链滴

绘画板 04——增加操作框（矩阵变化）

作者：crick77

原文链接：https://ld246.com/article/1474513045156

来源网站：链滴

github地址： https://github.com/wangyuheng/painter

DEMO地址： http://painter.crick.wang/

针对元素进行操作时，如果采用更改边框颜色的形式，会和修改边框冲突。所以采用增加4个操作边的形式，既能表示元素被选中，又能在后续增加拖拽，变形等操作。

增加了一个HandleBorder类，并扩展其prototype属性对象，增加方法

1. createShade 用于创建4个边框
2. judgeShade 用于将4个小矩形，放至所选元素的4个边角
3. showShade 显示操作边框
4. 隐藏操作边框

加入了一些距离计算和矩阵，详细代码如下

```
(function() {

    var sbw = 6;
    var bw = {
        width: 1
    };

    var HandleBorder = function(svgDoc) {
        this.init(svgDoc);
    }

    HandleBorder.prototype = {
        constructor: HandleBorder,
        init: function(svgDoc) {
            this.currentSvgDoc = svgDoc;
            this.createShade();
            return this;
        },
    };

    HandleBorder.prototype.createShade = function() {
        var _this = this;

        _this.transformerGroup = _this.currentSvgDoc.group();

        _this.blockGroup = _this.transformerGroup.group();

        _this.rectLT = this.blockGroup.rect(sbw, sbw).stroke(bw).attr({
            '_operate-type': 'scale',
            '_direction': 'lt'
        });
        _this.rectLB = this.blockGroup.rect(sbw, sbw).stroke(bw).attr({
            '_operate-type': 'scale',
            '_direction': 'lb'
        });
        _this.rectRT = this.blockGroup.rect(sbw, sbw).stroke(bw).attr({
```

```
            '_operate-type': 'scale',
            '_direction': 'rt'
        });
        _this.rectRB = this.blockGroup.rect(sbw, sbw).stroke(bw).attr({
            '_operate-type': 'scale',
            '_direction': 'rb'
        });

    };

    HandleBorder.prototype.judgeShade = function(bbox, matrix) {
        var x1 = bbox.x;
        var y1 = bbox.y;
        var x2 = bbox.x2;
        var y2 = bbox.y2;


        this.rectLT.move(x1 - sbw, y1 - sbw);
        this.rectLB.move(x1 - sbw, y2);
        this.rectRT.move(x2, y1 - sbw);
        this.rectRB.move(x2, y2);

        this.blockGroup.matrix(matrix);

    };

    HandleBorder.prototype.showShade = function(svgEle) {
        if (!svgEle) {
            return;
        }
        this.currentElement = svgEle;
        this.transformerGroup.show();

        this.judgeShade(svgEle.bbox(), new SVG.Matrix(svgEle));
    };

    HandleBorder.prototype.hideShade = function() {
        this.transformerGroup.hide();
    };

    this.HandleBorder = HandleBorder;

})();
```

在Element的扩展方法中，利用HandleBorder替换边框颜色变更操作，并将new的HandleBorder对绑定到元素中

```
_ele.on("click", function() {
    if (SVG.isPicked()) {
        if (!_ele.attr("picked")) {
            _ele.attr("picked", true);
            console.log(_ele.handleBorder);
            _ele.handleBorder = _ele.handleBorder || new HandleBorder(svgDoc);
```

```
            _ele.handleBorder.showShade(_ele);
            pickedElementList.push(_ele);
        } else {
            _ele.attr("picked", null);
            _ele.handleBorder && _ele.handleBorder.hideShade(_ele);
            pickedElementList.remove(_ele);
        }
    }
});
```

扩展了一个pickedElementList数组，并在SVG扩展中提供获取方法

```
    getPickedElementList: function(){
        return pickedElementList;
    }
```

为了方便移除操作，强化了数组操作，提供了remove方法

```
Array.prototype.indexOf = function(val) {
    for (var i = 0; i < this.length; i++) {
        if (this[i] == val) return i;
    }
    return -1;
};
Array.prototype.remove = function(val) {
    var index = this.indexOf(val);
    if (index > -1) {
        this.splice(index, 1);
    }
};
```

选择元素操作已经完成，可以在此基础上绑定颜色操作，以及拖拽及变形操作。