**v-enter-active:** Giriş için aktif durum. Tüm giriş aşaması boyunca uygulanır. Öğe eklenmeden önce eklenir, geçiş/animasyon bittiğinde kaldırılır. Bu sınıf, giriş geçişi için süreyi, gecikmeyi ve hızlanma eğrisini tanımlamak için kullanılabilir.

**v-enter-to:** Enter için son durum. Eleman yerleştirildikten sonra bir kare eklendi (aynı zamanda v-enter-from kaldırıldı), geçiş / animasyon bittiğinde kaldırıldı.

**v-leave-from:** Ayrilis için başlangıç durumu. Bir ayrılma geçişi tetiklendiğinde hemen eklenir, bir kareden sonra kaldırılır.

**v-leave-active:** Ayrilis için aktif durum. Ayrılma aşamasının tamamı boyunca uygulanır. Çıkış geçişi tetiklendiğinde hemen eklenir, geçiş/animasyon bittiğinde kaldırılır. Bu sınıf, ayrılma geçişi için süreyi, gecikmeyi ve hızlanma eğrisini tanımlamak için kullanılabilir.

**v-leave-to:** Ayrilis için bitiş durumu. Çıkış geçişi tetiklendikten sonra bir kare eklendi (aynı zamanda v-leave-from kaldırıldı), geçiş/animasyon bittiğinde kaldırıldı.

**Named Transition**

Adlandırılmış bir Transition için, Transition classlarinin önüne v yerine kendi adı eklenir. Örneğin, örnek Transition için uygulanan sınıf, v-enter-active yerine fade-enter-active olacaktır.

**CSS animations in Transitions**

Native CSS animasyonları, CSS geçişleriyle aynı şekilde uygulanır. Yani kendimiz de css animasyonlari verebiliriz.

**Custom Transition Classes(Ozel gecis siniflari)**

Aşağıdaki donanımları <Transition>'a props olarak geçirerek özel geçiş sınıfları da belirtebilirsiniz:

Bunlar, geleneksel sınıf adlarını geçersiz kılar. Bu, özellikle Vue'nun geçiş sistemini Animate.css gibi mevcut bir CSS animasyon kitaplığıyla birleştirmek istediğinizde kullanışlıdır.

**1 Nested Transitions and Explicit Transition Durations**

Transition sınıfları yalnızca <Transition> içindeki doğrudan alt elemana uygulansa da, iç içe geçmiş CSS seçicilerini kullanarak iç içe geçmiş elemanlara geçiş yapabiliriz.

Ayrıca, transition delay ekleyebiliriz. Ancak bu küçük bir sorun yaratıyor.

Bu sorunu şu şekilde çözebiliriz: :duration prop.

Fakat;

Süre = Geçiş Süresi + Geçiş Gecikme Süresi

Giriş ve çıkış zamanlarını farklı şekillerde ayırarak ekleyebiliriz:

<Transition:duration="{ enter: 2000, leave: 2000}">

<transition name="nested" duration-enter="1.5" duration-leave="2">

<transition name="nested" duration="{enter: 1.5, leave: 2 }">

<transition name="nested" duration="{enter: '1500ms', leave: '2s' }">

**2 Dynamic Transition Named**

Geçiş adını dinamik olarak tanımlayabiliriz. Bu işlem binding operation ile yapılır. Yani selectboxtan istedigimiz transition secip ona gore bir animasyon uygulayabiliriz.

**JavaScript Hooks**

<Transition> bileşenindeki olayları dinleyerek JavaScript ile geçiş sürecine bağlanabilirsiniz. JS Hook'larını transition CSS’leri ile kullanmak istiyorsanız, bu yöntemi kullanabilirsiniz.

JS Hook'larını transition CSS’leri ile kullanmak istemiyorsanız, ": css = false" prop'unu eklemelisiniz. Çünkü, Transition ilk başta css arar. Eğer css varsa, ilk önce o çalışır. Sonra JS Hooks calisir. Css olmadigini gorur. JS hooklari calistirir. Bu performans düşmesine neden olur., css kullanmak istemiyorsak performans düşüşü nedeniyle, ":css=false" prop eklemeliyiz.

