小谈游戏渲染技术

QQ:951907111
Web:http://www.D5Power.com
任浩[D5Power]
2012-02-25

主题

- Flash游戏渲染方法分析
- Flash渲染效率提高
- Flash游戏素材保存与优化

Flash游戏渲染方法分析

· 伟大的时间轴

动画：gotoAndStop/gotoAndPlay

传说中的纯位图·

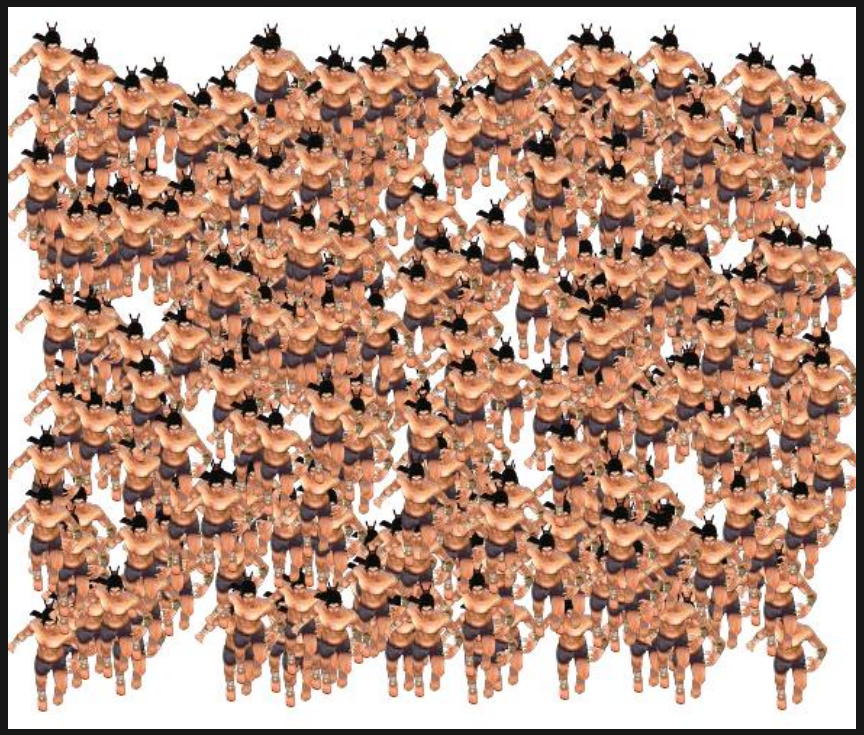
动画：copyPixels +(EnterFrame || Timer)

BitmapData

· 犀利的逐帧缓存

动画：bitmap.bitmapData +(EnterFrame || Timer)

Flash游戏渲染方法分析



所有动画逐帧存放于MC中
直接对MC进行实例化
var mc:MovieClip = new MC();

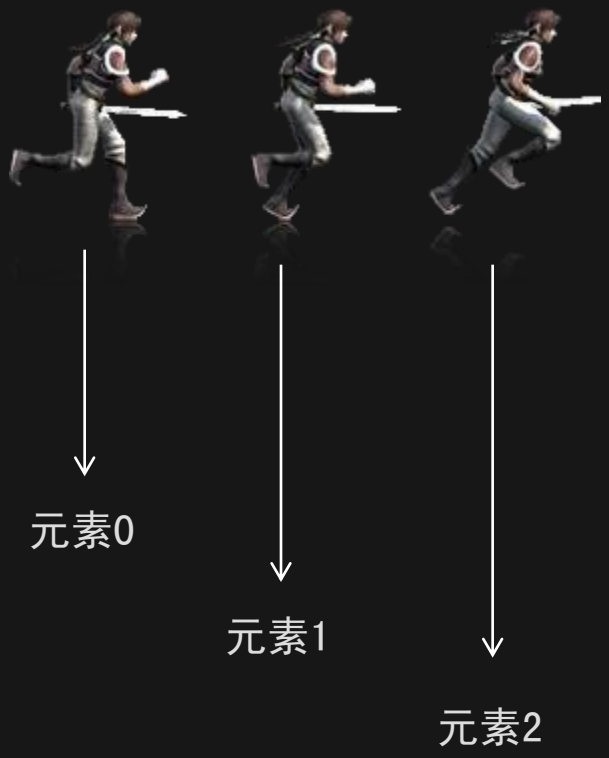
循环添加200个对象

cacheAsBitmap?
我们是动态内容-。-

渲染成绩：

353	Flash Player	renhoward	129.0	6	59.0 MB	Intel

Flash游戏渲染方法分析



将动画每帧以BitmapData的方式保存至数组

通过ENTER_FRAME或Timer定期进行切换

Bitmap.bitmapData = arr[frame];

