Gestures & Composers

Montage使用`Composer` API来支持常见的手势。 比如与设备相关的`DOM`事件click和touch。`Composer` API把这两个事件抽象为一个press事件，这样你就不需要设置多个事件监听。Montage已经支持以下事件：

- press / long press
- swipe
- key press
- drag

`PressComposer`

Press Composer处理press和long press手势。把鼠标click和touch抽象为Montage事件。已经支持的事件包括：

`pressStart`

当组件的`mousedown`或者`touchstart`触发的时候会被发送。

`press`

触发时间是在`pressStart`事件发送之后，鼠标键弹起(`mouseup` 事件) 或者手指离开屏幕（`touchend`事件）。 这个事件也会在`longPress`发送之后触发，但是是可以被撤销的。

`longPress`

当一个`press`手势超过`longPressTimeout`指定的时间会发送 `longPress`事件。如果要避免发送在`longPress`事件之后发送 `press`可以在`longPressHandler`中调用`cancelPress()`方法撤销`press`。

在 `longPressHandler` 中调用 `cancelPress()` 方法撤销 `press`。

`pressCancel`

当 `press` 被撤销之后会发送。两种方式可以撤销事件，一种是开发者调用 `cancelPress()` 方法，另外一种是其它的页面元素或者到事件焦点：

- 浏览器发送touch cancel 事件
- 用户通过在页面元素上按下鼠标之后，把鼠标从元素上移开，然后弹起鼠标键

`press` and `longPress` 手势实例

我们用一个例子来展示 `press` 事件，在例子中当 `longPress` 事件触发之后会改变页面元素的颜色和文字，并且显示一个JavaScript `alert` 弹出框。

组件设置

在Montage开发中如果要使用一个功能，你首先需要在Javascript文件中导入相关的模块。press和long press都在press-composer模块中：

```
var Montage = require("montage/core/core").Montage,
        Component = require("montage/ui/component").Component,
        PressComposer = require("montage/ui/composer/press-composer
").PressComposer;

You then have to create and add the `PressComposer`:

    exports.PressExample = Montage.create(Component, {
     didCreate: {
        value: function() {
            this._pressComposer = PressComposer.create();
            this.addComposer(this._pressComposer);
        }
    }
});
```

事件处理

导入 `PressComposer` 之后我们就可以添加事件监听器。在这个例子中同时添加 `press` 和 `longPress` 事件监听器:

```
prepareForActivationEvents: {
        value: function() {
            this._pressComposer.addEventListener("press", this, fal
se);
            this._pressComposer.addEventListener("longPress", this,
 false);
        }
    }
```

然后我们需要用Montage事件处理方式添加一个 `handle` 前缀的事件处理方法。

我们用两个方式来响应 `press` 事件， 一是修改元素的CSS来改变样式，另外一个是改变元素的 `innerHTML` :

```
handlePress: {
    value: function(event) {
        this.element.classList.toggle("press-active");
        if (this.element.classList.contains("press-active")) {
            this.element.innerHTML = "I’m active!";
        } else {
            this.element.innerHTML = "Now deactivated";
        }
    }
}
```

`longPress` 事件处理中弹出一个JavaScript `alert` 对话框，并且撤销 `press` 事件:

```
handleLongPress: {
    value: function(event) {
```

```
    value: function(event) {
        this._pressComposer.cancelPress();
        alert("Long press event fired.");
    }
}
```

在真实项目中你的案例可能是创建一个右键菜单来让用户使用。

实现功能

接下来你需要在模板中把HTML元素和JavaScript对象关联起来并定义CSS样式：

HTML:

```
<div data-montage-id="pressme" class="press-target">
    Click or long click me!
</div>
```

Script:

```
{
    "pressExample": {
        "prototype": "PressExample",
        "properties": {
            "element": {"#": "pressme"},
            "hasTemplate": false
        }
    }
}
```

如果你很快的点击元素 `handlePress` 被调用。如果你长按元素 `longPress` 被调用。

`SwipeComposer`

Montage当前只在手持设备上支持swipe手势，暂时还不支持桌面系

统。 `SwipeComposer` ，还在不断的完善，为Montage正式版本做准备。

`KeyComposer`

TBD

`TranslateComposer`

TBD