<u>light</u> <u>dark</u>	Harrier shoot shoot
Entity	Ursina cheat sheet
<u>Text</u>	This document lists most modules and classes in ursina. Each
Button	section is structured as follows:
mouse	Section 13 Structured as foctows.
raycaster	ClassName(BaseClass)
	module location
<pre>string_utilities</pre>	
ursinastuff	parameters
<u>curve</u>	How instantiate the class, ie. Button(text='',
<u>texture_importer</u>	**kwargs).
scene toyt	'**kwargs' in this case, means you can give it optional
<u>text</u>	keyword arguments.
window ursinamath	For example, Button('Start', scale=.25, color=color.blue, position=(1,.25)) also includes
camera	information on how big the button should be, its color
shader	and its position.
main	·
color	attributes
<u>input_handler</u>	Names of values we can get/set, sometimes followed by
<u>mesh_importer</u>	its starting value and a short explanation.
<u>duplicate</u>	For example, 'scale', 'color' and 'position' are
<u>build</u> <u>application</u>	attributes we gave the Button above. These are members of Entity, which Button class
<u>sequence</u>	inherits from, so the Button class can also access
<u>sequence</u>	these.
Vec3	
Empty	methods/functions
LoopingList	these ends with (), which means they are functions that
CubicBezier	can be called.
<u>MeshModes</u>	Also lists their parameters and default arguments.
Mesh Shadara	For example, Entity has a method called 'look_at()'.
Shader	You need to give it a
<u>Audio</u> Ursina	'target' (an Entity or position) to look at and optionally say
Color	which axis will be facing the target.
Vec4	milen axis mile so racing the tanget
HitInfo	example
Collider	·
<u>BoxCollider</u>	You can search the document with Ctrl+F for instant search
<u>SphereCollider</u>	results.
<u>MeshCollider</u>	
<u>Keys</u>	Entity()
Vec2 Texture	ursina/entity
<u>Light</u>	<del>,</del>
<u>DirectionalLight</u>	<pre>Entity(add_to_scene_entities=True, **kwargs)</pre>
PointLight	
AmbientLight	<pre>name = camel_to_snake(self.type)</pre>
<u>SpotLight</u>	enabled = True  # disabled entities will not be visible nor run
<u>Trigger</u>	code.
<u>FastMesh</u>	visible = True ignore = False
Func	eternal = False  # eternal entities does not get destroyed on
Sequence	scene.clear()
FileButton	ignore paused = False # if True, will still run when application
FileBrowser	is paused. useful when making a pause menu for example.
VideoRecorder	ignore_input = False
VideoRecorderUI	parent = scene # default parent is scene, which means it's in 3d
Cursor	space. to use UI space, set the parent to camera.ui instead.
<u>Draggable</u>	add_to_scene_entities = add_to_scene_entities # set to False to be
Tooltip	<pre>ignored by the engine, but still get rendered. model = None  # set model with model='model name' (without file</pre>
SynthGUI Manager Country	type extension)
<u>MemoryCounter</u> Panel	color = color.white
Tilemap	texture = None  # set model with texture='texture name'. requires
<u>Sprite</u>	a model to be set beforehand.
Sky	render_queue = 0
<u>DropdownMenuButton</u>	double_sided = False
DropdownMenu	collision = False # toggle collision without changing collider.
Animation	collider = None  # set to 'box'/'sphere'/'mesh' for auto fitted
FrameAnimation3d	collider.
<u>FirstPersonController</u>	<pre>scripts = list() # add with add_script(class_instance). will assign an 'entity' variable to the script.</pre>
<u>EditorCamera</u>	an entity variable to the script.  animations = list()
<u>Space</u> WindowPanel	hovered = False  # will return True if mouse hovers entity.
