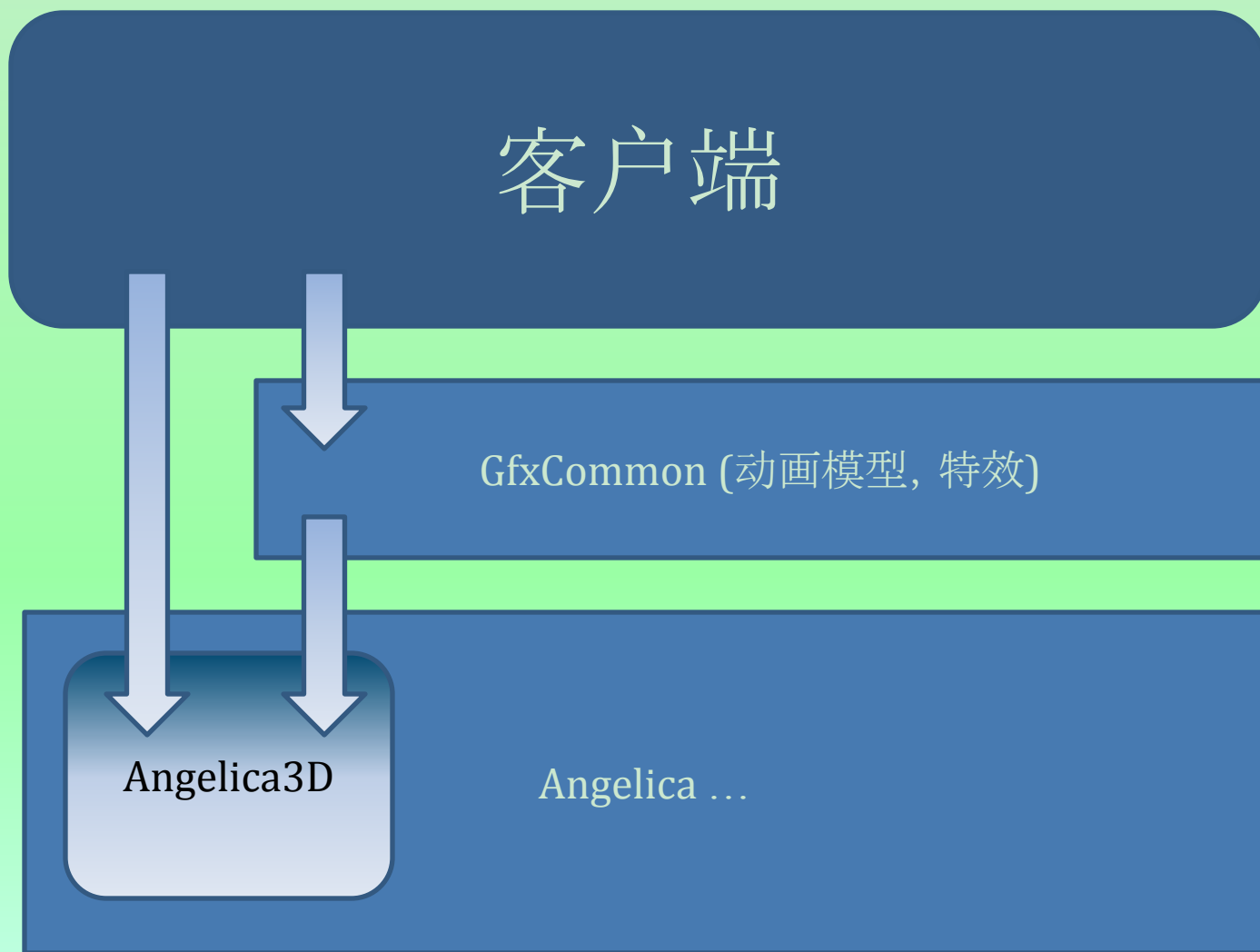


ECModel & GFX 使用

张亚川

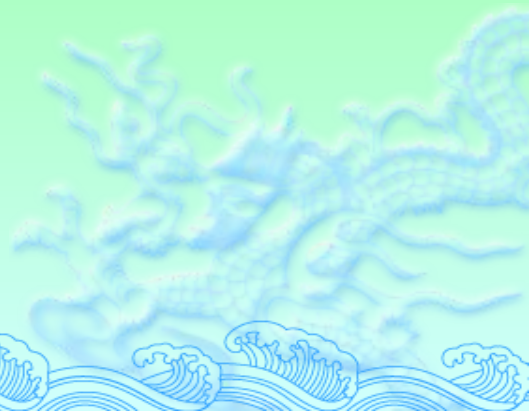


客户端、引擎、GfxCommon关系简图



目录

- ▣ A3DSkinModel 简介及动画相关
 - ▣ 3DMAX导出
 - ▣ 文件、类说明
 - ▣ 动作通道简介
 - ▣ 从 A3DSkinModel 到 CECModel
- ▣ CECModel 使用
 - ▣ 由.smd生成
 - ▣ 子模型, 组合动作
 - ▣ 动作事件
 - ▣ 物理相关
 - ▣ 注意事项
- ▣ GFX 使用
 - ▣ GFX各类元素的设计及GFX Container的概念
 - ▣ GFX的注册渲染机制

