第七章 二维游戏动画合成

上节回顾

- Chapter 7
  - 计算机动画概述
  - 常见计算机动画技术
  - Cocos2d-x中的动作类

本节内容

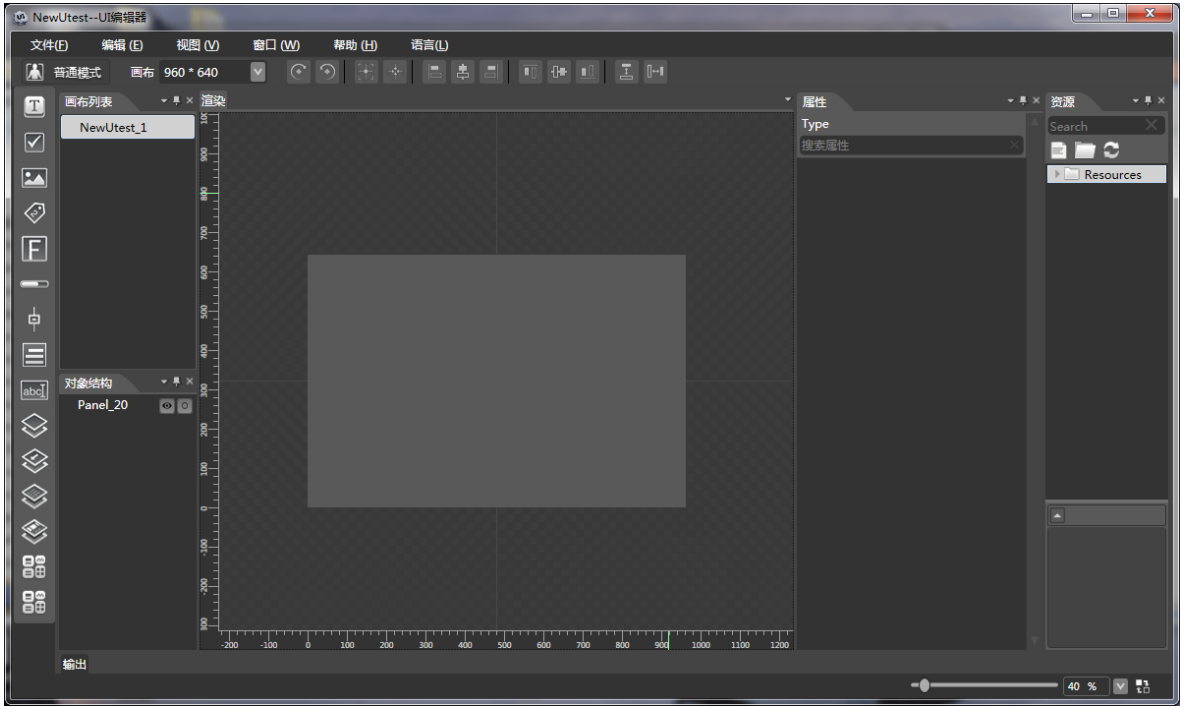
- Chapter 7
  - Cocos2d-x动画编辑器
    - 序列帧动画
    - 骨骼动画
  - 提高计算机动画效果的基本手法
  - Cocos2d-x中与动画相关的类

Cocos2d-x动画编辑器

- CocosStudio
  - 动画编辑器
    - 角色动画、特效动画、场景动画资源
    - 关键帧动画、序列帧动画、骨骼动画方式
  - UI编辑器
  - 数据编辑器
  - 场景编辑器

