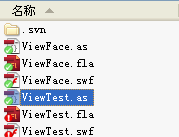
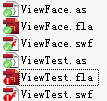
新建ViewTest.as



代码如下：

|  |
| --- |
| package {  import fl.core.UIComponent;  import com.snsoft.util.SkinsUtil;  import flash.display.MovieClip;  import fl.core.InvalidationType;  import flash.text.TextField;  import flash.events.TouchEvent;  import flash.events.Event;  import fl.livepreview.LivePreviewParent;  import com.snsoft.map.util.MapUtil;  import flash.display.Sprite;  import fl.events.ComponentEvent;  import flash.utils.getDefinitionByName;  public class ViewTest extends UIComponent {  private var mc:MovieClip;  private var \_varTitle:Array;  private var tfds:Sprite;  private var lp:LivePreviewParent;  public function set varTitle(varTitle:Array):void {  this.\_varTitle = varTitle;  this.drawNow();//设置可视属性时重绘组件，不然无法实时显示。  }  public function get varTitle():Array {  return this.\_varTitle;  }  public function ViewTest() {  }  override protected function configUI():void {  mc = com.snsoft.util.SkinsUtil.createSkinByName("View\_skin");  this.addChild(mc);  tfds = new Sprite();  this.addChild(tfds);  this.invalidate(InvalidationType.ALL,true);  this.invalidate(InvalidationType.SIZE,true);  super.configUI();  }  private function handler(e:Event):void {  this.drawNow();  }  /\*\*  \*  \*  \*/  override protected function draw():void {  this.mc.width = this.width;  this.mc.height = this.height;  com.snsoft.map.util.MapUtil.deleteAllChild(tfds);  if (this.varTitle != null) {  for (var i:int =0; i<this.varTitle.length; i++) {  var tfd:TextField = new TextField();  tfd.text = (this.varTitle[i] as String);  var cmc:MovieClip = com.snsoft.util.SkinsUtil.createSkinByName("View\_skin");  cmc.height = 20;  cmc.width = this.width;  cmc.y = i \* 20;  tfd.width = this.width;  tfd.y = i \* 20;  this.tfds.addChild(cmc);  this.tfds.addChild(tfd);  }  }  }  }  } |

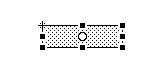
新建ViewTest.fla:



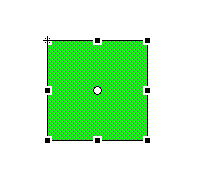
打开ViewTest.fla的库，建立如下目录和影片剪辑:



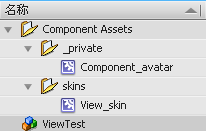
Component\_avatar 是组件在编辑状态下的辅助边框：

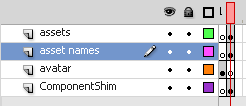


View\_skin是组件显示区域，也可以说是背景：



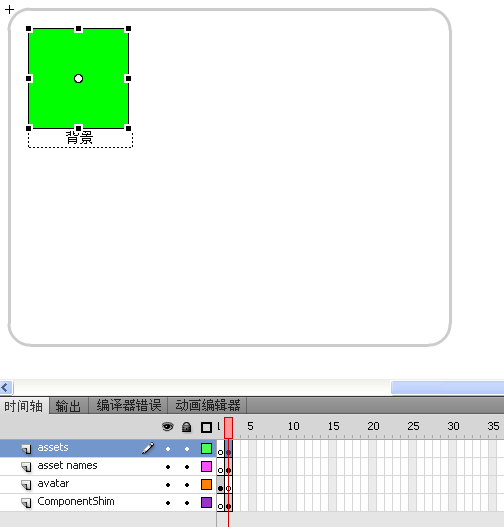
新建MovieClip 命名为ViewTest (和ViewTest.fla名称无关，可用其它名称)