同样200个

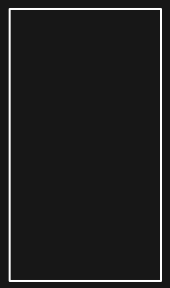
渲染成绩：

Flash Player	renhoward	73.3	6	44.2 MB	Intel

BitmapData

Flash游戏渲染方法分析

资源保存在一张BitmapData中
var shower:Bitmap;
shower.bitmapData.copyPixels
复制特定位置的图形，进行显示

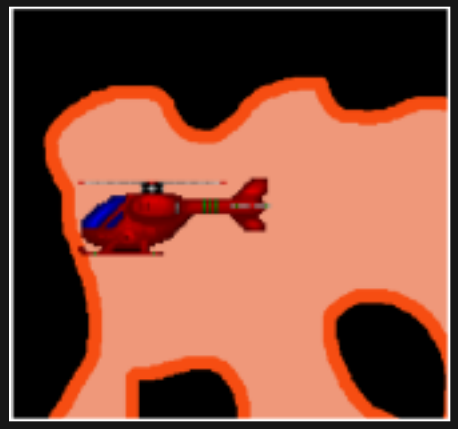


shower.bitmapData.copyPixels
复制对应位置的背景进行擦除

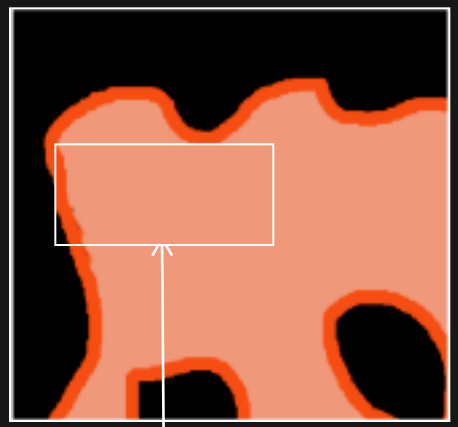
shower.bitmapData.copyPixels
复制新的位置的图形，进行显示
依此循环，形成动画

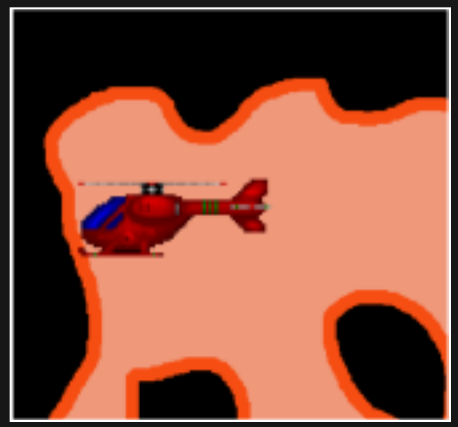
BitmapData

Flash游戏渲染方法分析



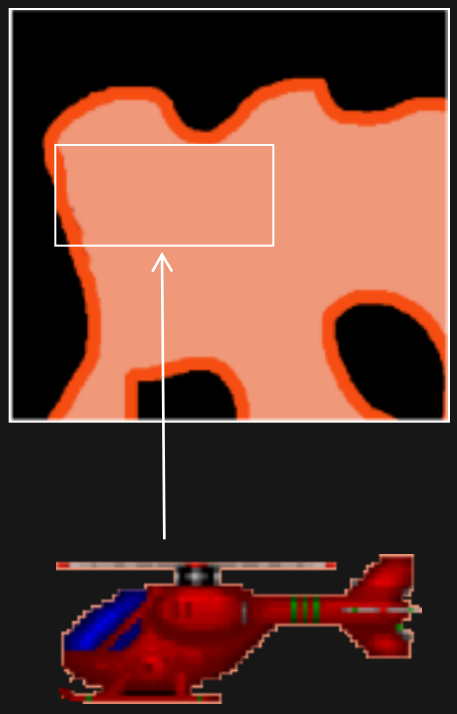
使用copyPixels渲染1000次所需要的时间
单位：毫秒





渲染耗时： 156
渲染耗时： 119
渲染耗时： 119
渲染耗时： 125
渲染耗时： 120
渲染耗时： 115
渲染耗时： 133
渲染耗时： 114
渲染耗时： 115
渲染耗时： 111
渲染耗时： 116