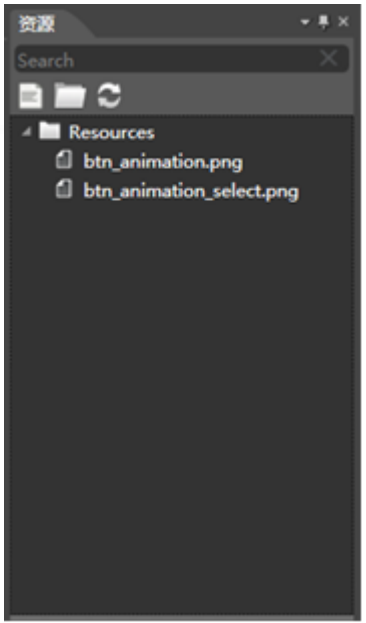
Cocos2d-x动画编辑器

- 1.创建项目
  - Animation Editor->文件->新建，创建新项目



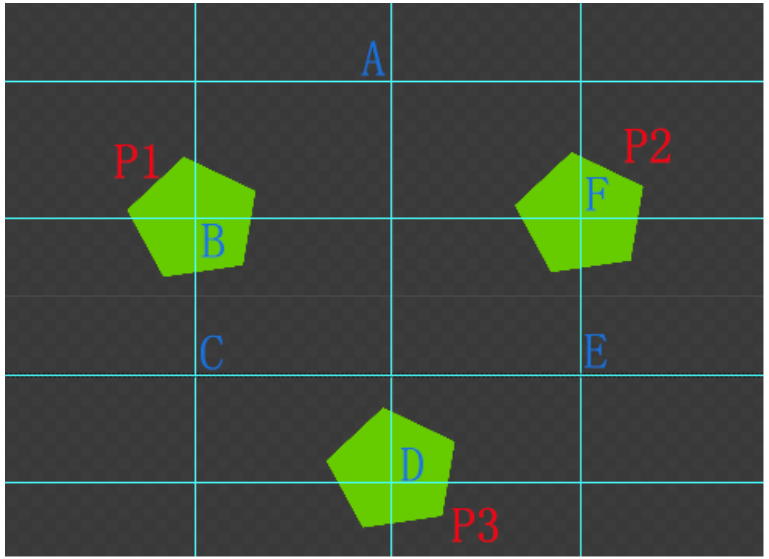
Cocos2d-x动画编辑器

- 2.导入资源
  - 资源面板导入图片button.png



Cocos2d-x动画编辑器

- 3.编辑动画
  - 在时间轴上在15帧和30帧处添加关键帧

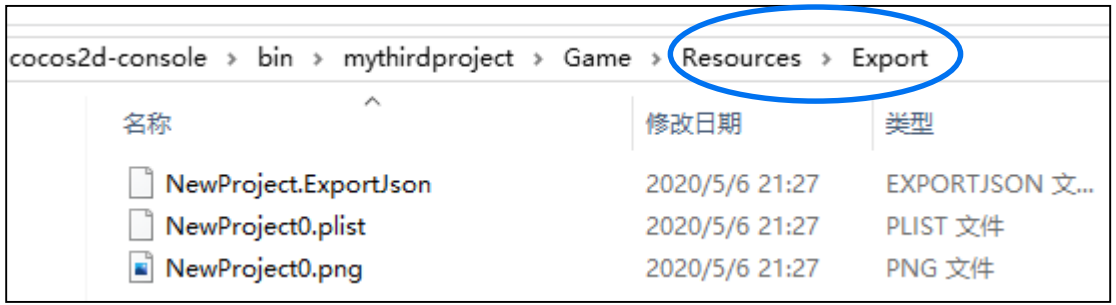


Cocos2d-x动画编辑器

- 4.导出动画
  - 选择文件->导出项目
  - 使用默认参数导出动画

导出项目

导出完毕后，把导出的文件夹拷贝到cocos2d-x project的Resource文件夹下，就能够在项目中使用了。



制作序列帧动画

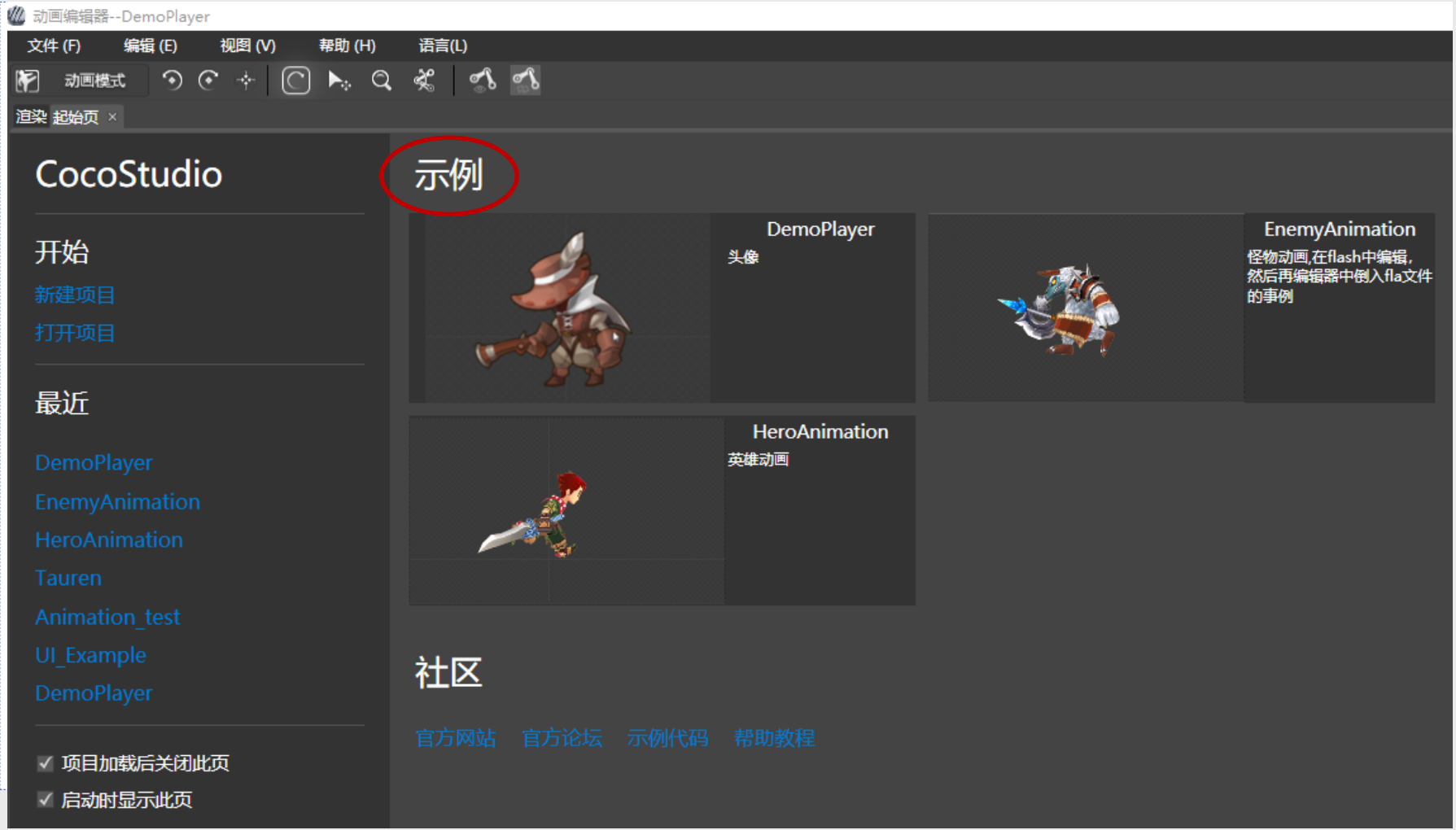
本节内容

- Chapter 7
  - Cocos2d-x动画编辑器
    - 序列帧动画
    - 骨骼动画
  - 提高计算机动画效果的基本手法
  - Cocos2d-x中与动画相关的类

