1 直接人工截断 如果是响应式布局好像没问题呀 (可是项目的文本里面有标签是以v-html的方式插入的, 所以人工阶段直接失败)

var userAgent = navigator.userAgent; // 取得浏览器的userAgent字符串 console.log(userAgent)

var isFirefox = navigator.userAgent.indexOf('Firefox') > -1 // 是否是火狐 ，火狐内核Gecko

console.log('火狐'+ isFirefox) // 返回 true 则是 ，false 则不是

var isWebKit = navigator.userAgent.indexOf('WebKit') > -1 // 是否是WebKit 内核

console.log('谷歌内核'+ isWebKit) // 返回 true 则是，false 则不是

var isChrome = navigator.userAgent.indexOf('Chrome') > -1 // 是否是谷歌

console.log('谷歌'+ isChrome) // 返回 true 则是 ，false 则不是

var isOpera = navigator.userAgent.indexOf('Opera') > -1 // 是否是opera ， opera内核 Presto

console.log('Opera'+ isOpera) // 返回 true 则是 ，false 则不是

var isTrident = navigator.userAgent.indexOf('Trident') > -1 // 是否是IE内核

console.log('IE内核'+ isTrident) // 返回 true 则是 ，false 则不是

var isIE = userAgent.indexOf("compatible") > -1 && userAgent.indexOf("MSIE") > -1 && !isOpera

console.log('IE '+ isIE) //判断是否IE浏览器

// 因为ie10-ie11的版本问题，不再支持document.all判断，所以ie判断函数要重新写

var isIeL = !!window.ActiveXObject || "ActiveXObject" in window

console.log('IELLQ '+ isIeL) //判断是否IE浏览器

var isIE9 = navigator.userAgent.indexOf("MSIE 9.0")>0

console.log('IE999 '+ isIE9) //判断是否IE9 ;如果是其他IE版本，则 MSIE 7.0 MSIE 8.0

// 判断是否为移动端

var browser = {

versions: function() {

var u = navigator.userAgent;

return {

trident: u.indexOf('Trident') > -1, //IE内核

presto: u.indexOf('Presto') > -1, //opera内核

webKit: u.indexOf('AppleWebKit') > -1, //苹果、谷歌内核

gecko: u.indexOf('Firefox') > -1, //火狐内核Gecko

mobile: !!u.match(/AppleWebKit.\*Mobile.\*/), //是否为移动终端

ios: !!u.match(/\(i[^;]+;( U;)? CPU.+Mac OS X/), //ios

android: u.indexOf('Android') > -1 || u.indexOf('Linux') > -1, //android

iPhone: u.indexOf('iPhone') > -1 , //iPhone

iPad: u.indexOf('iPad') > -1, //iPad

webApp: u.indexOf('Safari') > -1 //Safari };

}()

}var isPhone = browser.versions.mobile || browser.versions.ios || browser.versions.android || browser.versions.iPhone || browser.versions.iPad

console.log('是否为移动端' +isPhone )

// JS判断浏览器是否是IE9以下，处理可能遇到的兼容性问题if(navigator.appName == "Microsoft Internet Explorer"&&parseInt(navigator.appVersion.split(";")[1].replace(/[ ]/g, "").replace("MSIE",""))<9){

console.log("您的浏览器版本过低，请使用IE9及以上版本");

}

2 使用js代码阶段 由于里面有标签代码 所以也会有问题 , 会失去标签代码

3 先假设都是中文字符, 然后js代码算出是否需要 css来

根据有用的代码来判断长度

/\*\* !

\* linelip.js V1.0.0 | MIT License | https://github.com/JasonSubMara/linelip

\* Copyright 2019, JasonSubMara <jie.yr127@gmail.com>

\* @description Multiline text overflow ellipsis

\* @param {Object} line.

\* @param {Object} element.

\*/

function linelip(lineNum, tagele) {

var eles = document.getElementsByClassName(tagele);

var lineNum = lineNum || 2;

for (var i = 0; i<eles.length; i++) {

var element = eles[i]

/\* 获取原文本 \*/

var text = element.textContent;

var textarr = text.split("");

/\* 临时存放数值 \*/

var num1 = 0;

for (var j = 0; j < textarr.length; j ++){

var regzn = new RegExp("[\\u4E00-\\u9FFF]+$","g");

if(!regzn.test(textarr[j])) {

/\* 非中文字符\*/

num1 = num1 + 1;

}else {

num1 = num1 + 2;

}

}

/\* 获取原文本内容长度 \*/

var totalTextLen = num1;

/\* 获取文本容器宽度 \*/

var baseWidth = window.getComputedStyle(element).width;

/\* 获取文本字体大小 \*/

var baseFontSize = window.getComputedStyle(element).fontSize;

/\* 获取单行文本宽度 >> 即文本容器宽度css属性值转换成数字 \*/

var lineWidth = +baseWidth.slice(0,-2);

/\* 获取单行文本内容个数 \*/

var strNum = Math.floor(lineWidth/ +baseFontSize.slice(0,-2));

/\* 定义最终输出内容 \*/

var content = "";

/\* 定义最终文本容器可容纳文本长度--->统一转成英文字符计算 \*/

var totalStrNum = (Math.floor(strNum \* lineNum))\*2;

var lastIndex = totalTextLen - totalStrNum;

/\* 定义最终文本截取位置 \*/

var lastplace = 11 - lastIndex;

if (totalTextLen > totalStrNum) {

content = text.slice(0, lastplace).concat("...");

}else {

content = text;

}

element.innerHTML = content;

}

}