktadır. Tarayıcı üzerinden uygulamanın Portrait(Dikey), Landscape(Yatay) pozisyona göre ayarlamasını yapıp özizlemesini yapabiliyoruz. Geo Location, Platform vs. pek çok ayarlamayı da yine tarayıcı üzerinden yapıp özizleyebiliyoruz.

Device üzerindeki bir çok ayarı yapmanız mümkün eğer Android için Emulatör üzerinde çalışacaksanız bilgisayarınızda Android Studio kurulu olması gerekmektedir.

Git ve GitHub

Git; Günümüzde en çok kullanılan ve popüler (SVN – SubversioN) sürüm kontrolü sistemidir ve kısa sürede yazılım geliştiricilerin vazgeçilmezi haline gelmiştir. Git açık kaynak kodludur. Google, Facebook, Microsoft, Twitter, LinkedIn gibi firmalar tarafından kullanılmaktadır. Tom Preston-Werner tarafından 2007 yılında temeli atılmıştır.

Git

Git bireysel veya takım çalışması yaparken etkin bir şekilde yönetilebilen projeler oluşturmayı amaçlar. Takım olarak proje geliştirmeyi kolaylaştırır. Projeye yeni bir kişi ekleyebilir, kişilere görevler atayabilir, kişilerin analizlerini yapabilir, ana projeyi değiştirmeden yeni kodları test etmek için brachler oluşturabilir ve daha fazlasını yapabilirsiniz. Git ile projenizin yönetimini çok rahat yapabilirsiniz.

Yazdığımız projeleri, bilgisayarımızda ya da harici disklerde binbir tehlike altında değilde internet üzerinde tutmamızı ve yönetmemizi sağlayan bir sistemdir. Projemizi bilgisayarımızda, bir harici diskte ya da bir bulut sisteminde tuttuğumuzda, projede bir hata yapma veya projemizin başına bir sorun gelmesi gibi olasılıkları göz önünde bulundurarak sürekli projelerimizi yedekleme ihtiyacı duyarız. Bu sebeple projemizi versiyon versiyon kaydederiz. Projenizin büyüklüğüne göre bu işlemi yapmak oldukça karmaşık haller alabilir ve disk üzerindeki boyutu artacağından disk sıkıntısı çekmeye başlayabiliriz. Git bizi bu karmaşadan kurtarır. İstedığınız anda projenin son kayıt edilmiş haline ulaşabilir veya istediğimiz bir güne geri dönüş yapabiliriz.

Git aynı zamanda bize aynı proje üzerinde, birden fazla kişi ile eşzamanlı olarak ya da farklı zamanlarda çalışıyorsanız kodlarınızı birleştirmek ya da kod alışverişi yapmak oldukça sorun çıkartmaktadır. Fakat Git kullanıyorsanız bu işlemleri yapmanız hiç de zor olmayacaktır. Çünkü birden fazla kişi ile eşzamanlı veya farklı zamanlarda çalışıyorsak, kodları kolayca birleştirme imkanı sunar. Normal durumlarda kod alışverişi yapmak oldukça zordur fakat Git ile bu işlem oldukça kolaylaşmaktadır. Ayrıca Git bilgisayarda oldukça az yer tutar ve hızlı çalışır.

Git Kurulumu için <https://git-scm.com/downloads> bağlantısını ziyaret ederek sisteminize uygun kurulum dosyasını indirebilirsiniz.

Github

Github; versiyon kontrol sistemi için kullanılan bir havuzdur. Birçok yazılımcının bir araya geldiği, bir projenin kopyası üzerinde çalışarak projenin alt versiyonlarının çıkarıldığı sosyal bir kodlama alanıdır. Github üzerinde oluşturulmuş bir projenin kopyasını alıp, bilgisayarımız üzerinde değişiklikler yaparak proje sahibine gönderebiliriz. Tabiki github üzerinde bir projede oluşturabiliriz.

Github kullanabilmek için site üzerinde bir hesap oluşturmali ve giriş yapmalıyız. Tabiki bilgisayarımıza git'in kurulumunu yapmamız gerekmektedir.

Git Kurulumu ve Kullanımı

Git kullanabilmek için öncelikle bilgisayarımıza kurulum yapmamız gerekmekte ve bir github hesabı oluşturmamız gerekmektedir. Kurulum için <https://git-scm.com/> adresini ziyaret edebilirsiniz. Aynı zamanda bu adres üzerinden çeşitli dökümantasyonlarına ulaşabiliriz. Git Windows, Linux, Mac ve Solaris platformu için indirip kullanabilirsiniz. Github'a kayıt olmak için ise <https://github.com/> sitesini kullanabilirsiniz.

GitHub sitesine kayıt olduktan sonra kendi sayfamızı görebiliriz. Bu sayfa bizim projelerimizin bulunduğu sayfadır ve kullanıcılar bize kullanıcı adımızı kullanarak erişebilir. Github ilk açıldığında bilgisayara kurma, proje oluşturma, yönetme ve takım oluşturma gibi işlemleri anlatan ipuçları karşımıza çıkmaktadır.

Download sayfasından indirme yaptıktan sonra kurulumu tamamlayınız. Bir arayüzü bulunmayan git arkaplanda çalışmaktadır. Git'i kullanırken 'Git Shell' isimli bir console üzerinden kullanılmaktadır. Git kullanmak için gerekli komutları bilmemiz gerekmektedir. Kurulumu tamamladıktan sonra console ekranını açarak komutlarımızı yazabiliriz. Tabiki git shell dışında görsel olarak kullanmamızı sağlayan programlarda bulunmaktadır. İki yöntemide inceleyelim.

GitHub ücretsiz olarak kullanıldığında attığımız tüm projeler herkes tarafından görülebilir olur.Özel projeleriniz için her bir Repository oluşturduğunuzda aylık \$7 ödemeniz gerekmektedir.

Command Line Interface (CLI)

Cordova üzerinde bulunan Command Line Interface (Komut Satırı Arayüzü) herhangi bir IDE'ye ihtiyaç duymadan, kod satırlarını kullanarak proje oluşturmak, güncellemek, plug-in eklemek/çıkartmak, yeni platform eklemek çıkartmak gibi işlemlerin tek merkezden yönetilmesini sağlar. Kısacası Cordova komutu arayüzünü oluşturur.

Cordova 3.0 ile birlikte kullanılmaya başlanan CLI, <http://phonegap.com/install/> sayfasında Cordova kütüphanesinin indirilebilir halleri en son PhoneGap 2.9.1 sürüme kadar devam ettirilmiştir. Cordova 3.0 ile birlikte bu işlem tamamen CLI aracılığı ile yapılmaya başlanmıştır.

CLI, geliştirme aşamasında geliştiriciyi ortam bağımsız hale getirir. Örneğin Android platformu için bir Cordova projesi geliştiriyorken Eclipse editörünü hiç açmadan tüm işlemlerinizi CLI üzerinden yürütüp favori web geliştirme aracınızı kullanarak geliştirmeye devam edebilirsiniz. Özellikle 3.0 versiyondan önce Cordova çalışma ortamını kurmak oldukça zor bir işlemdi. Kurulumlar uzun süren ve karışık işler olduğundan geliştiriciler bu işlemlerden kurtulmak için Eclipse üzerinden kurulan PhoneGap plug-in'ini kullanıyordu. Artık bu plug-in'e ihtiyaç duymadan Cordova CLI ile tüm bu işlemleri gerçekleştirmek oldukça kolaylaşmıştır.

Kullanılabilir CLI Komutlarını Öğrenmek

CLI komutlarını kullanmak ve komutlar hakkında daha fazla bilgi edinmek için CLI üzerinde aşağıdaki komutu kullanabilirsiniz. Bu komut CLI üzerinde kullanabileceğiniz komut listesini vermektedir.

Cordova Komutlarını Öğrenmek

```
cordova help
```

CLI Kullanımına Hazırlık

Cordova CLI kullanmak için;

- <http://nodejs.org> adresinden **Node.js kurulum dosyasını indirin ve bilgisayarınıza kurulum yapınız.** Bu işleminden sonra "**npm**" komutu bilgisayarınızın komut satırından kullanılmaya hazır hale gelecektir.
- **Node.js** kurulumunu gerçekleştirdikten sonra "**npm install -g cordova**" komutu kullanılarak Cordova CLI komutları kullanıma hazır hale gelmiş olur.

- CLI'yi çalıştırmadan önce, kullanacağınız her platformun SDK'sının bilgisayarınızda kurulu olması gerekiyor. Bu sebeple hangi platform ya da platformlar üzerinden proje oluşturacaksınız, bilgisayarınızda bu SDK'ların bulunması ve tanımlanmış olması gerekiyor.

Önemli Not: Visual Studio Cordova Pack ile kurulum yaptıysanız, yukarıdaki adımları uygulamadan cordova komutlarını kullanamazsınız. Bu işlem birkaç dakika sürebilir

Cordova Komutlarını Kullanıma Hazırlamak

```
npm install -g cordova
```

Console ekranında projeyi hangi dizine oluşturmak istiyorsak, console ekranında ilgili dizine geçiş yapın. Console ekranında dizin değiştirmek için komut satırı içerisinde `dir` veya `cd` komutlarını kullanabilirsiniz.

Kullanabileceğiniz Ms Dos Komutları

`dir` klasorYolu //Command Prompt ekranında verdiğiniz klasöre gider.

`cd..` //Bulunduğunuz klasörden bir üst klasöre çıkar.

`cd` klasorAdi //Bulunduğunuz dizinde bulunan bir klasöre gider

Çalışacağımız klasörü oluşturduktan veya açtıktan sonra artık cordova projesini oluşturabiliriz ve geliştirmelerimizi yapabiliriz.

Bir Cordova Projesi Oluşturmak

CLI üzerinden yeni bir cordova projesi oluşturabilirsiniz. Cordova uygulamalarını visual studio olmadan geliştirmeniz gereken durumlarda CLI üzerinden proje oluşturmanız gerekmektedir.

Cordova Projesi Oluşturmak

```
cordova create ProjeKlasoruAdi com.etkiAlani.ProjeAdi ProjeAdi -d
```

Yukarıdaki komut satırını incelediğimizde sırasıyla;

- `cordova create` : Yeni bir cordova projesi oluşturmak için kullanılır.
- `IlkProjem` : Projenin bulunacağı dizinin adını belirtir.
- `com.example.IlkJem` : reverse domain-style (ters domain stili) tanımlayıcıdır. Bu tanım dikkatli yapılmalıdır, uygulama markete atıldığında görünmektedir.
- `IlkJemimiz` : Oluşturulan projemizin adıdır.
- `-d` : Bu parametre, komutun o anda hangi işlemlerin gerçekleştirildiğini bilgi olarak gösterir.

Cordova Uygulamasına Platform Ekleme

Önceki adımları takip ederek yeni bir proje oluşturduktan sonra, oluşturulan projeye platform eklememiz gerekmektedir. Platform ekleme işlemi için aşağıdaki komut satırlarını kullanabilirsiniz. Öncelikle oluşturulan projenin dizini içerisine giriniz. Daha sonra aşağıdaki komutları kullanarak projeye platform ekleme işlemi yapabilirsiniz. Daha detaylı ve güncel bilgi için Cordova dokümantasyonundaki **"Platform Guides"** konusunu inceleyebilirsiniz.

Ekleme istediğiniz platformu seçerek projeye ekleyebilirsiniz. Aşağıdaki komut satırları arasından seçimler yaparak istediğiniz platformları ekleyebilirsiniz.

Cordova

```
cordova platform add android
cordova platform add ios
cordova platform add amazon-fireos
cordova platform add blackberry10
cordova platform add firefoxos
cordova platform add wp7
cordova platform add wp8
cordova platform add windows8
```

Yukarıdaki komutları kullanarak bir platform ekledikten sonra platformu kaldırmak istediğimizde ise aşağıdaki iki komut satırından birini kullanabiliriz.

Cordova Projesinden Platform Kaldırmak

Projenize eklediğiniz bir platformu kaldırmak istediğimizde CLI komutlarından remove komutunu kullanabiliriz.

Platform Kaldırma

```
cordova platform rm ios //ios yerine kaldırmak istediğiniz platform adını yazın.
cordova platform remove ios //ios yerine kaldırmak istediğiniz platform adını yazın.
```

Birden fazla platformu aynı anda kaldırmak için aşağıdaki komut satırlarından birini kullanabilirsiniz.

Birden Fazla Platform Kaldırma

```
cordova platform rm android ios//android, ios yerine kaldırmak istediğiniz platform adını yazın.
cordova platform remove android ios //android, ios yerine kaldırmak istediğiniz platform adını yazın.
```

Cordova'da Kurulu Platformlar

Cordova içerisinde kurulu olan platformları listelemek için aşağıdaki komut satırını kullanabilirsiniz. Platformları görüntülemenin iki yöntemi bulunmaktadır. İlk olarak Cordova içerisindeki platform klasörü içerisinde platformun olup olmadığını kontrol edebilirsiniz. Bu işlemi CLI kullanarak yapmak istediğinizde aşağıdaki komut satırlarından birini kullanabilirsiniz.

NOT : Platform kontrolü yaparken cd ile proje dizinine gitmeniz gerekmektedir.

Kurulu Platformları Öğrenmek

```
cordova platform ls
cordova platform list
cordova platforms
```

Cordova Versiyonu Öğrenmek

Kullandığımız Cordova versiyonunu öğrenmek için aşağıdaki komutlardan birini kullanabilirsiniz.

Cordova Versiyonunu Öğrenmek

```
cordova -version
```

```
cordova -v
```

Cordova Sürümünü Güncellemek

Kullandığınız Cordova sürümünü, en son cordova versiyonuna güncellemek için aşağıdaki komut satırını kullanabilirsiniz. Bu işlemten sonra Cordova en son versiyona güncellenecektir. Bunun dışında dilerseniz cordovayı istediğiniz bir versiyonada güncellemeniz mümkündür.

Cordova Sürümünü Güncellemek

```
npm update -g cordova //Cordova versiyonunu en güncel versiyona günceller.
```

Cordova sürümünü, en son sürüm yapmak yerine istediğimiz bir sürüme güncellemek istiyorsak. Öncelikle bu işlem için ilgili Cordova sürümünü sistemimize yüklememiz gerekmektedir. Aşağıdaki komutu kullanarak bu işlemi yapabilirsiniz.

Cordova Sürümünü İstedığınız Versiyona Güncellemek

```
npm install -g cordova@3.1.0-9-2-0 //İstedğimiz bir versiyona güncellemek için kullanılır.
```

Yukarıdaki komut Cordova 3.0.0 sürümünü sisteminize yükleyecektir. Eğer farklı bir sürüm yüklemek istiyorsanız yalnızca sürüm numarasını değiştirerek komutu kullanabilir ve istediğiniz versiyonu yükleyebilirsiniz.

