VIDEO: https://www.youtube.com/watch?v=gy7X1IZKeQQ&list=PLdo4fOcmZ0oUBAdL2NwBpDs32zwGqb9DY&index=7

|  |  |
| --- | --- |
| **ENGLISH** | **INDONESIA** |
| Today we're continuing our journey, learning how to build beautiful native cross-platform desktop and mobile apps with .Net MAUI.  Now, so far we've done a lot in our application. We've learned about what is in a file new project.  We started to build a user interface in XAML. We added data binding in an MVVM architecture pattern, and we even did some navigation.  Now next up is how to access Native API features. Now when we think about native APIs, things that are on iOS, Android, Mac, and Windows, these are things that are platform specific.  Now you can access these directly inside of C# thanks to the underlying systems of iOS and Android for.Net and Windows App SDK and UI3.  Now what's great about.Net MAUI is that the team abstracted a whole bunch of common platform APIs into a single API, for you to use as a developer.  This means when you want to go access common things such as geolocation, sensors, or connectivity, you don't have to learn a bunch of different APIs.  There's a single API that you can use.  Similar to the user interface, that abstracted common user interface controls into a single API, that's what these APIs do as well.  Now what we're going to do is go back to our application and learn how to integrate connectivity into our application, which is a very common feature in nearly every single application.  Let's head over there now. In this .Net MAUI Beginner Series, we've been building the My Task application.  It's a simple task application that allows us to add different tasks, and also allows us to swipe to delete, or tap on one and navigate to get details and go back.  All with the XAML user interface, data binding, and the built-in navigation system.  The last thing we want to do is add some platform capabilities here.  Now it would be pretty awesome to not allow people to add new tasks or synchronize them unless they have Internet connectivity, with .Net MAUI it's super simple to do that with the built-in APIs that abstract the underlying APIs of the platform.  Let's go ahead and do that. Here again, we have all of the applications that we've been building out.  I haven't added anything to it, so if you're following along at home, you can just start coding along with me.  Of course, you can find a full final version of the app in the show notes.  Here we have a main page and what we can see up top here is that we have a simple grid, an image, an entry, and this button called Add Command.  Below we have a CollectionView, we have SwipeViews and we have our view models here using full MVVM.  Now, if we want to get access specifically to different services or different APIs, there are tons that are built into .Net MAUI itself.  For example, let's say we go up to the Add Command here. What maybe you want to be able to do is check the Internet here.  So I can come in and say Microsoft Maui. Inside here there are a bunch of different namespaces such as application models, which would give me access to different things like app actions, app info, and launching the browser.  Here's communication, for example, which would give me access to contacts, e-mails, phone dialers, and a whole bunch of things.  There are also additional ones that we can see inside here, like authentication.  We of course have built-in controls and devices. There are all things that are inside here that you can get access to directly.  Now, the one that we need access to is going to be connectivity. When I type in connectivity, we can see it's in  Microsoft.Maui.Networking.Connectivity.  Now associated with the connectivity is also an interface too. I could come in and just check the current access state here so I can say what is  the current network access?  But if you're like me and you want your view models to be testable, you could be able to pass it an interface.  Let's go ahead and set that up. I'm going to say IConnectivity. Here we go. Then what I'm going to do is simply fill this in.  Of course, Visual Studio writes all the code for me, which is spectacular. Then what I'm able to do here is say if connectivity. and I'm going to say network access does not equal network access to the Internet or then I don't have Internet and I'm going to return.  What I can then do is say, await in here. We've gone ahead and modified this to say asynchronous, and we'll modify this to a task.  I'm going to say, Shell.Current.DisplayAlert and I'll say no Internet and say okay. Just with one line of code, I'm accessing the network access to see if I have Internet access or not and allowing me to add that new item to the list or not.  If I don't, I'll pop up a message. Now, we need to register that connectivity service with the main application, and we can do that in the MAUI program.  