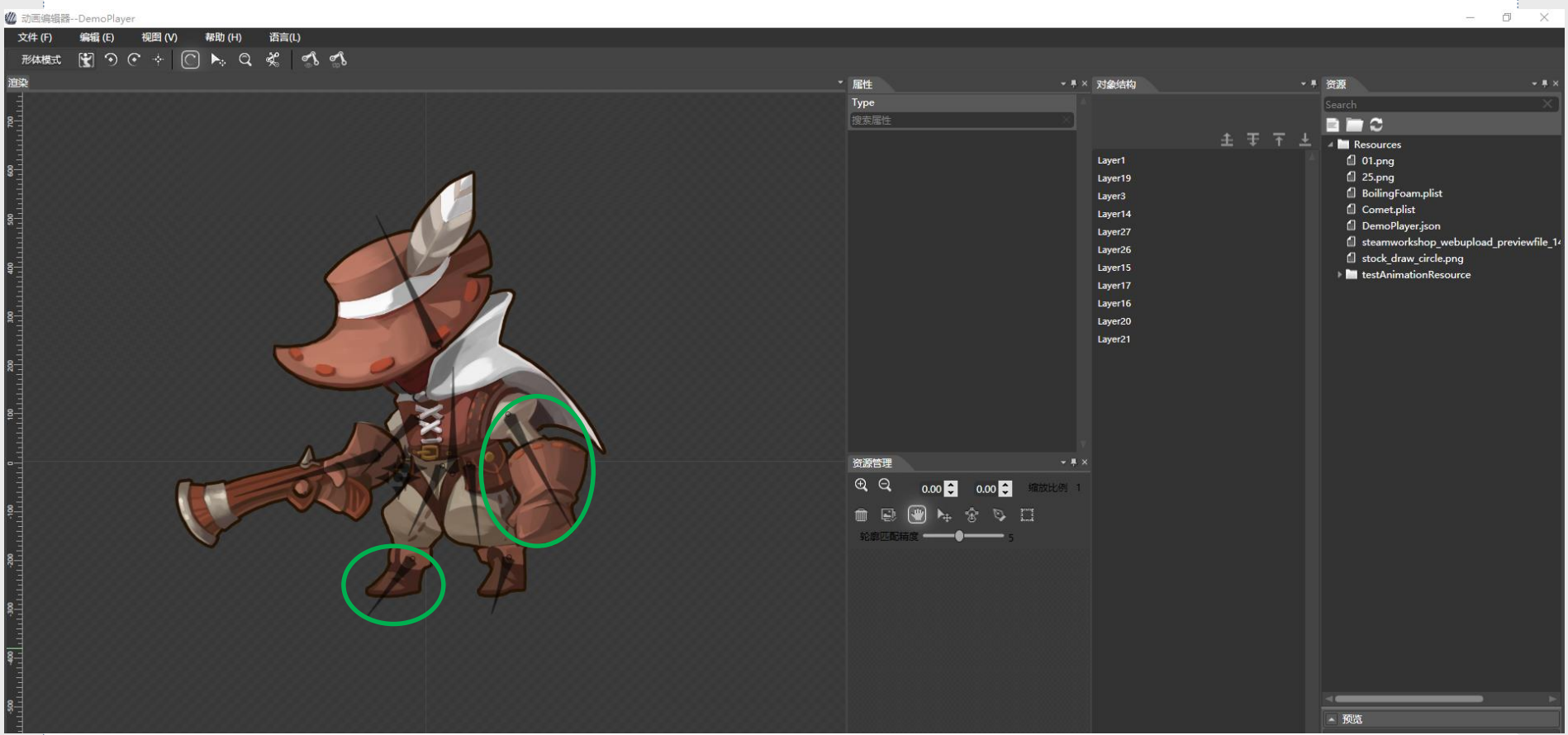
Cocos2d-x动画编辑器

- CocosStudio
  - 动画编辑器
    - 角色动画、特效动画、场景动画资源
    - 关键帧动画、序列帧动画、骨骼动画方式
  - UI编辑器
  - 数据编辑器
  - 场景编辑器