**JavaScript Hooks with GSAP(Green Sock Animation Platform)**

Ayrıca, JavaScript Hooks ile animasyon olusturmak için Gsap kullanabilirsiniz.

GSAP nedir?

GSAP, geliştiricileri animasyon süper kahramanlarına dönüştüren sağlam bir JavaScript araç setidir.

**Reusable Transitions**

Transitionlar, Vue'nin component sistemi aracılığıyla yeniden kullanılabilir.

Yeniden kullanılabilir bir geçiş oluşturmak için <Transition> componentini saran ve slot içeriğini aktaran bir bileşen oluşturabiliriz.

MyTransition.vue: bizim transition componentimizdir.

Artık MyTransition, içe aktarılabilir ve kullanılabilir.

Ayni transition slot kismini degistirerek farkli yerlerde tekrardan kullanabilirz. Bu da dynamic bir yapi olusturur.

**Transition Between pages**

Routing işleminde Sayfadan Sayfaya geçerken de Transition kullanabiliriz.

**Transition Appear**

Sayfa ilk yuklendiginde transition islemi istersek eger appear propunu kullanmamiz lazim.

Appear islemine disaridan animasyon classi da ekleyebiliriz.

**Transition Between Elements**

Bir öğeyi v-if / v-show ile değiştirmenin yanı sıra, herhangi bir anda sadece bir öğenin gösterildiğinden emin olduğumuz sürece, v-if / v-else / v-else-if kullanarak iki öğe arasında geçiş yapabiliriz.

Geçiş modları olmadan ogeler arasında geçiş kullanırsak, bir sorun oluşur. Cunku biri leave islemi yaparken yeni eklenen enter islemine baslar. Birbirlerini beklemeleri gerekir.

**Transition Modes**

Leave ogesinin animasyonu bittikten sonra enter ogesinin animasyonunun başlamasını isteyebiliriz. Veya, enter öğesinin animasyonu bittikten sonra leave öğesinin animasyonunun başlamasını isteyebiliriz. Kısacası, her iki animasyonun da aynı anda olmasını istemeyebiliriz. Geçiş modlarını kullanabiliriz:

**Out-in:** ilk basta leave ogesinin transition islemi biter daha sonra enter elemaninin transition islemi yapilir.

**Out-in**: ilk basta enter ogesinin transition islemi biter daha sonra leave elemaninin transition islemi yapilir.

**Transition Between Components**

Transition componentler arasinda gecislerde de kullanilabilir. Ornegin elimizde 2 adet component olsun ve hangi butona basilirsa o componentin sayfada olmasini isteyebiliriz.

**Transition Group**

<TransitionGroup> bir v-for listesine bir öğe veya bileşen eklendiğinde, listeden çıkarıldığında veya içinde taşındığında animasyonları uygulamak için kullanilir.

Butun transition ozellikleri aynidir. Ornegin named islemi gibi.

**Transition Groups ve Transition farki**

<TransitionGroup> <Transition> ile aynı prop'ları, CSS geçiş sınıflarını ve JavaScript Hooklari dinleyicilerini aşağıdaki farklılıklarla destekler:

Varsayılan olarak, bir sarmalayıcı öğe(Root) oluşturmaz.

Transition modes kullanilmaz. Cunku birbirlerinin yerini alan ogeler kullanilmaz. Between elements te oldugu gibi.

İçerideki öğelerin her zaman benzersiz bir anahtar özelliğine sahip olması gerekir.

Transition grubun kapsayicina yani rootuna uygulanirken TransitionGroupta listedeki ayri ayri her bir elemana uygulanir.

**Enter/Leave/Move TransitionGroup with Named**

Transition da olduğu gibi ozel isimler ile de kullanabiliriz..

Ek olarak Move Transition, shuffle veya sort işlemleri için kullanılır.

Önceki demonun bazı belirgin kusurları var: bir öğe çıkarıldığında, çevresindeki öğeler sorunsuz bir şekilde hareket etmek yerine anında yerine "atlar". list-leave-active css bu sorunu çözmek için kullanılır.

**Js Hooks and Gsap in TransitionGroup**

Ayrıca, Geçiş Gruplarında Js Hooks ve Gsap'ı Geçiş Gruplarında kullandığımız gibi kullanabiliriz.