0、软键盘顶起：

安卓window.resize ios监听onblur方法
改变输入框位置 可以调用scrollintoviewifneeded（）方法让输入框保持在可视区域，
或者监听document.documentelement.offsiteheight和dom的实际高度做比较 让它隐藏

1、click300ms延迟?

- fastclick可以解决在手机上点击事件的300ms延迟
- zepto的touch模块，tap事件也是为了解决在click的延迟问题
- 触摸事件的响应顺序为 touchstart --> touchmove --> touchend --> click,也可以通过绑定ontouchstart事件，加快对事件的响应，解决300ms延迟问题
- 若移动设备兼容性正常的话（IE/Firefox/Safari(IOS 9.3)及以上），只需加上一个meta标签

```
<meta name="viewport" content="width=device-width">
```

即把viewport设置成设备的实际像素，那么就不会有这300ms的延迟。

2、移动端样式兼容处理

- 设置meta标签viewport属性，使其无视设备的真实分辨率，直接通过dpi，在物理尺寸和浏览器之间重设分辨率，从而达到能有统一的分辨率的效果。并且禁止掉用户缩放

```
 <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no" />
```

- 使用rem进行屏幕适配，设置好root元素的font-size大小，然后在开发的时候，所有与像素有关的布局统一换成rem单位。针对不同的手机，使用媒体查询对root元素font-size进行调整

```
<script>
    window.onload = function () {
        //window的onresize,窗口或框架改变时调用的事件
        window.onresize = setRemFont;
        function setRemFont () {
            //获取html元素
            var domIe = document.documentElement;
            //获取设备宽度
            var width = domIe.clientWidth;
            //此375是你想固定设备开发页面的标准（我375），100是你想让1em等多少px（我100px）
            domIe.style.fontSize = width/375 * 100 + 'px';
        }
        setRemFont();
    }

</script>
```


3、阻止旋转屏幕时自动调整字体大小

移动端开发时，屏幕有竖屏和横屏模式，当屏幕进行旋转时，字体大小则有可能会发生变化，从而影响页面布局的整体样式，为避免此类情况发生，只需设置如下样式即可

```
1 * {
2   -webkit-text-size-adjust: none;
3 }
```

4、修改移动端难看的点击的高亮效果，iOS和安卓下都有效

```
1 * {
2   -webkit-tap-highlight-color: rgba(0,0,0,0);
3 }
```

不过这个方法在现在的安卓浏览器下，只能去掉那个橙色的背景色，点击产生的高亮边框还是没有去掉，有待解决！

一个CSS3的属性，加上后，所关联的元素的事件监听都会失效，等于让元素变得“看得见，点不着”。IE到11才开始支持，其他浏览器的当前版本基本都支持。详细介绍见这里：https://developer.mozilla.org/zh-CN/docs/Web/CSS/pointer-events

```
pointer-events: none;
```

5、iOS下取消input在输入的时候英文首字母的默认大写

```
<input type="text" autocapitalize="none">
```

6、禁止 iOS 识别长串数字为电话

```
<meta name="format-detection" content="telephone=no" />
```

7、禁止 iOS 弹出各种操作窗口

```
-webkit-touch-callout: none;
```

8、禁止ios和android用户选中文字

```
-webkit-user-select: none;
```

9、calc的兼容处理

CSS3中的calc变量在iOS6浏览器中必须加-webkit-前缀，目前的FF浏览器已经无需-moz-前缀。 Android浏览器目前仍然不支持calc，所以要在之前增加一个保守尺寸：

```
1 div {
2   width: 95%;
3   width: -webkit-calc(100% - 50px);
4   width: calc(100% - 50px);
5 }
```

10、fixed定位缺陷

iOS下fixed元素容易定位出错，软键盘弹出时，影响fixed元素定位，android下fixed表现要比iOS更好，软键盘弹出时，不会影响fixed元素定位 。iOS4下不支持position:fixed

解决方案： 可用iScroll插件解决这个问题

11、一些情况下对非可点击元素如(label,span)监听click事件，ios下不会触发

针对此种情况只需要对不触发click事件的那些元素添加一行css代码即可

```
cursor: pointer;
```

12、消除transition闪屏问题

```
/*设置内嵌的元素在 3D 空间如何呈现：保留 3D*/
-webkit-transform-style: preserve-3d;
/*(设置进行转换的元素的背面在面对用户时是否可见：隐藏)*/
-webkit-backface-visibility: hidden;
```

13、CSS动画页面闪白,动画卡顿

解决方法:

1.尽可能地使用合成属性transform和opacity来设计CSS3动画，不使用position的left和top来定位

2.开启硬件加速

```
-webkit-transform: translate3d(0, 0, 0);
   -moz-transform: translate3d(0, 0, 0);
    -ms-transform: translate3d(0, 0, 0);
        transform: translate3d(0, 0, 0);
```

14、iOS系统中文输入法输入英文时，字母之间可能会出现一个六分之一的空格

解决方法：通过正则去除

```
this.value = this.value.replace(/\u2006/g, '');
```

15、input的placeholder会出现文本位置偏上的情况

input 的placeholder会出现文本位置偏上的情况：PC端设置line-height等于height能够对齐，而移动端仍然是偏上，解决方案时是设置css

```
line-height: normal;
```

16、浮动子元素撑开父元素盒子高度

解决方法如下：

```
父元素设置为 overflow: hidden;
父元素设置为 display: inline-block; 等
```

这里两种方法都是通过设置css属性将浮动元素的父元素变成间接变成BFC元素，然后使得子元素高度可以撑开父元素。这里需要注意的时，最好使用方法1， 因为inline-block元素本身会自带一些宽高度撑开其本身。

17、往返缓存问题

点击浏览器的回退，有时候不会自动执行js，特别是在mobilesafari中。这与往返缓存(bfcache)有关系。 解决方法：

```
window.onunload = function () {};
```

18、overflow-x: auto在iOS有兼容问题

解决方法：

```
-webkit-overflow-scrolling: touch;
```

二、vue移动开发特有坑以及小技巧分享

1、iOS原始输入法问题

iOS原始输入法，中文输入时，无法触发keyup事件，且keyup.enter事件无论中英文，都无法触发

解决方法：

1. 改用input事件进行监听
2. 将keyup监听替换成值的watch
3. 让使用者安装三方输入法，比如搜狗输入法（不太现实）

2、input元素失焦问题

业务场景重现： 项目中需要写一个搜索组件，相关代码如下


```
<template>
  <div class="y-search" :style="styles" :clear="clear">
    <form action="#" onsubmit="return false;">
      <input type="search"
        class="y-search-input"
        ref="search"
        v-model='model'
        :placeholder="placeholder"
        @input="searchInputFn"
        @keyup.enter="searchEnterFn"
        @foucs="searchFocusFn"
        @blur="searchBlurFn"
      />
      <y-icons class="search-icon" name="search" size="14"></y-icons>
    </form>
    <div v-if="showClose" @click="closeFn">
      <y-icons class="close-icon" name='close' size='12'></y-icons>
    </div>
  </div>
</template>
```


其中我需要在enter的时候进行对应的搜索操作并实现失焦，解决方法其实很简单，在enter时进行DOM操作即可

```
1 searchEnterFn (e) {
2   document.getElementsByClassName('y-search-input')[0].blur()
3   // dosomething yourself
4 }
```

对了，这里还有一个坑，就是在移动端使用input类型为search的时候，必须使用form标签包裹起来，这样在移动端呼出键盘的enter才会是搜索按钮，否则只是默认的enter按钮。