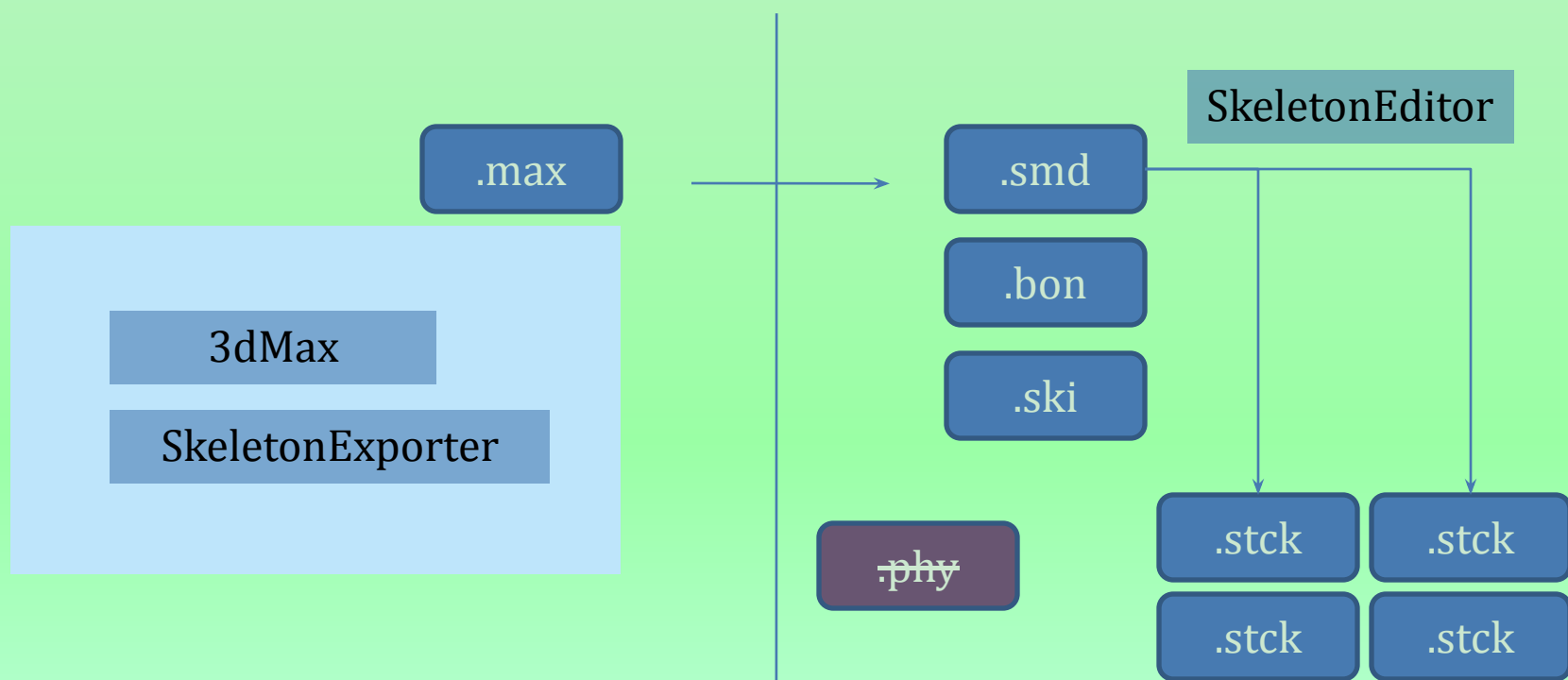


美术制作流程

- 3dmax导出(骨骼,动画,皮肤)
 - .smd, .bon, [.stck], .ski [参考实际导出的文件](#)
- 动作合并(.bon->.stck)——将动作信息存入.smd (在A3DSkinModel中添加基础动作:
[A3DSkinModelActionCore](#))
- 根据.smd创建.ecm文件, 创建游戏当中实际需要的组合动作, 配上特效, 音效等

注:美术导出动作的时候, 是以.bon的形式导出出来的, 其中包含了骨架以及动画信息, 需要经过SkeletonEditor处理之后才实际生成.stck文件

A3DSkinModel 从max导出



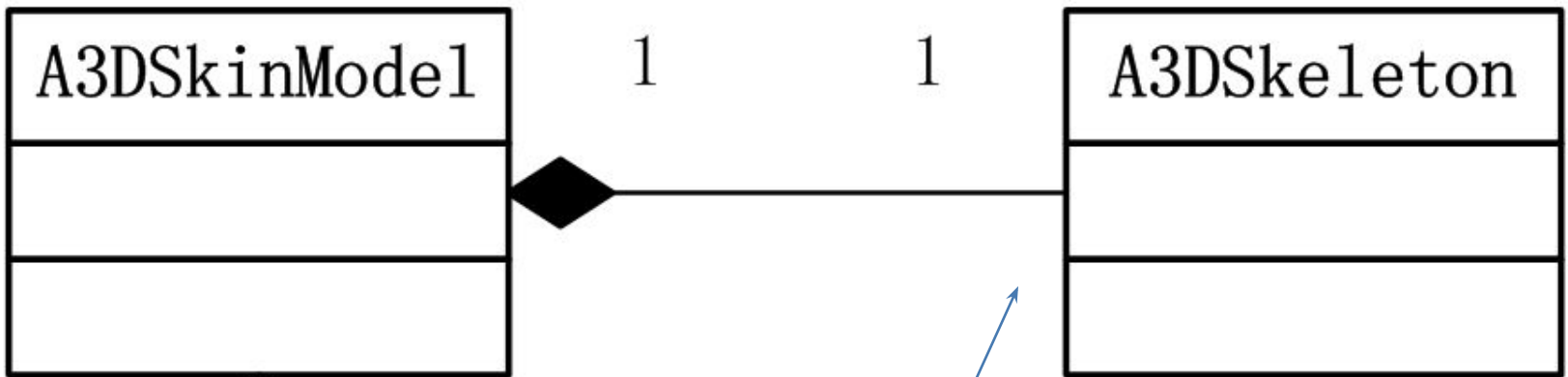
.Max可以分别导出.smd, .bon, .ski, 一个动作导出一个.bon(经过SkeletonEditor转换为.stck)

注:.phy文件已经不再使用了, 曾经是用来做物理信息的, 后采用基于PhysX的Aphys库提供物理功能

A3DSkinModel 文件说明

- **.smd -> [A3DSkinModel](#)**
 - 综合了SkinModel所需的各类信息(骨架, 动作, 皮肤等)
- **.bon -> [A3DSkeleton](#)**
 - 骨架, 骨骼, 挂点, 骨骼的父子关系等
- **.stck -> [A3DSklTrackSet](#)**
 - 动画数据, 骨架上每个骨骼的位置朝向随时间变化的关键帧数据

每个A3DSkinModelActionCore包含一个A3DSklTrackSet
- **.ski -> A3DSkin**
 - 蒙皮, Mesh数据(顶点,索引,贴图等)



