Tutorial 3:

Welcome back!, this time we’ll cover position types.

Remember the weird “posinframex” and “posinframey” variables from the first tutorials we skipped back then? Well, we’re going to talk about those in this tutorial.

[CODE]

square.posinframex=px;

square.posinframey=px;

[/CODE]

Let’s get some terms and macros down. Uiz has quite a few of its own macros.. The one we’ve seen so far is “px”. What px stands for is pixels. Just “pixels”.

What “posinframex/y” does is it changes the way “posvalx/y” behave and work. There are a few different basic modes:

-dp: stands for density pixel.

-fc: stands for factor.

-uiz\_snapleft

-uiz\_snapright

-uiz\_top

-uiz\_bottom

-uiz\_center

-uiz\_fill

**DP:** This means “density pixel” and those who have developed before on mobile platforms might be familiar with this term (also know as dpi). If you’ve never heard of dpi before, look at the beginners tutorial.

Example using dp:

(See example 7)

[CODE]

square.posinframex=dp;

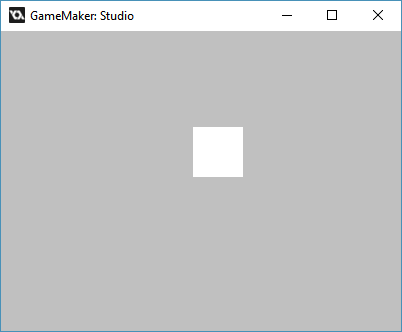
square.posinframey=dp;

square.posvalx=2;

square.posvaly=1;

[/CODE]

This gives:



**FC:** Fc can be very useful when working with the scaling, animating, etc. of objects. What it does is it takes the size of your parent (which is the size of you window/screen when no parent has been set), and when it has that size, it multiplies that size by a given value and uses that value. For example, if we want to put an object in the middle of the screen, we’d have to give it an fc value of 0.5 fc. This gives us the middle of the screen. An fc of 0 would give you the left side of the screen, and a value of 1 fc the right of the screen. You can say the same about an object when it has a parent, just replace “screen” with “parent”.

Example: (see example 8)

[CODE]

//initialize uiz

uiz\_init()

//create our square object

square=uiz\_c(obj\_uiZ\_square)

//setup some variables

square.posinframex=fc;

square.posinframey=fc;

square.posvalx=0.3;

square.posvaly=0.6;

square.posvalwtype=px;

square.posvalhtype=px;

square.posvalw=50;

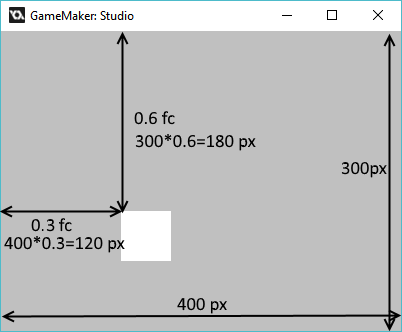
square.posvalh=50;

//fix our square object.

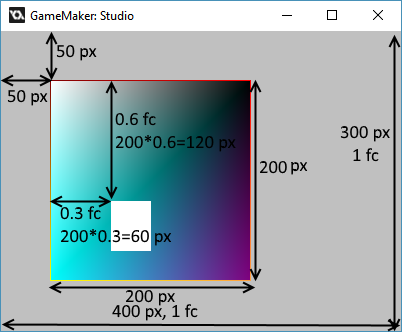
uiz\_fixgeneralpos(square)

[/CODE]

IMAGE 9:



This principle works on children the following way:



**Modes that disable the use of posvalx and/or posvaly**

Some values for posinframex and posinframey disable the use of posvalx and posvaly, as they aren’t needed for the purpose of what posinframex/y is set to. Some examples of these values are:

-uiz\_snapleft

-uiz\_snapright

-uiz\_top

-uiz\_bottom

-uiz\_center

-uiz\_fill.

These modes do pretty much what the name says.