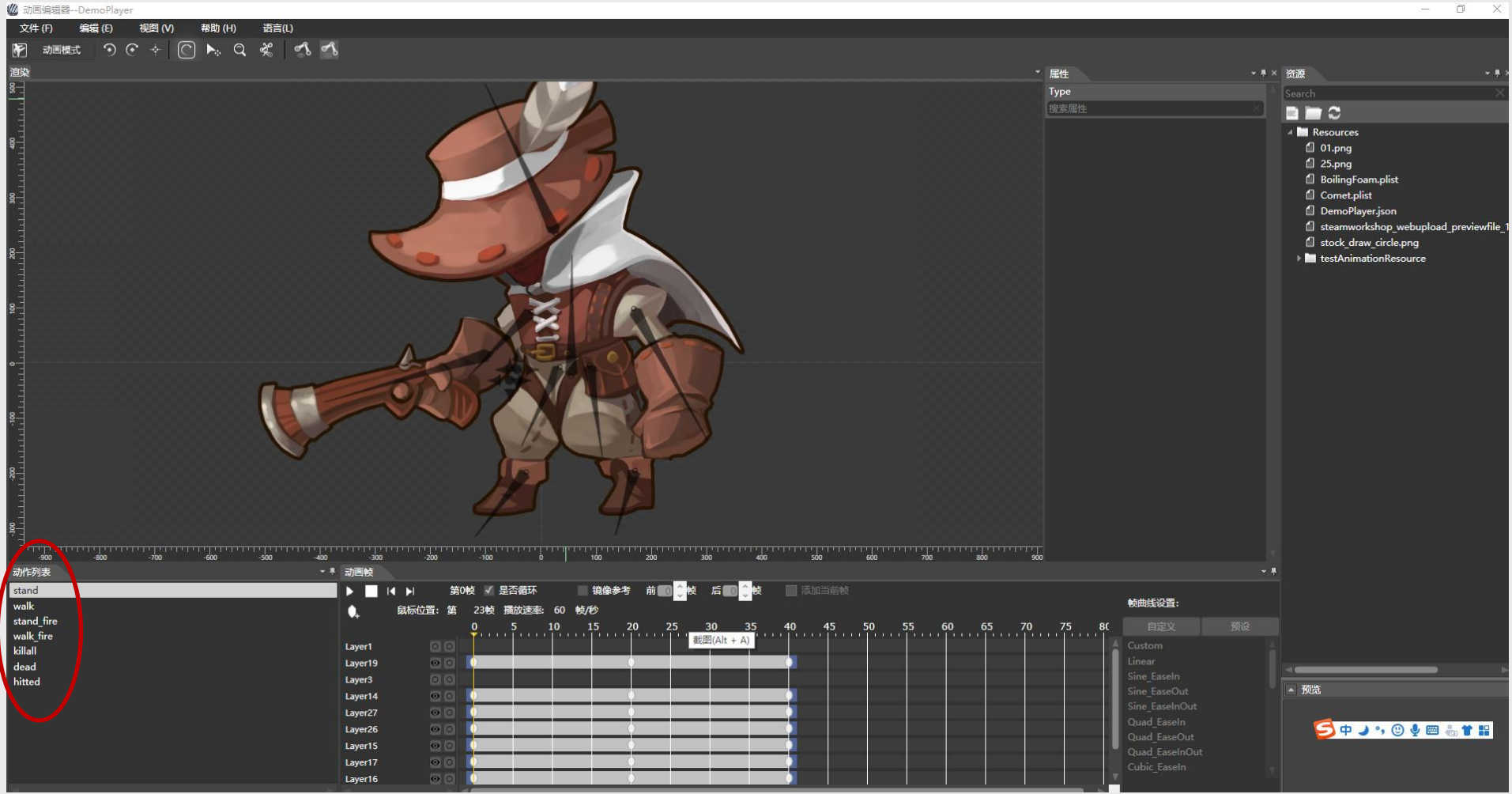
Cocos2d-x动画编辑器



Cocos2d-x动画编辑器

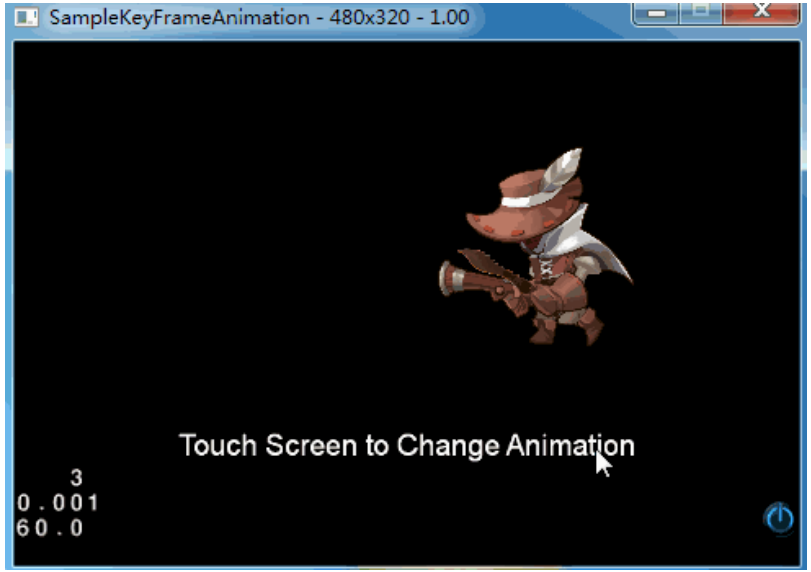


Cocos2d-x动画编辑器



Cocos2d-x动画编辑器

- 骨骼动画绘制步骤
  - 创建骨骼
  - 绑定骨骼
  - 绑定父子关系
  - 编辑动画



骨骼动画制作

本节内容

- Chapter 7
  - Cocos2d-x动画编辑器
    - 序列帧动画
    - 骨骼动画
  - 提高计算机动画效果的基本手法
  - Cocos2d-x中与动画相关的类

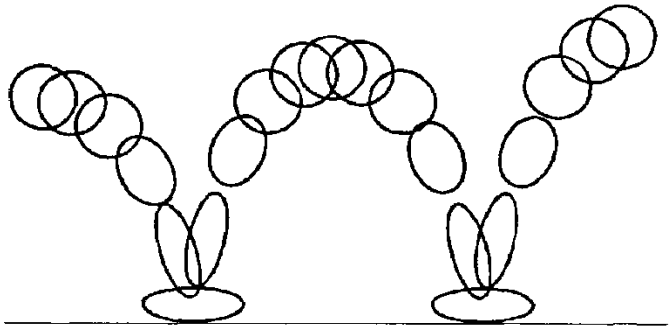
提高计算机动画效果的基本手法

- 挤压与拉伸（Squash and Stretch）
- 时间分配（Timing）
- 预备动作（Anticipation）
- 场景布局（Staging）
- 惯性动作与交迭动作（Follow Through and Overlapping Action）
- 连续动作与重点动作（Straight Ahead Action and Pose-To-Pose Action）
- 慢进和慢出（Slow In and Out）
- 弧形运动（Arcs）
- 夸张（Exaggeration）
- 附属动作（Secondary Action）
- 吸引力（Appeal）