对应.smd文件

对应.bon文件

1

1

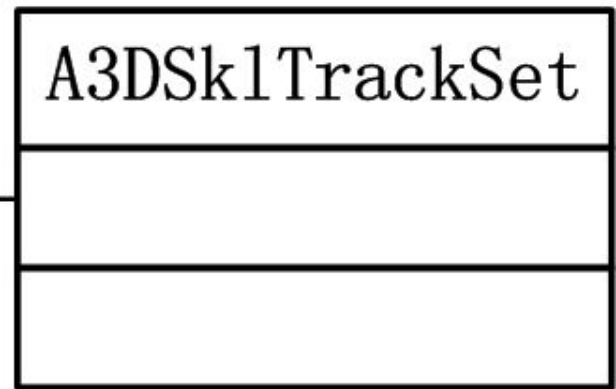
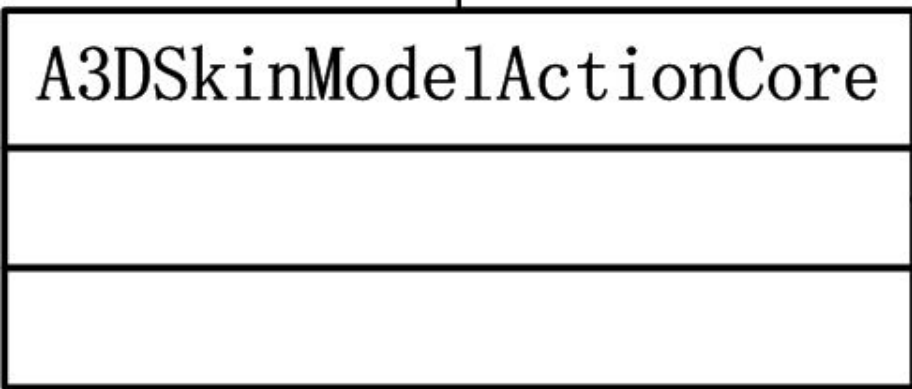
1

A3DSkinModel类结构简图

0..*

基础动作

对应.stck文件



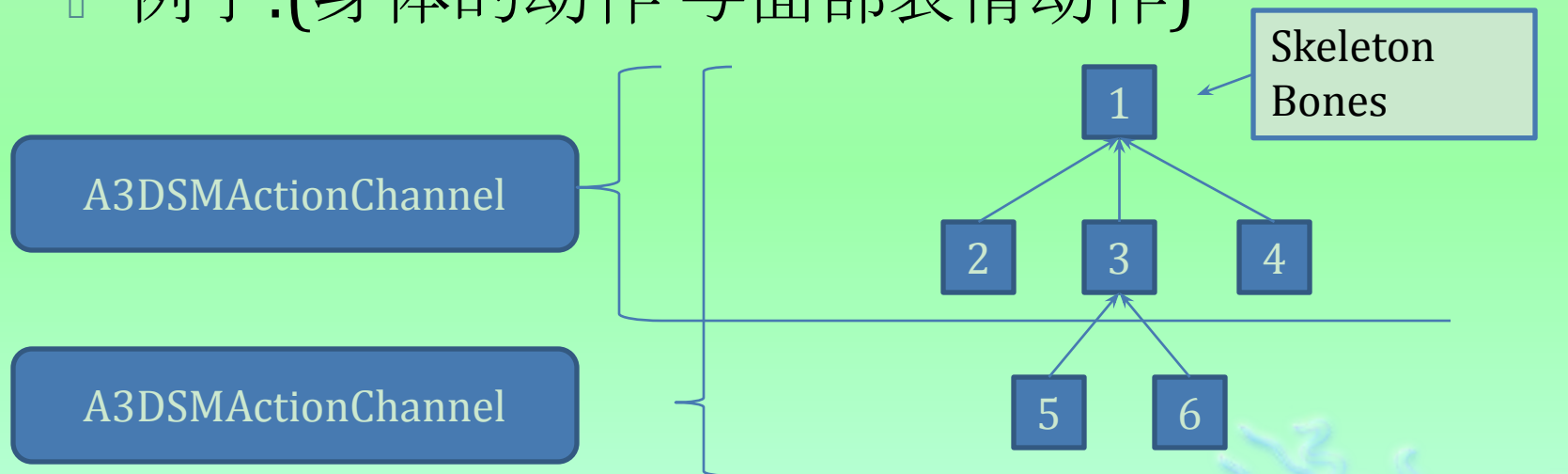
1

1

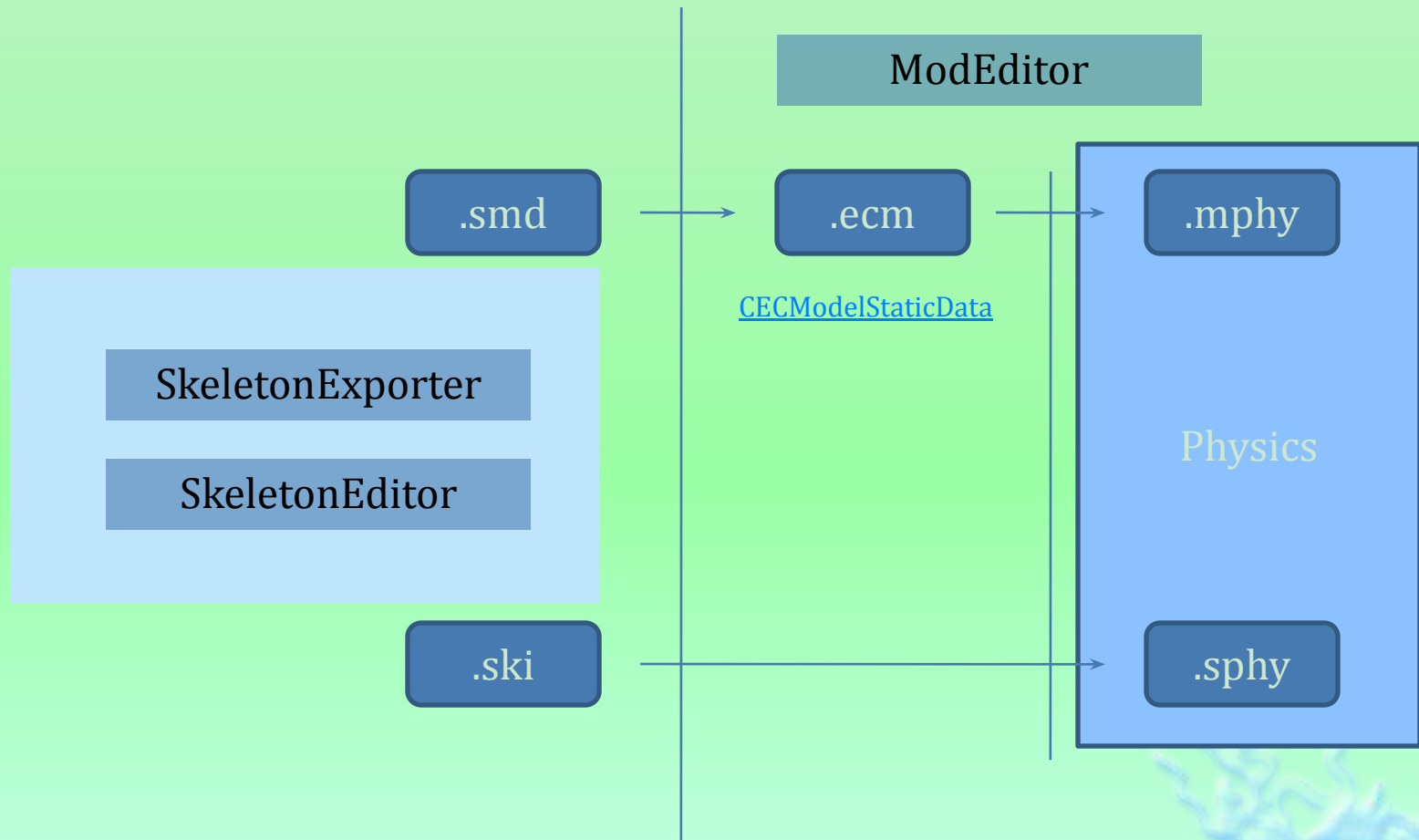


A3DSkinModel 动作通道

- 动作通道是用来解决什么问题的？
 - 多个动作同时播放，按照权重混合
 - 例子:(身体的动作 与 面部表情动作)