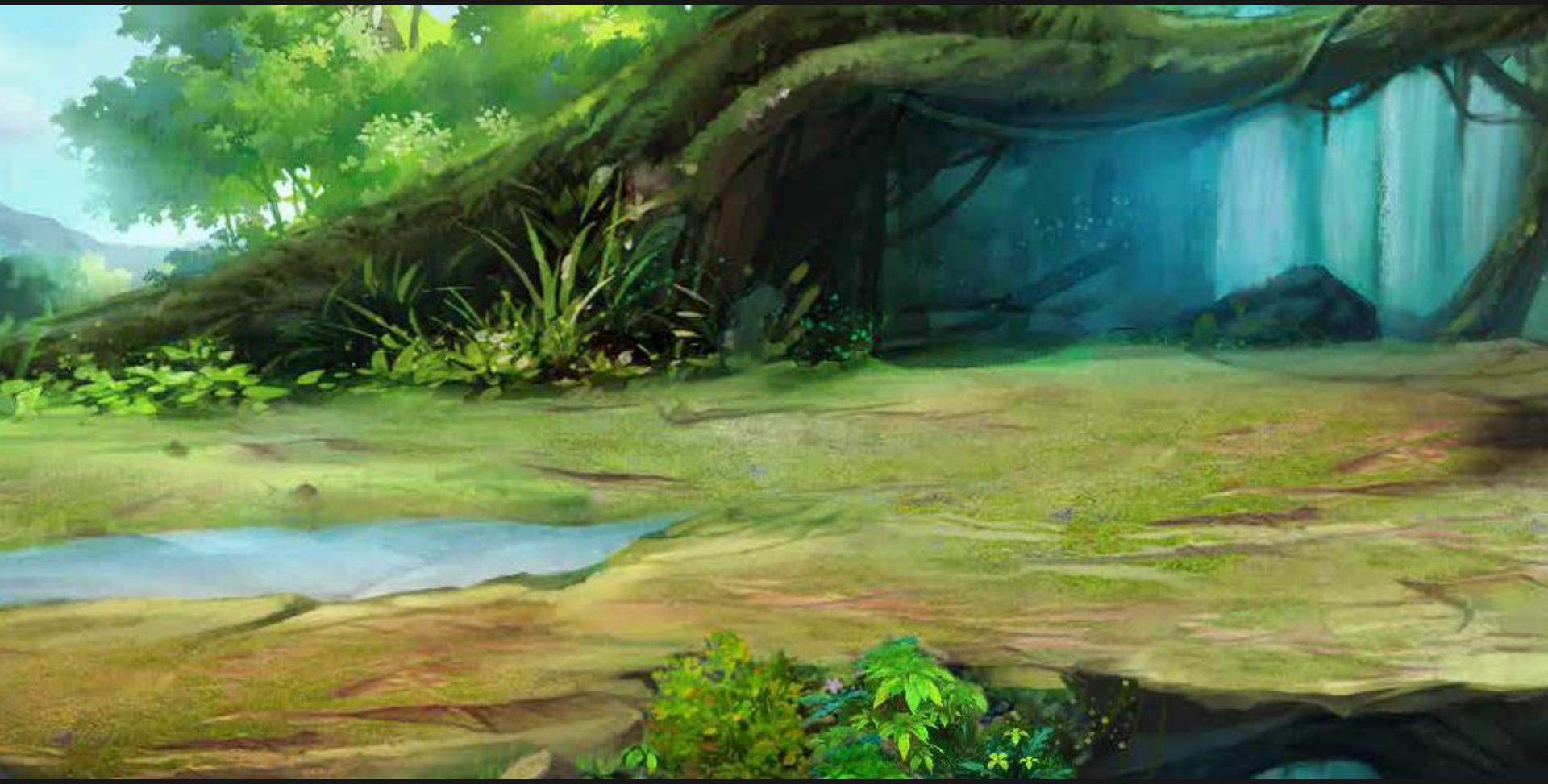
Flash游戏渲染方法分析



使用bitmap.bitmapData渲染1000次的时间
单位：毫秒

渲染耗时： 77
渲染耗时： 46
渲染耗时： 49
渲染耗时： 47
渲染耗时： 45
渲染耗时： 51
渲染耗时： 46
渲染耗时： 45
渲染耗时： 46
渲染耗时： 47
渲染耗时： 45