I'm going to do up the top and say, Builder.Services.AddSingleton and I'm going to say IConnectivity, and then what I can do is I can pass it,  the Connectivity.Current as what to use when it registers and resolves the IConnectivity.  Now, I'm going to go ahead and show this on Android here, but it works on obviously all operating systems. There we go.  What's great about the Android emulator is that I can flip on and off the Internet here very easily.  This is best done in the non-debug mode because the debugger does rely on some of the underlying network connectivity to actually debug your application.  But here's my application running and I should be able to add Apples in there. There we go.  Now if I go in and actually toggle on, for example, "Airplane Mode" that's going to toggle off the Internet and I'm going to go ahead and say Oranges and I'm going to add it's going to say, "Uh-Oh, No Internet."  Just like that, we're using native APIs, but completely cross-platform in the same exact code that we have inside of our main view model to check Internet access.  It's going to work 100 percent across iOS, Android, Mac OS, and Windows. The last thing I want to do is head over to my Mac.  We spent a lot of time inside Visual Studio 2022 on Windows, but I took the same exact My Task application and I loaded them up on the Mac.  You can go ahead and run your applications from Visual Studio 2022 on Mac or from the command line.  Here we have the Mac and iOS versions of the app. Here you can add items, get swipe to delete, you can tap and navigate and just like that, we have our full applications ready to go on both Mac, iOS, and, of course, Android and Windows.  There you've done it. You've integrated some platform-specific APIs, 100 percent cross-platform, thanks to .Net MAUI.  Now we're going to continue our journey in the.Net MAUI Beginner Series by taking a look at all of the other things that you need to know and resources to continue your learning journey.  Stay tuned for more in the .Net MAUI Beginner Series. | Hari ini kita akan melanjutkan perjalanan kita, mempelajari cara membuat aplikasi desktop dan mobile lintas platform yang indah dengan .Net MAUI.  Sekarang, sejauh ini kita telah melakukan banyak hal dalam aplikasi kita. Kita telah belajar tentang apa yang ada di dalam file proyek baru.  Kita mulai membangun antarmuka pengguna di XAML. Kita telah menambahkan pengikatan data dalam pola arsitektur MVVM, dan kita bahkan telah melakukan beberapa navigasi.  Sekarang selanjutnya adalah cara mengakses fitur Native API. Sekarang ketika kita berpikir tentang native API, hal-hal yang ada di iOS, Android, Mac, dan Windows, ini adalah hal-hal yang spesifik untuk setiap platform.  Sekarang Anda dapat mengaksesnya secara langsung di dalam C# berkat sistem yang mendasari iOS dan Android for.Net dan Windows App SDK dan UI3.  Sekarang, yang hebat dari for.Net MAUI adalah bahwa tim ini mengabstraksikan sejumlah besar API platform umum ke dalam satu API, untuk Anda gunakan sebagai pengembang.  Ini berarti ketika Anda ingin mengakses hal-hal umum seperti geolokasi, sensor, atau konektivitas, Anda tidak perlu mempelajari banyak API yang berbeda.  Ada satu API yang bisa Anda gunakan.  Mirip dengan antarmuka pengguna, yang mengabstraksikan kontrol antarmuka pengguna umum ke dalam satu API, itulah yang dilakukan oleh API ini juga.  Sekarang yang akan kita lakukan adalah kembali ke aplikasi kita dan mempelajari cara mengintegrasikan konektivitas ke dalam aplikasi kita, yang merupakan fitur yang sangat umum di hampir setiap aplikasi.  Mari kita pergi ke sana sekarang. Dalam Seri Pemula .Net MAUI ini, kita telah membangun aplikasi My Task.  Ini adalah aplikasi tugas sederhana yang memungkinkan kita untuk menambahkan tugas yang berbeda, dan juga memungkinkan kita untuk menggesek untuk menghapus, atau mengetuk salah satu tugas dan menavigasi untuk mendapatkan detail dan kembali.  Semuanya dengan antarmuka pengguna XAML, pengikatan data, dan sistem navigasi bawaan.  Hal terakhir yang ingin kami lakukan adalah menambahkan beberapa kemampuan platform di sini.  Sekarang akan sangat luar biasa untuk tidak mengizinkan orang menambahkan tugas baru atau menyinkronkannya kecuali mereka memiliki konektivitas Internet, dengan .Net MAUI, sangat mudah untuk melakukannya dengan API bawaan yang mengabstraksikan API yang mendasari platform.  Mari kita lanjutkan dan lakukan. Di sini sekali lagi, kita memiliki semua aplikasi yang telah kita bangun.  Saya belum menambahkan apa pun ke dalamnya, jadi jika Anda mengikutinya di rumah, Anda bisa mulai membuat kode bersama saya.  Tentu saja, Anda bisa menemukan versi final aplikasi yang lengkap di catatan acara.  Di sini kita memiliki halaman utama dan apa yang bisa kita lihat di atas adalah kita memiliki kisi-kisi sederhana, gambar, entri, dan tombol yang disebut Add Command.  