从A3DSkinModel到CECModel



ECModel

- 为什么需要CECModel?
 - 子模型管理
 - 多个简单动作组合成一个复杂动作(技能等)
 - ◆ 播放动作到特定时刻播放音效(需要由美术控制)
 - ◆ 播放动作到特定时刻播放特效(同上)
 - ◆ 播放动作到特定时刻触发脚本
 - ◆ etc.
 - 游戏里需要预设一些不同骨骼缩放系数的模型(缩放)
 - 游戏需要对动画模型进行碰撞检测(凸包)
 - 动画模型与物理引擎关联(物理)
 - 其他与客户端逻辑相关的(脚本)接口(逻辑)

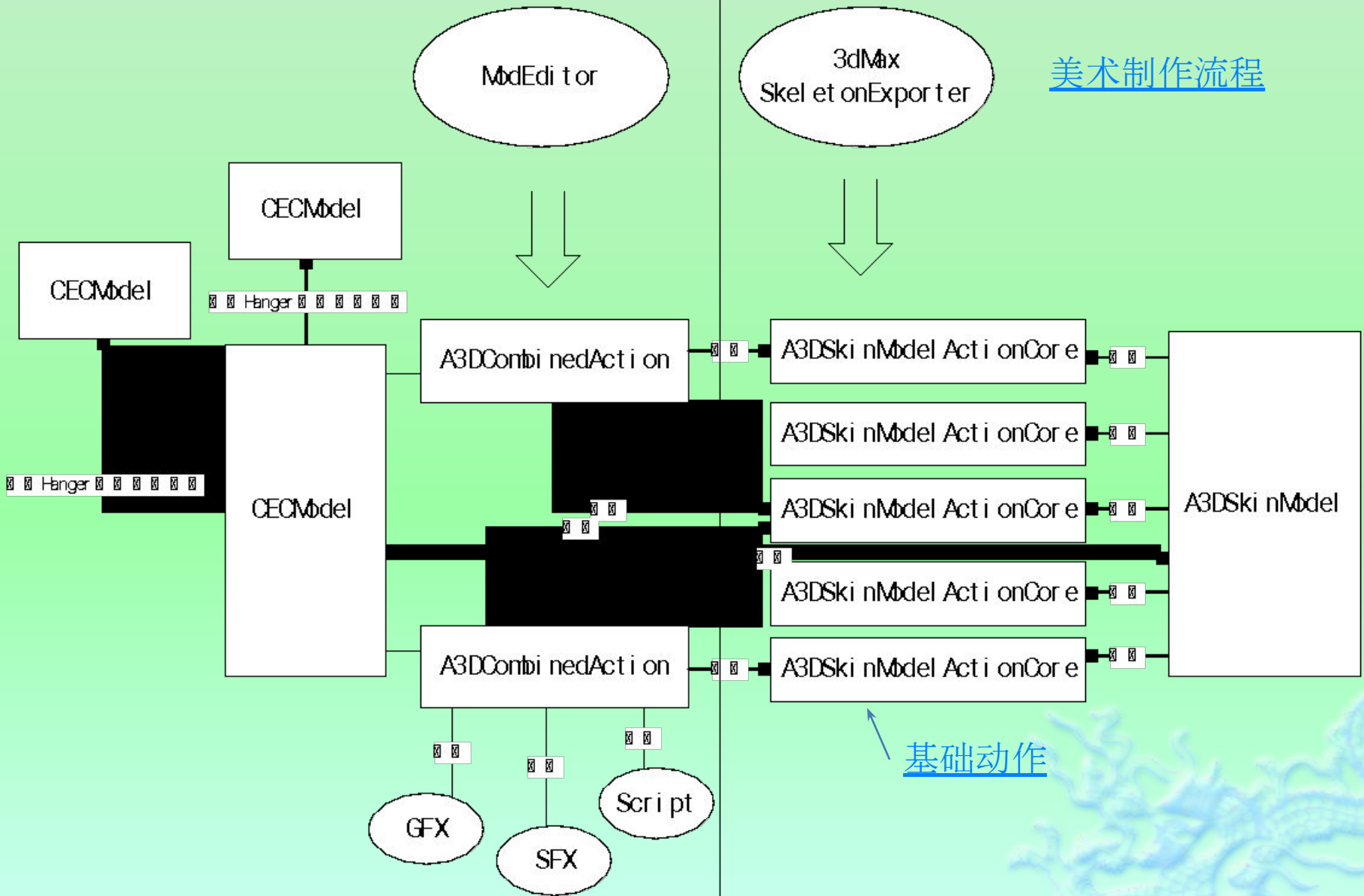
ECModel

类: CECModel
文件: .ecm

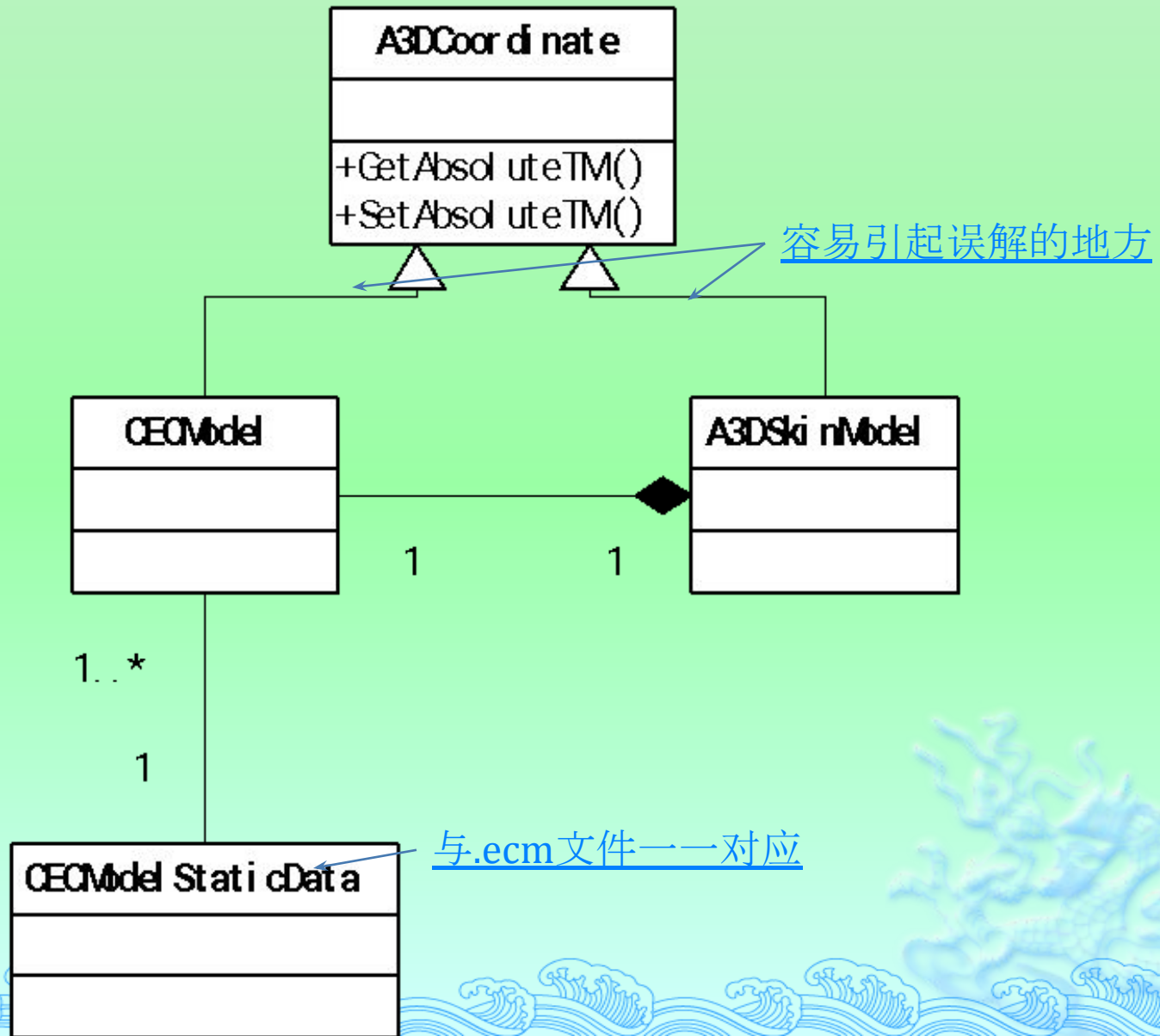
- 与A3DSkinModel的关系
- 类结构简介
- 子模型
- 组合动作
- 动作事件
- 脚本
- 物理
- 其他注意事项



ECModel与A3DSkinModel的关系



ECModel 结构简图



ECModel 子模型

- 子模型用在什么场合？
 - 武器
 - 骑乘
- 父子模型间通过挂点连接 HH_ <-> CC_(注释1)
- **Hanger:** 用来索引子模型的字符串, 由客户端与美术约定
 - 美术可指定一个效果在父模型动作的时间, 以及子模型的挂点上播放(通过Hanger)<例子:父模型播放一个火枪开枪的动作, 枪口冒烟的特效>

ECModel 组合动作

[返回:为什么要
有ECModel](#)

- 什么是组合动作？([编辑器-动作](#))
- 组合动作解决了什么问题？
 - 节约数据:一个动作在多个技能中用到, 那么只需要一份实际的动作数据, 在多个组合动作中调用即可
 - 程序, 策划, 美术, 音效的结合
 - ◆ 程序: 播放动作;
 - ◆ 策划:技能动作名称;美术:动作, 特效, 色彩, 子模型动作;

ECModel 组合动作 数据

- A3DCombinedAction & A3DComActDynData
 - A3DCombinedAction 静态数据:
 - ◆ 由编辑器创建
 - ◆ 包含需要存/读的信息, 各个基本动作名称, 各事件信息等
 - A3DComActDynData 动态数据:
 - ◆ 由 [PlayActionByName / QueueAction / PlayAttackAction](#) 创建
 - ◆ 包含运行时数据
 - ◆ 参与Tick, 负责运行过程中的状态更新
 - ◆ 判断动作是否播完

QueueAction 不会自动保证加入的动作不重复, 因此无限制的调用QueueAction会导致无限的往动作列表中添加动作

ECModel 组合动作 播放