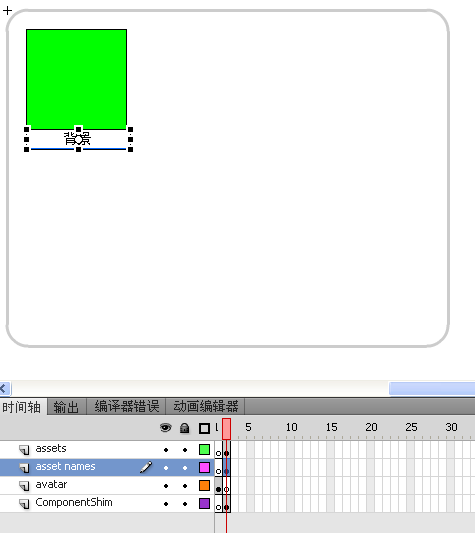


每层每帧的子对象分别为：

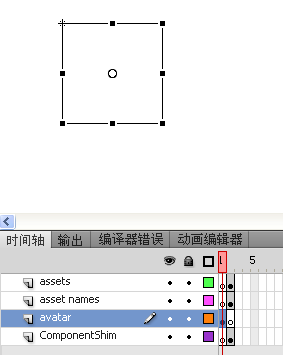
View\_skin



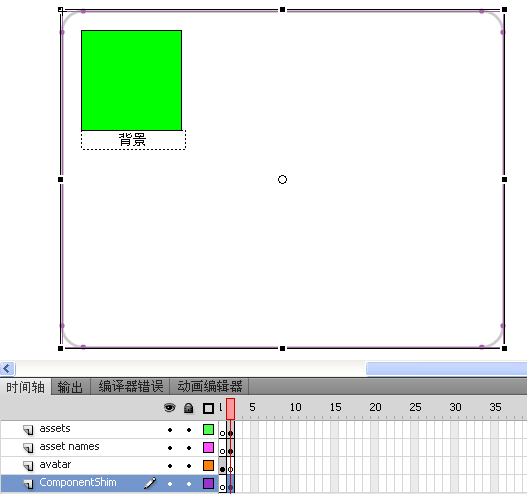
说明文字（随便写，不写也行）

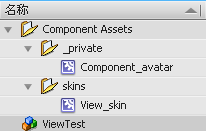


Component\_avatar

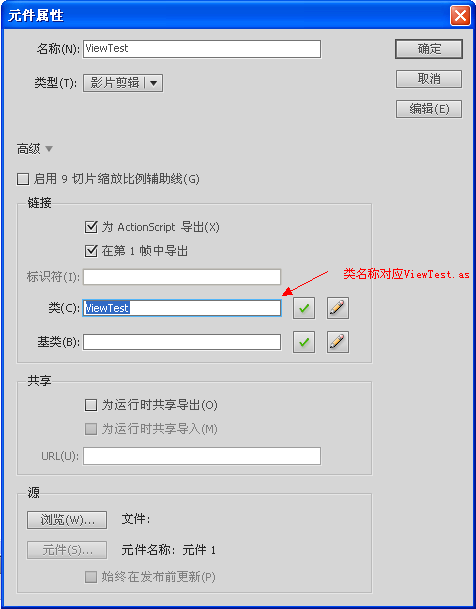


组件皮肤框（没什么用，就是为了说明框里面的是皮肤skin）：

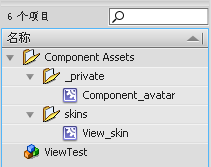




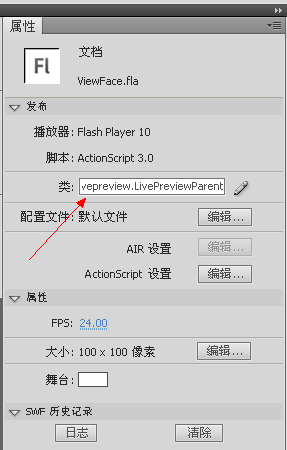
ViewTest右键属性：



新建 ViewFace.fla，把ViewTest.fla库里面所有项目拷到ViewFace.fla库里:



