|  |  |
| --- | --- |
| Canvas常用命令 | |
| ctx.moveTo( x, y ) 落笔  ctx.lineTo( x, y ) 起笔 | ctx.fillRect( x, y, w, h );  ctx.strokeRect( x, y, w, h ); 绘制矩形  ctx.clearRect( x, y, w, h ); |
| ctx.stroke(); 描 |
| ctx.fill(); 涂颜色 | ctx.arc(x, y, r, startA, endA, clockwise); 绘制圆 |
| ctx.closePath(); | ctx.font = “30px 黑体” 绘制文字  ctx.textAlign = left / center / right;  ctx.measureText( text content );  ctx.fillText( text content, x, y );  ctx.strokeText( text content, x, y ); |
| ctx.beginPath(); |
| ctx.setLineDash([5,10]);  ctx.getLineDash(); 虚线  ctx.lineDashOffset = -2; |
| ctx.lineWidth = 5;  ctx.fillStyle = ; 颜色  ctx.strokeStyle = ; | ctx.drawImage( img, sx, sy, sw, sh, x, y, w, h ); |
| ctx.translate(x,y);  ctx.rotate(); 坐标变换  ctx.scale(); |
| ctx.save(); 恢复画面状态  ctx.restore(); |
|  |

<script>

var canvas = document.getElementsByTagName("canvas")[0];

var ctx = canvas.getContext("2d");

</script>

1. 落笔、起笔画的只是路径，可以看成印好的凹槽字帖，然后stroke是用指定的颜色按照路径描一遍。
2. canvas里面所有涉及度数的函数参数都是**弧度**。
3. 清除整个画布：

① ctx.clearRect( 0, 0, cas.width, cas.height ); ② cas.width = cas.width;这种设置width属性的方法相当于让canvas重新渲染了一遍，所有的东西全部擦除。

1. 以下代码有什么不同？

<canvas width = "600" height="400"></canvas>

//这里的width / height是canvas的属性，可以改变canvas的真实像素。

canvas{

width: 600px;

height: 400px;

}

//这里的width / height是canvas的样式，不会改变真实像素，但会改变画布大小，相当于canvas是张照片，把它放大或缩小都不会改变它自身的分辨率。

1. 只要使用了变换，后续的绘制都会受到影响（即使用beginPath开启了新路径也是在新画布中画）
2. 在绘制圆圈移动动画的时候，为什么每次擦除之后还要再beginPath()？因为擦除只能擦除描上的颜色，擦除不了路径，所以后续的stroke()或fill()会重新把擦除过的描绘出来。
3. 在绘制矩形移动动画的时候，为什么每次擦除之后不用beginPath()？因为fillRect()内部已经beginPath()过了。
4. 第6行的img换成window也可以执行，但是把整个js代码都包在window.onload中就不能绘制图片，因为window.onload执行时机是整个dom树加载完成，而在第一种情况下，虽然dom已经加载完成，但是img还未创建出来，所以无法绘制。

<script>

var cas = document.getElementsByTagName(“canvas”)[0];

var ctx = cas.getContext(“2d”);

var img = new Image();

img.src = “2.jpg”;

img.onload = function () {

ctx.drawImage( img, 9, 9 );

};

</script>