VIDEO: <https://www.youtube.com/watch?v=l5qJRYwhyOY&list=PLdo4fOcmZ0oUBAdL2NwBpDs32zwGqb9DY&index=4>

|  |  |
| --- | --- |
| **ENGLISH** | **INDONESIA** |
| Today we're continuing our journey of building native, cross-platform, desktop, and mobile applications with .NET MAUI.  Now, previously we learned what .NET MAUI is, how to install it, and everything that's in a file new project.  Next up is creating a user interface and taking advantage of all the different layouts and controls built into .NET MAUI.  We're going to build a small application that is a to-do list that enables us to keep track of all the things that we want to do and also how to update our user interface when the users interact with it.  So let's hop over to Visual Studio 2022 and get started.  This is the application that we're going to build, it's called My Tasks.  Here on the Home Screen, we'll be able to enter a task, click "Add" and add it to this list.  There's going to be a Collection View, which is a nicely optimized list display that enables a whole bunch of new features and grouping capabilities.  So we're going to check that out today.  Now, as of current, our application looks like this because all we've done is created a brand new .NET MAUI application.  Now, what's great about the template is that it's fully responsive.  So here we have a light theme and a dark theme, and you can click to increase it, which is awesome.  Let's go ahead and turn this into my tasks.  The first thing that we need to do is go ahead and head over to the code.  Now, this is the default template.  So here we have some images, some labels, and the button.  Now, what we're going to do is delete all of it.  Now, we don't need a scroll view here because if you remember from the photo, this down, that has apples, bananas, and oranges is going to be a CollectionView, and that already scrolls.  What we're going to use in the base is going to be a grid.    I like grids a lot because they are rows and columns and I can put that into my head that here there'll be three specific rows that image, the enter tasks and add button, and then the CollectionView.  There'll also be two columns.  Here we're going to have a column for the center task and then another one for the button and both the logo and also CollectionView will span both of them.  We've deleted all the code, but we also want to take a look at the code behind here, this main page, XAML CS. Now there's code behind it.  At one point when we did file new represented some interactions with the main user interface.  So we're going to go out and delete those because we're not going to use those button clickers there.  Let's go and delete them.  I'm going to hit "Debug."  The reason I'm going to do that is that I'm going to start building my user interface directly with XAML reload so we can make changes and see it update in real-time.  Our application is launched.  It's a blank screen now, and all we have is a grid.  Let's go ahead and define those different rows and columns.  So I'm going to say row definition and the IntelliSense helps me out here, I want to say 100.  Those are three density pixels that automatically adjust based on the resolution.  For the second row, I'm just going to use auto that's just going to tell it to use as much space as it possibly needs for the controls.  Then the last row or row 2, so 012 of index 0, that one's going to fill it.  We're also going to use the column definitions.  Here again, there are going to be two columns.  Now, if I wanted them to just size themselves, I could say auto, auto.  But I want them to be a little bit different size.  I want the first one to be 75 percent of the second one.  So I'm going to say 0.75 stars and 0.25 stars, and that's going to give a ratio of 75 percent and 25 percent.  Now, so far, we still don't see anything because we haven't put anything in it.  The first thing that we're going to do is we're going to put an image in here.  Let's put an image and then I'm going to go ahead and say Grid.ColumnSpan so it spans both columns, set it to two.  The source here, well, we don't have any images in here yet,  but we could use that .NET bar under resources. So dotnet\_bot.png.  So let's go and put that in there, Dotnet bot.png.  