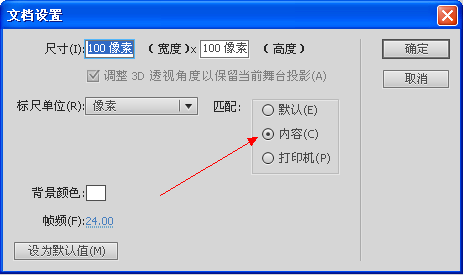
ViewFace.fla 点击场景，修改文档类为fl.livepreview.LivePreviewParent



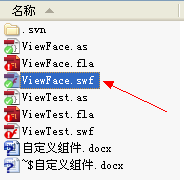
点击修改->文档



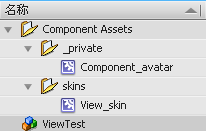
选中“内容”项



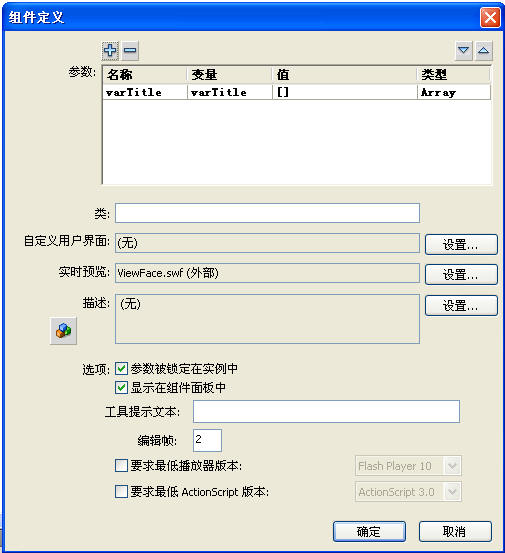
发布生成ViewFace.swf



打开ViewTest.fla

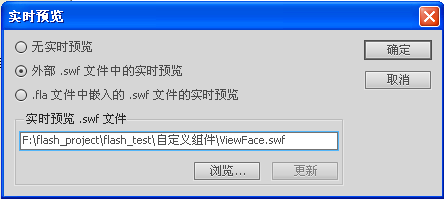


ViewTest右键属性，增加参数varTitle 对应ViewTest中的 varTitle :





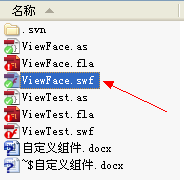
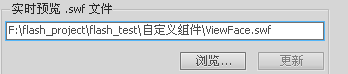
点击设置



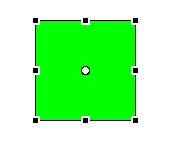
选中第二项



点击浏览，选中刚才生成的ViewFace.swf



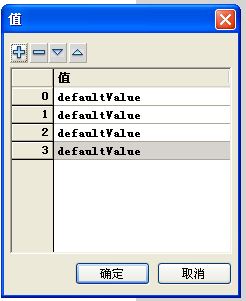
把ViewFace 拖到场景中，点击选中



属性面版



点击



点击增加几个列表项，然后确定：



看，自定义组件，并且实时显示成功。

注意

虽然组件是成功的，但为了简单易懂，例子里没有用标准的skin机制，而我是

mc = com.snsoft.util.SkinsUtil.createSkinByName("View\_skin");

这是我定义一个类，从文档中动态加载一个类并创建对象。

|  |
| --- |
| **package** com.snsoft.util  {  **import** flash.display.MovieClip;  **import** flash.utils.getDefinitionByName;  **public** **class** SkinsUtil  {  **public** **function** SkinsUtil()  {  }    /\*\*  \* 通过皮肤名称动态创建皮肤对象  \* @param skinName 皮肤名称  \* @return  \*  \*/  **public** **static** **function** createSkinByName(skinName:String):MovieClip {  **var** main:MovieClip = **new** MovieClip();  **var** mc:MovieClip;  **try** {  **var** MClass:Class = getDefinitionByName(skinName) **as** Class;  mc = **new** MClass() **as** MovieClip;  } **catch** (e:Error) {  **trace**(**" error SkinsUtil.createSkinByName() 动态加载找不到类："** +skinName);  }  **return** mc;  }  }  } |