- PlayActionByName & PlayAttackAction
 - PlayActionByName 播放组合动作(可设置 Channel)
 - PlayAttackAction 播放技能动作(需传入自身ID及目标ID, 组合动作上挂载的Attack事件会播放飞行GFX及击中目标GFX, 过程中回调客户端提供的函数, 由客户端根据ID及其他附属信息计算位置)
 - [参考攻击事件ATT](#)

ECModel 组合动作 各种参数1

<code>bool PlayActionByName(</code>	<code>int nChannel,</code>	动作通道
<code>const char* szActName,</code>		动作名称
<code>float fWeight,</code>		动作通道权重 <code>SetWeight</code>
<code>bool bRestart=true,</code>		重新开始(当前如果已经是该动作)
<code>int nTransTime=200,</code>		过渡时间(从上一个动作的国度)
<code>bool bForceStop=true,</code>		强制停止
<code>DWORD dwUserData=0,</code>		用户数据
<code>bool bNoFx=false);</code>		不播放任何附加事件

<code>bool PlayAttackAction(</code>	<code>int nChannel,</code>	动作通道
<code>const char* szActName,</code>		动作名称
<code>unsigned char SerialId,</code>		回调客户端的参数
<code>clientid_t nCasterId,</code>		发起者Id
<code>clientid_t nCastTarget,</code>		目标Id
<code>const A3DVECTOR3* pFixedPoint,</code>		击中效果定点播放
<code>DWORD dwUserData = 0,</code>		用户数据
<code>bool bNoFx = false);</code>		不播放任何附加事件

*FixedPoint用于没有目标玩家或怪,但是有一个特定位置的情况,该状况下nCastTarget可为0

ECModel 组合动作 各种参数2

CECModel::AddOneAttackDamageData(

int nChannel,

动作通道

clientid_t nCaster,

发起者id

clientid_t nTarget,

目标id

unsigned char SerialId,

序列Id (回调客户端参数)