EXAMPLE 11:

[CODE]

//initialize uiz

uiz\_init()

//create our square object

square=uiz\_c(obj\_uiZ\_square)

//setup some variables

square.posinframex=uiz\_snapright;

square.posinframey=uiz\_snapbottom;

square.posvalwtype=px;

square.posvalhtype=px;

square.posvalw=50;

square.posvalh=50;

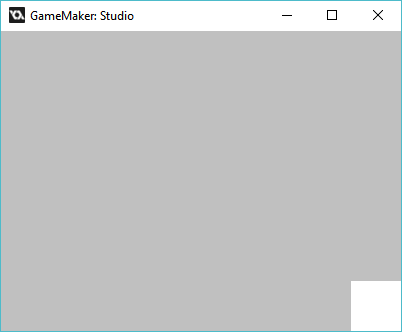
//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]

Which looks like:

IMAGE 8



Uiz\_center would look like this:

EXAMPLE 11:

[CODE]

//initialize uiz

uiz\_init()

//create our square object

square=uiz\_c(obj\_uiZ\_square)

//setup some variables

square.posinframex=uiz\_snapright;

square.posinframey=uiz\_snapbottom;

square.posvalwtype=px;

square.posvalhtype=px;

square.posvalw=50;

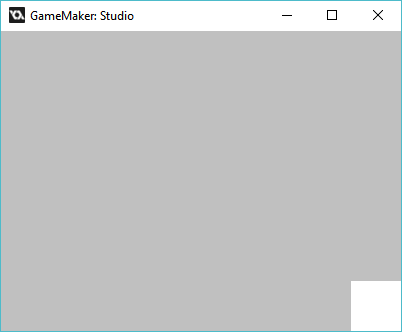
square.posvalh=50;

//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]

Which looks like:



**Uiz\_center**

There is also an macro called uiz\_center, which as the name suggests centers the object. A code like this will give following results:

EXAMPLE 12:

[CODE]

///Example 12:

//initialize uiz

uiz\_init()

//create our square object

square=uiz\_c(obj\_uiZ\_square)

//setup some variables

square.posinframex=uiz\_center;

square.posinframey=uiz\_center;

square.posvalwtype=px;

square.posvalhtype=px;

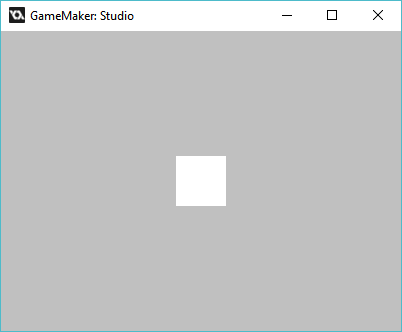
square.posvalw=50;

square.posvalh=50;

//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]



And at last but not least, uiz\_fill:

A special thing to note about uiz\_fill, is that it not only “disables” the posvalx and posvaly variables, it also disables posvalwtype, posvalw (for posinframex=uiz\_fill) and posvalhtype, posvalh (for posinframey=uiz\_fill)

[CODE]

//initialize uiz

uiz\_init()

//create our square object

square=uiz\_c(obj\_uiZ\_square)

//setup some variables

square.posinframex=uiz\_center;

square.posinframey=uiz\_fill;

square.posvalwtype=px;

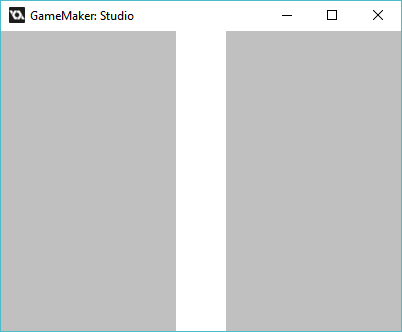
square.posvalw=50;

//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]

IMAGE 14:



Well, that was it for this relatively long tutorial, Next time we’ll look at how to use the types (you we used on posinframex/y) on posvalw/htype.