I'm going to close that tag and sure enough, there's a Dotnet bot. Hey, Dotnet bott.  Now, what's great here is that I could also set the background color if I wanted to so see what exactly is filling.  Here we can see it fills the entirety of it, which is pretty great and it's directly in the center as the default.  I like that so far, but we can see that is filling every single corner, which means that everything is going to be filled.  Let's go ahead on this grid and put a padding and I'm going to put a padding of ten as a little inset here.  So the Dotnet bot is a little bit inside and not hugging that home bar.  Next up, we're going to put in an entry.  So here's an entry and I'm going to put in a placeholder.  An entry is a text entry field and you can do all sorts of great things, like make a password and things like that.    The placeholder here will be entering the task and I'm going to make sure that it is at a "grid.row" equals 1. There we go.  I can say hello world. Now, the important thing to note here is that the automatic default is 0, 0.  So I didn't need to define a grid dot row or react column on the image because it's already at 0, 0.  Same thing here on the "grid.column" on the entry.  I didn't have to enter it because it defaults to zero.  I guess I'll put it there if I wanted to, totally your choice.  Let's add a button here. Now we're going to add a button and set the text to add.  Now we're going to set that "grid.row" to one and "grid.column" to one as well.  We'll close the tag and now our button's there.  Now you're probably saying, James, wow, those things are all close to each other.  Now, I could add a margin to each of these and the margin says, Hey, stay a little bit further away from me.  But built into the grid are row spacings and column spacings too. Here we go.  I want to say 10, 10 and hit "Save," we can see that now we get a little bit more space there, which is great.  Next up is going to be our CollectionView.  So this is going to be a way for us to hold a bunch of items.  I want to put in a CollectionView.  Here we're going to say "grid.row" 2 and "grid.columnspan" 2 as well,  and we're going to go ahead and open this up because we're going to put some things inside of it.  The first thing that we're going to do is give us some hard-coded data, and this is just so we can design the user interface.  Later on, we're going to fill it in with real data.  So whatever we enter adds it to the CollectionView.  Here we're going to say CollectionView.  ItemSource and then what we can do is fill it with an array.  Now this is one of my favorite features because you can just actually say "X:Array", and then I'll create an array in the XAML and here I can say, Hey, array, please be a string.  Now we're going to create an array of strings.  Now, I can put my string in here and I can say apples, and then go ahead and copy these, and I'll say bananas and I'll say oranges, and hit "Save."  Now we can see that apples, bananas, and oranges showed up, which is great and of course, I can fix my typos too.  Now we have all our collection and you're totally good to go.  Now, one thing I want to note here is that they're not looking super duper great because it's just going and using the default template, which is just a label, and that's what it's doing  here for the string.  So we can go ahead and give it something called an item template to spice it up a little bit.  Here we go underneath the item source, we're going to say CollectionView.ItemTemplate.  We're going to be doing some data binding here, and we're going to get into it a little bit more in our next video.  But what it means is I want to take these apples and say, please use that inside of my item template.  So by default, we'll use the data template here, and then what we can do is put in, let's say, a label inside of it, maybe inside of a nice little card view frame, and we'll be good to go.    The default I'm going to use here is a grid and I'm going to give it a little padding of 0, 5. Inside that grid, we're going to give it a frame, and then what we're going to do is give it a label.  What we're going to do is create a label that is the text that has a binding to dot, which says, bind to me.  