Watchers are a fundamental feature of Vue.js that allow you to reactively perform custom logic when a specific property or data object changes. They provide an alternative approach to computed properties and methods by providing finer-grained control over how your component reacts to changes in its state.

```
watch: {
  myData(newVal, oldVal) {
    // Perform some logic here
  }
}
```

In this example, we're watching the myData property for any changes. Whenever myData is updated (i.e., its value changes), the function specified as the watcher's handler will be called with two arguments: newVal, which represents the new value of myData, and oldVal, which represents its previous value

By default, watchers are not executed until after the initial render cycle has completed. However, you can use the immediate option if you want your watcher to execute immediately upon creation:

```
watch: {
  myData(newVal, oldVal) {
    // Perform some logic here
  },

  immediate: true,
}
```

You can also specify whether your watcher should watch only for direct mutations on a given property (shallow) or recursively observe all nested properties within it (deep). The default behavior is shallow:

```
watch: {
  myData(newVal, oldval) { /* ... */ },

  // Watch all nested properties of "myNestedObject"
  myNestedObject:{
    handler(val){
      console.log('Changed Detected');
    },
    deep: true
  }
}
```