Cordova Platform Versiyonunu Güncelleme

Uygulamada belli bir platforma göre güncelleme yapmak içinse aşağıdaki komutlardan birini kullanabilirsiniz.

Platform Versiyonlarını Güncellemek

```
cordova platform update android //android yerine güncellemek istediğiniz platform adını yazın.
```

```
cordova platform up android //android yerine güncellemek istediğiniz platform adını yazın.
```

Cordova Plug-in'leri Nasıl Yüklenir?

Cordova API'lerinden kullanmak istediğiniz pluginleri yüklemeniz gerekiyor. Bunun için aşağıdaki komut satırını kullanabilirsiniz. Biz device plug-in'i üzerinden örnekleme yapacağız.

API : Açılımı Application Programming Interface olan API; bir yazılımın başka bir yazılımda tanımlanmış fonksiyonlarının kullanılması için oluşturulmuş bir tanım bütünüdür.

Kullanmak üzere bir plugin araması yapmak için aşağıdaki komutu kullanabilirsiniz. Cordova sitesinde arama yaptığınız "camera" kelime geçen tüm pluginler tarayıcınızda görüntülenecektir.

İsme Göre Plugin Araması Yapmak

```
cordova plugin search camera
```

Açılan sayfada bulduğunuz plugin başlığını aşağıdaki komutta 'cordova-plugin-camera' yerine yazarak istediğiniz plug-in'in yüklenmesini sağlayabilirsiniz.

Plugin'i Projeye Dahil Etmek

```
cordova plugin add cordova-plugin-camera
```

CLI Üzerinden Plugin Kullanımı

CLI komutları ile projemize plugin dahil edebiliyoruz. Kullanacağımız pluginlerin adreslerini, dökümantasyonlarını ve plugin araması yapabileceğimiz adresleri inceleyelim.

Plug-in bulma ve hakkında bilgi edinme işlemi oldukça basit bir işlemdir. Bu işlem için aşağıdaki plug-in sayfasına giriş yapınız.

<https://cordova.apache.org/plugins/?platforms=cordova-android>

Yukarıdaki sayfası ziyaret ettikten sonra aşağıdaki filtreleri kullanarak platformlara göre plug-inleri filtreleyebilirsiniz. Plug-in altında hangi platformu desteklediklerini görebilirsiniz.



Cordova Plug-in Nasıl Kaldırılır

Uygulamanızda kullandığınız pluginlerden herhangi birini kaldırmak için aşağıdaki komut satırı ile kaldırabilirsiniz. Adres parametresi olarak kaldırmak istediğiniz plug-in adresini yazmanız yeterlidir.

Plugin Kaldırmak

```
cordova plugin remove cordova-plugin-camera
```

Kurulu Plug-in Listelemek

Uygulamamızda kurulu olan plug-in listesini görmek için aşağıdaki komut satırlarından birini kullanabilirsiniz.

Projemizdeki Kurulu Pluginleri Listelemek

```
cordova plugin ls  
cordova plugin list
```

Projeyi Derlemek

Uygulamamızı, eklediğimiz platformlara göre belirli bir platformda hazırlamak/derlemek için aşağıdaki komutu kullanabilirsiniz. Aşağıdaki komut projenizi yükle olan tüm platformlara göre build edecektir.

Projeyi Derlemek

```
cordova build
```

Tabiki oluşturduğunuz projeyi sadece istediğiniz bir platforma görede derleyebilirsiniz. Bu işlem için aşağıdaki komut satırını kullanabilirsiniz.

Projeyi İstedığımız Platform için Derlemek`cordova build android`

Not: Root içerisinde yapılan işlemleri platformlar içerisine kopyaladığı için platform altında özel bir işlem yaptıysanız bu dosyalar ezilecektir.

Uygulamayı Yerel Sunucu Üzerinde Çalıştırmak

Oluşturduğunuz ve geliştirme aşamasını tamamladığınız uygulamayı yerel sunucu üzerinde çalıştırabilirsiniz. Eğer bir Windows işletim sistemi kullanıyorsanız uygulamanızı yayınlamak için IIS (Internet Information Service) hizmetinin kurulu olması gerekmektedir. Bir MAC bilgisayar kullanıyorsanız Apache Tomcat kullanabilirsiniz. Local sunucu üzerinden uygulamalarınızı çalıştırmak için aşağıdaki komutu kullanabilirsiniz.

Yerel Sunucu Üzerinde Çalıştırma`cordova serve`

Uygulamayı tek bir platform için çalıştırmak istediğinizde aşağıdaki komutu kullanabilirsiniz.

Tek Bir Platformu Yerel Sunucu Üzerinde Çalıştırma`cordova serve android`

Yukarıdaki komutu, komut satırına girdikten sonra web tarayıcınızın adres satırına <http://localhost:8000> portu üzerinden uygulamayı çalıştırabilirsiniz. Tabiki bu portu kullanabilmeniz için bu porta izin vermeniz gerekiyor.

Uygulamayı Emulator Üzerinde Çalıştırmak

Yazdığınız uygulamaları Emulator üzerinde test edebilirsiniz. Burada dikkat edilmesi gereken nokta kullanacağınız platforma ait emulatörü çalıştırabilmek için platforma ait emülatörlerin tanıtımlarının yapılmış olması gerekmektedir. Örneğin Android ile bir uygulama geliştirdiğinizde, uygulamanın Emulator üzerinde çalışabilmesi için bilgisayarınızda Android Studio kurulu olmalıdır. Projeyi bir platform üzerinde test etmek için, hangi emulatörde çalıştırmak istiyorsanız aşağıdaki komut satırını kullanabilirsiniz.

Emulator Üzerinden Çalıştırmak`cordova emulate android`

Uygulamanızı bir mobil cihaz üzerinde test etmek istiyorsanız bu işlem için aşağıdaki kod satırında yazdığınız platforma ait cihazın bilgisayarınıza USB portu üzerinden bağlanması gerekmektedir. Dikkat etmeniz gereken nokta cihazınızı device üzerinde test ederken mobil cihazınız bilgisayarım içerisinde görünmelidir. Android SDK üzerinden google'ın usb bağlantılarına izin ve yetkisi verilmiş olması gerekmektedir. Ayrıca kullandığınız cihaz üzerinde projeniz açılmıyorsa, cihazınızın usb kablosu zayıf olduğundan çalışmama durumu ortaya çıkabilir. Farklı cihazlarda test ediniz.

Cihaz Üzerinde Çalıştırmak`cordova run android`

Yukarıdaki komutta android parametresi yerine istediğiniz platformu parametre olarak kullanabilirsiniz. Verdiğiniz parametreye göre uygulamanız emulator üzerinde çalıştırılacaktır.

Cordova Dizin Yapısı

Cordova CLI ile bir proje oluşturduğunuzda aşağıdaki dizin yapısına göre projeniz oluşacaktır. Bu bölümde dizin yapılarını tanıyarak başlamalıyız.

- myProject/merges/
- myProject/www/
- myProject/platforms/
- myProject/plugins

Cordova ile bir proje oluşturduğunuzda yukarıdaki klasörlerden www klasörü içinde dosyalar bulunmaktadır. Diğer klasörlere göz attığınızda klasörlerin boş olduğunu görebilirsiniz. Peki bu klasörler ne işe yarar? Projemiz ilerledikçe hangi klasör, hangi dosyaya ev sahipliği yapacak?

‘www’ Klasörü

Projemiz içerisinde sayfalarımız ile ilgili dosyaların (html, css, js vb.) bulunduğu klasördür. Daha önce incelediğimiz CLI komutlarından “cordova prepare” komutunu kullanarak www klasörünün içerisindeki öğreleri platformlara ait www klasörünün içerisine taşır. Yani www klasörü, tüm platformlara ait dosyaların bulunduğu ortak klasördür. Bu klasör üzerinde CLI komutlarını kullandığımızda, bu dosyalar platformlara ait www klasörlerine taşınır.

“platforms” Klasörü

Projemize yeni bir platform eklediğimizde, platforma ait native öğeler bu klasör içerisinde oluşur. Dikkat edilmesi gereken nokta oluşan öğeler platform isminde taşımaktadır. Projemizin IOS platformunu xCode üzerinden oluşturmak istersek, xCode platforms klasörü içerisindeki ios klasörünü kullanacaktır. Öğelerin olduğu klasör yapıları aşağıdaki gibidir.

- myProject/platforms/android/
- myProject/platforms/ios/

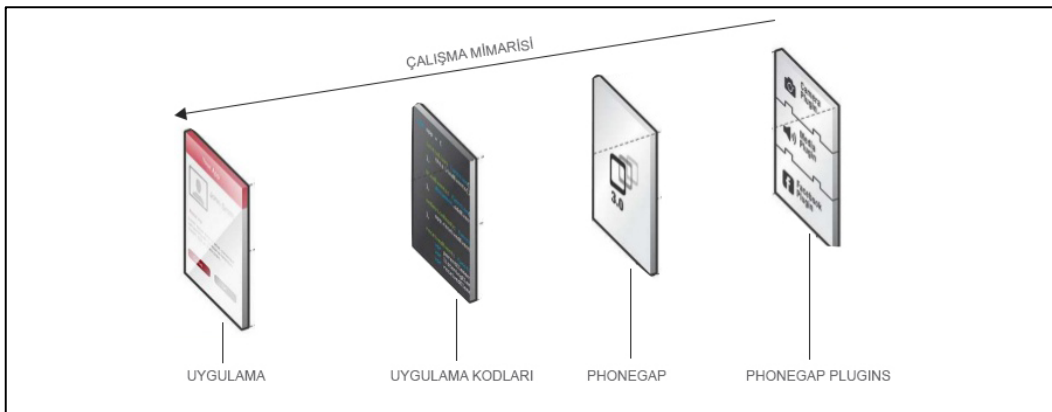
Projemize CLI kullanarak veya başka yöntemlerle yeni bir platform eklediğimizde merge ve plugins klasörü içerisinde platforma ait dosyalar ve klasörler oluşmaktadır.

Merge klasörü içerisinde ios ve android klasörleri içleri boş olarak oluşturulurken, plugins içerisine android.json ve ios.json dosyaları oluşturulacaktır.

Plugins Klasörü

Projemize eklediğimiz tüm eklentiler (plugin), eklentiler klasörü altında toplanır. Tabiki sadece eklentiler değil, projemize bir platform eklediğimizde, platforma ait eklentilerin tutulduğu json dosyalarıda plugins klösörü altında oluşturulur. Bunun dışında önemli bir detay olarak; Cordava 3.0 çekirdeğinde plugin bulunmamaktadır. Bu sayede projede bulunmasını istediği pluginleri geliştirici kurmalıdır.

Cordova mimarisini aşağıdaki resimden inceleyebilirsiniz.



Uygulamamıza yeni bir pluing eklediğimizde, eklediğimiz eklentinin plugins klasörü içerisinde bir klasör oluşturduğunu gözlemleyebiliriz. Daha sonraki konularımızda plugin ekleme işlemi yapacağız. Bir plugin eklediğimizde **plugins** klasörü altındaki **android.json** ve **ios.json** dosyalarımız içerisinde değişiklikler olacaktır. Yapılan değişiklikler eklenen plugin ile ilgili bilgilerim json dosyalarına işlenmesidir.

“Merges” Klasörü

Bildiğiniz gibi Cordova, hybrid uygulamalar oluşturmamızı sağlar yani pek çok mobil platform için uygulama geliştirmemizi sağlamaktadır. Eğer bir çok platform kullanıyorsak merge klasörü çok büyük bir önem kazanmaktadır. Farklı platformlarda farklı özellikler/dosyalar (Örneğin; android ve ios için geliştirme yaparken yazı rengi, boyutu ve tipi platforma göre değişiklik gösteriyorsa) kullanmak istiyorsak merges klasörünü kullanabiliriz.

Apache Cordova ile Proje Geliştirmek için Kullanılan Teknolojiler

Apache Cordova'nın hybrid mobil geliştirme çatısı olduğundan ve web standartları üzerine kurulu olduğundan bahsetmiştik. Apache Cordova ile proje geliştirirken kullanacağımız teknolojileri tanıyalım.

Html

Hyper Text Markup Language (HTML), web sayfaları geliştirmek için kullanılan metin işaretleme dilidir. İlk olarak 1993 yılında ortaya çıkmıştır ve günümüzde geliştirilmeye devam etmektedir. Html son sürümü olan HTML5 ile günümüz web sayfaları yapılmaktadır ve HTML5, W3C tarafından geliştirilmektedir. Html bir programlama dili değildir yani HTML ile kendi başına çalışan programlar yazılamaz. HTML5 ile birlikte bir çok yenilik ve standart gelmiştir.

Css

Cascading Style Sheets (Basamaklı Stil Şablonları), HTML ile birlikte çalışan ve HTML etiketlerin üzerinde görsel değişiklikler ve düzenlemeler yapmak için kullanılan bir web teknolojisidir. CSS kullanarak Html etiketlerine renk, boyut, arkaplan rengi gibi bir çok özellik kazandırılabilir. Css'in en son sürümü CSS3'dür. Css3 ile birlikte CSS ile basit animasyon yapımı gibi özellikler gelmiştir.

Javascript

1995 yılında Mocha ismiyle çıkan, daha sonra LiveScript ve Javascript isimlerini alan bir script dilidir. Script dili olduğundan HTML olmadan çalışamaz ve kendi başına bir anlam ifade edemez. Client Side bir dil olduğundan Javascript kullanıcı bilgisayarında çalışmaktadır.

AngularJS

AngularJS google firmasının destek verdiği, javascript MVC kütüphanesidir. MVC prensibi Model-View-Controller mantığında verinin, kullanıcı arayüzünün ve işlemleri yapacak yapıların birbirinden ayrılması prensibine dayanmaktadır.

AngularJS'in çalışabilmesi için jQuery gibi kütüphanelere ihtiyaç yoktur. AngularJS'in en önemli farkı çift yönlü işlem yapabilmesidir. Veride bir değişiklik olduğunda görünüme, görünümde bir değişiklik olduğunda veriye müdahale edebilir. AngularJS kod satırlarını azaltmak ile birlikte kod karmaşasına engel olmaktadır. Yaklaşık 100 KB'lık bir kütüphane olan AngularJS'i <https://angularjs.org> adresinden indirebilirsiniz. AngularJS bileşenleri içeren modüller olarak düzenlenmiştir. Bileşenleri Direktifler (Directives), Hizmetler (Services), Sağlayıcılar (Providers), Tipler (Types) olarak sıralayabiliriz. Bu bileşenler AngularJS'in genel apileridir. Angular kütüphanelerini aşağıdan inceleyebilirsiniz. **ng** : angularJS nin temel modüllerini içerir. Bir angularJS uygulaması başlatıldığında bu modül varsayılan olarak yüklenir.

