30 ActionSheet弹出式菜单组件的设计与开发

更新时间：2019-08-05 11:10:53

> 生活永远不像我们想像的那样好，但也不会像我们想像的那样糟。
>
> ——莫泊桑

我们这一节来实现一个弹出式的菜单组件，这个组件也会经常用到，比如在微信中发起语音和视频通话选项的菜单：

这个组件在不使用的时候会隐藏在页面的底部，激活的时候从底部弹出来，并带有一个蒙版。移动端中因为屏幕尺寸比较小，所以通常会在页面上只显示一些重要操作，一些次要操作就会集合在一起，放在这种需要有一步操作才能出现的菜单里。下来我们讲一下这种菜单的需求。

弹出式菜单组件的需求

我们这一节准备做一个如下样式的弹出式菜单：

我们分两部分来说明这个组件的需求，一方面是菜单的静态效果，另一方面是菜单的进场和离场的效果。

先来说下弹出式菜单的静态效果：

1. 菜单分为上中下三部分，头部用来放菜单的说明。
2. 菜单头部里的文字在竖直方向上居中，最多两行，超长的话自动截断。
3. 菜单中间的部分是主要操作区，用来放菜单功能按钮，多个按钮用边框隔开。
4. 菜单的尾部和主操作区留有一小段距离，里面通常只放一个取消按钮。
5. 菜单显示的时候，固定在页面的底部，且在菜单和页面内容区之间用半透明蒙版隔开。
6. 菜单不显示的时候，隐藏在页面的最下面。

然后是菜单一些动态的样式：

1. 菜单显示的时候从底部向上滑出，底层的蒙版使用淡入的效果。
2. 菜单消失的时候，向下滑出屏幕，底层的蒙版使用淡出的效果。

这些就是对弹出式组件的要求，接下来我们进入设计和开发阶段。

弹出式菜单组件的设计与开发

一、文件的建立

这个组件需要在蒙版层以上，所以它的 HTML 结构应该和内容区并列，可以参考模态框的做法。我们先建立 HTML 文件 /demo/action-sheet.html：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="initial-scale=1, width=device-width, maximum-scale=1, user-scalable=no">
    <link rel="stylesheet" href="../src/tuitui-ui.css">
    <title>推推UI-ActionSheet弹出式菜单</title>
  </head>
  <body>
    <div class="tt-content">
      <h1 class="tt-panel-title">ActionSheet弹出式菜单</h1>
      <div class="tt-panel-body">
        <a class="tt-btn" id="js-show">显示弹出式菜单</a>
      </div>
    </div>
    <div class="tt-action-sheet">
      <div class="tt-mask"></div>
      <div class="tt-action-sheet-wrap">
        <div class="tt-action-sheet-header">
          <h1 class="tt-action-sheet-title">你需要做什么操作？ </h1>
        </div>
        <div class="tt-action-sheet-body">
          <a class="tt-action-sheet-menu">收藏</a>
          <a class="tt-action-sheet-menu">关注</a>
          <a class="tt-action-sheet-menu">分享给好友</a>
        </div>
        <div class="tt-action-sheet-footer">
          <a class="tt-action-sheet-menu" id="js-close">取消</a>
        </div>
      </div>
    </div>
  </body>
</html>
```

在这个页面的 HTML 结构里，我们在内容区添加了一个按钮，用来演示弹出菜单的效果。然后和内容区并列的就是弹出菜单的容器 tt-action-sheet。弹出菜单的蒙版还是使用公共的 tt-mask 来实现。而在菜单容器中，分为头部、内容区和尾部三部分，其中头部里放了一个标题，内容区放了三个元素作为菜单的按钮，最后在尾部里面放了一个元素当做取消按钮。

这样整个页面的结构就清楚了，下面建立 CSS文件 /src/action-sheet.css：

```
/*
 * @Author: Rosen
 * @Date: 2019-08-03 10:37:24
 * @Last Modified by: Rosen
 * @Last Modified time: 2019-08-03 20:10:41
 */

/* 弹出菜单的样式文件 */
```

最后，在主文件中引入这个样式文件，在 /src/tuitui-ui.css 中添加如下样式；

```
/* actionSheet可选菜单 */
@import './action-sheet.css';
```

这样，所有文件就建好了。

二、弹出式菜单的设计与开发

下来我们进行弹出式菜单的开发。这个弹出式菜单和 Toast 有些相似，它会有显示和隐藏两个状态，我们还是使用“show”这个 class 来控制组件是否显示。这个组件中，我们先来完成菜单容器的样式，在 /css/action-sheet.css 中添加如下样式：