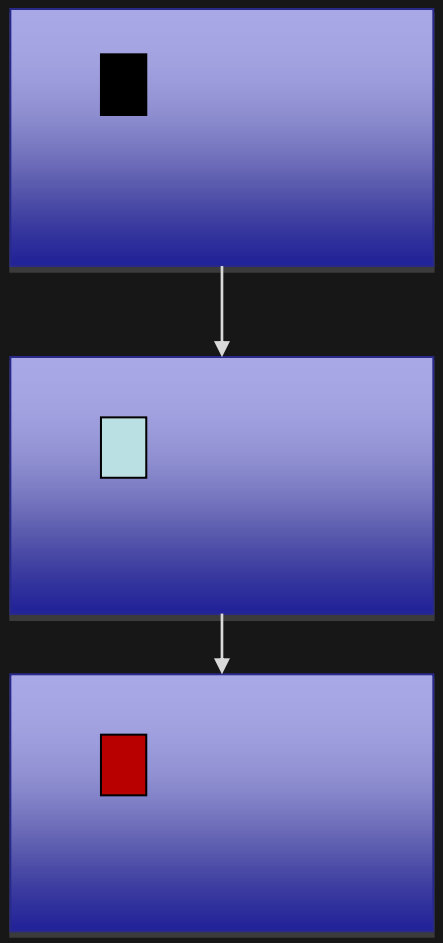
Flash游戏渲染优化



Flash游戏渲染优化

Flash游戏渲染优化

bitmapData.fillRect()



取背景对应区域的像素擦除上一帧

1000个对象，进行1001次绘制

1000个对象，进行2000次绘制

Flash游戏渲染优化

渲染耗时： 921
渲染耗时： 881
渲染耗时： 816
渲染耗时： 833
渲染耗时： 869
渲染耗时： 860
渲染耗时： 825
渲染耗时： 845
渲染耗时： 842
渲染耗时： 869
渲染耗时： 937

渲染耗时： 156
渲染耗时： 119
渲染耗时： 119
渲染耗时： 125
渲染耗时： 120
渲染耗时： 115
渲染耗时： 133
渲染耗时： 114
渲染耗时： 115
渲染耗时： 111
渲染耗时： 116