- **ngRoute** : Hybrid uygulamada Url yönetimi yapılmasını sağlamak için kullanılır. (angular-route.js)
- **ngAnimate**: Uygulamamızda animate metotlarını kullanmamızı sağlar. (angular-animate.js)
- **ngResource** : Rest api veri yönetimi için kullanılır. (angular-resource.js)
- **ngCookies**: Cookie yönetimini sağlar. (angular-cookies.js)
- **ngTouch**: Mobil tarayıcılar için geliştirilen uygulamalarda kullanılır. (angular-touch.js)

- **ngSanitize**: HTML verilerini güvenli bir şekilde ayrıştırmak ve işlemek için kullanılır. (angular-sanitize.js)
- **ngMock**: Test modülleri için kullanılır. (angular-mock.js)

Node.JS

Açık kaynak kodlu, server side çalışan ve ağ teknolojisini kullanan uygulamalar geliştirmek için kullanılan bir framework'dür. Node.js uygulamaları genelde client side çalışan bir dil olan Javascript ile kullanılır. Bu teknoloji gerçek zamanlı web uygulamaları geliştirmek için kullanılmaktadır bu sebeple kullanım alanı her geçen gün artmaktadır.

Node.js ile gerçek zamanlı sohbet uygulamaları, push notification sistemleri geliştirmek mümkün kılınmaktadır.

Mobil Projelerde Performans İpuçları

Geliştirdiğimiz uygulamalarda kullanılabilirlik ve kullanıcı deneyimi kadar performansda önemlidir. Mobil proje geliştirirken performans dikkat edilmesi gereken en önemli noktayı oluşturmaktadır. Projelerde performans sağlamak için aşağıdaki başlıkları incelemeliyiz.

DTO Kullanımı

Bir web api üzerinden veri çekeceğimiz durumlarda, tüm bilgileri client'a göndermek sayfalarımızda gecikmelere neden olacaktır. Örneğin entity framework ile bir yansıma aldığımızda tamamen class'ı client'a göndermek yerine bir Data Transfer Object (Veri Transfer Nesnesi) oluşturarak sadece gönderilmesini istediğimiz verilerin propertylerini oluşturabiliriz. Bu yöntemle aktarılabilecek veri miktarı düşürüldüğünden performans kazanılabilir.

Uygulamaya Göre Yaklaşım

Mobil bir proje oluştururken projenin büyüklüğüne göre bir tasarım yaklaşımı seçilmektedir. Günümüzde oldukça popüler olan Single Page yaklaşımlar genellikle küçük projelerde tercih edilmelidir. Daha büyük projelerde Multi Page yaklaşımını kullanmalıyız.

Single Page uygulamalar tüm işlemleri tek bir sayfa altında tutacağından dolayı uygulamamızda yavaşlama söz konusu olacaktır.

Fakat uygulama ne kadar büyük olursa olsun, multi page uygulamalarda her sayfa kendi işlerinden sorumlu olacağından dolayı hız konusunda bir problem yaşanmayacaktır.

FastClick.js

Daha önce Hybrid uygulamalardan bahsederken Chromeless web browser mantığında çalıştığından bahsetmiştik. Bilgisayarlarımızda kullandığımız mouse hareketleri farklı olduğundan hybrid uygulamalarda Html kontrollerini kullandığımızda bu butonun çift tıklama olarak algılanmaması için 300ms'lik bir gecikme verilir. Web uygulamalarında bu gecikme rahatsız edici olmasada Hybrid uygulamalarda daha akıcı geçişler elde etmek için bu gecikmeyi ortadan kaldırma amacı ile FastClick.js kullanabiliriz.

Framework 7

<http://www.framework7.io> adresine girdiğimizde, Framework 7 kütüphanesi bizi karşılıyor. Framework 7 kütüphanesi, hybrid teknolojilerle mobil uygulama geliştirmemizi sağlayan bir kütüphanedir. Bir mobil uygulama yazdığımızda IOS için Objective C, Android için Java, Windows Phone için C# gibi native diller kullanmalıyız. Günümüzde uygulama ihtiyacının artması sebebiyle Framework 7 gibi yapılar ortaya çıktı. Bu yapı sadece Html, Css ve Javascript'e özel bir yapı değil. Bu cross platform dediğimiz bir yapıdır. Örneğin Xamarin kullanarak C# ile uygulama yazarak, birden fazla platforma mobil uygulama geliştirebiliyorsunuz.

Biz içeriğimizde Html, Css ve Javascript üzerine yoğunlaşıyor olacağız. Burada Html5'in gücüyle beraber bir çok css ve js kütüphanelerini kullanarak uygulama geliştireceğiz. Framework 7 kütüphanesi MIT lisanslı bir open source kütüphanedir. Framework 7 bize bir css ve bir js kütüphanesi verir. Belli tasarım tercihleriyle, çeşitli fonksiyonları kolayca kullanmamızı sağlar.

Framework 7'nin sayfasına ilk giriş yaptığımızda, karşımıza çeşitli bağlantılar gelecektir. Sayfamızda Framework 7 template'ini önizleyebileceğimiz bir cihaz görünecektir. Yanındaki IOS ve Android butonlarına tıklayarak tema tasarımını değiştirebilirsiniz.

UI Components

Framework 7 ile birlikte pop-up, liste görünümüleri, medya listeleri, sekmeler, yan paneller, etkinlik göstergesi, form kontrolleri gibi UI üyeleri ve Widget eklentileri hazır olarak gelmektedir. Framework 7'nin en büyük avantajı Widget kullanımları için javascript kullanımına gerek kalmamaktadır. UI araçları içerisinde Swipe Back, Dynamic bar gibi bir çok özellik bulunmaktadır.

Göze Çarpan Özellikler

Framework 7 içerisinde bir çok göze çarpan özellik bulunmaktadır. Hybrid uygulamalarda kolaylıkla yapabileceğimiz işlemlerden bir kaçını aşağıdan inceleyebilirsiniz.

Swipe Back

Bir önceki sayfaya geçmek için kullanılan bir özelliktir. Ekran üzerinde IOS Jest hareketlerini kullanarak uygulamalarımızda bir önceki sayfaya geçmek için ekran üzerinde el hareketi yapmak yeterlidir. Swipe Back Native bir uygulama gibi mükemmel olarak çalışır. Animasyon parmak hareketinizi takip eder ve mükemmel bir swipe back deneyimi sağlar.

Swipe Actions

Uygulamamız üzerindeki bir seçeneği sola doğru parmak hareketi ile sürüklediğimizde sağ tarafta bir seçenek penceresi çıkacaktır. Framework7 aynı mükemmel animasyon ve dokunmatik etkileşimi, herhangi bir liste elemanları için tamamen aynı işlevselliğe sahiptir!

Dynamic NavBar

Uygulamalarımızda ki en önemli özelliklerden birisi NavBar kullanımıdır. Etkin olarak NavBar kullanımı uygulamalarınızda ki kullanıcı deneyimini arttıracaktır. Dinamik NavBar özelliği ile Gezinti çubuğunu sayfalar arası geçiş için kaydırabilirsiniz.

Pull To Refresh


Bir çok yerel uygulamada kullanılan bu özellik sayesinde uygulamada verileri yenileme işlemini ufak bir parmak hareketiyle yönetebilirsiniz. Uygulama ekranını yukarıdan aşağıda doğru çektiğinizde uygulamada ki verileri güncelleyebilir farklı işlemler yapabilirsiniz. Framework 7'de her şey tek parmak hareketiyle..


Messages

İtici veya PubNub gibi bazı gerçek zamanlı senkronizasyon veri servisini kullanan kullanıcılar veya cihazlar arasındaki mesajlaşma uygulamanıza entegre edebilir. Framework 7 bunun gibi Widget özellikleriyle birlikte geliyor. Framework 7 ile birlikte Swiper ve Template 7 isimli iki kütüphane daha kullanılabilir.

BuildPhoneGap ile Proje Derleme

Build Phone Gap, cordova projelerimizi online ortamda derlemek ve test etmek için kullanılan bir web uygulamasıdır. Oluşturduğumuz cordova projesini <http://www.build.phonegap.com> adresine giriş yaparak, adobe kullanıcı hesabımız ile oturum açtıktan sonra build.phonegap üzerine yükleyerek derleme işlemlerinin yapılmasını sağlayabiliriz. Eğer bir hesabınız yoksa öncelikle bir adobe hesabı oluşturmanız gerekmektedir. Build phonegap web sayfasının anasayfasını aşağıdaki resimde görebilirsiniz.

 Home Plugins Docs Blog FAQ Support

Sign in Sign up 



Adobe® PhoneGap™ Build

Package mobile apps in the cloud.

[Get started!](#)

Bir kullanıcı hesabınız yoksa sign up sayfasına giderek yeni bir hesap oluşturun. Build.PhoneGap üzerinden 1 projeyi ücretsiz olarak build edebilirsiniz. Kayıt olma sayfasından gerekli tabloyu inceleyerek hesap planınızı seçmeniz gerekmektedir.


| | Free Plan | Paid Plan | Adobe Creative Cloud Membership ? |
|------------------|-----------|---|-----------------------------------|
| open source apps | | ∞ unlimited must be pulled from a public Github repo | |
| private apps ? | 1 | 25 | |
| max app size | 50 MB | 100 MB | 1 GB |

Plan seçiminden sonra hesap bilgilerimizi gireceğimiz sayfaya geçiş yapıyoruz. Ücretsiz olarak oluşturduğunuz deneme hesabında (Free Plan) bir projeyi ücretsiz olarak derleyebilirsiniz. Free Plan seçerek devam ediyoruz.


Açılan giriş yapma sayfasında Get an Adobe ID seçeneğini seçerek bir hesap oluşturacağımız sayfaya giriş yapıyoruz ve hesap bilgilerimizi dolduruyoruz.

Kullanım şartları ve gizlilik anlaşmasını onayladığınızı belirten checkboxları işaretledikten sonra işleme devam edebilirsiniz.

Tüm işlemleri tamamladıktan sonra Sign Up butonuna tıkladığımızda profil sayfamıza geçiş yapacağız. Kayıt olduğunuz E-Posta adresine bir onay maili gelmeyeceğinden doğrudan hesabınızı kullanabilirsiniz.

 Adobe ID

Sign up to continue

 PhoneGap Build

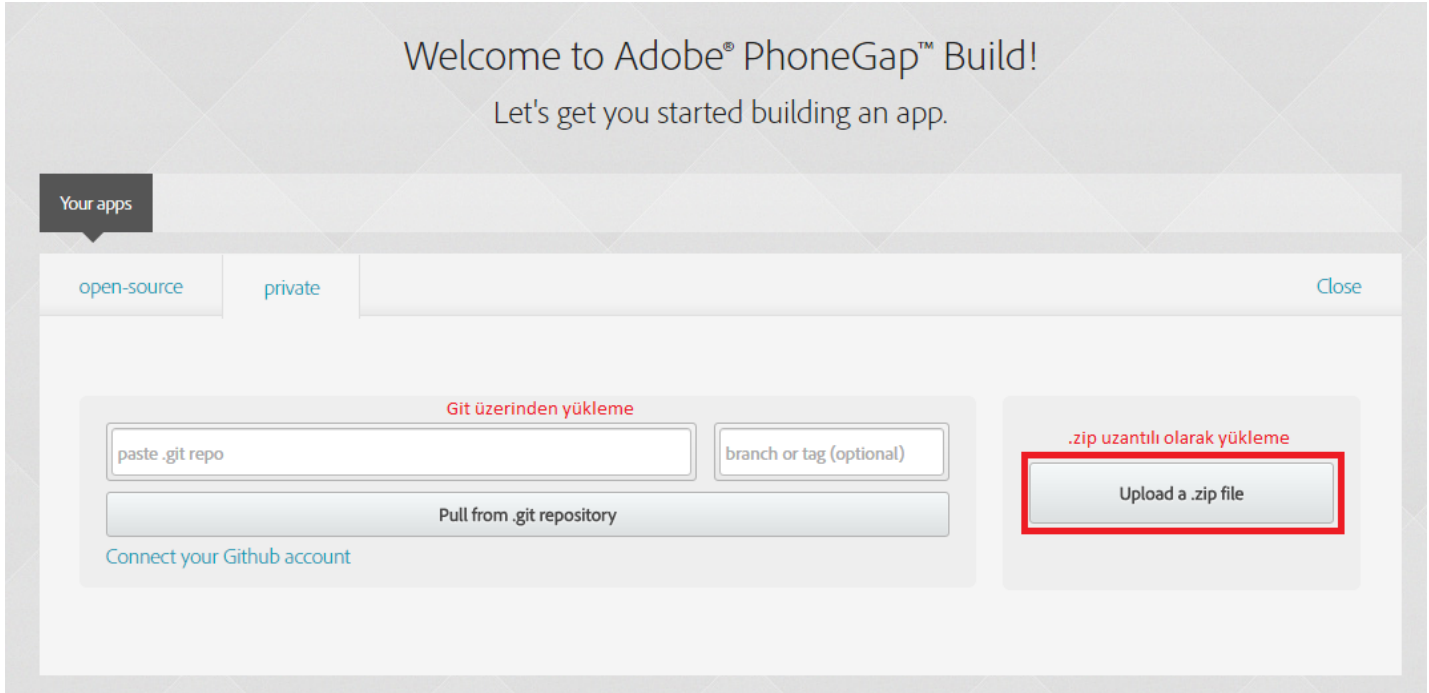
☒ Stay informed about Adobe products and services. [Learn more.](#)

☒ I have read and agree to the [Terms of Use](#) and [Privacy Policy](#).

[Sign up](#)

Already have an Adobe ID? [Sign In](#)

Hesap oluştuktan sonra repository sayfamıza yönleneceğiz ve bu sayfa altından projemizi yükleyerek derleme işlemlerini yapacağız.



Projeyi yüklerken uygulamanız gereken adımlar aşağıda yer almaktadır. Aşağıdaki adımları takip ettikten sonra Upload a .zip file butonu ile dosyanızı build.phonegap sistemine göndererek derleyebilirsiniz.

- Proje ana dizininde bulunan config.xml dosyasını www dizini içerisine atın.
- www klasörü içerisindeki dosyaları zip uzantısı ile arşivleyin. Dikkat etmeniz gereken nokta www klasörünü değil, klasörün içerisindeki dosyaları ziplemeniz gerektirir.
- Zip dosyanızı Build.PhoneGap.com adresine yükleyin.