When I hit save here, we're going to see that we have apples, bananas, and oranges and we can also set that text to be a little bit bigger.  So let's say font size 24, for example.  Now we're talking apples, bananas, and oranges and everything resizes for us automatically.  Now we're able to add our new items here or will be in a little bit but what if we wanted to delete them?  We could add a delete button.  We could try to like click on one.  But what if we could also swipe to delete?  That sure would be nice. Well, Dynamo includes all the necessary features and capabilities  What we'll do is we're going to wrap this grid inside of a swipe view.  Let's go ahead and do that.  So when I say swipe view. There we go.  What that does is says, please put everything inside of this grid into a swipe view and then we'll add some swipe items.  I'm going to say swipe view dot right items.  You could do left, top or bottom, but we're going to do right and then I'm going to give it a list of swipe items.  Now, you can have completely customized swipe items, but I'm going to use the one that's built-in.    This enables me to set text like delete in the background color of red.  I'm going to go ahead and save again.  Now we can see that we have to delete it right there.  Look at that. That's awesome. Very cool.  Well, we still don't have our image fixed up, so let's go ahead and remove that orange.  Now we're back to Transparent. Let's put our logo in there.  I'm going to stop debugging for this and what we're going to do is we're going to go into our resources, and images, right-click on "Add" and then do the existing item.  Now over on my desktop, I'm going to select "All files" and I have my logo.png.  Now I've added it in here and we can see it referenced what I can do at this point is come in and say logo.png.  Now I'll go ahead and recompile the application and what this will do is take that resource and automatically recompile it to display it inside of my application.  Now, this time, it's not going to be as quick.  It may take a few additional seconds as it compiles up those resources.  Let's give it a few seconds here.  Our application is launching and now what we see is my task logo right there can now come in, enter my tasks, enter a task, and soon we'll be able to add it to the list and get rid of that hard-coded data.  That's what we're going to do next when we learn how to use MVVM and data binding inside the My Task app that we just started to create.  There you have it. We've started to build out our user interface completely in XAML.  Again, this is just one of the options that developers have when creating user interfaces with .NET MAUI.  You can also create your user interfaces directly in C# and there's a great community toolkit, in fact, the .NET MAUI Community Toolkit, which is built by the community and supported by Microsoft.  There's a whole bunch of great .NET MAUI goodness in it, including C# markup extensions that help you be even more productive when you're building user interfaces with C#.  I'll put links in the shown-up below.  Now, since we are building our user interface with XAML, it comes with great developer productivity features like data binding, which enables your UI to automatically react when you update code in the code-behind and vice versa when you update code in the code-behind will automatically update your user interface.  Next what we're going to do is we will go ahead and start using an MVVM "model-view-view-model" architecture and data binding to update our application to be reactive and responsive.  Stay tuned for the next video. | Hari ini kita akan melanjutkan perjalanan kita dalam membuat aplikasi native, lintas platform, desktop, dan mobile dengan .NET MAUI.  Sekarang, sebelumnya kita telah mempelajari apa itu .NET MAUI, cara menginstalnya, dan semua yang ada di dalam file proyek baru.  Selanjutnya adalah membuat antarmuka pengguna dan memanfaatkan semua tata letak dan kontrol yang berbeda yang ada di dalam .NET MAUI.  Kita akan membuat sebuah aplikasi kecil yang merupakan daftar tugas yang memungkinkan kita untuk melacak semua hal yang ingin kita lakukan dan juga cara memperbarui antarmuka pengguna kita ketika pengguna berinteraksi dengannya.  