提高计算机动画效果的基本手法

- 挤压与拉伸（Squash and Stretch）

面积 / 体积不变

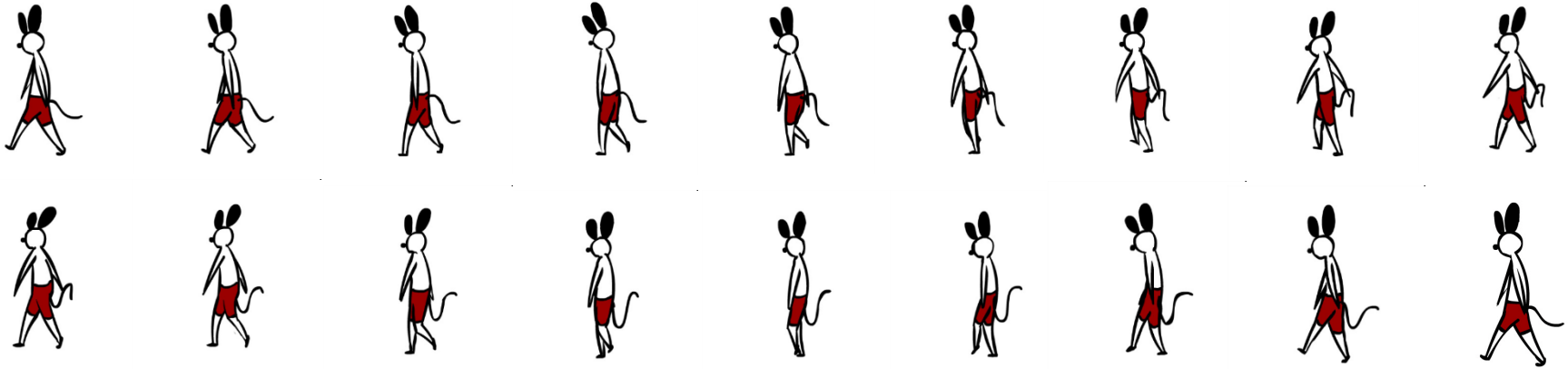


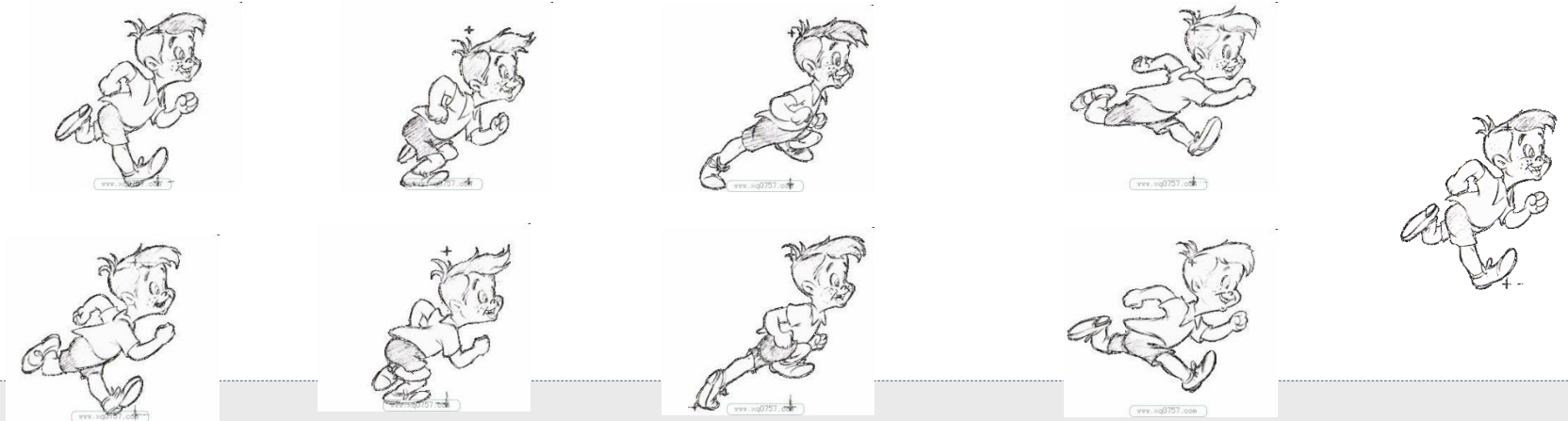
提高计算机动画效果的基本手法

- 时间分配（Timing）

提高计算机动画效果的基本手法

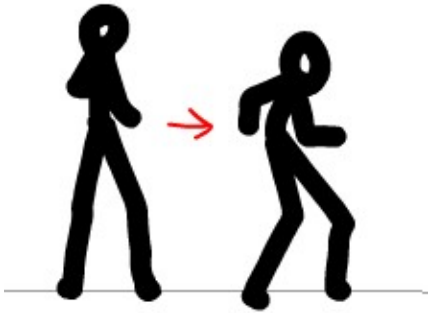
- 时间分配（Timing）





提高计算机动画效果的基本手法

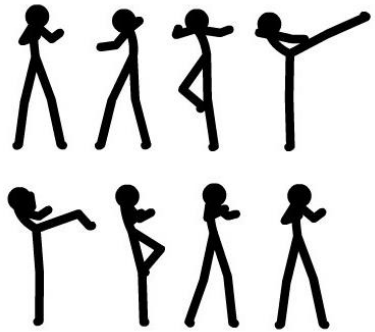
- 预备动作（Anticipation）
- 场景布局（Staging）

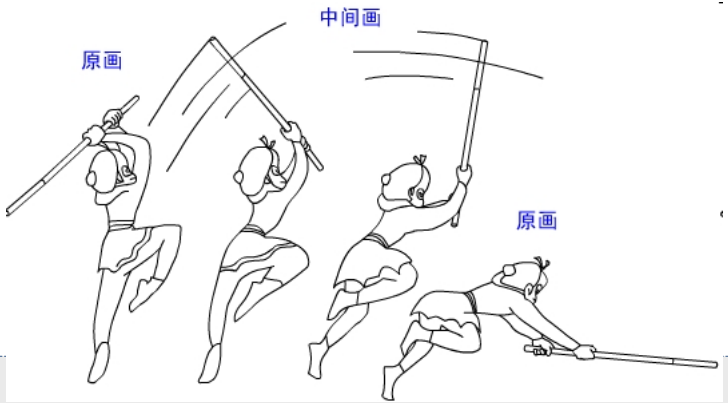


引导观众在正确的时机
关注到屏幕的正确位置

提高计算机动画效果的基本手法

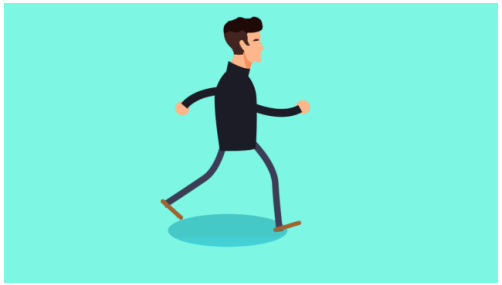
- 惯性动作与交迭动作（Follow Through and Overlapping Action）
- 连续动作与重点动作（Straight Ahead Action and Pose-To-Pose Action）