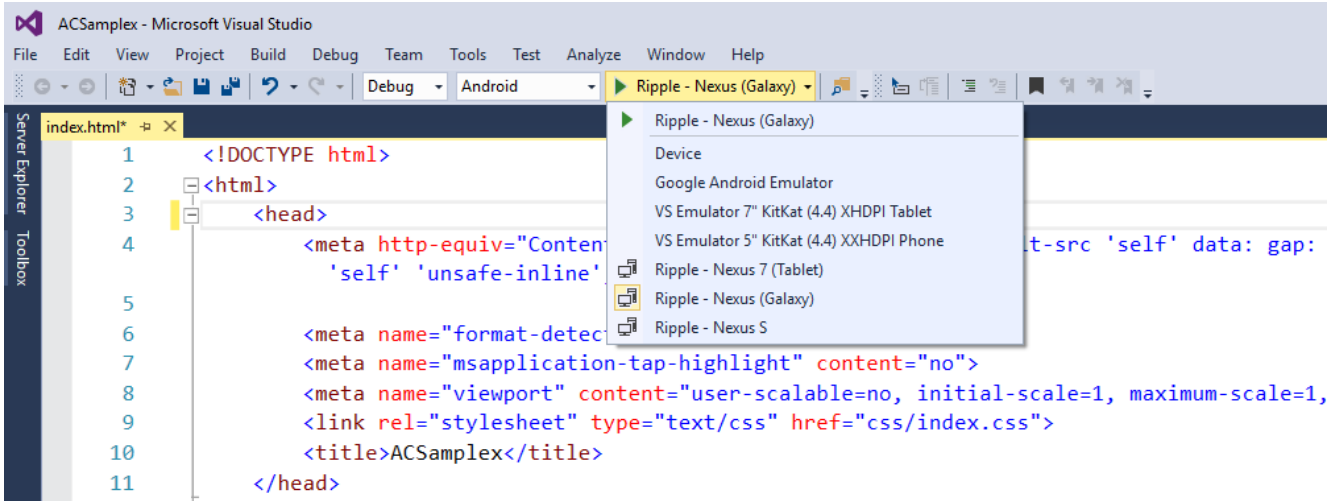
Derleme işlemlerinden sonra proje sayfasında verilen QR kodu okutarak derlenen projeyi mobil cihazımıza indirerek test edebiliriz.

Uygulamayı yüklemeyi önce mobil cihazınızdan 'Harici Kaynaklara İzin Ver' ayarını yapmalısınız.

Visual Studio ile bir proje oluşturduğumuzda projemizi derlerken dikkat etmemiz gereken noktalar bulunmaktadır. Visual studioda geliştirme yaparken tüm yazdığımız kodların www içerisinde olmasına dikkat etmeliyiz. Proje tamamlandıktan sonra aşağıdaki adımları takip ederek projemizi derleyebiliriz.

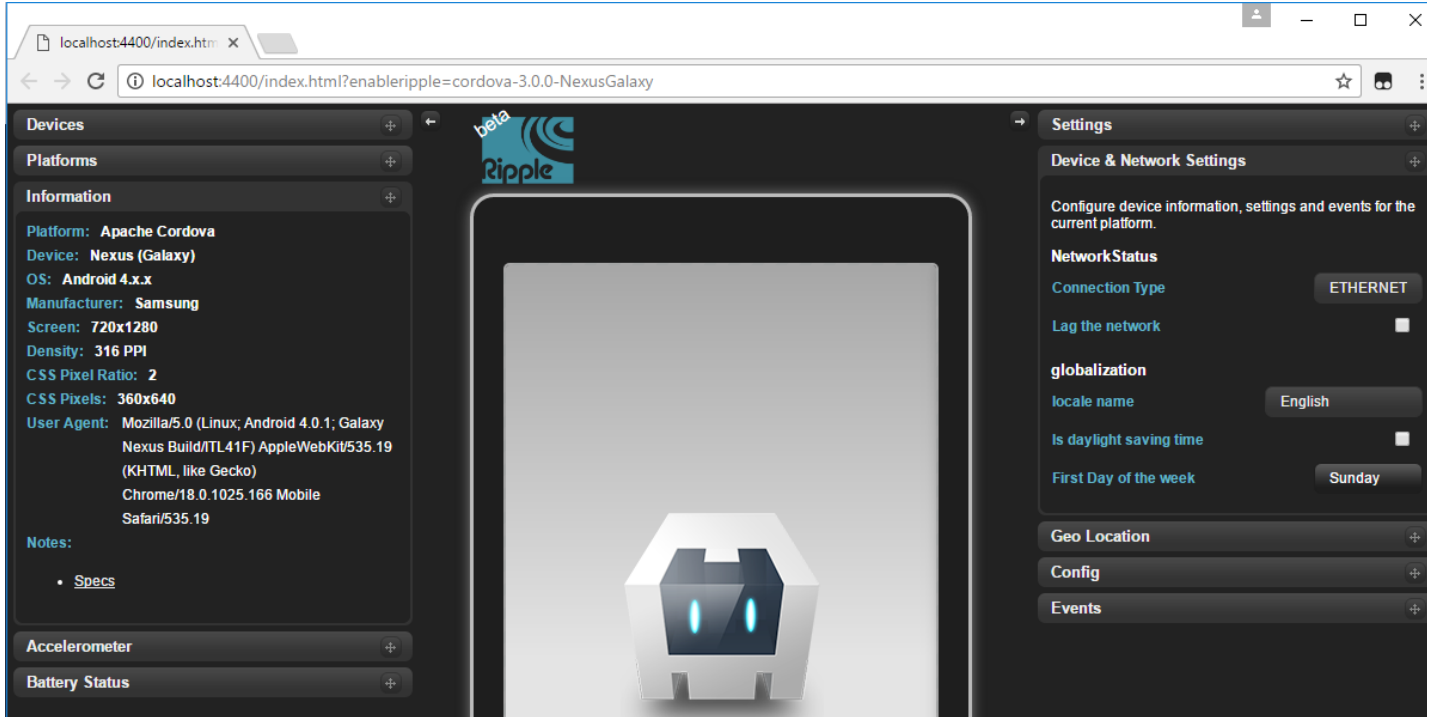
Emulator Ayarları ve Cordova Projesini Çalıştırmak

Apache Cordova projelerimizi çalıştırmak için emulator kullanmamız gerekmektedir. Apache Cordova Tool içerisinde emulatorler ve projeleri çalıştırmak için Device seçenekleri bulunmaktadır. Device bir emulator olmamak ile birlikte geliştirme yapılırken ve geliştirme sonrasında test amaçlı kullanılmaktadır. En sağlıklı geliştirme ortamı her zaman device ortamıdır. Apache Cordova ile oluşturduğunuz bir projenin hangi emulator ile veya device ile çalışacağını start seçenekleri altından değiştirebilirsiniz.



Ripple Mode

Google Chrome tarayıcısı üzerinde çalışmaktadır. Cihaz özelliklerinin bir çoğunu desteklemekle birlikte, yapılan mobile tasarımlarda hatalı görüntülemeler yapabilmektedir. Ayrıca tüm plug-inleri çalıştırmadığından (camera plugin'i gibi..) dolayı tercih edilmeyebilir. Aşağıda ripple emulatörün resmini görebilirsiniz. Ripple emulatörü bir çok uygulamayı sorunsuz çalıştırabilmektedir. Sayfa yanında bulunan panellerden emulatör cihaz ile ilgili bir çok durum ve ayar yönetilebilmektedir.



Device Mode

Geliştirilen projeyi doğrudan USB üzerinden bilgisayara bağlanan cihaz üzerinde çalıştırmayı sağlar. Bu yöntem en sağlıklı yöntemdir. Fakat farklı cihazlarda test yapmak için birden fazla cihaz gerekecektir. Ayrıca cihaz üzerine doğrudan deploy etmek için kullanılan telefonun, sistem tarafından tanıtılması gerekmektedir. Bu işlem için Pdanet isimli bir program kullanmaktayız.

Android Studio Emulator

Android Studio içerisinde bulunan Emulator'un üzerinden projeleri çalıştırma yöntemi bulunmaktadır. Bu yöntem diğer seçeneklere göre en fazla tercih edilen yöntemdir. SDK Manger üzerinden yeni bir Emulator oluşturulur,

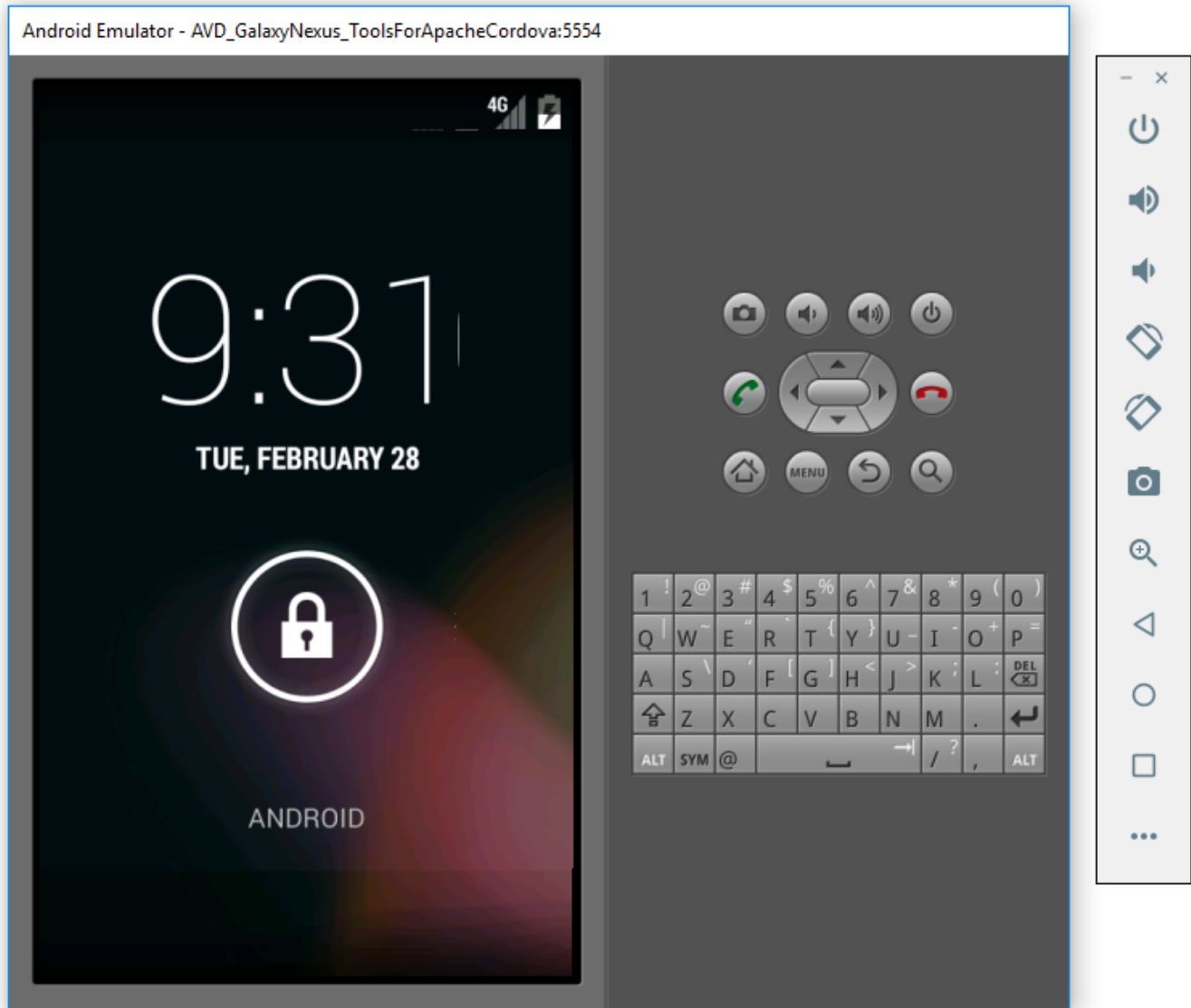
oluşturulan emulator (virtual machine) istediğimiz cihaz özelliklerini tanımaktadır. Android Emulatörü kullanıldığında TV, Phone, Wear ve Tablet seçenekleride oluşturacağımız emulatörler arasında bulunmaktadır. Cihaz seçiminden sonra kullanılacak android sürümü seçimi gibi ayarlar dışında daha detaylı ve özel ayarlar yapabilmekteyiz. Örneğin kamerayı kullanarak bir proje geliştireceğimizde, emulator'e bilgisayarımızın webcam'ini kullanabiliriz. Bunun dışında cihazın ram, işlemci, sd card boyutu gibi tüm ayarlamalar detaylı olarak yapılabilir.

Emulator oluşturma işleminden sonra çalışmıyorsa, bilgisayarımızda bulunan windows phone çalıştırmak için kullanılan Hyper-V açık olduğundan olabilmektedir. Böyle bir durumla karşılaştığınızda Hyper-V'nin kapalı (disable) olduğundan emin olunuz..

Ayrıca BIOS üzerinden Intel'in Virtual Machine özelliğini açmamız gerekmektedir.

Visual Studio eğer Device Mode'dan çalıştırmaya izin vermezse Android Studio > Tools > Android > SDK Manager > Extras >Google USB Drives seçeneğinin yüklendiğine emin olun!

Yukarıdaki ayarlardan sonra kurulum işlemimiz tamamlanmıştır. Artık Cordova platformu üzerinden geliştirmeler yapabiliriz.



Android Device Manager ile Device Oluşturma

Oluşturacağımız projeleri test etmek ve çalıştırmak için Android Studio üzerinde çalışacak sanal cihazlarımızı kullanacağız. Çalışmaya başlamadan önce Android Studio üzerinden cihaz oluşturmamız ve cihaz özelliklerimizi ayarlamamız.

- Android Studio'yu açın.

- Tool > Android > AVD (Android Virtual Device) Manager ekranını açarak device ekleyebilirsiniz ve oluşturulmuş cihazları yönetebilirsiniz.
- Create Virtual Device butonuna tıklayın.
- Oluşturmak istediğimiz cihaz seçimini yapıyoruz ve Next butonuna tıklayarak devam ediyoruz.
- Oluşturduğumuz cihaza isim veriyoruz. Kullanacağımız android versiyonu, ekran çözünürlüğü gibi ayarları yapıyoruz ve Next butonuna tıklayarak devam ediyoruz.
- Açılan pencerede, Advanced Settings'den oluşturulan cihazın Ram, Sd Kartı, Ekran özellikleri gibi gelişmiş ayarlarını yapabilirsiniz.
- Create diyerek sanal makineyi oluşturma işlemini tamamlıyoruz.
- Start diyerek sanal makinamızı çalıştırabiliriz.

Cordova Plug-in Mantığı

Cordova kodlaması javascript'i etkin olarak kullanabilen herkes tarafından kolaylıkla yapılabilir. Titreşim, Kamera, Şarj durumu gibi tüm özelliklerin her biri plugin'dir. Plugin'ler ikiye ayrılır.