Jadi, mari kita buka Visual Studio 2022 dan mulai.  Ini adalah aplikasi yang akan kita buat, namanya My Tasks.  Di sini, di Layar Awal, kita akan dapat memasukkan tugas, klik "Tambah" dan tambahkan ke daftar ini.  Akan ada Tampilan Koleksi, yang merupakan tampilan daftar yang dioptimalkan dengan baik yang memungkinkan sejumlah fitur baru dan kemampuan pengelompokan.  Jadi kita akan memeriksanya hari ini.  Sekarang, untuk saat ini, aplikasi kita terlihat seperti ini karena yang telah kita lakukan adalah membuat aplikasi .NET MAUI yang baru.  Nah, yang hebat dari templat ini adalah bahwa templat ini sepenuhnya responsif.  Jadi di sini kita memiliki tema terang dan tema gelap, dan Anda dapat mengklik untuk meningkatkannya, yang luar biasa.  Mari kita lanjutkan dan ubah ini menjadi tugas saya.  Hal pertama yang perlu kita lakukan adalah membuka kodenya.  Sekarang, ini adalah template default.  Jadi di sini kita memiliki beberapa gambar, beberapa label, dan tombol.  Sekarang, yang akan kita lakukan adalah menghapus semuanya.  Sekarang, kita tidak memerlukan tampilan gulir di sini karena jika Anda ingat dari foto, yang di bawah ini, yang memiliki apel, pisang, dan jeruk akan menjadi CollectionView, dan itu sudah bergulir.  Apa yang akan kita gunakan di bagian dasar akan menjadi grid.    Saya sangat menyukai grid karena mereka adalah baris dan kolom dan saya dapat menempatkannya di kepala saya bahwa di sini akan ada tiga baris tertentu yaitu gambar, tombol masukkan tugas dan tambah, dan kemudian CollectionView.  Juga akan ada dua kolom.  Di sini kita akan memiliki satu kolom untuk tugas tengah dan satu lagi untuk tombol dan kedua logo dan juga CollectionView akan menjangkau keduanya.  Kita telah menghapus semua kode, tapi kita juga ingin melihat kode di belakangnya, halaman utama ini, XAML CS. Sekarang ada kode di belakangnya.  Pada satu titik ketika kita membuat file baru yang merepresentasikan beberapa interaksi dengan antarmuka pengguna utama.  Jadi kita akan menghapusnya karena kita tidak akan menggunakan tombol-tombol tersebut di sana.  Mari kita hapus saja.  Saya akan menekan "Debug."  Alasan saya melakukan itu adalah karena saya akan mulai membangun antarmuka pengguna secara langsung dengan memuat ulang XAML sehingga kita dapat membuat perubahan dan melihatnya diperbarui secara real-time.  Aplikasi kami diluncurkan.  Sekarang layarnya kosong, dan yang kita miliki hanyalah kisi-kisi.  Mari kita lanjutkan dan tentukan baris dan kolom yang berbeda.  Jadi saya akan mengatakan definisi baris dan IntelliSense membantu saya di sini, saya ingin mengatakan 100.  Itu adalah tiga piksel densitas yang secara otomatis menyesuaikan berdasarkan resolusi.  Untuk baris kedua, saya hanya akan menggunakan auto yang hanya akan memerintahkannya untuk menggunakan ruang sebanyak yang dibutuhkan untuk kontrol.  Kemudian baris terakhir atau baris 2, jadi 012 dari indeks 0, yang akan mengisinya.  Kita juga akan menggunakan definisi kolom.  Di sini sekali lagi, akan ada dua kolom.  Sekarang, jika saya ingin ukurannya sama saja, saya bisa mengatakan otomatis, otomatis.  Tapi saya ingin ukurannya sedikit berbeda.  Saya ingin yang pertama berukuran 75 persen dari yang kedua.  Jadi saya akan mengatakan 0,75 bintang dan 0,25 bintang, dan itu akan memberikan rasio 75 persen dan 25 persen.  Sekarang, sejauh ini, kami masih belum melihat apa pun karena kami belum memasukkan apa pun ke dalamnya.  Hal pertama yang akan kita lakukan adalah menaruh gambar di sini.  Mari kita taruh gambar dan kemudian saya akan langsung mengatakan Grid.ColumnSpan sehingga menjangkau kedua kolom, atur ke dua.  Sumbernya di sini, yah, kita belum memiliki gambar di sini,  tapi kita bisa menggunakan bilah .NET di bawah sumber daya. Jadi dotnet\_bot.png.  Jadi mari kita taruh di sana, Dotnet bot.png.  Saya akan menutup tag itu dan tentu saja, ada bot Dotnet. Hei, Dotnet bott.  Nah, yang hebat di sini adalah saya juga bisa mengatur warna latar belakangnya jika saya mau, jadi kita bisa melihat apa yang sebenarnya terisi.  Di sini kita bisa melihat bahwa bot ini mengisi seluruh bagian, yang cukup bagus dan langsung berada di tengah sebagai default.  Sejauh ini saya suka itu, tetapi kita bisa melihat bahwa itu mengisi setiap sudut, yang berarti semuanya akan terisi.  