Flash游戏渲染优化

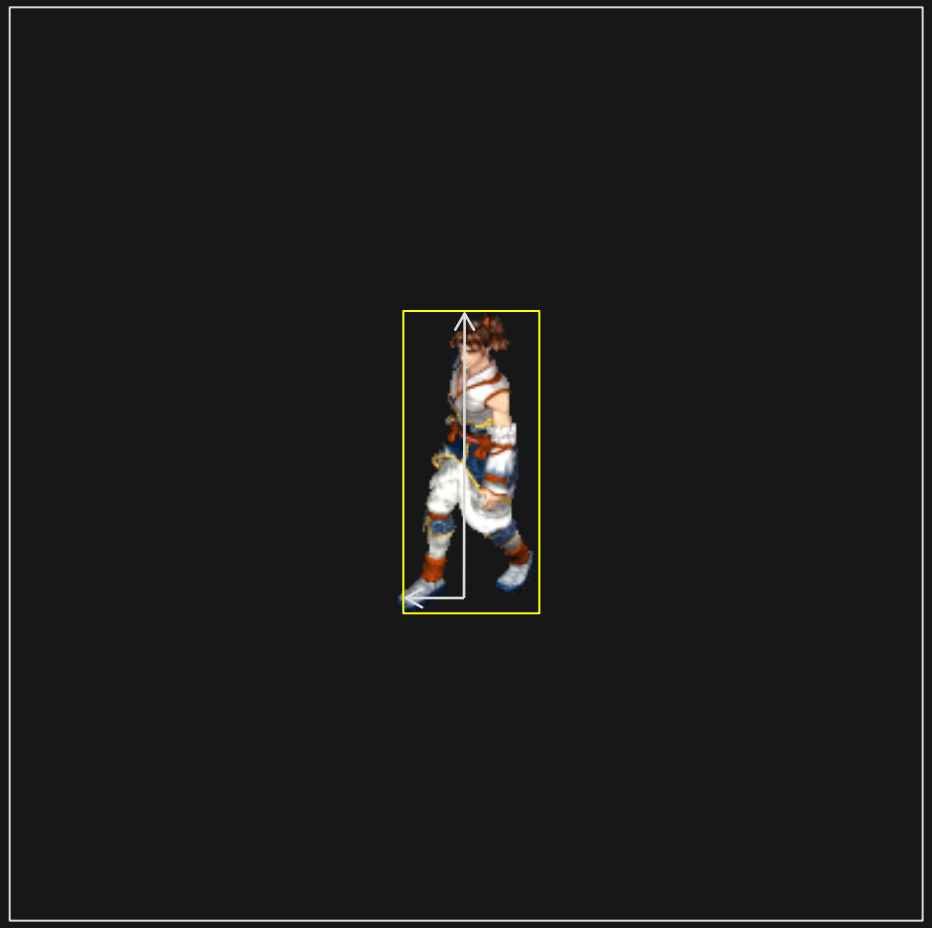
重绘面积越小 渲染速度越快

Flash游戏渲染优化

中间存在大量的透明像素
玩家看的见么？
反正计算机是“看”的见的

中间隔着太平洋啊太平洋

Flash游戏渲染优化



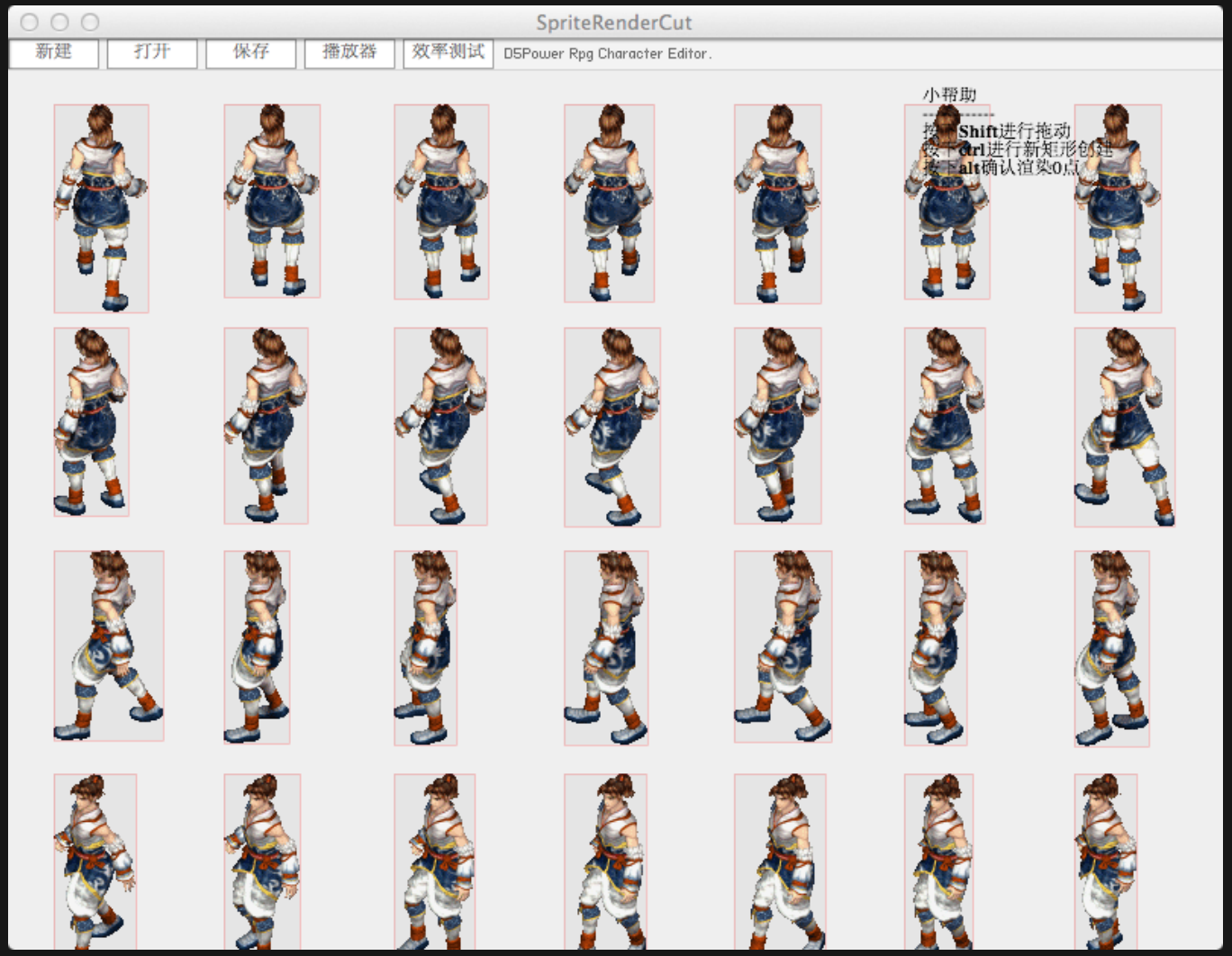
记录下重心点到最小有效矩形的偏移量，即可在渲染时将素材渲染到正确位置。

如何获取最小有效矩形？

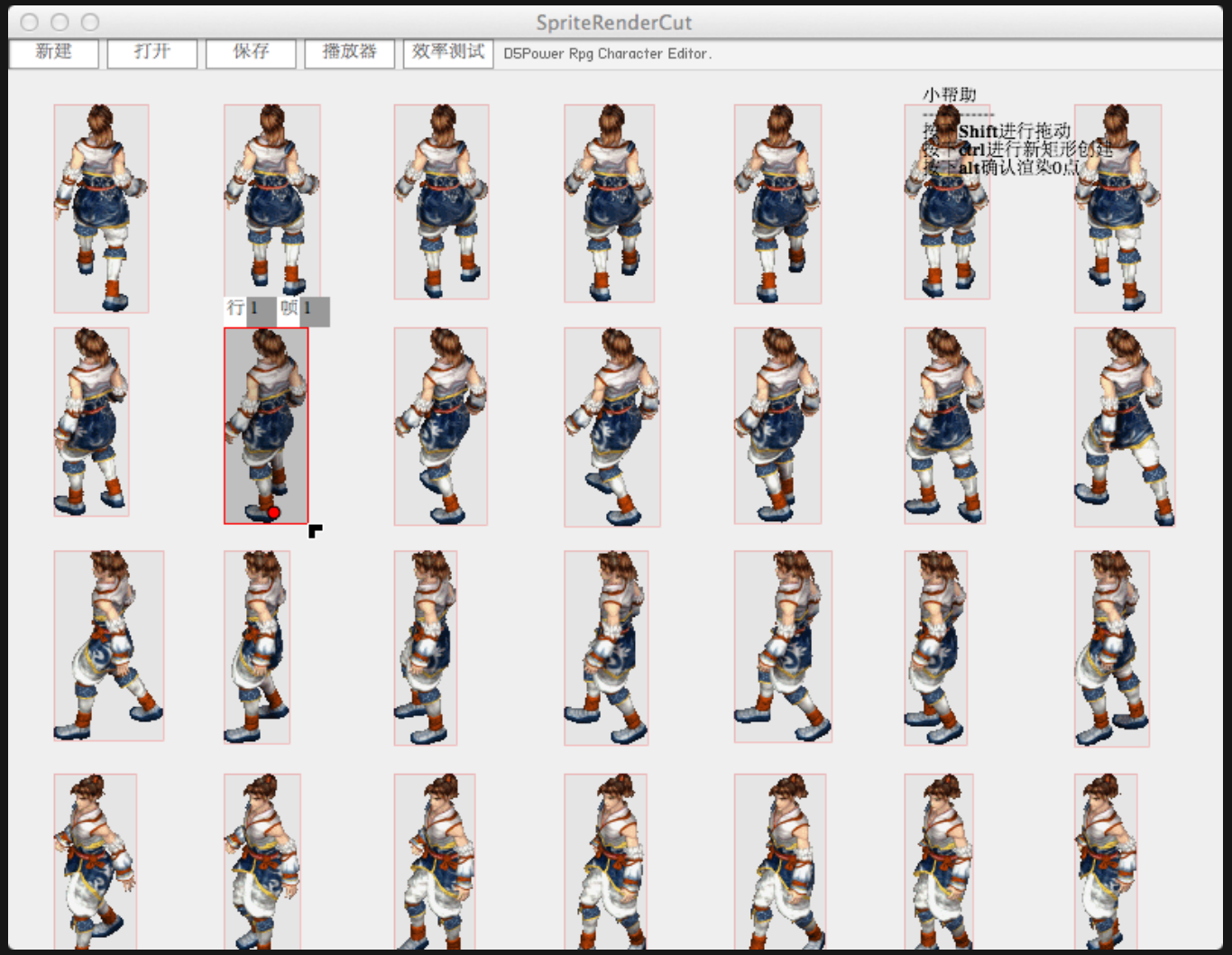
```
BitmapData.getColorBoundsRect(
            0xff000000,
            0x00000000,
            false
);

Value & 0xff000000 != 0x000000;
```

```
0xffd5d5d5          0x00d5d5d5
0xff000000          0xff000000
&---------          &---------
0x11000000          0x00000000
```