```
/* 弹出菜单容器，默认隐藏在屏幕的下面 */
.tt-action-sheet > .tt-action-sheet-wrap{
    position: fixed;
    bottom: 0;
    left: 0;
    right: 0;
    max-width: 640px;
    margin: auto;
    background: #eee;
    transition: transform .3s ease;
    transform: translateY(100%);
    z-index: 301;
}
/* 菜单弹出的时候，改变容器位移 */
.tt-action-sheet.show .tt-action-sheet-wrap{
    transform: translateY(0);
}
```

在对容器的定位中，我们使用固定定位来安排菜单的位置，使用 bottom 值把菜单容器放在了页面的最下面，然后默认情况下又通过 transfrorm 属性把菜单向下移动了菜单高度，也就把菜单隐藏在了屏幕的下边缘。当容器显示的时候，再把位移值归零，这样容器就回到了页面中。这个过程我们还是用了“transition: transform .3s ease;”属性来添加菜单容器的入场和出场效果。

接下来要实现背景的控制，我们上一节在做 Toast 提示组件的时候，是给元素设置了入场和出场动画后把元素移除掉了。但是弹出菜单这个组件通常是藏在页面下方，不会用的时候再加载，所以要让它一直存在于 DOM 中。这样就会造成一个问题，后面的蒙版层我们可以用透明度 opacity 属性来实现淡入淡出效果。当淡出以后蒙版的透明度是 0，但这个元素还是遮盖着后面的内容区的，导致内容区的操作不能进行。遇到这种情况，就要介绍一下“pointer-events”这个属性了。

> **@ Tips:**
> **pointer-events** 这个属性用来指定是否为某个元素触发鼠标点击事件。这个属性主要用于 SVG，但是在 HTML 中也是可以用的，只不过可以取的值只有“auto”和“none”这两个，下来说下这两个取值的含义；
> **auto**，pointer-events 属性默认的取值就是 auto，使用这个属性值的情况下，HTML 元素就是正常的触发点击事件，通常只有为了覆盖不同取值的时候才会使用这个值。
> **none**，给元素用上这个属性值的话，这个元素就变成点不中的了，无论这个元素是什么样式，点击事件都会忽略它而去触发它底层元素的点击事件。

刚才所讲的 pointer-events 这个样式正好符合，我们要处理的蒙版的状态。当蒙版出现的时候我们把它的 pointer-events 属性值设置成“auto”来挡住内容区的元素；当蒙版透明的时候把 pointer-events 属性值设置成“none”，就能让蒙版不再阻挡内容区的操作，变成真正透明的元素了。

下面就是处理蒙版的代码：

```
/* 默认隐藏蒙版 */
.tt-action-sheet > .tt-mask{
    opacity: 0;
    /* 屏蔽元素的点击事件 */
    pointer-events: none;
    transition: opacity .3s ease;
}
/* 菜单弹出的时候显示蒙版 */
.tt-action-sheet.show > .tt-mask{
    opacity: 1;
    pointer-events: auto;
}
```

到这里，最复杂的部分就过去了，后面就是菜单容器里面几部分内容的布局了，这些都是我们之前做过的。

先来说菜单头部的标题，它最多有两行，超长时自动截断，这正好是多行截断的样式。然后要求文字水平竖直居中，这里面因为文本行数不固定，高度也不固定，所以可以使用弹性布局中的 align-items 属性来实现。最终代码如下：

```
/* 弹出菜单头部 */
.tt-action-sheet .tt-action-sheet-header{
    padding: 0 2rem;
    display: flex;
    align-items: center;
    text-align: center;
    height: 3rem;
    background: #fff;
}
/* 头部标题，用来描述菜单作用 */
.tt-action-sheet .tt-action-sheet-header > .tt-action-sheet-title{
    flex: 1;
    font-size: .7rem;
    line-height: 1rem;
    font-weight: normal;
    color: rgba(0, 0, 0, .3);
    overflow: hidden;
    display: -webkit-box;
    -webkit-line-clamp: 2;
    -webkit-box-orient: vertical;
}
```

最后，是菜单的内容区和菜单尾部，这两部分里面放的样式很相似，所以我们把这两部分一起实现。这两个部分中间用深色背景来隔开，因为前面已经设置过了容器的背景色，所以这里的值需要使用 margin 空出来一点距离就可以了。再就是紧邻的按钮之间要使用边框来隔开，这里我们使用兄弟选择器，就可以很容易的实现，边框我们采用了一个不太明显的半透明的颜色。最后的代码如下：