- Core Plugins
- Custom Plugins

Plugin çalışma mantığında; eğer bir özelliği projede kullanacaksak, özelliği kullanabilmemiz için gerekli eklenti kurmamız gerekmektedir. Plugin kullanımını jQuery kütüphanesini projemize eklemek gibi düşünebilirsiniz. Web Projelerimizde jQuery kullanacaksak, jQuery.com adresinden jQuery'nin güncel versiyonunu indirerek projemize dahil ediyoruz ve daha sonrada kullanabiliyoruz. Pluginlerde aynı mantıkla çalışmaktadır. Visual Studio plugin yönetimi konusundada bize oldukça yardımcı olan bir IDE'dir.

Bir plugin kullanırken hangi platformları destekleyip, desteklemediği kontrol edilmelidir. Core pluginler genelde tüm platformlar tarafından desteklenirken, Custom pluginler tüm platformlar tarafından desteklenmeyebilir.

Core Plugins

Cordova içerisinde bulunan çekirdek pluginlerdir. Kamera, Titreşim, Rehber Erişim, Ekran Görüntüsü Alma, Dosya Transferi, Geolocation, Oryantasyon, Media Notification, Network vb. pluginler core plugin olarak adlandırılır. Bu pluginlerin tamamı çekirdek pluginidir. (Çekirden pluginler Config.xml içerisinde yüklenir.)

Custom Plugin

Bir çoğu MIT tarafından yazılmış özel pluginler vardır. Bu pluginler bir çok iyi geliştirici tarafından yazılmış pluginlerdir. Facebook, twitter gibi pluginidir. <http://www.pluginreg.com> isimli adrese girerek, güvenilir geliştiricilerin hazırladığı custom pluginleri kullanabilirsiniz. Custom plugin kullanırken desteklediği platformları incelemeyi unutmayınız.

Cordova Life Cycle Events

Apache Cordova öğrenirken ilk incelenmesi gereken nokta Life Cycle'dır. Apache Cordova yaşam döngüsü, eventlerden oluşmaktadır. Cordova içerisinde deviceReady, back, pause gibi çeşitli eventler bulunmaktadır. Uygulama başladığında cihazın tamamen hazırlandığı durumlarda deviceReady event'i çalışmaktadır.

Cihaz üzerindeki back tuşuna tıklandığında çalışacak event'in oluşturulması veya uygulama duraklatıldığında yapılacak işlemlerin bulunacağı pause eventinin oluşturulması Cordova life cycle'ı oluşturmaktadır. Aşağıda bir cordova projesi oluşturduğumuzda varsayılan olarak gelen eventleri bulabilirsiniz. Cordova Life Cycle eventleri hakkında daha detaylı bir bilgi edinmek için <https://cordova.apache.org/docs/en/latest/cordova/events/events.html> bağlantısını tıklayarak gerekli dökümantasyon sayfasına ulaşabilirsiniz. Aşağıda örnek bir life cycle bulunmakta ve örnek üzerinden açıklanmaktadır.

Apache Cordova Life Cycle Events (index.js)

```
(function () {  
    "use strict";  
  
    //Cihaz hazır olduğunda onDeviceReady metodunu tetikleyecektir. addEventListener içerisinde  
    verilen ilk isim önemlidir. İsim birebir benzerlik göstermektedir. İkinci parametremiz cihaz
```

hazır olduğunda çalışacak olan metodun adıdır. Son parametre ise useCapture parametresidir. Kullanım yakalamayı buradaki boolean ifade (true) ile açabilirsiniz.

`document.addEventListener('deviceready', onDeviceReady, false);` //onDeviceReady yanında bulunan bind özelliğini hatalara sebebiyet verdiğinden kaldırıyoruz.

//Cihaz hazır olduğunda yapılacak işlemleri onDeviceReady metodu içerisinde yazabiliriz.

```
function onDeviceReady() {
```

//Back, Pause ve Resume olayları cordovanın life cycle eventleri arasındadır. Cihaz üzerindeki tuşlardan Back tuşuna basıldığında yapılması gereken işlemleri, ana ekran düğmesi ile program arka plana alındığında yapılacakları ve görev yöneticisinden uygulama seçildiğinde devam ederken yapılması gereken işlemleri bu eventleri kullanarak yapabiliriz. deviceready'de olduğu gibi aşağıdaki ilk parametreler önemlidir. Farklı bir isim verememekteyiz.

`document.addEventListener('pause', onPause, false);`//onDeviceReady yanında bulunan bind özelliğini hatalara sebebiyet verdiğinden kaldırıyoruz.

`document.addEventListener('resume', onResume, false);`//onDeviceReady yanında bulunan bind özelliğini hatalara sebebiyet verdiğinden kaldırıyoruz.

//event tanımlamalarından sonra bulunan bir kaç satır kodu siliyoruz.

//Bir Cordova projesi oluşturduğunuzda yukarıdaki eventler oluşturulur. Örneğimizde kendimiz bir event ekleyelim. Aşağıdaki back eventini ekledikten sonra bu eventin çalıştıracağı metodu seçiyoruz.

```
document.addEventListener('back', onBack, false);
```

```
};
```

//Uygulama durdurulduğunda yapılacak işlemleri yazacağımız javascript fonksiyonu

```
function onPause() {
```

```
};
```

```
function onResume() {
```

//Uygulamaya devam edildiğinde yapılacak işlemleri yazacağımız javascript fonksiyonu

```
};
```

//Tarafımızdan oluşturulan onBack metodu içerisinde cihaz üzerinde bulunan back tuşuna basıldığında yapılacak işlemleri giriyoruz.

```
function onBack()
```

```
{
```

```
}
```

```
} )();
```

Cordova Local & Session Storage Kullanmak

Cordova hybrid çatısı altında gelen html 5 ile birlikte gelen localStorage ve sessionStorage özelliklerini kullanarak veri saklama işlemleri yapabilirsiniz. Bu işlemler için öncelikle html 5 ile birlikte gelen local ve session storage özelliklerini incelemeliyiz.

Html5 Local Storage

Html5 ile birlikte gelen localStorage veri saklamak için kullanılmaktadır. localStorage, cookie gibi yapıların yerine gelmiştir. LocalStorage üzerinde veri tutabiliriz, veriler kullanıcı silene kadar veya tarayıcı geçmişi silinene kadar kaybolmaz. Bu işlemler için localStorage nesnesinin metotları kullanılır ve veriler kullanıcının bilgisayarında depolanır. Verilerin depolama boyutu sınırsızdır. Kullanıcının kullandığı tarayıcı saklanacak veri boyutunu sınırlayabilir. Verilerin son geçerlilik tarihi yoktur. Bir örnek üzerinden inceleyelim;

- Content Security Policy özelliğini yorum satırı haline getirin.
- Index.html sayfasını aşağıdaki gibi düzenleyin.

index.html

```
<div>
  <input id="txtValue" name="txtValue" type="text" value="" />
  <br /><br />
  <button id="btnRead" onclick="Read()">LocalStorage Oku</button>
  <button id="btnWrite" onclick="Write()">LocalStorage Yaz</button>
</div>
```

Yazdığımız javascript fonksiyonlarını tetikleyecek olan buttonlarımızı oluşturuyoruz. Çalıştıracakları javascript fonksiyonlarımızın isimlerini belirliyoruz. Bu işlemleri javascript ile yapacağız fakat jQuery kullanmak istediğimizde jQuery kütüphanesinde bir problem çıkartmayacaktır. Metodlarımız index.js içerisinde.

- Cordova life cycle eventlerinin bulunduğu index.js dosyasına giriş yaparak javascript içerisindeki GLOBAL alana tarayıcının localStorage kullanabilme yeteneğinin olup olmadığını kontrol eden bir fonksiyon yazıyoruz.

index.js

```
function isStorageSupport() {
  try {
    return 'localStorage' in window && window['localStorage'] !== null;
  } catch (e) {
    return false;
  }
}
```

Yukarıda ki komutta localStorage içerisine bir değer atmaya çalışıyoruz. Eğer işlem sırasında hata verirse try catch bu hatayı yakalayacak ve geriye false değer döndürecektir. Bu metodu çağırarak kullanıcının tarayıcısının localStorage desteği olup olmadığını kontrol edebiliriz.

- Son olarak deviceReady metodu içerisinde yazdığımız fonksiyonu çağırarak tarayıcı desteğini kontrol ediyoruz.

index.js > DeviceReady

```
if (isStorageSupport) {
  alert("Tarayıcınız localStorage'yi destekliyor!");
}
```

Destek veriliyorsa bir alert ile mesajı gösteriyoruz. Ardından okuma ve yazma işlemlerinin yapılacak fonksiyonları yazıyoruz.

index.html

```
<script type="text/javascript">
  //LocalStorage içerisindeki veriyi okuyacak olan read metodunu tanımladık.
  function Read() {
    //LocalStorage içerisinde bulunan değeri okuyoruz.
    var name = window.localStorage.getItem("isim");
    //Eğer localStorage içerisinde değer yoksa
```

```
if (name == '' || name == null)
    alert("Bir isim bulunmamaktadır!"); //İsim bulunmamaktadır mesajını veriyoruz.
else
    alert(name); //localStorage içerisindeki değeri yazdırıyoruz.
}

//LocalStorage içerisine veri yazacak olan write metodunu tanımladık.
function Write() {
    //Textbox içerisinden okuduğu değeri localStorage içerisine atıyoruz.
    window.localStorage.setItem("isim", txtValue.value);
    alert("Success"); //İşlemin başarılı olduğunun mesajını veriyoruz.
}
</script>
```

Html5 Session Storage

Html5 ile birlikte gelen sessionStorage tarayıcı üzerindeki oturum sırasında veri saklamak için kullanılmaktadır. Veriler tarayıcı veya sekme kapatılana kadar kaybolmaz. sessionStorage nesnesinin metotları kullanılır ve veriler kullanıcının bilgisayarında depolanır. Verilerin depolama boyutu sınırsızdır. Kullanıcının kullandığı tarayıcı saklanacak veri boyutunu sınırlayabilir. SessionStorage tarayıcı kapana kadar saklanır. Tarayıcı yeniden açıldığında tüm verilerin kaybolduğunu gözlemleyebilirsiniz.

- Content Security Policy özelliğini yorum satırı haline getirin.
- Index.html sayfasını aşağıdaki gibi düzenleyin.

index.html

```
<div>
    <input id="txtValue" name="txtValue" type="text" value="" />
    <br /><br />
    <button id="btnRead" onclick="Read()">SessionStorage Oku</button>
    <button id="btnWrite" onclick="Write()">SessionStorage Yaz</button>
</div>
```

Yazdığımız javascript fonksiyonlarını tetikleyecek olan buttonlarımızı oluşturuyoruz. Çalıştıracakları javascript fonksiyonlarımızın isimlerini belirliyoruz. Bu işlemleri javascript ile yapacağız fakat jQuery kullanmak istediğimizde jQuery kütüphanesinde bir problem çıkartmayacaktır. Metotlarımız index.js içerisinde yer almaktadır.

- Cordova life cycle eventlerinin bulunduğu index.js dosyasına giriş yaparak javascript içerisindeki GLOBAL alana tarayıcının localStorage kullanabilme yeteneğinin olup olmadığını kontrol eden bir fonksiyon yazıyoruz.

index.js

```
function isStorageSupport() {
    try {
        return 'sessionStorage' in window && window['sessionStorage'] !== null;
    } catch (e) {
        return false;
    }
}
```



```
}  
}
```

Yukarıda ki komutta sessionStorage içerisine bir değer atmaya çalışıyoruz. Eğer işlem sırasında hata verirse try catch bu hatayı yakalayacak ve geriye false değer döndürecektir. Bu metodu çağırarak kullanıcının tarayıcısının sessionStorage desteği olup olmadığını kontrol edebiliriz.

- Son olarak deviceReady metodu içerisinde yazdığımız fonksiyonu çağırarak tarayıcı desteğini kontrol ediyoruz.

index.js > DeviceReady

```
if (isStorageSupport) {  
    alert("Tarayıcınız sessionStorage'yi destekliyor!");  
}
```

Destek veriliyorsa bir alert ile mesajı gösteriyoruz. Ardından sessionStorage içerisine değer atabileceğimiz ve içerisindeki değeri okuyabileceğimiz javascript kodlarımızı yazıyoruz.

index.html

```
<script type="text/javascript">  
    //LocalStorage içerisindeki veriyi okuyacak olan read metodunu tanımladık.  
    function Read() {  
        //LocalStorage içerisinde bulunan değeri okuyoruz.  
        var name = window.localStorage.getItem("isim");  
        //Eğer localStorage içerisinde değer yoksa  
        if (name == '' || name == null)  
            alert("Bir isim bulunmamaktadır!"); //İsim bulunmamaktadır mesajını veriyoruz.  
        else  
            alert(name); //localStorage içerisindeki değeri yazdırıyoruz.  
    }  
  
    //LocalStorage içerisine veri yazacak olan write metodunu tanımladık.  
    function Write() {  
        //Textbox içerisinden okuduğu değeri localStorage içerisine atıyoruz.  
        window.localStorage.setItem("isim", txtValue.value);  
        alert("Success"); //İşlemin başarılı olduğunun mesajını veriyoruz.  
    }  
</script>
```

LocalStorage ve Json Kullanarak Telefon Defteri

Yukarıda kullandığımız LocalStorage ve Json formatını kullanarak telefonumuzun hafızası üzerinde kalıcı şekilde veri saklayabilmekteyiz. Bu işlem için öncelikle yeni bir Cordova projesi oluşturmalıyız. Index.html sayfamızın içeriğini aşağıdaki gibi oluşturun.

index.html

```
<h2>Contacts</h2>

<div class="AddGroup">
  <span>Name :</span> <input type="text" id="txtName" /> <br />
  <span>Phone :</span> <input type="text" id="txtPhone" /> <br />
  <button onclick="KisiEkle()">Add Contacts</button>
  <button onclick="Temizle()">Clear Contacts</button>
</div>

<ul id="contacts"></ul>
```

Yukarıdaki işlemleri tamamladıktan sonra javascript kodlarını yazmaya başlayabiliriz. Oluşturduğumuz rehberimizi localStorage üzerinde saklayacağız. Bu işlem için öncelikle tarayıcımızın localStorage özelliğinin desteklenip, desteklenmediğini kontrol etmemiz gerekmektedir.

Index.js > jQuery İçerisine

```
function isStorageSupport() {
  try {
    return 'localStorage' in window && window['localStorage'] !== null;
  } catch (e) {
    return false;
  }
}
```

Ardından deviceReady metodu içerisinde oluşturduğumuz metodu çağırarak kullanıcıya bilgi veriyoruz.

Index.js > OnDeviceReady

```
function onDeviceReady() {
  if (isStorageSupport) {
    alert("Tarayıcınız localStorage'yi destekliyor!");
  }
};
```

Son olarak rehberde kişi ekleme işleminin yapılacağı metodumuzu yazıyoruz.

