第一章：VUE基础

超全面　　极细致　　更深入

计算属性

模板内的表达式是非常便利的，但是它们实际上只用于简单的运算。

在模板中放入太多的逻辑会让模板过重且难以维护。

适用场景：

对于任何复杂逻辑，都应当使用**计算属性**的原因。

```
<div id="example">
  {{ message.split('').reverse().join('') }}
</div>
```

计算属性

可以像使用普通变量一样使用计算属性。

```
var vm = new Vue({
  el: '#example',
  data: {
    message: 'Hello'
  },
  computed: {
    // a computed getter
    reversedMessage: function () {
      // `this` points to the vm instance
      return this.message.split('').reverse().join('')
    }
  }
})
```

计算属性 vs Methods

我们可以将同一函数定义为一个 method 而不是一个计算属性。对于最终的结果，两种方式确实是相同的。然而，不同的是计算属性是基于它们的依赖进行缓存的。计算属性只有在它的相关依赖发生改变时才会重新求值。

相比而言，只要发生重新渲染，method 调用**总会**执行该函数。

计算属性 vs Watched 属性

如果一个逻辑涉及到多个变量，使用Watched需要监视多个变量，且代码冗余。

而计算属性会自动监视所有使用到的变量。并且当变量改变时自动进行重新计算。

计算 setter

在需要时VUE也可以提供一个 setter。

```
computed: {
  fullName: {
    // getter
    get: function () {
      return this.firstName + ' ' + this.lastName
    },
    // setter
    set: function (newValue) {
      var names = newValue.split(' ')
      this.firstName = names[0]
      this.lastName = names[names.length - 1]
    }
  }
}
```

Thank You