Tutorial 1:

Hi, welcome to this 14 part basic uiZ tutorial. This tutorial is intended for those who bought the uiZ asset over on the yoyo games marketplace. This is a very extensive tutorial for those who are beginner up to intermediate in using game maker. Still, a basic understanding of gml is required. For more advanced tutorials see:

-coming up

-coming up

-coming up.

Let dive right into the first tutorial, shall we? We start off by creating A simple square. There should be an object called “obj\_uiZ\_square”. Which is a simple object displaying a square of given color with an optional outline. To start off, create a new object and place it in a empty room. We need initialize uiZ. We can easily do this by calling the script “uiz\_init()” without any arguments inside the create event of the newly created object. To create an object, we can either place it inside the room, or just create using instance\_create(). No special script are needed for this. But there is one special script called uiz\_c(), which is a very simple version of instance\_create. Whereas instance\_create takes an object, x and a y, uiz\_c only takes an object. So calling uiz\_c(obj\_uiZ\_square) in a create event should be fine. Uiz\_c (and instance\_create as well) should return an “instance id” about which you can read more in the game maker manual. But practically it’s a reference to our created object.

Example 1:

[CODE]

//create event of a newly created object.

//initialize uiz

uiz\_init()

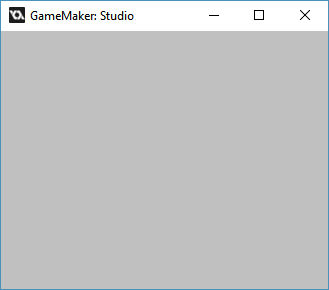
//create our square object

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

[/CODE]

We have created our square, and the result should look like this:

IMAGE 1

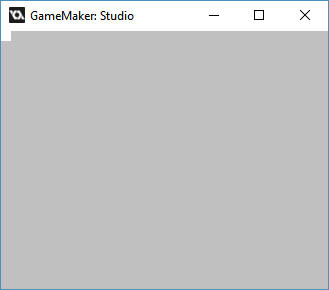


Not a lot of stuff… But, try to resize your window. (if this isn’t working for you go to: Global Game Setting -> Windows -> Graphics and make sure “Allow the player to resize the game window” is checked.)

After you resized your window just A bit, you should see something like this:

IMAGE 2:

.



You can now see a tiny white square on the top-left of your screen.

Why isn’t this square being drawn immediately? Good question. The answer is that we forgot to “fix” our object. Practically fixing is making sure that our object will we put into the right place and all object variables are the value of what they are supposed to be. The reason why things worked when we resized the window is because every uiZ object will be automatically “fixed” when we do that.

How do we “fix”? Very easy. Just call the script “uiz\_fixgeneralpos()”. This is where the handle to our object comes in. Remember? It was the “Square” part in “square=uiz\_c(obj\_uiZ\_square)” that now refers to the obj\_uiZ\_square object. We use this by putting inside the first argument of uiz\_fixgeneralpos, like this:

Example 2:

[CODE]

//create event of a newly created object.

//initialize uiz

uiz\_init()

//create our square object

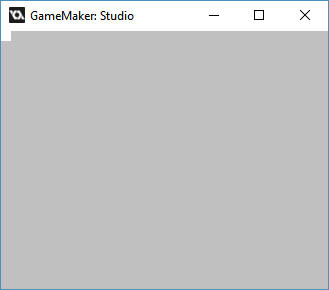
square=uiz\_c(obj\_uiZ\_square)

//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]

Now our square will look like this from the beginning of our run:

.

There are generally 4 parts in uiz object generation:

-Creation

-Setting parents

-variables/settings

-fixing.

“creation” and “fixing” is already covered and we’ll cover parents in a different tutorial.