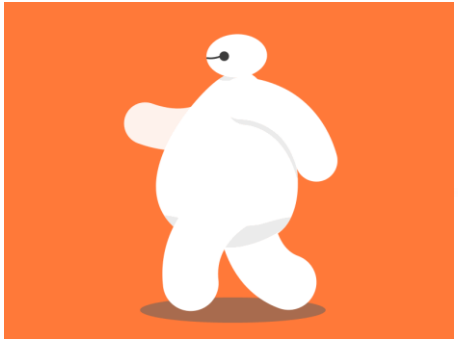


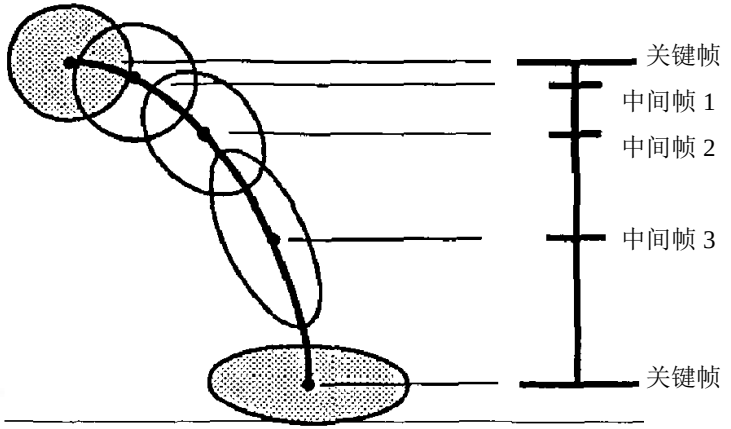


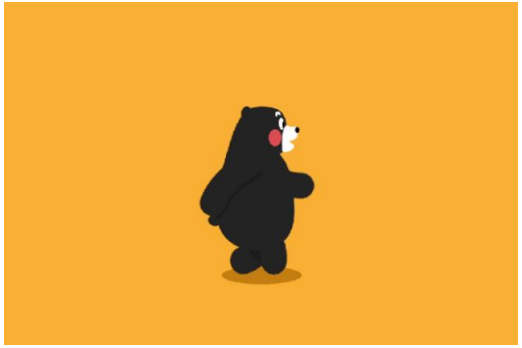
提高计算机动画效果的基本手法

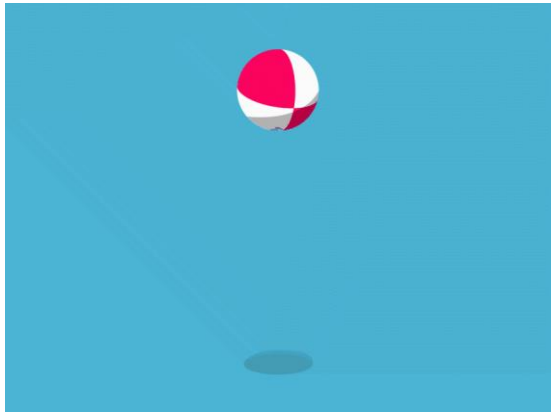
- 慢进和慢出（Slow In and Out）
- 弧形运动（Arcs）











提高计算机动画效果的基本手法

- 夸张（Exaggeration）
- 附属动作（Secondary Action）
- 吸引力（Appeal）

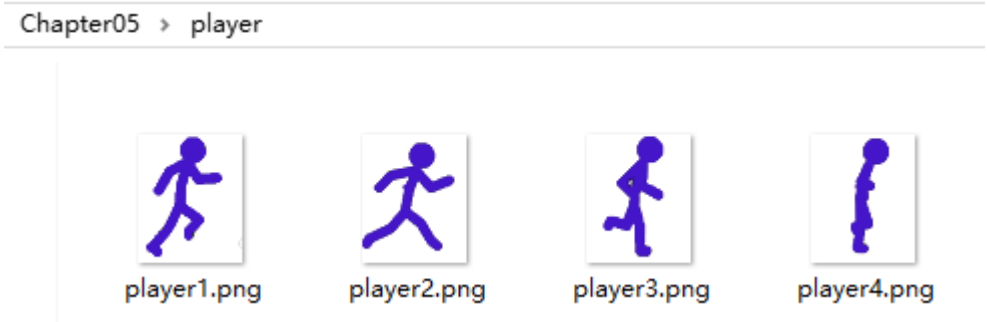
细节决定成败

本节内容

- Chapter 7
  - Cocos2d-x动画编辑器
    - 序列帧动画
    - 骨骼动画
  - 提高计算机动画效果的基本手法
  - Cocos2d-x中与动画相关的类

英雄快跑

- 英雄动画
- Animation
  - 动画
    - auto animation=Animation::create();
  - 封装一系列动画帧
    - animation->addSpriteFrameWithFile(“player1.png”);
    - animation->addSpriteFrameWithFile(“player2.png”);
  - 可设定动画帧的播放间隔
    - animation->setDelayPerUnit(0.1f);



英雄快跑

- 英雄动画
- Animate

Animation 只是存放了动画需要的数据信息，而执行动画还是需要 Animate

  - 动画呈现
  - 专门用来呈现动画的持续动作类

    - auto animate=Animate::create(animation);
    - sprite->runAction(animate);

英雄快跑

- 英雄动画

```
25      // player img path
26      static string PLAYER_IMG_PATH[4] = {
27          "Chapter05/player/player1.png",
28          "Chapter05/player/player2.png",
29          "Chapter05/player/player3.png",
30          "Chapter05/player/player4.png"
31      } ;
```

cene.h | Config.h | MapScene.cpp

me | MapScene | update(float t)

```
169
170     void MapScene::addPlayer(Vec2 pos)
171     {
172         // 玩家跑动动画
173         Vector<SpriteFrame* >frameVector;
174         for(int i=0;i<4;i++)
175         {
176             auto spriteFrame = SpriteFrame::create(PLAYER_IMG_PATH[i], Rect(0,0,PLAYER_WIDTH,PLAYER_HEIGHT));
177             frameVector.pushBack(spriteFrame);
178         }
179         auto animation = Animation::createWithSpriteFrames(frameVector);
180         animation->setDelayPerUnit(0.07f);
181         auto animate = Animate::create(animation);
182
183         // 添加玩家
184         auto player = Sprite::create();
185         player->setTag(PLAYER_TAG);
186         player->runAction(RepeatForever::create(animate));
187         this->addChild(player,10);
188         player->setPosition(pos);
189     }
```

精灵

精灵的创建

可以使用一张图像来创建精灵，*PNG*, *JPEG*, *TIFF*, *WebP*, 这几个格式都可以。当然也有一些其它的方式可以创建精灵，如使用 **图集** 创建，通过 **精灵缓存** 创建，我们会一个一个的讨论。本节介绍通过图像创建精灵。

使用图像创建

`Sprite` 能用一个特定的图像去创建:

```
auto mySprite = Sprite::create("mysprite.png");
```

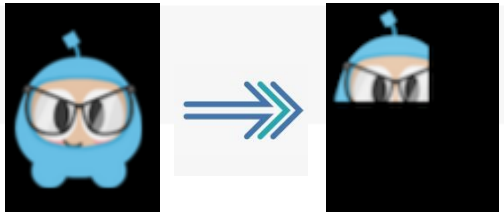
上面直接使用了 **mysprite.png** 图像来创建精灵。精灵会使用整张图像，图像是多少的分辨率，创建出来的精灵就是多少的分辨率。比如图像是 200 x 200， `Sprite` 也是 200 x 200。

精灵

使用矩形

上一个例子，精灵和原始图像的尺寸一致。但是如果你想创建一个尺寸只有原始图像一部分的精灵，那你可以在创建的时候指定一个矩形，指定矩形需要四个值，初始 x 坐标，初始 y 坐标，矩形宽，矩形高。

```
auto mySprite = Sprite::create("mysprite.png", Rect(0,0,40,40));
```

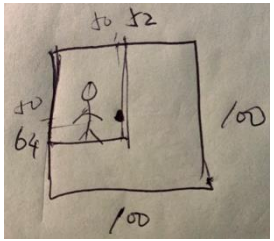


矩形的初始坐标，从图形的左上角开始算，即左上角的坐标是 (0, 0)，不是从左下角。因此结果精灵是图像左上角的一小块，从左上角开始算起，40 x 40 的大小。

尺寸设置摘要

原始尺寸:	52 x 64 像素
调整后尺寸:	52 x 64 像素

```
static int PLAYER_WIDTH = 100;
static int PLAYER_HEIGHT = 100;
```