Di bawah ini kita memiliki CollectionView, kita memiliki SwipeView dan kita memiliki model tampilan di sini menggunakan MVVM penuh.  Sekarang, jika kita ingin mendapatkan akses secara khusus ke layanan yang berbeda atau API yang berbeda, ada banyak sekali yang dibangun ke dalam .Net MAUI itu sendiri.  Sebagai contoh, katakanlah kita pergi ke Add Command di sini. Apa yang mungkin ingin Anda lakukan adalah memeriksa Internet di sini.  Jadi saya bisa masuk dan mengatakan Microsoft Maui. Di dalam sini ada banyak ruang nama yang berbeda seperti model aplikasi, yang akan memberi saya akses ke berbagai hal seperti tindakan aplikasi, info aplikasi, dan meluncurkan peramban.  Di sini ada komunikasi, misalnya, yang akan memberi saya akses ke kontak, email, pemanggil telepon, dan banyak hal lainnya.  Ada juga yang tambahan yang bisa kita lihat di dalam sini, seperti autentikasi.  Tentu saja kami memiliki kontrol dan perangkat bawaan. Ada banyak hal yang ada di dalam sini yang bisa Anda akses secara langsung.  Sekarang, salah satu yang perlu kita akses adalah konektivitas. Ketika saya mengetikkan konektivitas, kita bisa melihatnya di  Microsoft.Maui.Networking.Connectivity.  Nah, yang terkait dengan konektivitas juga merupakan sebuah antarmuka. Saya bisa masuk dan memeriksa status akses saat ini di sini sehingga saya bisa mengatakan apa  akses jaringan saat ini?  Tetapi jika Anda seperti saya dan Anda ingin model tampilan Anda dapat diuji, Anda dapat memberikannya sebuah antarmuka.  Mari kita lanjutkan dan mengaturnya. Saya akan mengatakan IConnectivity. Ini dia. Kemudian yang akan saya lakukan adalah mengisinya.  Tentu saja, Visual Studio menulis semua kodenya untuk saya, dan ini sangat spektakuler. Kemudian apa yang dapat saya lakukan di sini adalah mengatakan jika konektivitas. dan saya akan mengatakan akses jaringan tidak sama dengan akses jaringan ke Internet atau saya tidak memiliki Internet dan saya akan kembali.  Apa yang bisa saya lakukan adalah mengatakan, tunggu di sini. Kami telah memodifikasi ini untuk mengatakan asinkron, dan kami akan memodifikasi ini menjadi tugas.  Saya akan mengatakan, Shell.Current.DisplayAlert dan saya akan mengatakan tidak ada Internet dan mengatakan oke. Hanya dengan satu baris kode, saya mengakses akses jaringan untuk melihat apakah saya memiliki akses Internet atau tidak dan mengizinkan saya untuk menambahkan item baru tersebut ke dalam daftar atau tidak.  Jika tidak, saya akan memunculkan sebuah pesan. Sekarang, kita perlu mendaftarkan layanan konektivitas tersebut dengan aplikasi utama, dan kita dapat melakukannya di program MAUI.  Saya akan melakukan di bagian atas dan mengatakan, Builder.Services.AddSingleton dan saya akan mengatakan IConnectivity, dan kemudian apa yang dapat saya lakukan adalah saya dapat meneruskannya, Connectivity.Current sebagai apa yang harus digunakan ketika mendaftar dan menyelesaikan IConnectivity.  Sekarang, saya akan menunjukkannya di Android di sini, tetapi ini bisa digunakan di semua sistem operasi. Ini dia.  Yang hebat dari emulator Android ini adalah saya bisa menyalakan dan mematikan Internet dengan sangat mudah.  Hal ini paling baik dilakukan dalam mode non-debug karena debugger bergantung pada beberapa konektivitas jaringan yang mendasari untuk benar-benar men-debug aplikasi Anda.  Tapi di sini aplikasi saya berjalan dan saya seharusnya bisa menambahkan Apel di sana. Ini dia.  Sekarang jika saya masuk dan benar-benar mengaktifkan, misalnya, "Mode Pesawat" yang akan mematikan Internet dan saya akan melanjutkan dan mengatakan Jeruk dan saya akan menambahkannya akan mengatakan, "Uh-Oh, Tidak Ada Internet."  Dengan begitu, kita menggunakan API asli, tetapi sepenuhnya lintas platform dengan kode yang sama persis dengan yang kita miliki di dalam model tampilan utama untuk memeriksa akses Internet.  Ini akan bekerja 100 persen di iOS, Android, Mac OS, dan Windows. Hal terakhir yang ingin saya lakukan adalah membuka Mac.  Kami menghabiskan banyak waktu di dalam Visual Studio 2022 di Windows, tetapi saya mengambil aplikasi My Task yang sama persis dan saya memuatnya di Mac.  Anda bisa langsung menjalankan aplikasi Anda dari Visual Studio 2022 di Mac atau dari command line.  Di sini kami memiliki versi aplikasi Mac dan iOS. Di sini Anda dapat menambahkan item, menggesek untuk menghapus, Anda dapat mengetuk dan menavigasi dan begitu saja, kami memiliki aplikasi lengkap yang siap digunakan di Mac, iOS, dan, tentu saja, Android dan Windows.  Anda sudah berhasil melakukannya. Anda telah mengintegrasikan beberapa API khusus platform, 100 persen lintas platform, berkat .Net MAUI.  Sekarang kita akan melanjutkan perjalanan kita di Seri Pemula .Net MAUI dengan melihat semua hal lain yang perlu Anda ketahui dan sumber daya untuk melanjutkan perjalanan belajar Anda.  Nantikan lebih banyak lagi di Seri Pemula .Net MAUI. |
|  |  |