DWORD dwModifier,

回调客户端参数

int nDamage,

回调客户端参数

bool bIsAttDelay,

攻击事件是否有延迟效果

int nAttDelayTimeIdx)

延迟时间

客户端实现的函数:

AfxSkillGfxAddDamageData(nCaster, nTarget, SerialId, dwModifier, nDamage);

通过传入参数, 取出A3DSkillGfxEvent, 加入新的TARGET_DATA

如果返回真, 则不再调用下面两个函数

AfxSkillGfxShowDamage(nCaster, nTarget, nDamage, 1, dwModifier);

AfxSkillGfxShowCaster(nCaster, dwModifier);

ECModel 动作事件

- 什么是动作事件？([编辑器-事件](#))
- 动作事件解决了什么问题：
 - 在动作的特定时间点做某件事的自由性(GFX, SFX, ATT, CHILDACT, etc.)(注1)
 - 举例：技能效果的播放，音效的播放，子模型动作，[脚本事件](#)

ECModel 动作事件 ATT

[返回ECModel播放攻击动作](#)

- 攻击事件 ATT
 - 飞行、击中等GFX效果
 - 飞行效果播放过程中回调客户端

[攻击事件的流程的说明](#)

```
virtual GetPositionByID() = 0
```

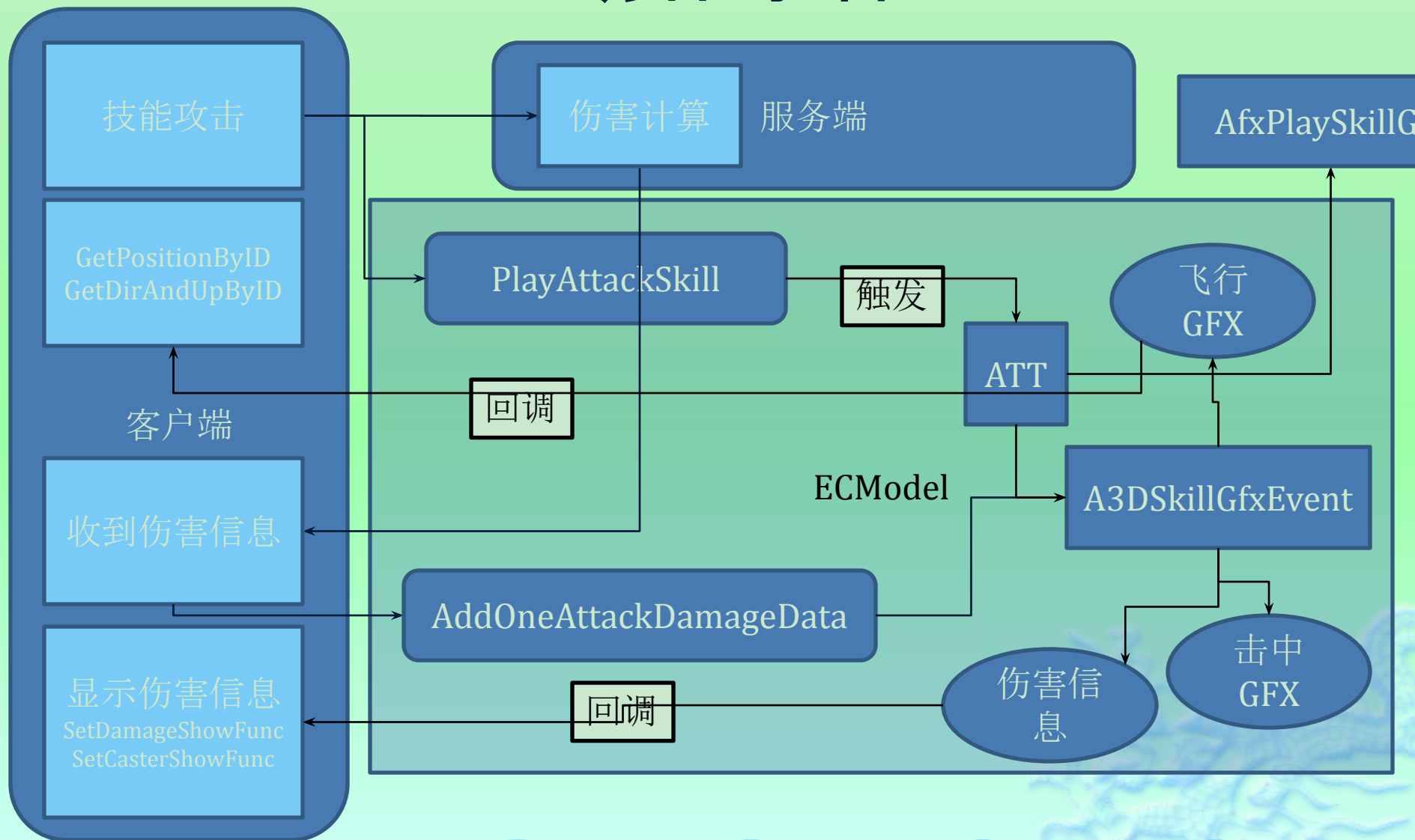
```
GfxCommon::A3DSkillGfxMan
```

```
virtual GetDirAndUpByID() = 0
```

```
ElementClient::SkillGfxMan
```



ECModel 动作事件 ATT - 2



ECModel 脚本(lua)

[返回:为什么要
有ECModel](#)

- 模型发生某些事件时触发 ([编辑器-脚本](#))
 - 加载(初始化)
 - 播放动作(根据动作切换武器挂点)
 - 更换装备(根据装备ID播放特效)
 - 物理破碎
- [动作脚本事件](#)
 - 播放动作到特定时刻触发