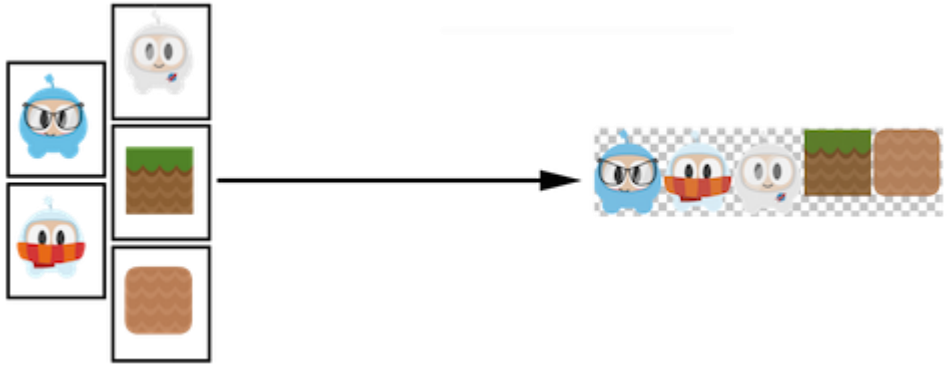



```
auto spriteFrame = SpriteFrame::create(PLAYER_IMG_PATH[i], Rect(0,0,PLAYER_WIDTH,PLAYER_HEIGHT));
frameVector.pushBack(spriteFrame);
```

精灵

使用图集

图集(Sprite Sheet) 是通过专门的工具将多张图片合并成一张大图，并通过 plist 等格式的文件索引的资源，使用图集比使用多个独立图像占用的磁盘空间更少，还会有更好的性能。这种方式已经是游戏行业中提高游戏性能的标准方法之一。

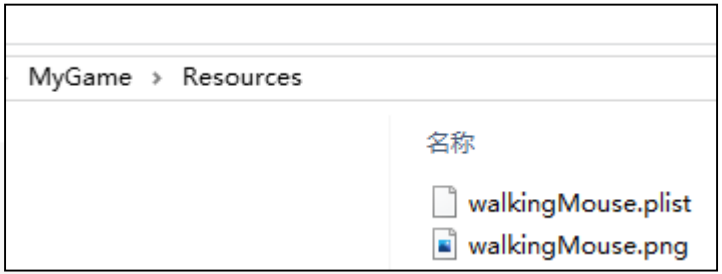


在使用图集时，首先将其全部加载到 `SpriteFrameCache` 中，`SpriteFrameCache` 是一个全局的缓存类，缓存了添加到其中的 `SpriteFrame` 对象，提高了精灵的访问速度。`SpriteFrame` 只加载一次，后续一直保存在 `SpriteFrameCache` 中。

精灵

加载图集

获取到 `SpriteFrameCache` 的实例，把图集添加到实例中。



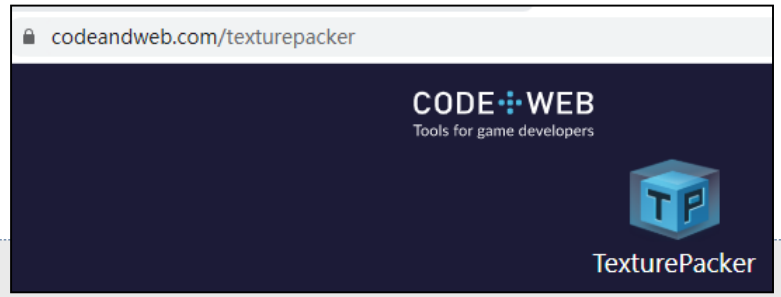

```
// load the Sprite Sheet
auto spritecache = SpriteFrameCache::getInstance();

// the .plist file can be generated with any of the tools mentioned below
spritecache->addSpriteFramesWithFile("sprites.plist");
```

这样我们就完成了，将一个图集添加到 `SpriteFrameCache` 中，现在我们就能利用这个对象创建精灵了！

创建图集

- Texture Packer



精灵

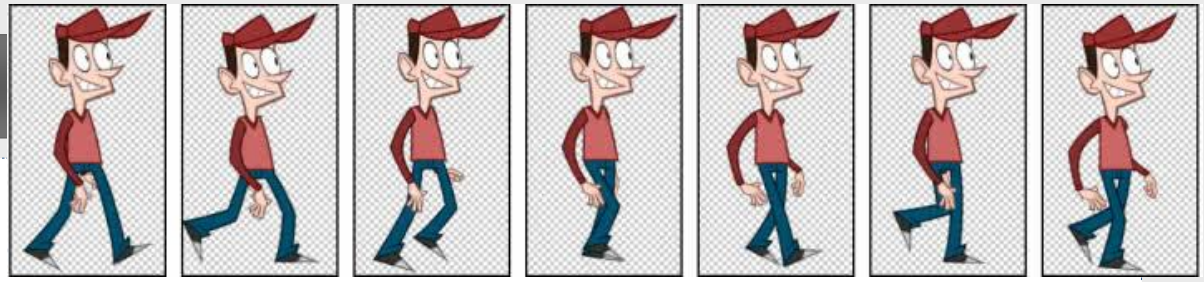
创建图集

- Texture Packer

Tutorials for Cocos2d-X

精灵图集

精灵



使用精灵缓存

精灵缓存是 Cocos2d-x 为了提高精灵的访问速度，提供的一个精灵的缓存机制。

我们可以创建一个精灵并把精灵放到精灵的缓存对象 `SpriteFrameCache` 中：

```
// Our .plist file has names for each of the sprites in it.  We'll grab
// the sprite named, "mysprite" from the sprite sheet:
auto mysprite = Sprite::createWithSpriteFrameName("mysprite.png");
```

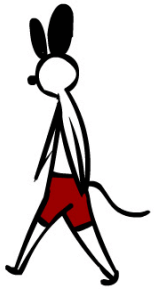
相对的，我们也可以从精灵的缓存对象 `SpriteFrameCache` 访问一个精灵，访问方法是先从缓存对象中获取对应的 `SpriteFrame`，然后从 `SpriteFrame` 创建精灵，方法：