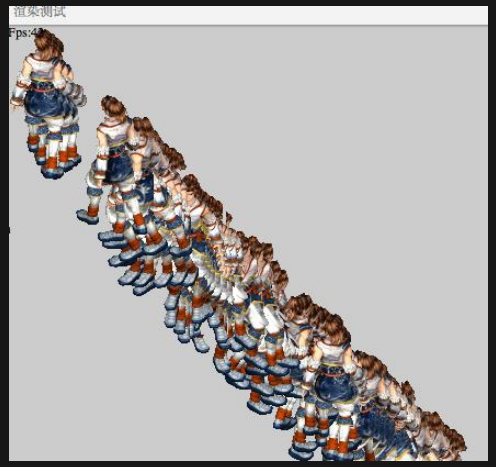
Flash游戏渲染优化



Flash游戏渲染优化



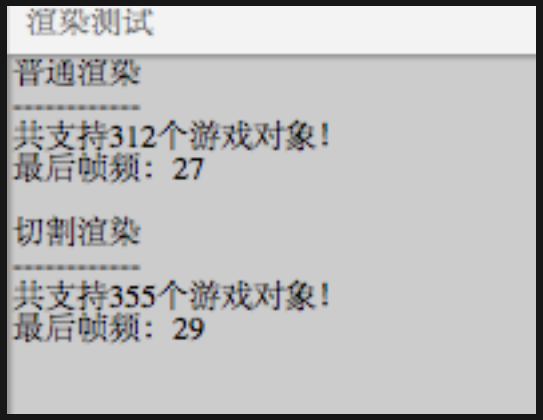
Flash游戏渲染优化

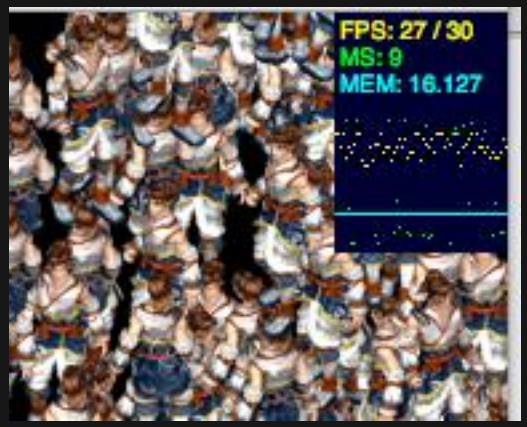


脑残测试方法：

不停向场景中增加游戏对象
不停更改游戏对象的位置
并监测瞬时帧频
当帧频下降至30fps时，计算游戏对象个数。

>_<





2000个游戏对象同时渲染

无裁剪的D5Rpg2.1则只能维持在400-500个左右

Flash游戏渲染优化

- var i:uint=0,j=arr.length;i<j;i++
- var i:uint=0;i<arr.length;i++
- 尽量减少重复的计算

- 分段计算以减轻某一时刻的大规模计算带来的压力。

- 尽量减小重绘的面积

- 独孤九剑：CPU换内存 OR 内存换CPU

- 适当关闭鼠标事件mouseEnabled，mouseChildren等，减轻事件监测的压力。优化鼠标碰撞监测（四叉树等优化算法）

Flash游戏素材存储

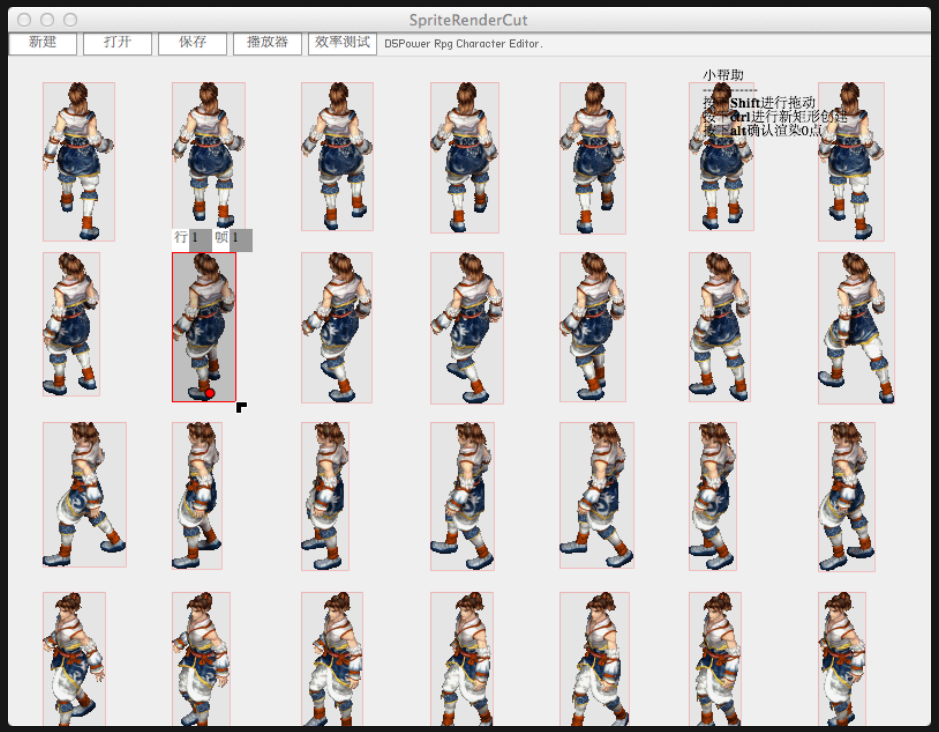
- 将素材合并到一张图片上，代替多次加载

- 将素材逐帧导入时间轴，导出为swf文件，利用Flash本身对图片的压缩，大幅降低文件体积。

```
Bitmapdata.draw();
