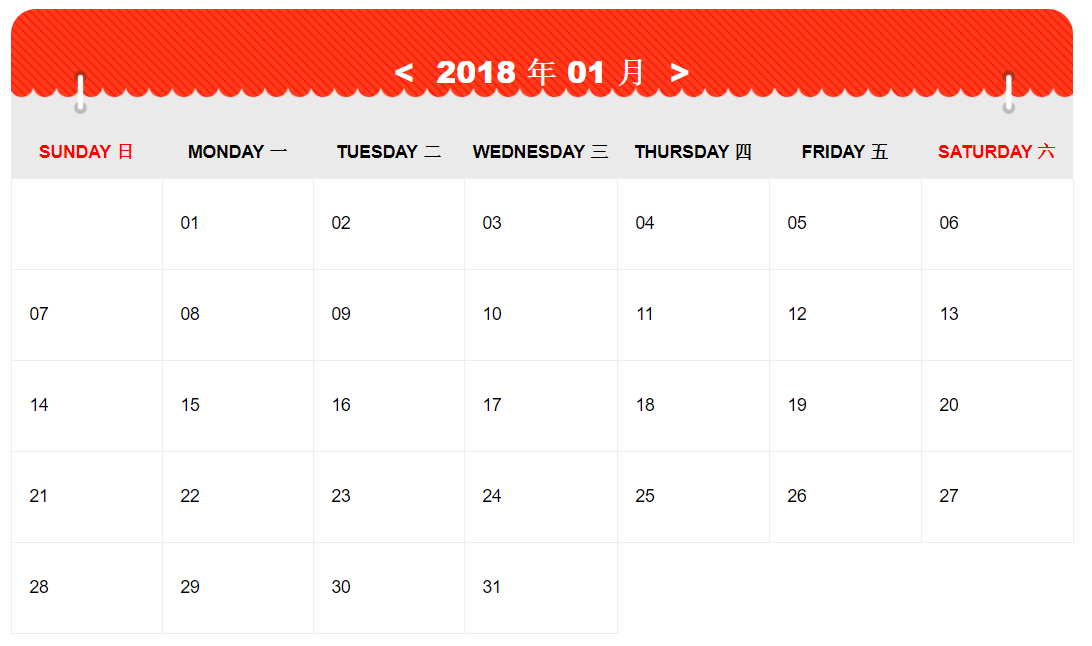
**Vue实现日历插件开发**

在前端开发中，日历插件是非常常用的一个组件，我们如何去实现这个组件呢？有的同学会感觉很头疼，不知道如何去做。我们可以百度一下，但是即使我们找到了一个比较合适的日历，在应用到我们的项目中，我们还是要花费很多的时间去理解其代码结构，这样我们有很多的时间浪费在这。即使最后能看懂别人写的代码，在后面再遇到一个这样的需求还是要使用日历插件时就不知道如何去开发了，那我们怎么办那？自己写又不知原理也不知道该从何下手去写这个日历插件，更别说使用Vue了。那接下来我就教大家使用Vue先来实现一个比较简单的日历插件吧。

## 界面分析

其实使用Vue写插件还是比较简单的，首先我们需要对日历插件进行分块处理，一点一点进行开发，日历有这样几个功能（参照图1）1）显示当前的月份；2）显示两个按钮可以点击上一个月和下一个月；3）星期 每列都有一个值对应着星期\*；4）当前月份有多少天根据当前日期是星期\*按顺序排列到页面上；

从上面的分析我们看出来是可以分为4部分，这四部分中，最好在页面上显示的是前3部分，这些值不需要计算就可以写出来了。那我们接下来先写一下这几部分的代码首先我们需要在代码中components文件夹下创建dmCalendar文件夹，并在这个文件夹下创建一个Vue的文件为：index.vue文件。



