

叶落为重生

每片落下的叶子都是为了下一次的涅槃...^ ^

关于base64编码的原理及实现

我们的图片大部分都是可以转换成base64编码的data: image。这个在将canvas保存为img的时候尤其有用。虽然除ie外，大部分现代浏览器都已经支持原生的基于base64的encode和decode，例如btoa和atob。（将canvas画布保存成img并强制改变mimetype进行下载，会在下一篇记录）

但是处于好奇心，还是驱使我去了解下base64编码的原理。以便也在不支持原生base64编码的ie下可以得以实现。

【Base64】

- base64的编码都是按字符串长度，以每3个8bit的字符为一组，
- 然后针对每组，首先获取每个字符的ASCII编码，
- 然后将ASCII编码转换成8bit的二进制，得到一组3*8=24bit的字节
- 然后再将这24bit划分为4个6bit的字节，并在每个6bit的字节前面都填两个高位0，得到4个8bit的字节
- 然后将这4个8bit的字节转换成10进制，对照Base64编码表（下表），得到对应编码后的字符。

（注：1. 要求被编码字符是8bit的，所以须在ASCII编码范围内，\u0000-\u00ff，中文就不行。

导航

[博客园](#)[首页](#)[联系](#)[订阅](#) XML[管理](#)

统计信息

随笔 - 78

文章 - 0

评论 - 897

Trackbacks - 0

NEWS

昵称: 岑安

园龄: 3年11个月

荣誉: 推荐博客

粉丝: 697

关注: 8

[+加关注](#)

搜索

常用链接

[我的随笔](#)

2. 如果被编码字符长度不是3的倍数的时候，则都用0代替，对应的输出字符为=)

Base64 编码表							
Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+

[我的评论](#)
[我的参与](#)
[最新评论](#)
[我的标签](#)

最新随笔

1. 【自己给自己题目做】之一：椭圆可点击区域
2. 我在想，技术博不能荒废
3. About me [my way]
4. 半年拾遗
5. context2D上的texture mapping
6. Animations In Canvas
7. 快3个月没写blog了
8. 【NodeCC】nodejs版本的脚本压缩和compo工具
9. 基于【双密度松弛算法】的二维流体粒子模拟
10. 为什么我推荐事件委托而不是批量绑定

我的标签

[js\(17\)](#)
[javascript\(9\)](#)
[3D\(7\)](#)
[焦点图\(5\)](#)
[轮播\(4\)](#)
[canvas\(4\)](#)
[淡入淡出\(3\)](#)
[幻灯片\(3\)](#)
[html\(3\)](#)
[css\(3\)](#)
[更多](#)

随笔档案

[2013年7月 \(2\)](#)
[2013年2月 \(2\)](#)
[2012年8月 \(3\)](#)

15	P	31	f	47	v	63	/
----	---	----	---	----	---	----	---

比如举下面2个例子：

a) 字符长度为能被3整除时：比如“Tom”：

	T	o	m	
ASCII:	84	111	109	
8bit字节:	01010100	01101111	01101101	
6bit字节:	010101	000110	111101	101101
十进制:	21	6	61	45
对应编码:	V	G	9	t

所以，btoa("Tom") = VG9t

b) 字符串长度不能被3整除时，比如“Lucy”：

	L	u	c	y	
ASCII:	76	117	99	121	
8bit字节:	01001100	01110101	01100011	01111001	
	00000000	00000000			
6bit字节:	010011	000111	010101	100011	
	011110	010000	000000	000000	
十进制:	19	7	21	35	30
	16	(异常)	(异常)		

- 2012年5月 (3)
- 2012年3月 (4)
- 2012年2月 (1)
- 2012年1月 (4)
- 2011年12月 (4)
- 2011年11月 (1)
- 2011年10月 (3)
- 2011年9月 (3)
- 2011年8月 (2)
- 2011年6月 (2)
- 2011年5月 (2)
- 2011年4月 (4)
- 2011年3月 (3)
- 2011年2月 (2)
- 2011年1月 (3)
- 2010年12月 (5)
- 2010年11月 (7)
- 2010年10月 (13)
- 2010年9月 (5)

