02

OPEN ORIENTED

凹 凸 实 验 室

浅谈vue数据绑定

penn

```
<h3>双向数据绑定demo</h3>
<div id="wrap">
  <input type="text" v-model='demo'>
  <h4 v-text='demo'></h4>
</div>
```

```
new Vue({
  el:'#wrap',
  data:{
    demo:'beidan'
  }
})
```

双向数据绑定原理

02

1. 脏值检查（angular.js）　轮询检测数据变动

- DOM事件，譬如用户输入文本，点击按钮等。(ng-click)
- XHR响应事件 ($http)
- 浏览器Location变更事件 ($location)
- Timer事件($timeout , $interval)
- 执行 $digest() 或 $apply()

2、Object.defineProperty劫持对象的get、set，从而实现对数据的监控。　(vue)

3、发布／订阅者模式实现数据与视图的自动同步。

Object.defineProperty优点

“脏值检测”——数据发生变更后，对于所有的数据和视图的绑定关系进行一次检测，识别是否有数据发生了改变，有变化进行处理，可能进一步引发其他数据的改变，所以这个过程可能会循环几次，一直到不再有数据变化发生后，将变更的数据发送到视图，更新页面展现

Object.defineProperty() 监控对数据的操作，可以自动触发数据同步。并且，由于是在不同的数据上触发同步，可以精确的将变更发送给绑定的视图，而不是对所有的数据都执行一次检测。

Object.defineProperty

兼容性： es5新增的，支持ES5的浏览器才可使用该方法 ie9+

```
var a= {}
Object.defineProperty(a,"b",{
    value: 123
})
console.log(a.b);//123
```

传人3个参数

第一个： a对象

第二个： a对象里面的b属性

第三个： 属性比较多，列举有用的 value，set，get，configurable

Object.defineProperty


```
var a = {};
Object.defineProperty(a, "b", {
  set: function (newValue) {
    console.log("我被赋值了！" + newValue);
  },
  get: function () {
    console.log("我被取值了！");
    return 2    }
})
a.b = 3; //我被赋值了！
console.log(a.b);    //我被取值了！          //打印 2
```

实现一个简单的数据绑定

1、实现一个数据监听器Observer，能够**对数据对象的所有属性进行监听**，如有变动可拿到最新值并通知更新视图数组【】

2、实现一个指令解析器Compile，对每个元素节点的指令进行扫描和解析，根据指令模板替换数据

3、实现一个更新视图数组【】，能够订阅并收到每个属性变动的通知，执行指令绑定的相应回调函数，更新视图

observer


```
var data = {name: 'beidan'};
observe(data);
data.name = 'test'; // 监听到值变化了  beidan 变成  test

function observe(data) {
  if (!data || typeof data !== 'object') {
    return;
  }
  // 取出所有属性遍历
  Object.keys(data).forEach(function(key) {
    defineReactive(data, key, data[key]);
  });
}
```

```
function defineReactive(data, key, val) {

 Object.defineProperty(data, key, {
   enumerable: true, // 可枚举
   configurable: false, // 不能再define
   get: function() {
     return val;
   },
   set: function(newVal) {
     console.log('监听到值变化了 ', val, ' 变成 ', newVal);
     val = newVal;
   }
 });
}
```

维护一个数组，数据变动触发notify，改变视图


```
function Dep() {
    this.subs = [];
}
```

```
Dep.prototype = {
    addSub: function (sub) {
        this.subs.push(sub);
    },
    notify: function (val) {
        this.subs.forEach(function (sub) {
            sub.update(val)
        });
    }
};
```

```
function defineReactive(data, key, val) {
  Object.defineProperty(data, key, {
    enumerable: true, // 可枚举
    configurable: false, // 不能再define
    set: function(newVal) {
     if (val === newVal) return;
     console.log('监听到值变化了 ', val, ' 变成 ', newVal);
     val = newVal;
     dep.notify(val); // 通知所有订阅者
  } });}
```

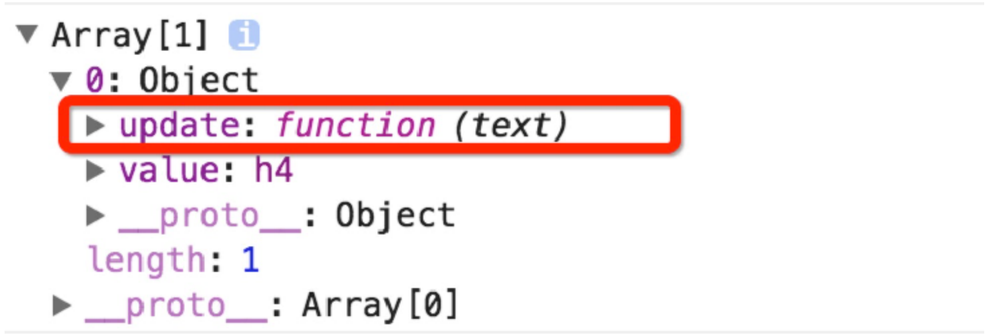
Compile


```
compile: function () {

    this.bindText();

    this.bindModel();

},
```




```
bindText: function () {
    var textDOMs = this.el.querySelectorAll('[v-text]'),
        bindText,_context = this;

    for (var i = 0; i < textDOMs.length; i++) {
        bindText = textDOMs[i].getAttribute('v-text');
        textDOMs[i].innerHTML = this.data[bindText];

        var val = textDOMs[i]

        var up = function (text) {
            val.innerText = text
        }

        _context.dep.addSub({
            value: textDOMs[i],
            update: up
        });
    }
},
```

THANKS

FOR YOUR WATCHING

02

OPEN ORIENTED

凹 凸 实 验 室