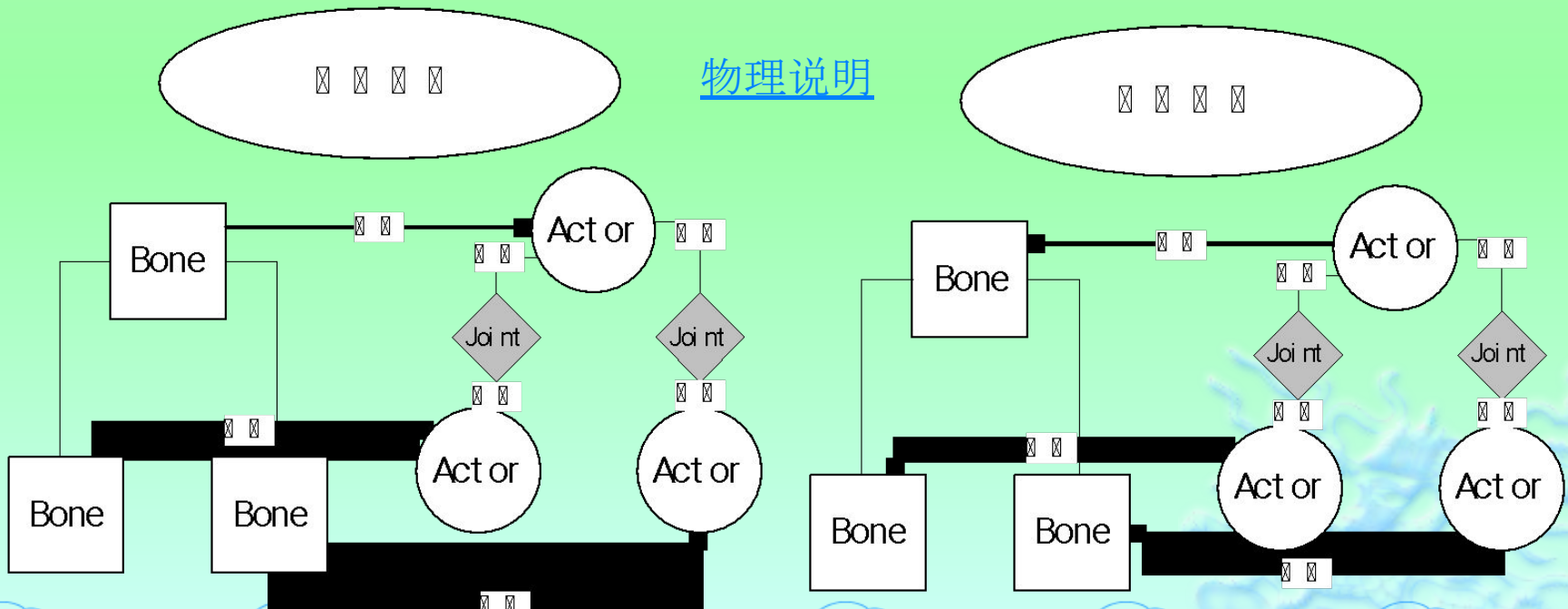


ECModel 物理

[返回:为什么要有ECModel](#)

- APhysX与A3DSkinModel的粘合
 - APhysXActor -> A3DBone
 - APhysXCloth -> A3DSkin

物理参看编辑器



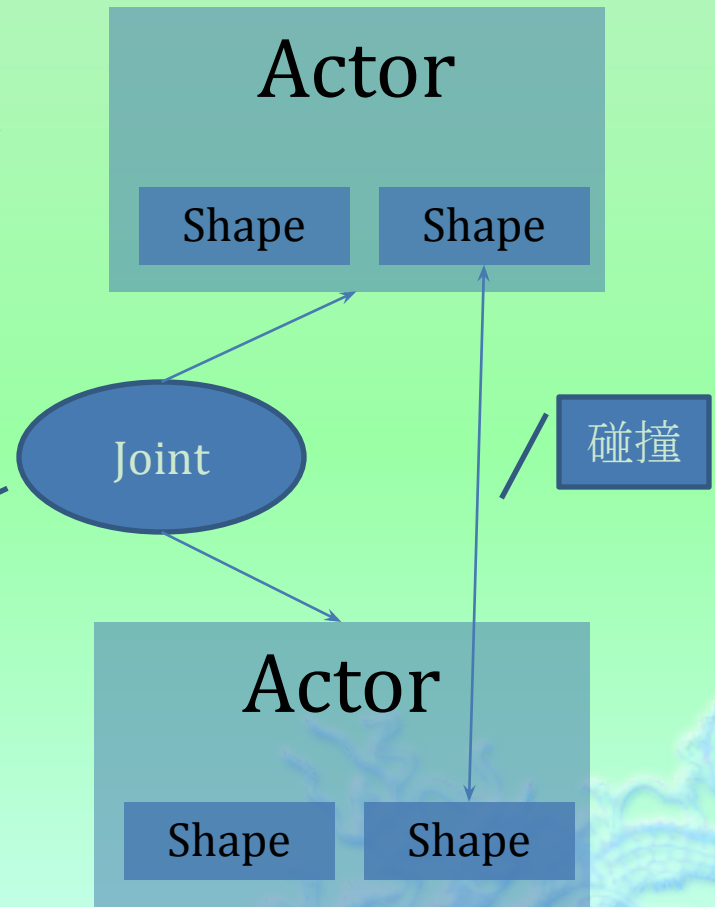
关于物理的简介

[返回ECModel物理](#)

- Actor – 刚体
 - Shape 实际参与物理运算的形体
 - 由美术根据实际形体编辑

- Joint – 约束(针对刚体)

- 1. 绕轴旋转角度限制
- 2. 移动距离限制



ECModel 其他 – 位置

- 模型的逻辑位置与实际位置
 - CECModel 继承自 A3DCoordinate(注1)
 - CECModel 包含 A3DSkinModel
 - A3DSkinModel 也继承自 A3DCoordinate([参考图](#))

```
CECModel::GetAbsoluteTM() // 逻辑的位置与朝向  
!=  
CECModel::GetA3DSkinModel()->GetAbsoluteTM() // 动画的位置与朝向
```

部分特效<标明相对ecm原点的>会依赖于CECModel的A3DCoordinate, 而非A3DSkinModel的A3DCoordinate, 需要注意两者的区分

ECModel 其他 – 各种设置

- **SetId**
 - 在脚本中使用, 于回调客户端时提供;
 - 与Gfx中的ECModel元素配合使用
- **SetGfxUseLod**
 - 模型上挂的GFX里的粒子喷射器受其影响
- **SetDisableCamShake**
 - 禁止模型上挂GFX中的相机振动起作用



ECModel 其他 – 附加皮肤

- 关于ECModel附加皮肤
 - 只用于ModEditor编辑器显示
- 客户端中使用的A3DSkin的两个来源：
 - SkeletonEditor中附加到A3DSkinModel上
 - 客户端中自行加载的
 - (与ECModel中编辑的附加皮肤无关)