积分与排名

积分 - 155526

排名 - 804

最新评论

1. Re:关于base64编码的原理及实现

@良村

这个可以处理中文：


--曾是土木人

2. Re:【备忘】bounce ease

想问一下您，要想达到您这种程度需要学习哪方面的数学知识，这些和动画有关的书算法那本书会讲到呢？

对应编码: T H V j e


Q = =



--PeakFish

由于Lucy只有4个字母，所以按3个一组的话，第二组还有两个空位，所以需要0来补齐。这里就需要注意，因为是需要补齐而出现的0，所以转化成十进制的时候就不能按常规用base64编码表来对应，所以不是a，可以理解成为一种特殊的“异常”，编码应该对应“=”。

有了上面的理论，那我们实现一个base64编码就容易了。



```
/**
 * base64 encoding & decoding
 * for fixing browsers which don't support Base64 | btoa |atob
 */

(function (win, undefined) {

    var Base64 = function () {

        var base64hash =
        'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/';

        // btoa method

        function _btoa (s) {

            if (/([\u0000-\u00ff]).test(s)) {
```

3. Re:canvas保存为data:image扩展功能的实现

@亦秋

多谢

--颜海镜

4. Re:canvas保存为data:image扩展功能的实现

@颜海镜html5 的 有 download属性var a = document.createElement('a');a.download = '我是文件名' + _suffix;a.href = _canvas.toDataURL();a.click();_suffix表示图片后缀_canva.....

--亦秋

5. Re:关于浮动与清除浮动

虽然这篇文章好几年的了，但还是忍不住说两句，关于博主说的“float的诡异之处”，其实不妥。真正的原因是：块级元素和float元素之间放置的时候是重合在一起放的。块级元素并不跟随float之后放置。博主文中的我是文案 这个代码,博主没有加背景色，和宽度高度，加上这些提示信息后，自然就知道“我是文案.....

--cymheart

阅读排行榜

1. 正因为我们是前端，所以代码更需要优雅(15847)
2. 关于base64编码的原理及实现(13966)
3. webkit-box & translate 的组合--流畅的滑动体验(11072)
4. 请允许我说：数学是如此美丽！(10590)
5. 关于浮动与清除浮动(7295)

评论排行榜

1. 请允许我说：数学是如此美丽！(117)
2. 我依旧喜欢折腾浏览器...(54)
3. 正因为我们是前端，所以代码更需要优雅(50)
4. 借着别人的创意，做自己的东西...(48)

```
        throw new Error('INVALID_CHARACTER_ERR');
    }
    var i = 0,
        prev,
        ascii,
        mod,
        result = [];

    while (i < s.length) {
        ascii = s.charCodeAt(i);
        mod = i % 3;

        switch(mod) {
            // 第一个6位只需要让8位二进制右移两位
            case 0:
                result.push(base64hash.charAt(ascii >>
2));

                break;
            //第二个6位 = 第一个8位的后两位 + 第二个8位的前4
            case 1:
                result.push(base64hash.charAt((prev & 3)
<< 4 | (ascii >> 4)));

                break;
            //第三个6位 = 第二个8位的后4位 + 第三个8位的前2位
            //第四个6位 = 第三个8位的后6位
            case 2:
                result.push(base64hash.charAt((prev &
```

5. 【前端应该知道的那些事儿】运动学基础(46)