Index.js > jQuery Dışarısına

```
localStorageOku();

function KisiEkle() {
  var userList = JSON.parse(window.localStorage.getItem('contacts'));
  userList.list.push({ "Name": txtName.value, "Phone": txtPhone.value });
  window.localStorage.setItem("contacts", JSON.stringify(userList));
  localStorageOku();
}
```

```
function localStorageOku() {
    contacts.innerHTML = '';

    var userList;
    if (localStorage.contacts) {
        userList = JSON.parse(window.localStorage.getItem('contacts'));
    }
    else {
        window.localStorage.setItem("contacts", JSON.stringify({
            "list": [
            ]
        }));
        userList = JSON.parse(window.localStorage.getItem('contacts'));
    }

    for (var i = 0 ; i < userList.list.length ; i++) {
        contacts.innerHTML += '<li>' + userList.list[i].Name + '<br /> <span>' +
        userList.list[i].Phone + '</span></li>';
    }
}

function Temizle() {
    window.localStorage.clear();

    localStorageOku();
}
```

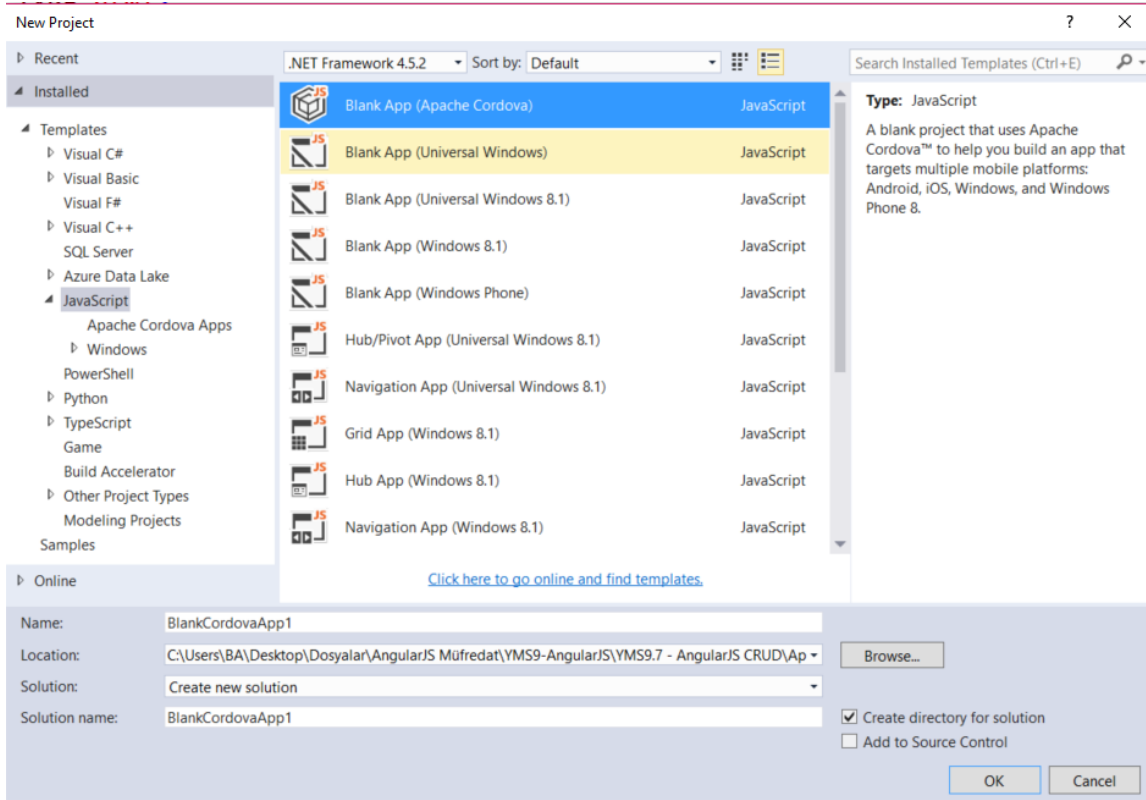
Cordova Projelerinde Türkçe Karakter Sorunu

Cordova projenizde türkçe karakter sorunu olduğunda bu sorunu çözmek için, türkçe karakter problemi yaşadığınız sayfayı yedek aldıktan sonra açın.

- File > Advanced Save Options yolunu takip edin.
- Encoding bilgisini 'Unicode (UTF-8 with signature) – Codepage 65001' olarak seçin.
- Sayfanızı kaydedin.

Cordova Projesi Oluşturmak

Visual Studio 2015 üzerinde yeni bir Cordova projesi oluşturuyoruz. Visual Studio 2015'i açın ardından sol panelden Javascript seçeneğini seçin. Açılan proje şablonlarından Apache Cordova template'ini seçiniz.



Yeni bir proje oluşturduğumuzda solution explorer içerisinde çeşitli klasör ve yapılar olacaktır.

| Dosya Adı | Açıklaması |
|-------------------|--|
| Platforms | Cordova projesine eklenen platformlara ait dosyaları içeren klasördür. |
| Merges | Platformlar için farklı dosyaları kullanmamızı sağlayan klasördür. |
| Plugins | Cordova uygulamalarında kullanılacak eklenti dosyalarının bulunduğu klasördür. Uygulamaya bir plugin indirdiğinizde plugin dosyaları bu klasör içerisine kopyalanır. |
| www | Cordova projemizin içerik sayfalarının barındırıldığı klasördür. Html, css ve javascript kodları www klasörü içerisinde bulunmaktadır. |
| index.html | İçerisinde listeleme, ekleme vb. İşlemleri yapabileceğimiz ve cihaz özelliklerine erişebileceğimiz sayfadır. Birden fazla html dosyamız olabilir. Kodlarımızı html sayfalarımıza yazacağız. İçerisinde varsayılan olarak verilen javascript ve css referansları bulunmaktadır. |
| config.xml | Plugin yönetmek, proje ayarlarını yapmak ve proje bilgilerini düzenlemek için kullanacağımız dosyadır. Sağ tıklayıp View Code seçeneği ile Config.xml'in kodlarını görebilirsiniz. |

Index.html sayfamızın içeriğini inceleyelim. Sayfamız bir html sayfasında olması gerektiği gibi head ve body etiketlerinden oluşuyor. Body etiketini incelediğimizde içerisinde iki javascript referansı verildiğini göreceğiz. İki referans dışında birde 'cordova.js' isimli bir referans olduğunu görüyoruz. Solution explorer içerisinde aradığımızda cordova.js dosyasını bulamıyoruz. Bunun sebebi Cordova.js'nin Server Side javascriptler olup, çalışma anında derleniyor olmasıdır. Tüm kodlarımız html dosyamız içerisinde yazılıyor.

Head Tag'leri arasında policy meta tag'i bulunmaktadır. Burdaki javascriptler çalışmayacağından bu policy tag'ini kaldırmamız gerekmektedir.

Kaldırılacak Policy MetaTag'i

```
<meta http-equiv="Content-Security-Policy" content="default-src 'self' data: gap:
```

```
https://ssl.gstatic.com 'unsafe-eval'; style-src 'self' 'unsafe-inline'; media-src *">
```

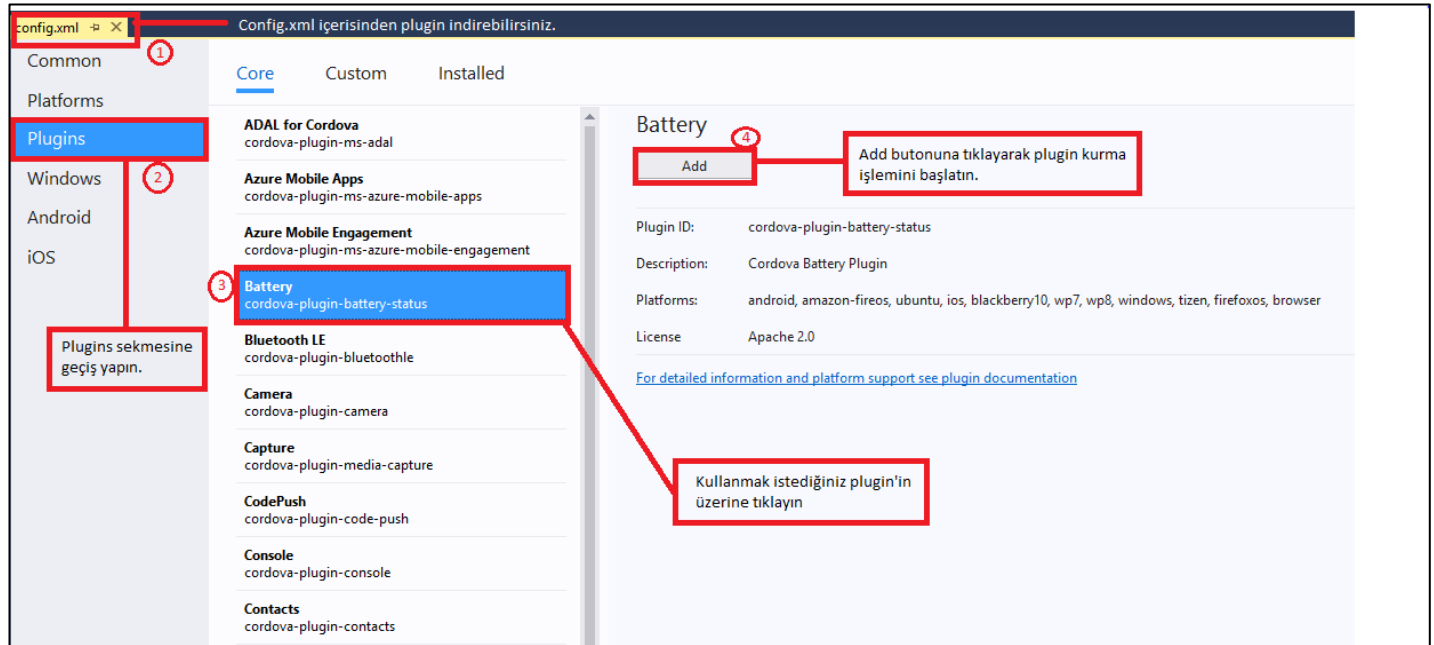
Plugins

Apache Cordova'nın çalışma yapısında device üzerindeki özelliklere erişmek için sistem üreticileri tarafından paylaşılan pluginlerin kullanılmalıdır. Sadece cihaz üzerindeki Native özelliklere erişmek için değil, aynı zamanda farklı işler içinde pluginler kullanılıyor. Plugin yapısını incelediğimizde Apache Cordova içerisinde iki çeşit plugin bulunmaktadır.

- Core Plugins
- Custom Plugins

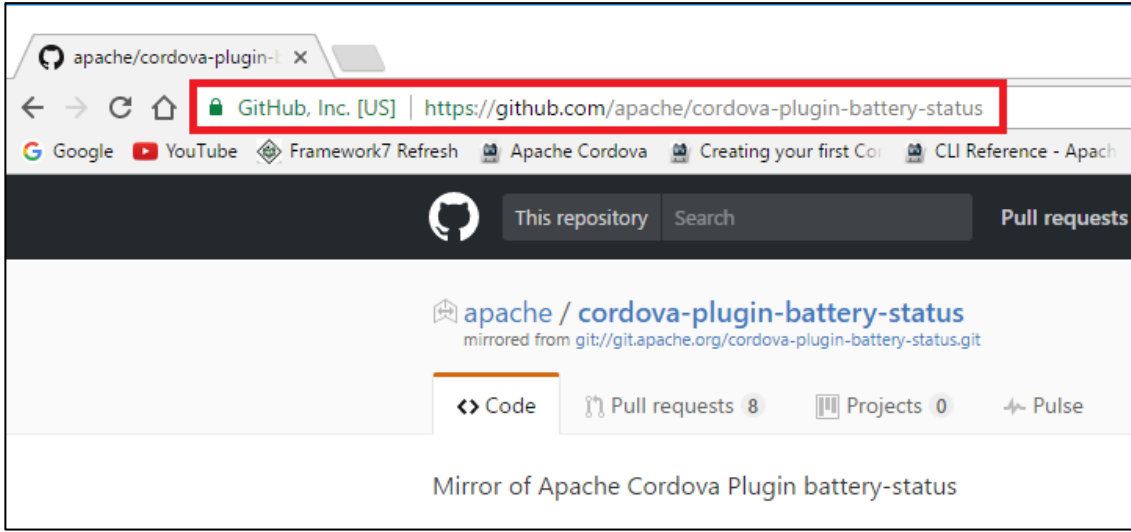
Plugin Yüklemek

Core pluginleri projeye yüklemek için iki farklı yöntem kullanılmaktadır. Visual Studio kullanan bir geliştirici config.xml dosyasını açarak plugin sekmesi altından çekirdek pluginleri kolayca indirebilir ve kurabilir. Aşağıdaki resimde Visual Studio 2015 üzerinden core plugin yükleme adımlarını takip edebilirsiniz.

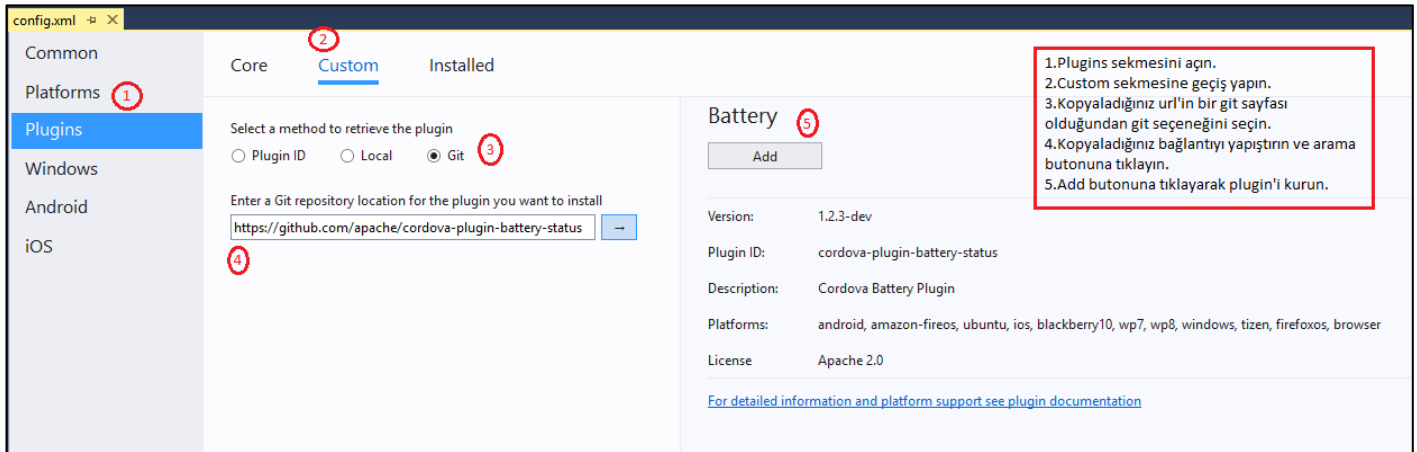


İkinci yöntem ise, github üzerinden plugin sayfasına giderek, plugin url'ini kopyalayarak kullanabilirsiniz. Plugin yüklemek için gerekli adımları aşağıdan takip edebilirsiniz.

