# Understanding Vue.js Transitions

alligator.io/vuejs/understanding-transitions

Despite having worked with Vue.js on countless projects, and having learned the ins-and-outs of various parts of it, for some reason understanding the transition system has always left me confused. Hopefully this article can help shed some light on what is actually a fairly intuitive system.

For the sake of sticking with the basics, we'll mostly be focusing on using CSS Transitions with Vue, but Vue also supports using CSS Animations or plain JavaScript transitions. We'll also ignore transition groups and event hooks for now.



## How It Works.

So basically, Vue works by adding and removing classes on components / elements during the transition process. There are six different classes for you to worry about, but thankfully, once you get the basic idea down, it all **starts to make sense...**

These classes are:

- `...-enter`
- `...-leave`
- `...-enter-active`
- `...-leave-active`
- `...-enter-to`
- `...-leave-to`

The `...` is just the name of your transition. I'm not exactly sure the reasons behind the decision, but Vue appends these class names to the name of your transition. So, if your transition name is "slither", then the classes are `slither-enter`, `slither-leave`, `slither-leave-active`, etc.

So what do we use these classes for?

## Handling Timing

Well, let's start with the easy ones. Yeah, those ones. `...-enter-active` & `...-leave-active`.

These honestly confused me the longest, but they're actually really simple to understand.

Use these classes to define which properties to transition and the duration and timing functions for them. That's it.

Before Vue 2.1.8, you had to use this class to define the ending state of the transition as well, which complicated things.

So, you would usually use it like this:

```
.slither-enter-active, .slither-leave-active {
  transition: transform 3s;
}
```

Not too shabby. Now we just need to define how it starts and ends.

## Handling Start & End Styles

`...-enter` and `...-leave` handle the start states for entering and leaving, respectively. `...-enter-to` and `...-leave-to` handle the end states for entering and leaving, respectively.

(Just to clarify, The `enter` transition happens when the component is being enabled or displayed. `leave` is the transition when the component is being disabled or removed.)

So, let's have the `slither` transition start with a transform that's way off to the left, then move to the normal position:

```
.slither-enter {
  transform: translateX(-100%);
}

.slither-enter-to {
  transform: translateX(0);
}
```

Now, usually, you want the leave transition to simply be the inverse of the enter transition. To do that, we simply add the opposite leave classes to the enter classes:

```
.slither-enter, .slither-leave-to {
  transform: translateX(-100%);
}

.slither-enter-to, .slither-leave {
  transform: translateX(0);
}
```

If you’re writing different transitions for entering and leaving, you can separate the classes however you’d like.

Putting it all together, you get this:

```
<template>
  <transition name="slither">
    <alligator v-if="alligatorDefinition"></alligator>
  </transition>
</template>

<script>
export default {
  data() {
    return {
      alligatorDefinition: null
    }
  },

  mounted() {
    alligatorDefinition = {}
  }
}
</script>

<style>
.slither-enter-active, .slither-leave-active {
  transition: transform 3s;
}

.slither-enter, .slither-leave-to {
  transform: translateX(-100%);
}

.slither-enter-to, .slither-leave {
  transform: translateX(0);
}
</style>
```

This will cause the alligator component to slide in from the left, and if you set `alligatorDefinition` to a falsey value, if will slide out to the left over three seconds.

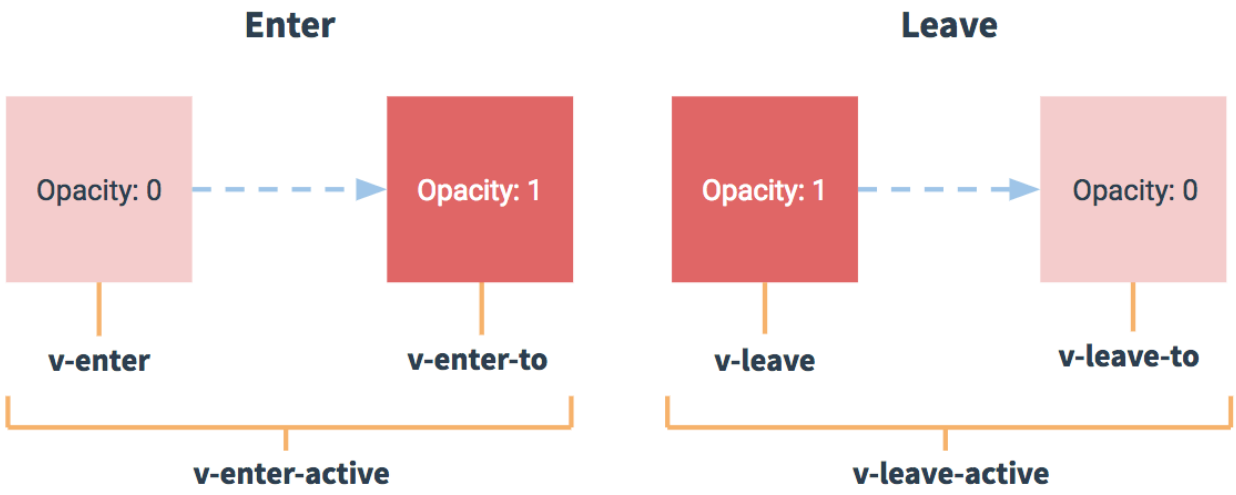
## Recap

The important things to remember are:

- `...-enter-active` & `...-leave-active` should only be used to define transition properties. Anything else will make things confusing.

- `...-enter` should contain start styles for when the component starts to appear.
- `...-enter-to` should contain end styles for when the component finishes appearing.
- `...-leave` is used in the same way as `...-enter`, but for when the component starts disappearing. It should usually contain the same properties as `...-enter-to`
- `...-leave-to` is used in the same way as `...-enter-to`, but for when the component finishes disappearing.

If that's still confusing, here's an image (courtesy of the Vue.js official docs) that might clear things up a bit.




There's plenty more to learn about transitions, it would be a good idea to take a look at the official docs for more information.