推荐排行榜

1. 请允许我说：数学是如此美丽！(127)
2. 正因为我们是前端，所以代码更需要优雅(37)
3. 【前端应该知道的那些事儿】运动学基础(35)
4. 我依旧喜欢折腾浏览器...(31)
5. 借着别人的创意，做自己的东西...(24)

```
0x0f) << 2 | (ascii >> 6)));  
        result.push(base64hash.charAt(ascii &  
0x3f));  
  
        break;  
    }  
  
    prev = ascii;  
    i ++;  
}  
  
    // 循环结束后看mod, 为0 证明需补3个6位, 第一个为最后一个8  
    位的最后两位后面补4个0。另外两个6位对应的是异常的“=”;  
    // mod为1, 证明还需补两个6位, 一个是最后一个8位的后4位补两  
    个0, 另一个对应异常的“=”  
    if(mod == 0) {  
        result.push(base64hash.charAt((prev & 3) << 4));  
        result.push('==');  
    } else if (mod == 1) {  
        result.push(base64hash.charAt((prev & 0x0f) <<  
2));  
        result.push('=');  
    }  
  
    return result.join('');  
}  
  
// atob method  
// 逆转encode的思路即可
```

```
function _atob (s) {
    s = s.replace(/\s|=/g, '');
    var cur,
        prev,
        mod,
        i = 0,
        result = [];

    while (i < s.length) {
        cur = base64hash.indexOf(s.charAt(i));
        mod = i % 4;

        switch (mod) {
            case 0:
                //TODO
                break;
            case 1:
                result.push(String.fromCharCode(prev <<
2 | cur >> 4));
                break;
            case 2:
                result.push(String.fromCharCode((prev &
0x0f) << 4 | cur >> 2));
                break;
            case 3:
                result.push(String.fromCharCode((prev &
3) << 6 | cur));
                break;
        }
    }
}
```

```
        }

        prev = cur;
        i ++;
    }

    return result.join('');
}

return {
    btoa: _btoa,
    atob: _atob,
    encode: _btoa,
    decode: _atob
};
})();

if (!win.Base64) { win.Base64 = Base64 }
if (!win.btoa) { win.btoa = Base64.btoa }
if (!win.atob) { win.atob = Base64.atob }

})(window)
```



Base64 example

绿色通道:

好文要顶

关注我

收藏该文

与我联系



岑安

关注 - 8

粉丝 - 697

10

0

荣誉: 推荐博客

+加关注

(请您对文章做出评价)

« 上一篇: 追踪子弹-初中简单的物理和数学

» 下一篇: canvas保存为data:image扩展功能的实现

posted on 2012-01-14 11:47 岑安 阅读(13966) 评论(10) 编辑 收藏

Feedback

#1楼 2012-01-14 12:20 Homer_Simpson

暂时看不懂，但是感觉很NB。

支持(0) 反对(0)

#2楼 2012-01-14 19:47 深海沉

不错，喜欢

支持(0) 反对(0)

#3楼 2012-01-16 12:03 良村

好像无法处理中文

支持(0) 反对(0)

#4楼[楼主] 2012-01-16 13:09 岑安

@良村

(注: 1. 要求被编码字符是8bit的, 所以须在ASCII编码范围内, \u0000-\u00ff, 中文就不行。

支持(0) 反对(0)

#5楼 2012-03-28 20:36 Franky

把循环的step 改为3 .

在余数为1或2的情况,即最后产生的2-3个有效字符的逻辑,拿到循环体外处理. 然后每次push(char1,char2,char2,char4).

我想, 在处理 长字符串的时候, 可能会有2-3倍的性能提升吧.

支持(0) 反对(0)

#6楼[楼主] 2012-03-28 22:01 岑安

@Franky

嗯, 确实啊, 追求细节的教主, 赞 :)

支持(0) 反对(0)

#7楼 2012-03-28 22:10 Franky

@岑安
客气啦 :)

支持(0) 反对(0)

#8楼 2014-03-31 16:59 CN_SPIDER

不明觉厉

支持(0) 反对(0)

#9楼 2014-05-04 10:08 学习蚂蚁

这个方法根据图片地址把图片转换成base64decode的吗？

支持(0) 反对(0)

#10楼 2014-08-21 16:57 曾是土木人

@良村
这个可以处理中文：http://my.oschina.net/goal/blog/201032#OSC_h2_11

支持(0) 反对(0)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【免费课程】分享：从D2到D2(大话游戏开发实战技巧)
听云App——终结移动App性能黑洞



最新IT新闻:

- Twitch：在这里，任你插嘴
- 融资战记之四：大众点评张涛怎么花融资？
- 当李彦宏被问到是否担心微信时，他如是答道：当然担心...
- 搜易贷CEO何捷：搜狐为何要做互联网金融
- 德国法院命令Uber停止提供打车服务
- » 更多新闻...

最新知识库文章:

- 一些好的规则
- 千万别理程序员
- 送给程序员：关于性格内向者的10个误解
- 生于忧患而死于安乐：程序员如何走出自己的安逸环境
- 思考软件开发中的快与慢
- » 更多知识库文章...

Powered by:

[博客园](#)

Copyright © 岑安