图（1）

## 结构代码实现

首先我们需要添加上他的结构，具体的结构代码如下：

**HTML：**

我们直接根据现有的结构去渲染出对应的头部文本显示，其中需要大家注意的是在html中我们使用了部分语法、变量例如：currentTitle、weekName、v-for等这些的使用可参照**Vue API进行学习**，这里就不过多讲解，

<template>

<div class='calendar'>

<div class="calendar-title-box">

<span class="prev-month " id="prevMonth"></span>

<span class="calendar-title" id="calendarTitle">{{currentTitle}}</span>

<span id="nextMonth" class="next-month "></span>

</div>

<div class="calendar-body-box">

<table id="calendarTable" class="calendar-table">

<thead>

<tr>

<th v-for=" item in weekName ">{{item}}</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

</div>

</div>

</template>

**JS：**

为方便我们开发，针对时间对象我们添加了一个addDatePro函数，此函数给原生Date对象增加了一个Format方法，调用方式如：new Date().Format("yyyyMMdd"); 。

<script>

export default {

data() {

return {

weekName: [

"SUNDAY 日",

"MONDAY 一",

"TUESDAY 二",

"WEDNESDAY 三",

"THURSDAY 四",

"FRIDAY 五",

"SATURDAY 六"

],

currentDay: "",

currentMonth: "",

currentTitle: "",

dayKey: []

};

},

methods: {

// index 0 代表当前月数据，-1 为当前月的上一个月，1为当前月下一个月的数据

init(index) {

this.currentDay = new Date(

this.currentDay.getFullYear(),

this.currentDay.getMonth() + index,

this.currentDay.getDate()

);

// 更新当前月数据和当前月显示title数据信息

this.currentMonth = this.currentDay.Format("yyyyMM");

this.currentTitle = this.currentDay.Format("yyyy 年 MM 月 ");

},

// 给Date对象增加Format方法

addDatePro() {

// 判断此函数是否存在，如果存在直接返回，不执行后续代码

if(Date.prototype.Format){return};

Date.prototype.Format = function(fmt) {

var o = {

"M+": this.getMonth() + 1, //月份

"d+": this.getDate(), //日

"H+": this.getHours(), //小时

"m+": this.getMinutes(), //分

"s+": this.getSeconds(), //秒

"q+": Math.floor((this.getMonth() + 3) / 3), //季度

S: this.getMilliseconds() //毫秒

};

if (/(y+)/.test(fmt))

fmt = fmt.replace(

RegExp.$1,

(this.getFullYear() + "").substr(4 - RegExp.$1.length)

);

for (var k in o)

if (new RegExp("(" + k + ")").test(fmt))

fmt = fmt.replace(

RegExp.$1,

RegExp.$1.length == 1

? o[k]

: ("00" + o[k]).substr(("" + o[k]).length)

);

return fmt;

};

/\*\*

\* 重写toJSON方法

\* returns {string}

\*/

Date.prototype.toJSON = function() {

var format = this.Format("yyyy-MM-dd HH:mm:ss");

return format.replace(" ", "T") + ".000Z";

};

}

},

mounted() {

this.currentDay = new Date();

this.addDatePro();

this.init(0);

}

};

</script>

**CSS：**

以下为less语法，在项目中已经配置自动编译插件，运行之后不会出现样式不能使用的问题，如对less语法不是很了解可**参照less API进行学习** 。

<style lang="less" >

/\*\*

\* 设置日历的大小

\*/

.calendar {

display: block;

margin: 0;

padding: 0;

margin-bottom: 20px;

/\*background: #cccccc;\*/

/\*\*

设置日历顶部盒子

\*/

.calendar-title-box {

position: relative;

width: 100%;

height: 125px;

line-height: 36px;

text-align: center;

border-radius: 20px 20px 0 0;

background: url("img/calendarTitle.png");

background-size: 100%;

color: #ffffff;

font-weight: 900;

font-size: 24px;

.calendar-title,

.next-month,

.prev-month {

line-height: 100px;

margin: 5px;

cursor: pointer;