Mari kita lanjutkan pada kisi-kisi ini dan letakkan padding dan saya akan meletakkan padding sepuluh sebagai inset kecil di sini.  Jadi bot Dotnet sedikit masuk ke dalam dan tidak memeluk bilah beranda.  Selanjutnya, kita akan memasukkan entri.  Jadi, inilah entri dan saya akan memasukkan placeholder.  Entri adalah bidang entri teks dan Anda dapat melakukan berbagai macam hal hebat, seperti membuat kata sandi dan hal-hal seperti itu.    Placeholder di sini akan memasukkan tugas dan saya akan memastikan bahwa "grid.row" sama dengan 1. Ini dia.  Saya bisa menyapa dunia. Sekarang, hal penting yang perlu diperhatikan di sini adalah bahwa default otomatisnya adalah 0, 0.  Jadi saya tidak perlu mendefinisikan grid dot row atau react column pada gambar karena sudah ada di 0, 0.  Hal yang sama di sini pada "grid.column" pada entri.  Saya tidak perlu memasukkannya karena secara default adalah nol.  Saya kira saya akan menaruhnya di sana jika saya mau, terserah Anda.  Mari tambahkan sebuah tombol di sini. Sekarang kita akan menambahkan sebuah tombol dan mengatur teks yang akan ditambahkan.  Sekarang kita akan mengatur "grid.row" menjadi satu dan "grid.column" menjadi satu juga.  Kita akan menutup tag dan sekarang tombol kita sudah ada di sana.  Sekarang Anda mungkin berkata, James, wow, semua itu berdekatan satu sama lain.  Sekarang, saya bisa menambahkan margin ke masing-masing dan margin mengatakan, Hei, menjauhlah sedikit dari saya.  Tetapi di dalam grid juga terdapat jarak baris dan jarak kolom. Ini dia.  Saya ingin mengatakan 10, 10 dan tekan "Simpan," kita dapat melihat bahwa sekarang kita mendapatkan sedikit lebih banyak ruang di sana, dan itu bagus.  Berikutnya adalah CollectionView kita.  Jadi ini akan menjadi cara bagi kita untuk menyimpan banyak item.  Saya ingin memasukkan CollectionView.  Di sini kita akan mengatakan "grid.row" 2 dan "grid.columnspan" 2 juga,  dan kita akan membuka ini karena kita akan menaruh beberapa hal di dalamnya.  Hal pertama yang akan kita lakukan adalah memberikan beberapa data yang sudah dikodekan, dan ini agar kita dapat mendesain antarmuka pengguna.  Nantinya, kita akan mengisinya dengan data yang sebenarnya.  Jadi, apa pun yang kita masukkan akan ditambahkan ke CollectionView.  Di sini kita akan mengatakan CollectionView.  ItemSource dan kemudian yang bisa kita lakukan adalah mengisinya dengan array.  Ini adalah salah satu fitur favorit saya karena Anda bisa mengatakan "X: Array", dan kemudian saya akan membuat array di XAML dan di sini saya bisa mengatakan, Hei, array, tolong jadikan ini sebuah string.  Sekarang kita akan membuat sebuah array dari string.  Sekarang, saya bisa meletakkan string saya di sini dan saya bisa mengatakan apel, dan kemudian silakan salin ini, dan saya akan mengatakan pisang dan saya akan mengatakan jeruk, dan tekan "Simpan."  Sekarang kita bisa melihat apel, pisang, dan jeruk muncul, dan tentu saja, saya bisa memperbaiki kesalahan ketik saya juga.  Sekarang kita sudah memiliki semua koleksi kita dan Anda sudah siap untuk melanjutkannya.  Sekarang, satu hal yang ingin saya catat di sini adalah bahwa mereka tidak terlihat sangat bagus karena hanya menggunakan templat default, yang hanya berupa label, dan itulah yang dilakukannya  di sini untuk string.  Jadi kita dapat melanjutkan dan memberikan sesuatu yang disebut template item untuk membumbui sedikit.  Di sini kita pergi di bawah sumber item, kita akan mengatakan CollectionView.ItemTemplate.  Kita akan melakukan beberapa pengikatan data di sini, dan kita akan membahasnya lebih lanjut di video kami berikutnya.  Tapi maksudnya adalah saya ingin mengambil apel-apel ini dan mengatakan, silakan gunakan itu di dalam template item saya.  Jadi secara default, kita akan menggunakan templat data di sini, dan kemudian apa yang dapat kita lakukan adalah memasukkan, katakanlah, label di dalamnya, mungkin di dalam bingkai tampilan kartu kecil yang bagus, dan kita siap melakukannya.  Default yang akan saya gunakan di sini adalah grid dan saya akan memberinya sedikit padding 0, 5. Di dalam grid itu, kita akan memberinya bingkai, dan kemudian apa yang akan kita lakukan adalah memberinya label.  Apa yang akan kita lakukan adalah membuat label yang merupakan teks yang memiliki pengikatan ke titik, yang bertuliskan, ikat ke saya.  