Öncelikle github üzerinden kullanmak istediğiniz plugin sayfasına giriş yapın. Örneğimizde Battery plugin üzerinden anlatımı gerçekleştireceğiz. Github üzerinden veya Google üzerinden yapacağınız Battery Plugin Cordova aramasından sonra ilgili github sayfasına ulaşabileceğiniz bağlantıyı kullanabilirsiniz. Plugin sayfasına giriş yaptığınızda aşağıdaki resimde işaretlenmiş olan url alanını kopyalayın.



Bağlantıyı kopyaladıktan sonra Apache Cordova projeniz içerisinde bulunan config.xml dosyasını açın. Pluings sekmesi altından Custom sekmesine geçiş yapın ardından kopyaladığınız github bağlantısını textbox'a yapıştırdıktan sonra arama butonuna tıklayın. Kısa bir bekleme süresinden sonra plugin'i indirebileceğiniz bağlantı alanı ekranda belirecektir.



Core ve core pluginler için bu yöntemleri kullanarak kurulum yapabilirsiniz. Kurmak istediğiniz her plugin için aynı adımı tekrar etmeniz gerekmektedir.

Plugin Kullanımında Dikkat Edilmesi Gerekenler

Core veya Custom plugin kullanırken dikkat edilmesi gereken ve geliştirme yaparken işlerimizi kolaylaştıran veya hata ile karşılaşmamıza sebep olacak noktalar vardır. Öncelikle plugin kullanırken ve indirirken plugin'in kullanılması için gerekli olan dökümantasyon sayfasına ulaşmalıyız.

Dökümantasyon içerisinde plugin kullanmak için gerekli kurulum adımlarını ve desteklediği platformları görebilmekteyiz. Ayrıca dökümantasyon içerisinde plugini kullanmak için gerekli metod ve yapıların hangi amaçlarla, nasıl kullanılabileceği hakkında bilgilerde bulabilmekteyiz. <https://cordova.apache.org> üzerinden gerekli dökümantasyonlara ulaşabilmekteyiz. Örnek bir dökümantasyon sayfasının resmi aşağıda paylaşılmaktadır.

batterystatus event

Fires when the battery charge percentage changes by at least 1 percent, or when the device is plugged in or unplugged. Returns an object containing battery status.

Example

```
window.addEventListener("batterystatus", onBatteryStatus, false);

function onBatteryStatus(status) {
  console.log("Level: " + status.level + " isPlugged: " + status.isPlugged);
}
```

Supported Platforms

- Amazon Fire OS
- iOS
- Android
- BlackBerry 10
- Windows Phone 7 and 8
- Windows (Windows Phone 8.1 only)
- Firefox OS
- Browser (Chrome, Firefox, Opera)

Quirks: Android & Amazon Fire OS

Warning: the Android and Fire OS implementations are greedy and prolonged use will drain the device's battery.

Plugin kullanmak için yukarıdaki dökümantasyonu detaylı olarak okumamız gerekmektedir. Yukarıda kullanılacak plugin'in kurulum aşamaları, eventleri ve desteklendiği platformlar gibi detaylı bilgiler verilmektedir. Cordova üzerinde bu dökümantasyonlardan yardım almadan geliştirme yapmak sağlıklı olacaktır.

Core Plugins

Mobil cihaz üzerindeki native özelliklere erişmek için core (çekirdek) pluginler bulunmaktadır. Çekirdek pluginler hemen hemen her platform tarafından desteklenmektedir. Apache Cordova platformunda bir özelliği kullanabilmek için kullanılacak özelliğe ait plugin'in projeye yüklenmesi gerekmektedir.

Vibration Plugin

Yukarıdaki adımları tamamladıktan sonra bir cordova uygulaması yazmaya başlayabiliriz. Oluşturulan her cordova projesinde yukarıdaki adımları sırayla uygulamalıyız. İlk geliştireceğimiz uygulama titreşim uygulaması olacak. Bu işlem için cordova vibration plug-in'ini kullanacağız. Config.xml dosyası içerisinde Plug-ins sekmesine girin ve Core Plug-ins altından Vibration Plug-in'in yüklemesini yapınız. Plug-in yüklemesi tamamlandıktan sonra html sayfamıza bir adet button ekleyelim.

Vibration Plug-in kullanmadan önce dikkat etmemiz gereken bazı noktalar bulunmaktadır.

Index.html içeriği

```
<body>
  <div class="app">
    <h1>Vibration</h1>
    <button onclick="defaultVibration()">Default Vibration</button>
    <button onclick="patternVibration()">Pattern Vibration</button>
  </div>
  <script type="text/javascript" src="cordova.js"></script>
  <script type="text/javascript" src="scripts/platformOverrides.js"></script>
```



```
<script type="text/javascript" src="scripts/index.js"></script>
</body>
```

Butonumuzu oluştururken onclick eventinde butona tıklandığında çalışacak javascript fonksiyonunun adını belirtiyoruz. Cihazımızın titreşimini sağlayan aşağıdaki javascript metodumuzu yazalım.

Index.js > jQuery tanımlamasının dışına

```
function defaultVibration() {
    //3s aralıksız titreşim uygula
    navigator.vibrate(3000);
}

function patternVibration() {
    //1s Titreşim, 2sn Dur, 500ms titreşim, 500ms dur, 500ms titreşim, 2s dur, 500ms titreşim, 2s
    //dur, 5s titreşim, 500ms dur, 2s titreşim
    navigator.vibrate([1000, 2000, 500, 500, 500, 2000, 500, 2000, 5000, 500, 2000]);
}
```

Daha sonra titreşim özelliğinin çalışıp çalışmadığını test ediyoruz. Titreşim özelliğini test etmek için Android Emulator'u kullanamıyoruz. Bunun için Ripple Mode veya Device Mod ile uygulamayı çalıştırmamız gerekmektedir. Device Mode kullanmak için cihazınızı bilgisayara bağlamayı ve tanıtmayı unutmayınız.

Device Plugin

Bu örneğimizde Device Plug-in'ini kullanacağız. Bu plug-in çekirdek plug-inler arasında yer almaktadır ve bize cihaz özelliklerine erişme imkanı vermektedir. Yeni bir Cordova projesi oluşturun. Config.xml içerisinden Plug-in sekmesini açarak Core Plugins altından Device plug-in kurulumunu yapın. Aşağıdaki device plug'in 'inin bize sunduğu özellikleri görebilirsiniz.

| | |
|------------------|--|
| Model : | Cihazın modelini bilgisini vermektedir. |
| Uuid : | Her android cihazın kendisine özel bir id bilgisi bulunmaktadır. Cihazın id bilgisini verir. |
| Version : | Cihazın android versiyonunu verir. |

Index.html içeriği

```
<body>

    <div class="app">
        <h1>Device Plugin</h1>

        <div>Cordova : <span id="cordova"></span></div>
        <div>Model : <span id="model"></span></div>
        <div>Platform : <span id="platform"></span></div>
        <div>UUID : <span id="uuid"></span></div>
        <div>Version : <span id="version"></span></div>
    </div>

    <script type="text/javascript" src="cordova.js"></script>
```

```
<script type="text/javascript" src="scripts/platformOverrides.js"></script>
<script type="text/javascript" src="scripts/index.js"></script>
</body>
```

Index.html dosyamızı yukarıda ki gibi şekillendirdikten sonra aşağıdaki javascript kodlarımızı yazarak device plug-in'i kullanabiliriz.

Index.js

```
function onDeviceReady() {
    document.addEventListener('pause', onPause, false);
    document.addEventListener('resume', onResume, false);

    //Device Plug-in eklendikten sonra device. ile device özelliklerine ulaşabiliriz.
    cordova.innerHTML = device.cordova;
    model.innerHTML = device.model;
    platform.innerHTML = device.platform;
    uuid.innerHTML = device.uuid;
    version.innerHTML = device.version;
};
```

Battery Plugin

Cihazımızın batarya özelliklerini test etmek için kullandığımız plug-in'dir. Bu plug-in core plug-inler arasında yer almakta ve tüm platformlar tarafından desteklenmektedir. Bu plug-in'i kullanarak cihazın şarj seviyesini ve şarjda olup olmadığını öğrenebilir, düşük (20%) ve kritik şarj (5%) seviyelerinde işlemler yapabilirsiniz. Bu plug-in örneği için bir Cordova projesi oluşturun. Oluşturduğunuz projenin Config.xml dosyası içerisinde Plug-in sekmesine gelerek Battery Plug-in'i projeye ekleyin. Aşağıdaki javascript kodunu kullanarak plug-in'i çalıştırabilirsiniz. Öncelikle kullanıcıya bilgi vereceğimiz html görünümünü hazırlayalım.

Index.html

```
<body>
    <h2>Battery Status</h2>
    <div>Pil Seviyesi : <span id="pil"></span></div>
    <div>Sarj Oluyor : <span id="sarj"></span> </div>
    <div>Pil Durumu : <span id="bilgi"></span> </div>

    <script type="text/javascript" src="cordova.js"></script>
    <script type="text/javascript" src="scripts/platformOverrides.js"></script>
    <script type="text/javascript" src="scripts/index.js"></script>
</body>
```

Battery Plugin içerisinde üç farklı event bulunmaktadır. Bu eventler batterystatus, batterylow ve batterycritical eventleridir. Pil seviyesi her değiştiğinde batterystatus event'i tetiklenmektedir. Batterylow event'i, pil seviyesi 20% olduğunda tetiklenmektedir. BatteryCritical event'i ise pil durumu 5% olunca tetiklenmektedir.

Index.js > onDeviceReady içerisine

```
window.addEventListener("batterystatus", onBatteryStatus, false);
```

```
//Batarya durumunu kontrol eden düşük batarya eventini oluşturunuz. Batarya yüzdesi 20%'nin altında olduğunda bu event tetiklenecektir.
```

```
window.addEventListener("batteryLow", onBatteryLow, false);
```

```
//Batarya durumunu kontrol eden kritik batarya eventini oluşturunuz. Batarya yüzdesi 5%'in altında olduğunda bu event tetiklenecektir.
```

```
window.addEventListener("batteryCritical", onBatteryCritical, false);
```

Pil durumlarını çeşitli eventler ile kontrol ediyoruz. Aşağıdaki metot içeriği ile pil durumunu ekrana yazdırabilirsiniz.

Index.js > jQuery Fonksiyonları içerisine

```
//Cihazın şarja takılı olup olmama durumu ripple üzerinden değiştirildiğinde bu event çalışmaktadır. Pil seviyesi 20% veya 5% altında iken şarj durumunu değiştirirseniz bu event çalışacak ve pil durumuna normal yazacaktır.
```

```
function onBatteryStatus(status) {  
    pil.innerHTML = status.level + '%';  
    sarj.innerHTML = status.isPlugged ? "Sarj Oluyor" : "Sarj Olmuyor";  
    bilgi.innerHTML = 'Normal';  
}
```

```
function onBatteryLow(status) {  
    pil.innerHTML = status.level + '%';  
    sarj.innerHTML = status.isPlugged ? "Sarj Oluyor" : "Sarj Olmuyor";  
    bilgi.innerHTML = 'Dusuk';  
}
```

```
function onBatteryCritical(status) {  
    pil.innerHTML = status.level + '%';  
    sarj.innerHTML = status.isPlugged ? "Sarj Oluyor" : "Sarj Olmuyor";  
    bilgi.innerHTML = 'Kritik';  
}
```

Bu işlemlerden sonra projenizi çalıştırdığınızda ekrana pil durumu ile ilgili bilgiler gelecektir.

Network Information Plugin

Network information eklentisi mobil cihazımızın ağ bilgilerini okumamıza ve bu bilgileri kullanarak işlem yapmamıza olanak vermektedir. Kullandığım bağlantı tipi veya internete bağlı olup olmadığımız gibi bilgileri Network Information eklentisini kullanarak öğrenebiliriz.

Index.js > jQuery Dışına

```
function checkConnection() {  
    var networkState = navigator.connection.type;  
  
    var states = {};
```

```
states[Connection.UNKNOWN] = 'Unknown connection';
states[Connection.ETHERNET] = 'Ethernet connection';
states[Connection.WIFI] = 'WiFi connection';
states[Connection.CELL_2G] = 'Cell 2G connection';
states[Connection.CELL_3G] = 'Cell 3G connection';
states[Connection.CELL_4G] = 'Cell 4G connection';
states[Connection.CELL] = 'Cell generic connection';
states[Connection.NONE] = 'No network connection';

durum.innerHTML = states[networkState];
}
```

Ripple Emulatrör üzerinden network bilgilerini deęiřtirebilirsiniz. Ripple üzerinden baęlantı yöntemini deęiřtirdięinizde ripple emulatorü F5 ile yenilemeniz gerekmektedir.

Kamera Plugin

Camera Plugin ile cihazımızın kamerasına erişebiliriz. Cihaz kamerasını kullanarak çekilen bir resmi programımız içerisine aktararak kullanabiliriz.. Bu işlem için plugin yükleme ekranından Camera Plugin'i indirmeniz gerekmektedir. Plugin kurulumundan sonra index.html sayfasını aşağıdaki gibi düzenleyebilirsiniz.

Index.html

```
<div>Smileeeee</div>
<input type="button" id="takePhoto" value="Fotoęraf Çek" />
<div id="photo"></div>
```

Html yapımızı düzenledikten sonra plugin'i kullanacaęımız javascript kodlarını ve fonksiyonlarımızı hazırlayabiliriz.

Index.js

```
document.getElementById('takePhoto').onclick = function () {
    navigator.camera.getPicture(function (imageUri) {
        var lastPhoto = document.getElementById('photo');
        alert('nice');
        lastPhoto.innerHTML = ''
    }, null, null);
}
```

Projenizi device üzerinde çalıştırarak test edebilirsiniz.

WhiteList Plugin

Bu eklenti cordova üzerinde web gezinimi için beyazlist yöntemini uygular. Oluřturduęunuz uygulamadan baęlanılacak web servislerin izinlerini ayarlayabilirsiniz. Android 4.0.0.0 üzerinde desteklenmektedir. WhiteList plugini ile ilgili ayarlar config.xml dosyasının code ekranında bulunmaktadır. Varsayılan olarak * kullanılmıřtır. Uygulamanız tüm servislere baęlanabilecek řekilde ayarlanmıřtır. Allow indent ile baęlanılacak olan servis yapılarını filtreleyebiliriz.