ECModel – 其他 透明度

- **SetTransparent/GetTransparent**
 - 设入的透明度和取出的透明度不等同的问题
 - 模型色彩改变[动作事件]的影响



GFX

类: A3DGfxEx

文件: .gfx

- 特效元素简介
- 特效关键帧简介
- 特效渲染机制简介
- 关于2D特效的说明

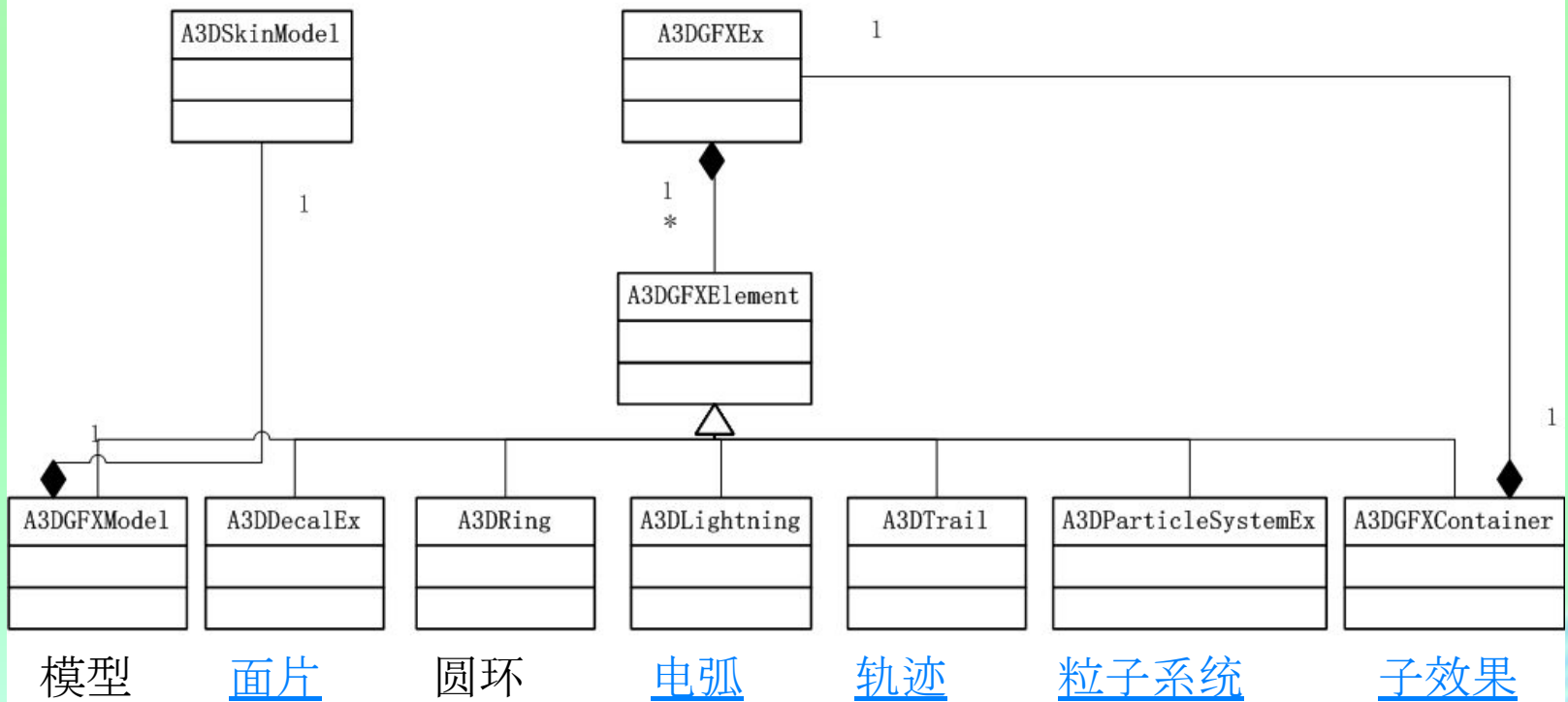


GFX 元素简介

- 3D面片
- 2D面片
- 圆环 (面片圈)
- 电弧 (面片节, 于径向使用PerlinNoise)
- 轨迹 (面片节, 于前端生成新的面片, 每节面片有TTL, 通过运动形成连续的效果)
- 粒子 (各种形状的发射器&面片, 可绑定其他元素)
- 模型 (.smd, 文件A3DSkinModel) ←
- 子效果 (把一个完整的GFX文件看成一个元素)
- 其他

为什么是A3DSkinModel而非CECModel, 注

GFX 结构简图



GFX 元素的例子-1

轨迹



类结构

3D面片



粒子



电弧



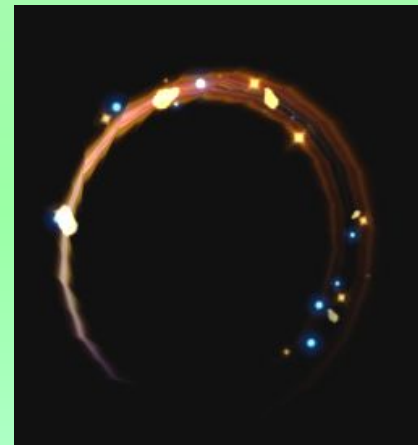
2D面片



GFX 元素的例子-2

添加一个效果为gfx的元素。可用于会多次使用的某一组效果或者元素。

子效果



子效果

类结构

GFX 关键帧

- 关键帧作用于A3DGFXElement
- 关键帧可以影响的属性
 - 移动
 - 旋转
 - 颜色(透明度)改变
 - 比例改变
 - Etc.
- [<参考编辑器>](#)



GFX的注册渲染机制

- **A3DGFXExMan::RegisterGfx (A3DGfxEx*)**
 - 渲染GFX, 首先放到一个容器(GfxRenderContainer)中, RenderAllGfx后该容器被清空
 - (注册机制1, 针对[A3DGFXEx](#))
 - 容器可以替换(可以用在一些特殊场合)
- **A3DGFXExMan::RenderAllGfx**
 - GFX的元素被按照渲染状态注册到渲染容器中
 - (注册机制2, 针对[A3DGFXElement](#))
 - 按照渲染状态的类别分别DrawPrimitive

2D GFX的使用

- 程序联入的2DGFX, 可以通过A3DGFXExMan提供的2DGFX接口进行加载, 渲染等操作(用于登录界面等场合)

参考编辑器2DGFX([诛仙登录界面](#))

- 非程序联入的2DGFX, 可以以3DGFX同样的方式渲染(RegisterGfx)。Flush采用Render2DGfx, 而非RenderAllGfx(注1)
- 关于Render2DGfx的参数(backLayer)——是为了配合界面, 分为前景和背景



在塵埃中活著
神氣以成身軀
存身
物往者存
存者不墮
固羅
只為信故
雖死之悔

Input field 1

Input field 2

强行登录

密码保护卡

进入游戏

服务器选择

内服XX00

退出



谢谢