}

.prev-month::before {

content: "<";

}

.next-month::after {

content: ">";

}

}

/\*

设置日历body 结构样式信息显示

\*/

.calendar-body-box {

background: #ffffff;

/\* 设置日历表格样式 \*/

.calendar-table {

width: 100%;

border-collapse: collapse;

text-align: center;

thead {

tr:first-child {

line-height: 30px;

height: 40px;

background: #eaeaea;

th:first-child,

th:last-child {

color: red;

}

}

}

}

}

}

</ style>



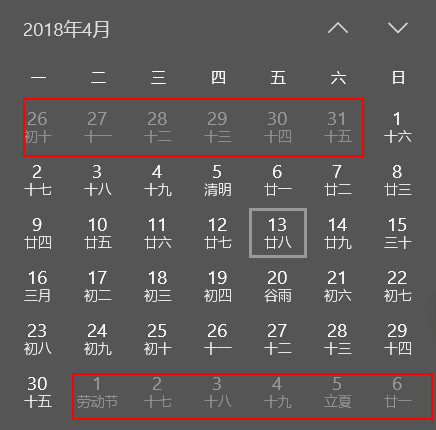
图（2）

## 日期分析和代码编写

添加完成后，最终的显示效果如图2 ，我们能看到头部需要显示的信息出现了。那接下来我们要做的是显示出tbody标签下面的子元素信息，这块改如何显示呢？我们可以先来分析一下。

1. 首先我们需要得到当前月中的日期个数，比如2018年1月有多少天；
2. 将我们的数据渲染到tbody标签下；

根据这两个点我们去考虑，首先我们需要得到当前月份的数据信息，比如说1月有31天，我们需要创建一个变量来保存这些数据变量叫“dayKey” 是一个数组类型的变量。在代码中先加上。



图（3）

接下来我们再看，我们假设已经得到了这个月的1-31日的数据，我们该如何渲染呢？我们都知道，日历显示是跟上个月的数据关联在一起显示的，例如我们电脑上的日历（图3）中红色框里面显示的日期，这些数据是上一个月或者下一个月的。也就是说我们每个月的1号是星期几是不确定的，我们在页面中渲染的话，对应的数据应该是当天是星期几就在星期几那列显示对应的日期。这样的话我们是不是需要知道没个日期的getDay()，这个信息。但是我们的daykey是个数组，那我们这里使用这样的一种计算方式，获取42天的数据，有同学会问，为什么是42天呢？从图3 中可以看出，这个日历中4月1日是周日，在我们日历显示中需要单独占一行，4月30日是周一也需要单独占一行，因此在一个日历中显示的数据最多的情况不会超过31天。加上两边多余的日期不会超过42天。因此我们在设计日历数据的时候，可以使用循环的方式得到对应的数据信息。具体代码如下：

getDateMap() {

let dayArray = [];

let firstDay = new Date(

this.currentDay.getFullYear(),

this.currentDay.getMonth(),

1

);

for (let i = 0; i < 42; i++) {

let time = new Date(

this.currentDay.getFullYear(),

this.currentDay.getMonth(),

i + 1 - firstDay.getDay()

).Format("yyyyMMdd");

dayArray.push(time);

}

this.dayKey = dayArray;

}

在methods方法中创建一个getDateMap函数，我们通过这个函数来获取日期数据信息，其中关键点在time = new Date(this.currentDay.getFullYear(),this.currentDay.getMonth(),i + 1 - firstDay.getDay()).Format("yyyyMMdd"); 大家知道new Date()中是可以传输参数信息的，第一个参数year，第二个参数是Month，第三个参数是Date。在这个调用中我们的年、月是当前日期的年、月信息，关键点在最后的date上，这个值是可以设置为负值，举个简单的例子：new Date(2018,0,1) 这段代码我们在浏览器控制台打印出来的话是Mon Jan 01 2018 00:00:00 GMT+0800 (中国标准时间) 也就是2018年1月1日，如果我们把date改成-1我们再来看结果是：Sat Dec 30 2017 00:00:00 GMT+0800 (中国标准时间)也就是2017年12月30日。这样我们可以推算出来，如果我们设定周日为当前月的第一个显示节点的话，对应每个日期的getDay() 是从0开始的，这样的话我们就可以使用这个特点来计算出我们要显示的这个月第一个节点位置是哪个日期。