Bu işlem için config.xml dosyası üzerine saę tıklayın ve View Code seęeneęi ile dosyayı açın. Ařaęıdaki tanımlama WhiteList plugin'i için kullanılan allow indent tanımlamalarıdır.

```
<allow-intent href="http://*/*" />
<allow-intent href="https://*/*" />
<allow-intent href="tel:*" />
<allow-intent href="sms:*" />
<allow-intent href="mailto:*" />
<allow-intent href="geo:*" />
```

Custom Plugin

Güvenli geliştiriciler tarafından oluşturulan pluginlerdir. <http://www.pluginreg.com> adresi üzerinden custom pluginler indirilebilir. Facebook entegrasyonlu uygulamalar, QR kod okuyucu pluginler gibi geniş bir yelpazede plugin bulunmaktadır.



The Plugin Registry

pluginreg allows Cordova / PhoneGap developers to search for existing plugins for their app projects. It also gives plugin authors additional exposure to their open source plugin(s).

At a glance, users are able to see the status of a plugin, the number of stars, open issues, version, supported platforms / engines and more.

There are currently **1613 plugins** from **1069 different authors** and growing all the time. Please **submit your own plugin(s)** if you haven't already.

Search All Indexed Cordova / PhoneGap Plugins:

Search... **Search**

Recently Added Plugins

- Ringtone Picker (2017-2-10)
- ITSAAppUsesNonExemptEncryption false (2016-11-11)

Authors with the Most Plugins

- MobileChromeApps (34)
- EddyVerbruggen (26)
- cranberrygame (24)

Plugins with the Most Stars

- Facebook Connect (1915)
- Facebook Connect (1915)
- LocalNotification (1469)

Kullanmak istediğiniz plugin'in dökümantasyonuna ve github bağlantısına pluginreg üzerinden erişebilirsiniz. Custom pluginlerin tamamının her platform tarafından desteklenmediğinden kullanımdan önce desteklenen platformların incelenmesi gerekmektedir.

Framework 7 Kullanımına Giriş

Framework 7, hybrid uygulama geliştirirken kullandığımız css ve javascript kütüphanesidir. Uygulama geliştirirken IOS veya Android metarial tasarımını kolayca yapmamızı sağlar. Framework 7 gibi kütüphanelere iki yaklaşım bulunmaktadır.

Single Page Application; Html içerisine baktığımızda view yapılarını göreceksiniz. Tüm uygulama aslında tek bir sayfa altında çalışmaktadır. Büyük bir uygulama geliştirdiğinizde tercih etmeniz tavsiye edilmez. Çünkü single page applicationlarda yapacağınız işlemler tek bir sayfada olduğundan uygulamanızda yavaşlamalar meydana gelecektir.

Multi Page Application; Her sayfanın kendi işlerinden sorumlu olduğu yaklaşımdır. Yapacağımız işlemleri tek bir sayfa altında toplamak yerine her iş için bir sayfa oluşturarak sayfa başı düşen iş yükünü azaltabilirsiniz. Büyük projelerde tavsiye edilen yöntem Multi Page uygulamalarıdır.

İlk Framework 7 Uygulaması

<http://www.framework7.io> sayfasına giriş yaparak download bağlantısından Framework 7'yi indirin. İndirdiğiniz kaynak dosyada;

- Example klasöründe kullanacağınız template'i seçin. (Tab Bar Template'ini kullanacağız.)
- İndirdiğiniz template'i bir cordova projesine giydirin.
- Referans yollarını kontrol edin css ve js referansları doğru verdiğinizden emin olun.

Framework 7 kurulumundan sonra doğru çalışmıyorsa dikkat edilmesi gerekenler

- Referans yollarının doğru verilmesi
- Left ve Right paneller
- Viewlar ve açılacak olan varsayılan view için active class'ı
- My-app dosyasının referansı

Örneklerimizde, Framework 7 içerisinde bulunan TabBar projesindeki index.html'de View-2 içerisinde bulunan liste yapısını kullanacağız.

Framework 7 farklı html sayfalarında hareket ederken bile tek bir program sayfasında farklı paneller kullanıyormuş hissi vermektedir. TabBar üzerindeki menülere tıkladığımızda aynı sayfada bulunan farklı viewlarda hareket ettiğini gözlemleyebiliriz. Fakat dikkat etmemiz gereken bir nokta View-2'de bulunan liste yapısını kullandığımızda farklı html sayfalarına geçiş yaptığımızdır.

About.html sayfasını açtığımızda html, body, head vb. etiketler bulunmamaktadır. Framework 7'nin farklı sayfalarla çalışması Asp.Net üzerinde kullanılan MasterPage/Layout mantığı ile çalışmaktadır. About.html sayfasına yönlendirildiğinde, about.html doğrudan açılmak yerine, index.html'de işaretli alanıyla değiştiriliyor.

Bu yapıyı sağlayan nokta, about.html içerisindeki div'de bulunan **Data-Page attribute** 'udur. Data-Page attribute içerisinde sayfa adını (uzantısı olmadan) yazıyoruz. Data-Page eventi about sayfasının açılmasıyla, o sayfanın init olduğu anı yakalamamızı sağlamaktadır. Bu şekilde sayfa açıldığında çeşitli işlemler yapabilmekteyiz.

Daha basit şekilde anlatırsak, jQuery ile bir geliştirme yaptığımızda document.ready ile sayfa hazır olduğunda çalışmayı bekleyen kodları yazarız. Framework 7 içerisinde durum biraz daha farklıdır, çünkü sayfa zaten yüklü durumdadır. About.html bağlantısına tıkladığımızda sadece sayfanın ekranda görünmesini sağlıyoruz. Bu sebeple sayfa açılış eventlerimiz için Data-Page attribute'unu kullanmalıyız.

Sayfayı çalıştırdığınızda dikkat ettiğiniz üzere about.html sayfasını açtığınızda, en altta bulunan nav bar sabit kalmaktadır. About.html sayfası içerisinde ise bu bar ile ilgili herhangi bir html kodu bulunmamaktadır.

My-app.js

My-app.js Framework yapımızın çalışması için gerekli ayarları barındırmaktadır. MyApp olarak Framework7'nin instance'ını aldıktan sonra artık myApp üzerinden uygulamamızı yönetebilmekteyiz. Viewların eklenmesi gibi işlemler my-app içerisinde yapılmaktadır.

Apache Cordova ile WebAPI Kullanımı

Apache cordova uygulamalarında sunucu üzerindeki veritabanından veri okumak veya veritabanına veri yazmak için servisleri kullanmaktayız. Apache Cordova içeriğimizde WebAPI üzerinden veri okuma ve veri yazma işlemlerini gerçekleştireceğiz. Bu işlemler için öncelikle kullanılacak olan WebAPI'yi oluşturmalıyız.

- **Web API Projesi oluştur**
- **Projeye empty WebAPI Ekle**
- **Orm ve DTO Klasörleri Oluştur**
- **Orm içerisine Northwind Database yansımalarını AI**
- **DTO içerisine Category classı oluştur ve propertyleri ver.**

WebApiConfig içerisinden webapi ye bağlanmak için gerekli route ayarını yapabiliriz. Api'ye bağlanmak için adres satırına api/Controlleradi yazarak girebiliriz.

WebApiConfig -> Route Ayarı

```
config.Routes.MapHttpRoute(  
    name: "DefaultApi",  
    routeTemplate: "api/{controller}/{action}/{id}", //Api yapısı api/Controller olarak  
    çalışıyor bu yapıyı değiştirerek action yapısının da url yapımıza dahil ediyoruz.  
    defaults: new { id = RouteParameter.Optional }  
);
```

WebApi varsayılan olarak geriye xml değer döndürür. Aşağıdaki ayarı WebApiConfig içerisine yazdığınızda değerler json tipinde dönecektir.

Json Döndürme Ayarı

```
//Default olarak global formatter olarak XML kullanılmaktadır. Biz bu desteklenen medya tiplerini  
temizlediğimizde geriye json formatında geri dönüş yapacaktır.  
GlobalConfiguration.Configuration.Formatters.XmlFormatter.SupportedMediaTypes.Clear();
```

Oluşturduğumuz WebApi controller içerisine aşağıdaki categories metodunu yazıyoruz. Bu metod geriye bir CategoryModel listesi döndürecektir. WebApi içerisinde gerekli ayarı yaptığımızdan bu liste json formatında verilecektir.

GetAllCategories isimli HttpGet Metodu

```
public List<CategoryModel> GetAllCategories()  
{  
    List<CategoryModel> cList = db.Categories.Select(x => new CategoryModel()  
    {  
        CategoryID = x.CategoryID,  
        CategoryName = x.CategoryName  
    }).ToList();  
  
    return cList;  
}
```

Yukarıdaki metod servisten veri çekmek için kullanılmaktadır. Metod ismi Get anahtar kelimesi ile başladığından metodun HttpGet Attribute'u ile işaretlememize gerek kalmayacaktır. Metod ismi başındaki Get bizim için metodun get işleminde kullanılacağını belirtmektedir.

Aşağıdaki metod ise bir Post metodu olarak kullanılacaktır ve bu sebeple HttpPost attribute'u ile işaretliyoruz. Veritabanına ekleme yapmak için gerekli bilgileri json formatında göndermemiz gerekmektedir.

InsertCategory isimli HttpPost Metodu

```
//{ "CategoryName" : "Kategori Adı", "Description" : "Açıklaması"}  
public IHttpActionResult InsertCategory(Category _model)  
{  
    db.Categories.Add(_model);  
    db.SaveChanges();  
    return Json("success");  
}
```


Servis metodlarımızı hazırladıktan sonra artık servisi Apache Cordova projemizden kullanabiliriz.

Servisden Data Çekmek

Servisimizi yazdıktan sonra artık servisimizi kullanabiliriz. Veri çekme işlemi Framework 7 üzerinden gerçekleştireceğimiz Servis üzerinden veri çekmek için jQuery ajax metodlarını kullanacağız.

- Framework 7 web sayfasını açın.
- Dökümantasyonlar sayfasına geçiş yapın.
- Dökümantasyon sayfasında ki Core sekmelerinden Pages sayfasına girin.
- Doküman içerisinde Page Data başlığını bulun. onPageInit kod örneğini kopyalayın.
- Projemize jquery.com üzerinden jquery dosyasını ekleyin ve proje klasörünüze atın.
- Index.html içerisinde **Cordova Life Cycle scriptleri yüklendikten sonraki** satıra jQuery'i referans olarak ekliyoruz.
- About.html sayfamızın adını kategoriler.html olarak değiştiriyoruz.
- Kategoriler.html sayfamızın adını **data-page attribute'una 'kategoriler'** olarak .html uzantısı olmadan yazıyoruz.
- Kodlarımızı yazacağımız 'MyCode.js' isimli bir javascript dosyası oluşturuyoruz.
- Kopyaladığımız onPageInit kodlarını 'MyCode.js' sayfasına yapııştırıyoruz.
- onPageInit içerisinde tanımlı olan about'u kategoriler olarak değiştiriyoruz.
- MyCode.js dosyamızı index.html dosyamız içerisinde jQuery referansından sonraki bir satıra referans olarak ekliyoruz.
- Index.html içerisindeki about.html sayfasına gidecek a etiketindeki href attribute'unu kategoriler.html olarak düzenliyoruz.

```
//kategoriler; kategoriler.html sayfasında bulunan data-page attribute'unda verilen değer.
myApp.onPageInit('kategoriler', function (page) {
    //Sayfa init olduğunda yapılacak işlemler
    alert('Sayfa açıldı');
})
```

Yukarıdaki işlemleri adım adım uyguladığınızda kategoriler.html sayfasını açmaya çalıştığınızda alert mesajını ekranda göreceksiniz. Artık bir sonraki adımda kategoriler.html sayfamıza veritabanımızdaki kategorileri WebApi kullanarak listelebiliriz.

- Index.html içerisinde bulunan list-block class'ına sahip liste yapısını (div ile birlikte) kopyalıyoruz.
- Kopyaladığımız listeyi, kategoriler.html sayfamızdaki paragrafları silerek class'ı content block olan div içerisine yapııştırıyoruz.
- UI içerisinde bulunan li etiketlerini silin.
- UI classına id="categoryList" ismini verin.
- MyCode.js sayfasını açın ve içerisindeki alert komutunu kaldırın.

Tüm bu işlemlerden sonra artık veri çekecek ajax kodumuzu oluşturuyoruz. Ajax üzerinden WebApi kullanarak veri listeleme işlemi yapacağız.

MyScript.js içerisinde

```
myApp.onPageInit('kategoriler', function (page) {
    $.ajax({
        url: 'http://localhost:50950/api/Category/ListCategories',
        type: 'GET',
        data: {
            q: 'Javascript', format: 'json', perTTY: 1
        }
    })
})
```

```

    },
    success: function (data) {
        $.each(data, function (index, value) {
            $("#categoryList").append('<li><a href="#" class="item-link"><div class="item-content"><div class="item-inner"><div class="item-title">' + value.CategoryName +
            '</div></div></div></a></li>');
        })
    }
})

```

- Ajax metodumuzu oluşturduktan sonra url olarak servis adresimizi veriyoruz.
- Type alanına işlemimizin bir get (veri çekme) işlemi olduğunu belirtiyoruz.
- Data property'sine gelecek data formatını yazıyoruz.
- Son olarak işlem başarılı olduğunda çalışacak callback metodumuzu oluşturuyoruz.

Service Data Göndermek

Service data göndermek için öncelikle web api projemizdeki Web.Config'de bir düzenleme yapmamız gerekmektedir. **System.webServer** etiketi altına aşağıdaki tanımlama satırını ekleyiniz.

Web.Config > System.webServer etiketi altında

```

<httpProtocol>
    <customHeaders>
        <add name="Access-Control-Allow-Origin" value="*" />
    </customHeaders>
</httpProtocol>

```

Sonraki adımda Ekleme işlemi yapılacak formu index.html içerisinde View-4'ü düzenliyoruz. View-4 içerisinde iki adet textbox ve bir buton olacak şekilde düzenlememizi bitiriyoruz.

Index.html > View-4

```

<input type="text" id="txtName" placeholder="Category Name">
<input type="text" id="txtDesc" placeholder="Description">
<input id="btnAdd" type="button" value="Save">

```

Ekle butonuna tıklandığı zaman çalıştırılacak olan jQuery ajax metodunu tetikliyoruz. Veriler json formatında gönderileceğinden dolayı dataType:json olarak işaretliyoruz.

Index.html e yazılacak javascript kodu

```

$("#btnAdd").click(function () {
    var c = new Object();
    c.CategoryName = $("#txtName").val();
    c.Description = $("#txtDesc").val();
    alert('dadada')
    $.ajax({
        type: "post",
        dataType: "json",

```

```
data: c,  
url: "http://localhost:50950/api/Category/InsertCategory",  
success: function (result) {  
    if (result == "success") {  
        alert("Eklendi")  
    }  
}  
});  
});
```