

# Vue.js Props

Topics : [Vue](#)

Written on [January 11, 2024](#)

In Vue.js, props are a way to pass data from a parent component to a child component. Props allow you to define custom attributes for a component and pass values into it when you use or instantiate the component. This facilitates communication between parent and child components. Here's a basic overview of using props in Vue.js:

## Passing Props:

### 1. Parent Component:

```
<template>
<div>
<child-component :message="parentMessage"></child-component>
</div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
  components: {
    'child-component': ChildComponent
  },
  data() {
    return {
      parentMessage: 'Message from parent'
    };
  }
};
</script>
```

In the example above, `:message="parentMessage"` binds the value of the `parentMessage` data property to the `message` prop of the `child-component`.

### 2. Child Component (ChildComponent.vue):

```
<template>
<div>
<p>{{ message }}</p>
```

```

</div>
</template>

<script>
export default {
  props: ['message']
};
</script>

```

The `props` option in the child component specifies an array of prop names that the child component is expecting to receive. In this case, it's expecting a prop named `message`.

## Prop Types and Validation:

You can also specify the type of the prop and add validation to ensure that the data passed as props meets certain criteria.

```

// Child Component
export default {
  props: {
    message: {
      type: String,
      required: true,
      default: 'Default message',
      validator: value => {
        // Custom validation logic
        return value.length < 50;
      }
    }
  }
};

```

In this example, the `message` prop is expected to be of type `String`, it is required, has a default value of `'Default message'`, and includes a custom validator function.

## Using Props in Templates:

You can use props directly in the template of the child component:

```

<template>
<div>
<p>{{ message }}</p>
</div>
</template>

```

Here, `{{ message }}` refers to the value of the `message` prop in the child component.

## Modifying Props in the Child Component:

Keep in mind that Vue.js follows a one-way data flow, which means that props should not be mutated directly in the child component. If you need to modify the prop value, consider using a data property in the child component that is initialized with the prop value.

```
<template>
<div>
<p>{{ modifiedMessage }}</p>
</div>
</template>
```

```
<script>
export default {
  props: ['message'],
  data() {
    return {
      modifiedMessage: this.message.toUpperCase()
    };
  }
};
</script>
```

In this example, `modifiedMessage` is a data property that is initialized with the uppercase version of the `message` prop.

© Copyright **Aryatechno**. All Rights Reserved. Written tutorials and materials by [Aryatechno](#)