```
/* 中间主要内容区 */
.tt-action-sheet .tt-action-sheet-body{
    border-top: 1px solid rgba(0, 0, 0, .1);
    background: #fff;
}
/* 菜单尾部，通常用来放取消按钮 */
.tt-action-sheet .tt-action-sheet-footer{
    margin-top: .3rem;
    background: #fff;
}
/* 每个菜单项 */
.tt-action-sheet .tt-action-sheet-menu{
    display: block;
    height: 2.8rem;
    line-height: 2.8rem;
    font-size: .8rem;
    text-align: center;
}
/* 菜单项的边框控制 */
.tt-action-sheet .tt-action-sheet-menu + .tt-action-sheet-menu{
    display: block;
    border-top: 1px solid rgba(0, 0, 0, .1);
}
```

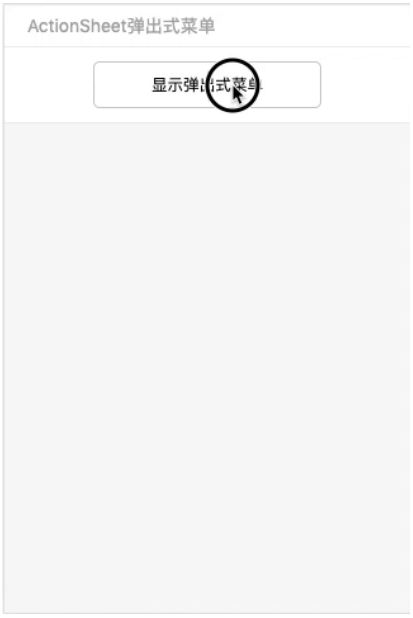
到这里我们整个弹出式菜单的样式就实现完了。最后还需要效果演示的处理，我们接下来把这部分也完成一下。

三、弹出式菜单的效果演示

由于这一节实现的组件也有入场和出场的效果，所以我们也借助下 JS 来提供一个效果演示的功能。在最开始的 HTML 里我们已经加了一个id为“js-show”的显示菜单的按钮，在菜单的尾部中，加入了一个 id 为“js-close”的取消按钮。我们就通过这两个按钮的点击事件来控制弹出式菜单的显示和隐藏。可以把下面这段 JS 代码添加到 HTML 文件中 body 元素的最后，来达到演示的效果。

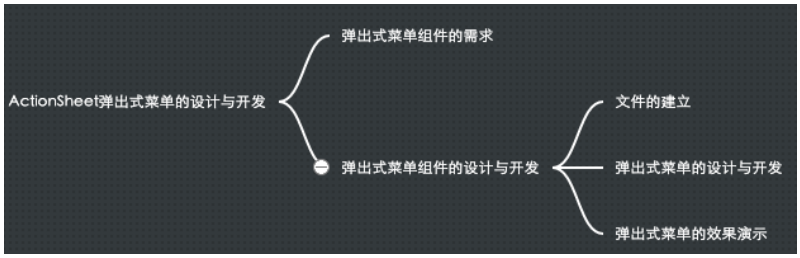
```
<script>
    window.onload = ()=>{
        // 弹出菜单
        document.querySelector('#js-show').onclick = (e) => {
            document.querySelector('.tt-action-sheet').classList.add('show');
        }
        // 收回菜单
        document.querySelector('#js-close').onclick = (e) => {
            document.querySelector('.tt-action-sheet').classList.remove('show');
        }
    };
</script>
```

这样这个菜单的效果也就完成了，最终的效果如下：



结语

我们这一节的内容到这里就结束了，这一节我们又用了一种新的方式来控制元素的显示和隐藏，这种方式在不需要JS 介入的情况下就能实现元素的入场和出场效果。这一节的内容结构如下：



我们这一节到这，同学们可以访问【Action　Sheet弹出式菜单在线预览】来查看这一节的演示效果，下一节将进行Article文本组件的开发。

}

29 Toast提示工具的样式与开发
31 Article文本样式的设计与开发