Let’s focus on the variables/settings part. To get a good idea of all general uiz variables, please open op the uiz documentation. This should be somewhere in your projects included files if you downloaded uiz form the assets market place. It should be called something like “uiZ.chm”. Open it up and click the “build in object variables” article. You should now see how much “basic” variables we’ve got for our object. About half of them have something to do with the position and size of our object. One thing NOT to do is change the x and y values of the object. There is a special “positionmode” in uiz called uiz\_static, but that won’t be covered in the basic tutorials.

Just focus on the “posvalx” and “posvaly” variables. Before we do that, please set posinframex and posinframey to px so you’ll get this code:

Example 3:

[CODE]

//create event of a newly created object.

//initialize uiz

uiz\_init()

//create our square object

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

//setup some variables

square.posinframex=px;

square.posinframey=px;

//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]

Ok, not so fast there! Let take a close look the syntax: “square.posinframex=px”. First of all remember our 4 points? Creation, parents, variables and fixing. Things are supposed to go in that order, thus we put our line which changes some setting of our object after setting the parent, which we didn’t do, so we have to put if between our create part, and our fixing part. Secondly game maker allows for us to change variables of an object if we know what instance we’re using. This can be done by saying: “instance\_id.variable\_name=new\_value”. The “instance\_id” part can be something returned by the uiz\_c function, which we got inside our square variable.

Now let’s return to “posvalx” and “posvaly”. Let’s just try them out. If you understand the way of setting variables for objects in uiz, you should know that setting the posvalx value to something can be done using “square.posvalx=100”. Set posvaly to some other value, and you’ll get something like this:

Example 4:

[CODE]

//create event of a newly created object.

//initialize uiz

uiz\_init()

//create our square object

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

//setup some variables

square.posinframex=px;

square.posinframey=px;

square.posvalx=100;

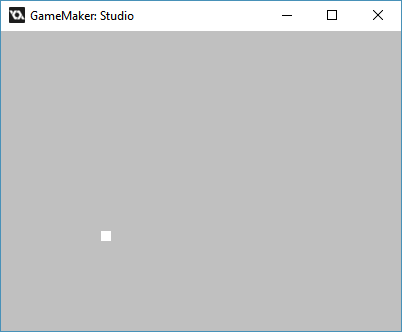
square.posvaly=200;

//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]

The result should look like:



Congrats! You placed the square in a different position.

Now let’s change the size of our object. Before we do this though, set the variables “posvalwtype” and “posvalhtype” both to “px”. You should know how to do this, because you are going to need this a lot when using uiz. With those variables coming later, we’ll now set the variables “posvalw” and “posvalh” to something like 50 by 50. We can do that like this:

Example 5:

[CODE]

//create event of a newly created object.

//initialize uiz

uiz\_init()

//create our square object

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

//setup some variables

square.posinframex=px;

square.posinframey=px;

square.posvalx=100;

square.posvaly=200;

square.posvalwtype=px;

square.posvalhtype=px;

square.posvalw=50;

square.posvalh=50;

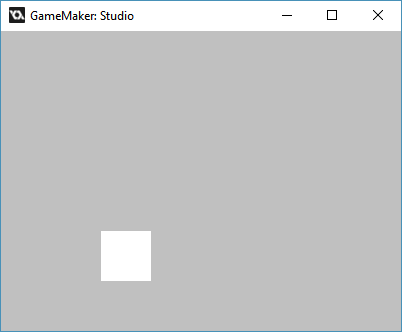
//fix our square object.

uiz\_fixgeneralpos(square)

[/CODE]

We’ll get something like this:

IMAGE 4:



Congratulations again!, you are now able to initialize uiz, create an object, set the position and sizing values for that object and you also know how to fix those objects. Just keep remembering the right syntax (“instance\_id.variable=new\_value”), and the little “Creation, parenting, variables and fixing“ row. Next tutorial we’ll be getting into parents.

If this tutorial was too easy for you, then consider following the advanced tutorial. Just know that you can always switch back and forth between the 2 versions since the same topics are covered every tutorial.