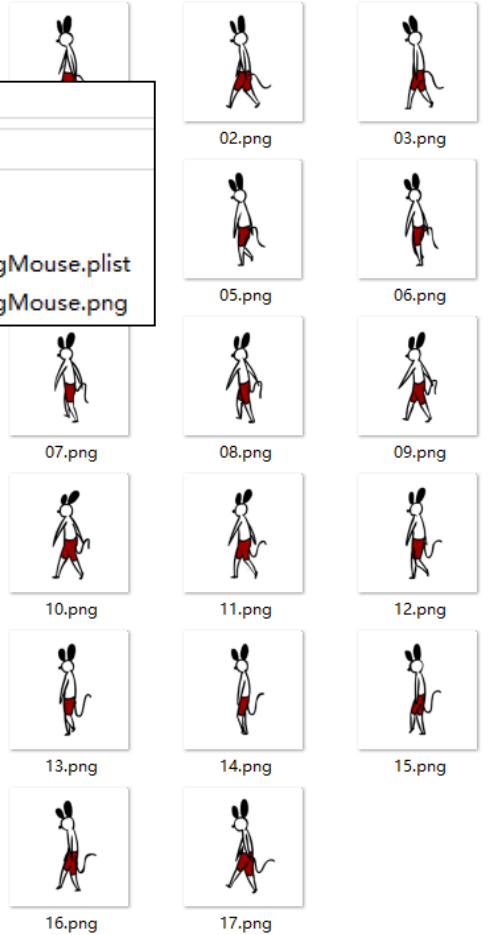
```
// this is equivalent to the previous example,
// but it is created by retrieving the SpriteFrame from the cache.
auto newspriteFrame = SpriteFrameCache::getInstance()->getSpriteFrameByName("Blue_Front1.png");
auto newSprite = Sprite::createWithSpriteFrame(newspriteFrame);
```

精灵

```
// load the Sprite Sheet
auto spritecache = SpriteFrameCache::getInstance();
// the .plist file can be generated with any of the tools mentioned below
spritecache->addSpriteFramesWithFile("walkingMouse.plist");
```

```
// walkingMouse animation demo
int num = 17;        // number of pictures for walking mouse
auto animation = Animation::create();
auto sprite = Sprite::create();
for (int i = 1; i < num; i++)
{
    char name[20];
    if (i < 10)
    {
        sprintf(name, "0%d.png", i);
    }
    else
    {
        sprintf(name, "%d.png", i);
    }
    auto newspriteFrame = spritecache->getSpriteFrameByName(name);
    animation->addSpriteFrame(newspriteFrame);
}
animation->setDelayPerUnit(0.1f);
auto animate = Animate::create(animation);
sprite->runAction(RepeatForever::create(animate));
```





精灵帧动画

骨骼动画设计

- 课后任务：

  ✓下载Cocos Studio并安装（已完成可忽略）

  ✓利用Animation Editor示例项目素材（或自行搜索、下载素材），完成Player、Enemy或Hero骨骼动画（三选一）

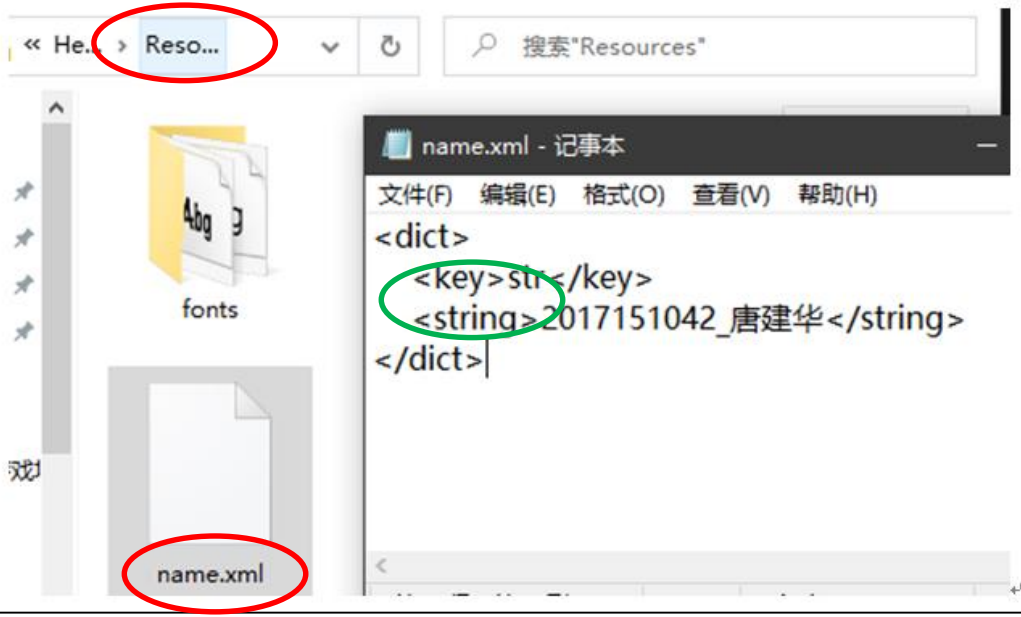
  ✓要求：

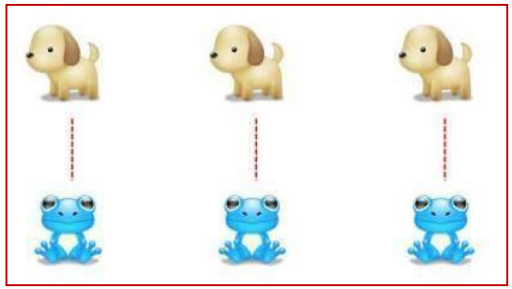
  - ✓完成一个完整运动周期的骨骼动画
  - ✓动画内容可以是走路、跑步、跳跃等
  - ✓将动画录屏文件上传至BB系统
  - ✓截止日期：2021. 11. 22晚11:59分

游戏标题栏中文显示问题

修改游戏显示名称

① 由于 cocos2dx 使用中文会出现乱码，所以需要写一个 xml 文件放在 Resources 文件夹内，如下





游戏标题栏中文显示问题

② 接着在 APPDelegate.cpp 中引入 xml，获取其中的文字内容赋值到 str，实现如下。

```
bool AppDelegate::applicationDidFinishLaunching() {
    // initialize director
    auto director = Director::getInstance();
    auto glview = director->getOpenGLView();
    auto *chnStrings = Dictionary::createWithContentsOfFile("name.xml");
    const char *str = ((String*)chnStrings->objectForKey("str"))->getCString();
    if(!glview) {
#if (CC_TARGET_PLATFORM == CC_PLATFORM_WIN32) || (CC_TARGET_PLATFORM == CC_PLATFORM_MAC) || (CC_TARGET_PLATFORM == CC_PLATFORM_LINUX)
        glview = GLViewImpl::createWithRect(str, cocos2d::Rect(0, 0, designResolutionSize.width, designResolutionSize.height));
#else
        glview = GLViewImpl::create(str);
#endif
        director->setOpenGLView(glview);
    }
```

③ 运行游戏，查看窗口名字，如下。

游戏标题栏中文显示问题