zAI 1.20 的非常陷阱 bug 修复说明：当浮点遭遇整数

假如我们图片的整数尺寸为:128*128
对应浮点尺度是:1*1

当我们的坐标系是 X:64 Y:64，将它换算浮点尺度
我们的常规写法：x:64/128,y:64/128
其实这种写法是错误的，并且非常常见，在 graphics32,fmx,vcl 各种库中都经常遇这类换算方式！

因为标准库的整数计数，都会将 0 作为起始计数，而在浮点中，没有 0 起始计数的概念
正确的写法：x:64/(128-1),y:64/(128-1)
这样计算浮点才能正确映射到准确的内存地址去

图形学的普片开发原则是：看起来没问题，多半没问题。
其实问题很严重，轻则，就像 zAI 早期版本，画图总有一点走样的感觉，重则，产生 range check 异常，而这类原因往往又很难查，这是因为换算公式看起来没问题而被忽略导致的。

其实这种错误本质上是一种计算机机制的陷进，属于冯洛伊曼陷进。

zAI 在 1.20 完整已经解决了该问题。因此也产生几个很抽象的命名，诸如
Width=整数机制宽
Width0=浮点机制宽
Width0i=讲浮点机制宽转成的整数

```
property Width: Integer read FWidth;
property Height: Integer read FHeight;
property Width0: TGeoFloat read GetWidth0;
property Height0: TGeoFloat read GetHeight0;
property Width0i: Integer read GetWidth0i;
property Height0i: Integer read GetHeight0i;
```

如果我们使用了浮点尺度，诸如三角，投影，填充，注意区分 0 计数器和浮点换算的陷进

诸如，在 MemoryRaster_Vertex.inc 这类内置库中，我们会看到很多转来转去整数和浮点，它们并不是要故意转来转去，而是要得到正确的结果，处于区分尺度空间，才有了这种转来转去的代码，全都放在一起，很容易搞乱，变得反人类。

2019-7
By.qq600585