既然我们得到dayKey，那就让我们显示出来吧，首先我们需要在tbody标签中添加如下代码：

<tbody>

<tr v-for=" x in dayKey.length/7">

<td v-for="y in dayKey.length/6" class=" ">

<div class="calendar-desc">

<div class="calendar-day">

{{ dayKey[(x-1)\*7+y-1]}}

</div>

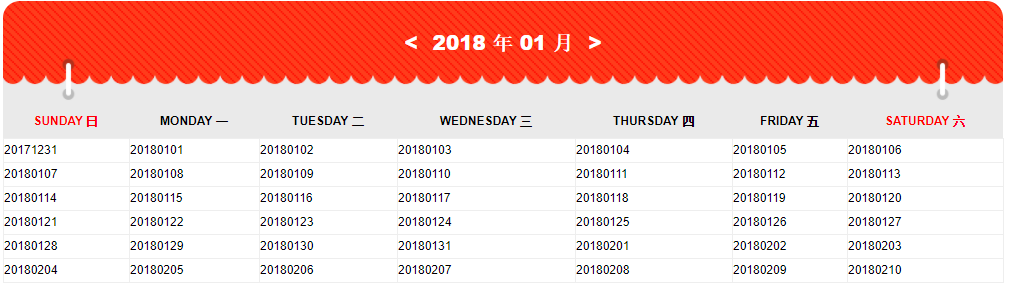
</div>

</td>

</tr>

</tbody>

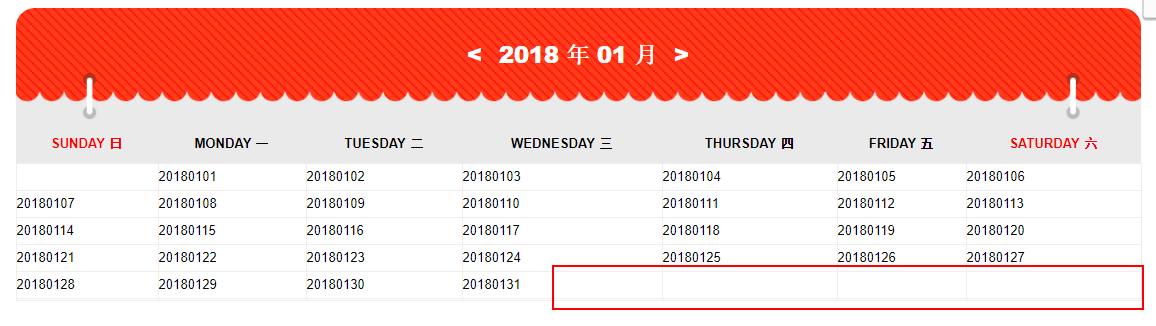
注意的是：<tr v-for=" x in dayKey.length/7"> 这行代码表示，每行显示7个总共显示的行数；<td v-for="y in dayKey.length/6" class=" "> 这行代码表示，每行显示的个数。{{ dayKey[(x-1)\*7+y-1]}} 这行代码表示，需要在每个td下显示的日期信息（如图4）。



图（4）

通过上图我们看到有很多数据不属于这个月，那我们可以判断其是否为当前月，然后进行显示。例如：<div class="calendar-desc" v-if="currentMonth== dayKey[(x-1)\*7+y-1].substr(0,6) “>