nextFrame();
```

```
frameList.push(Bitmapdata);
```

Flash游戏素材存储



PNG原图数据
XML记录每个帧单元的最小矩形，偏移坐标等等

为什么我们用XML？
因为除了AS3能读懂外，我们自己也能读懂。

问题：XML需要解析，特别是在移动设备上，据说速度比较慢。

Flash游戏素材存储

- FP10的时候，我们有了ByteArray
- XML作为开发存档。最终产品文件使用二进制数据来保存。
- 类似联机游戏中的数据通讯协议来构建自己的文件格式。

Flash游戏素材存储

7字节 D5Power	N字节 配置区	7字节 D5Power	N字节 原图素材

1字节 帧数	2字节 X	2字节 Y	2字节 宽	2字节 高	循环	1字节 结束符0

也可以用这种方法把多个文件连接在一起，减少文件加载次数

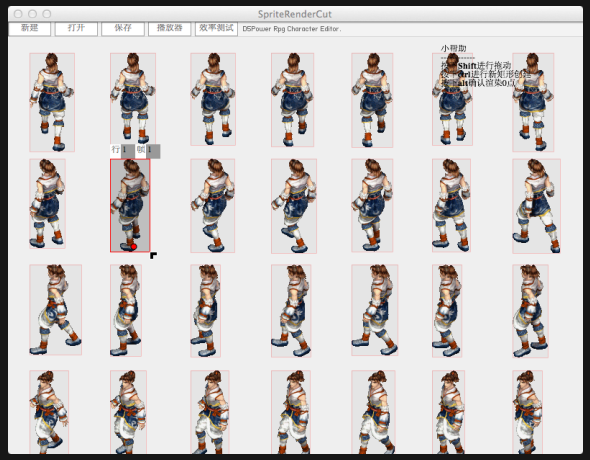
具备简单加密的“副作用”，由于破坏了原有的文件结构，因此不知道附加字节顺序无法进行解析，文件也无法直接浏览

解析速度快，带来的代价就是……这玩意谁看的懂啊 >_<

生产环境依旧保存XML，最终形成产品的时候再进行打包

地图配置文件同样适用：）

刚才提到的小工具？



D5Power角色素材编辑器

功能：最小渲染矩形识别，格式化文件，播放器，测试

配合D5 BitmapMacker单帧文件拼合器，效果更佳-。-

源码近期发布至天地会社区，敬请关注

谢谢！