* Blazor, uygulamaları daha hızlı oluşturmaya başlamanıza yardımcı olan Blazor Server ve Blazor WebAssembly olmak üzere iki şablonla birlikte gelir.
* Blazor WebAssembly, bağımsız ve barındırılan iki yapılandırmaya sahip daha karmaşık şablondur. Bağımsız, mevcut bir arka ucunuz olduğunda veya uygulamanız bir arka uç gerektirmediğinde yararlı olan tek bir Blazor WebAssembly projesi oluşturur. Barındırılan, bir ASP.NET Core arka ucu, Blazor ön ucu ve paylaşılan kod için bir .NET Standard sınıf kitaplığı ile tam bir yığın uygulaması sağlar.
* Blazor uygulamaları, Visual Studio gibi bir IDE ile veya .NET CLI kullanılarak komut satırından oluşturulabilir.
* Uygulamanızı oluşturmak, ihtiyaç duyduğu tüm bağımlılıkları otomatik olarak geri yükleyecektir.
* Blazor uygulamaları, Blazor uygulamasının oluşturulacağı HTML öğesini ve Blazor JavaScript çalışma zamanına bir bağlantı içeren bir ana bilgisayar sayfası kullanır.
* Blazor WebAssembly uygulamalarının bir Startupsınıfı yoktur, yalnızca bir Program sınıfı vardır, çünkü bir ara yazılım ardışık düzeni yoktur. Hizmet yapılandırması ve kaydı Program.cs.
* Blazor WebAssembly için, Program.csbir örneğini oluşturmak ve çalıştırmak için kullanılır WebAssemblyHost. aracılığıyla yapılır WebAssemblyHostBuilder. Oluşturucuyu, Blazor uygulamamızın kök bileşenleri ve hizmet kapsayıcısı gibi çeşitli yönlerini yapılandırmak için kullanırız.
* App.razor, varsayılan kök bileşendir ve Yönlendirici bileşenini içerir. Diğer tüm bileşenler, öğesinin alt öğeleri olarak işlenecektir App.
* Özellik klasörleri, uygulamanızdaki dosyaları düzenlerken bir dizi avantaj sağlayabilir. Bir özellikle ilgili her şey tek bir yerde bulunur ve güncellemeleri ve bakımı kolaylaştırır.
* Düzen bileşenleri, başlıklar ve gezinme menüleri gibi her sayfada tekrarlanacak olan ortak kullanıcı arabirimini tanımlamanın harika bir yoludur.
* Değerler, bir bileşen için API olarak düşünülebilecek parametreler aracılığıyla bileşenlere aktarılabilir. Parametreler genel mülk olmalıdır; özel olamazlar.