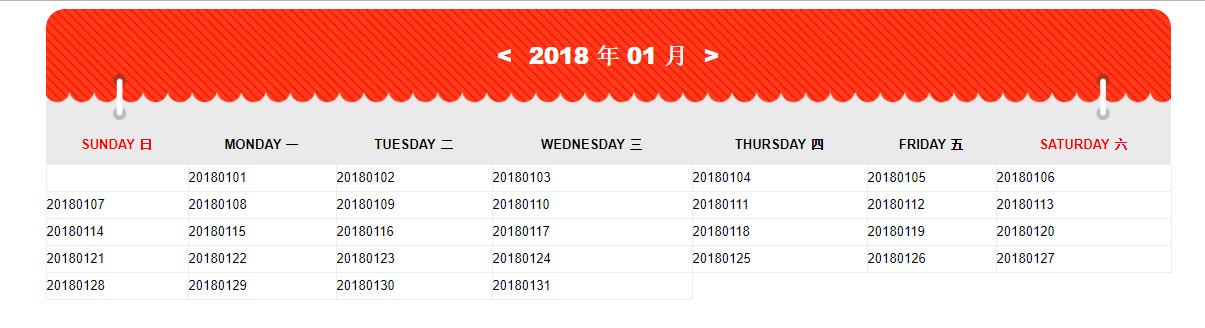
加上之后，我们页面上显示的效果如图（5）



图（5）

从图5中我们能看到，日历的右下角多了几个表格，那我们怎么处理呢？

我们需要给td增加这样的一段代码：v-if="currentMonth== dayKey[(x-1)\*7+y-1].substr(0,6) || ((x-1)\*7+y-1) < 7 "，这段代码的主要作用是，判断当前时间是否属于当前月，且日期小于7，这样我们前面显示的信息就不会减少，最终的效果如下图（6）：



图（6）

我们可以再优化一些，例如只显示对应的日期使用subStr对字符串进行截取，并增加上样式：/\* 设置日历表格样式 \*/

.calendar-table {

......

/\* 表格行高 \*/

tr {

line-height: 28px;

font-size: 14px;

td {

width: 170px;

height: 170px;

border: 1px solid #eeeeee;

.calendar-desc {

float: left;

width: 168px;

height: 170px;

.calendar-day {

width: 42px;

}

.calendar-list {

font-size: 12px;

}

.dm-card {

.calendar {

padding: 0;

}

}

}

}

}

}

显示效果如下图。



图7

## 4. 事件绑定

这样我们就把当前月的数据显示出来了，那如何切换月份呢，我们需要给上一个月（# prevMonth）和下一个月（# nextMonth）这两个标签添加两个点击事件：

<span class="prev-month " @click="clickPrev" id="prevMonth"></span>

<span id="nextMonth" class="next-month " @click="clickNext"></span>

方法示例如下：

clickPrev() {

// 点击上一个月的操作

this.init(-1);

},

clickNext() {

// 点击下一个月的操作

this.init(1);

},

我们看到，其实这个调用最主要的代码是init方法，那我们来看一下具体的呈现：

init(index) {

this.currentDay = new Date(

this.currentDay.getFullYear(),

this.currentDay.getMonth() + index,

this.currentDay.getDate()

);

// 更新当前月数据和当前月显示title数据信息

this.currentMonth = this.currentDay.Format("yyyyMM");

this.currentTitle = this.currentDay.Format("yyyy 年 MM 月 ");

},

这段代码中第一个数据改变的是currentDay，是我们当前时间的日期，通过传入的参数进行对Month的改变，从而实现月份的变化，同时我们可以根据currentDay来更新当前月份和当前月份的dayKey信息。从而实现月份的变化。

总体来说，日历显示到切换月份等功能我们已经实现了，那接下来我们要给这个日历添加一些私有的功能了，那如何添加呢？我们看下面的操作吧。

我们只需要在此页面中调用dmCard 组件，并进行渲染就可以，至于dmCard的数据，我们可以通过父节点传入到我们组件中，在组件中进行渲染即可。