Ketika saya menekan simpan di sini, kita akan melihat bahwa kita memiliki apel, pisang, dan jeruk dan kita juga dapat mengatur teks tersebut menjadi sedikit lebih besar.  Jadi, katakanlah ukuran hurufnya 24, misalnya.  Sekarang kita berbicara tentang apel, pisang, dan jeruk dan semuanya berubah ukurannya secara otomatis.  Sekarang kita dapat menambahkan item baru di sini atau akan segera ditambahkan, tetapi bagaimana jika kita ingin menghapusnya?  Kita bisa menambahkan tombol hapus.  Kita bisa mencoba untuk mengklik salah satunya.  Tetapi bagaimana jika kita juga bisa menggeser untuk menghapus?  Itu pasti akan menyenangkan. Nah, Dynamo menyertakan semua fitur dan kemampuan yang diperlukan  Yang akan kita lakukan adalah membungkus kisi-kisi ini di dalam tampilan swipe.  Mari kita lanjutkan dan lakukan.  Jadi, ketika saya mengatakan swipe view. Ini dia.  Apa yang dilakukannya adalah mengatakan, silakan letakkan semua yang ada di dalam grid ini ke dalam swipe view dan kemudian kita akan menambahkan beberapa item swipe.  Saya akan mengatakan swipe view dot right item.  Anda bisa melakukan kiri, atas atau bawah, tetapi kita akan melakukan kanan dan kemudian saya akan memberikan daftar item gesek.  Sekarang, Anda dapat memiliki item gesek yang sepenuhnya disesuaikan, tetapi saya akan menggunakan yang sudah ada di dalamnya.    Ini memungkinkan saya untuk mengatur teks seperti hapus dengan warna latar belakang merah.  Saya akan melanjutkan dan menyimpan lagi.  Sekarang kita bisa melihat bahwa kita harus menghapusnya di sana.  Lihat itu. Keren sekali. Sangat keren.  Nah, kita masih belum memperbaiki gambar kita, jadi mari kita hapus warna oranye itu.  Sekarang kita kembali ke Transparan. Mari kita taruh logo kita di sana.  Saya akan berhenti melakukan debug untuk ini dan apa yang akan kita lakukan adalah kita akan masuk ke sumber daya kita, dan gambar, klik kanan pada "Add" dan kemudian lakukan item yang ada.  Sekarang di desktop saya, saya akan memilih "Semua file" dan saya memiliki logo.png.  Sekarang saya telah menambahkannya di sini dan kita dapat melihatnya direferensikan apa yang dapat saya lakukan pada saat ini adalah masuk dan mengatakan logo.png.  Sekarang saya akan melanjutkan dan mengkompilasi ulang aplikasi dan apa yang akan dilakukannya adalah mengambil sumber daya tersebut dan secara otomatis mengkompilasinya untuk menampilkannya di dalam aplikasi saya.  Sekarang, kali ini, prosesnya tidak akan cepat.  Mungkin diperlukan beberapa detik tambahan saat mengkompilasi sumber daya tersebut.  Mari kita beri waktu beberapa detik di sini.  Aplikasi kita diluncurkan dan sekarang yang kita lihat adalah logo tugas saya di sana sekarang dapat masuk, memasukkan tugas saya, memasukkan tugas, dan segera kita akan dapat menambahkannya ke daftar dan menyingkirkan data yang sudah dikodekan.  Itulah yang akan kita lakukan selanjutnya saat kita mempelajari cara menggunakan MVVM dan pengikatan data di dalam aplikasi My Task yang baru saja kita buat.  Itu dia. Kita sudah mulai membangun antarmuka pengguna sepenuhnya di XAML.  Sekali lagi, ini hanyalah salah satu opsi yang dimiliki pengembang ketika membuat antarmuka pengguna dengan .NET MAUI.  Anda juga dapat membuat antarmuka pengguna langsung di C# dan ada toolkit komunitas yang hebat, pada kenyataannya, .NET MAUI Community Toolkit, yang dibangun oleh komunitas dan didukung oleh Microsoft.  Ada banyak sekali kebaikan .NET MAUI yang hebat di dalamnya, termasuk ekstensi markup C# yang membantu Anda menjadi lebih produktif ketika Anda membangun antarmuka pengguna dengan C#.  Saya akan menaruh tautan pada gambar di bawah ini.  Sekarang, karena kita membangun antarmuka pengguna kita dengan XAML, XAML hadir dengan fitur produktivitas pengembang yang hebat seperti pengikatan data, yang memungkinkan UI Anda bereaksi secara otomatis ketika Anda memperbarui kode di kode di belakang dan sebaliknya ketika Anda memperbarui kode di kode di belakang secara otomatis akan memperbarui antarmuka pengguna Anda.  Selanjutnya yang akan kita lakukan adalah kita akan melanjutkan dan mulai menggunakan arsitektur MVVM "model-tampilan-tampilan-model" dan pengikatan data untuk memperbarui aplikasi kita menjadi reaktif dan responsif.  Nantikan video berikutnya. |
|  |  |