Node	origin = $Vec3(0,0,0)$
HOUG	-

```
position = Vec3(0,0,0) # right, up, forward. can also set self.x,
                                              self.y, self.z
                                              rotation = Vec3(0,0,0) # can also set self.rotation_x,
<u>light</u> dark
                                              self.rotation_y, self.rotation_z
                                              scale = Vec3(\overline{1},1,1)
                                                                      # can also set self.scale x, self.scale y,
<u>Entity</u>
                                              self.scale z
                                              line definition = None # returns a Traceback(filename, lineno,
<u>Text</u>
                                              function, code context, index).
Button
<u>mouse</u>
                                             world parent
raycaster
                                                                     # get class name.
                                              type
                                              types
                                                                     # get all class names including those this
string_utilities
                                              inhertits from.
                                              visible self
                                                                     # set visibility of self, without affecting
ursinastuff
                                              children.
<u>curve</u>
                                              origin x
texture importer
                                              origin y
scene
<u>text</u>
                                              origin z
<u>window</u>
                                              world_position
                                             world x
<u>ursinamath</u>
                                              world_y
<u>camera</u>
                                              world z
shader
<u>main</u>
                                             Х
color
                                             у
input_handler
                                             Z
mesh importer
                                             Χ
                                                                     # shortcut for int(entity.x)
                                                                     # shortcut for int(entity.y)
<u>duplicate</u>
                                              Υ
<u>build</u>
                                              Ζ
                                                                     # shortcut for int(entity.z)
                                              world rotation
<u>application</u>
                                             world rotation x
sequence
                                              world_rotation_y
Vec3
                                              world rotation z
                                              rotation x
<u>Empty</u>
                                              rotation_y
<u>LoopingList</u>
CubicBezier
                                              rotation z
                                              quaternion
<u>MeshModes</u>
                                              world scale
<u>Mesh</u>
                                             world_scale_x
<u>Shader</u>
                                             world_scale_y
Audio
                                             world scale z
Ursina
                                              scale_x
<u>Color</u>
                                              scale_y
<u>Vec4</u>
<u>HitInfo</u>
                                              scale z
<u>Collider</u>
                                              transform
                                                                     # get/set position, rotation and scale
<u>BoxCollider</u>
                                              world transform
                                                                     # get/set world position, world rotation and
                                             world scale
<u>SphereCollider</u>
<u>MeshCollider</u>
                                              forward
                                                                     # get forward direction.
                                              back
                                                                     # get backwards direction.
<u>Keys</u>
                                                                     # get right direction.
Vec2
                                              right
                                                                     # get left direction.
<u>Texture</u>
                                              left
                                                                     # get up direction.
Light
                                             uр
<u>DirectionalLight</u>
                                              down
                                                                     # get down direction.
PointLight
                                              screen position
                                                                     # get screen position(ui space) from world space.
<u>AmbientLight</u>
                                              shader
                                              texture scale
SpotLight
                                              texture offset
<u>Trigger</u>
                                                                     # if the texture is a tileset, say how many tiles
<u>FastMesh</u>
                                              tileset size
Func
                                              there are so it only use one tile of the texture, e.g. tileset size=
                                              [8,4]
<u>Sequence</u>
                                              tile coordinate
                                                                     # set the tile coordinate, starts in the lower
FileButton
                                              left.
                                              alpha
<u>FileBrowser</u>
VideoRecorder
                                              always on top
<u>VideoRecorderUI</u>
                                              unlit
<u>Cursor</u>
                                              billboard
                                                                     # set to True to make this Entity always face the
<u>Draggable</u>
                                              camera.
                                              model\_bounds
<u>Tooltip</u>
SynthGUI
                                              model center
                                              bounds
<u>MemoryCounter</u>
Panel
                                              children
                                              attributes
                                                                     # attribute names. used by duplicate().
Tilemap
<u>Sprite</u>
                                              enable()
<u>DropdownMenuButton</u>
                                              disable()
                                              set_shader_input(name, value)
<u>DropdownMenu</u>
                                              generate sphere map(size=512,
<u>Animation</u>
FrameAnimation3d
                                              name=f'sphere_map_{len(scene.entities)}')
<u>FirstPersonController</u>
                                              generate cube map(size=512, name=f'cube map {len(scene.entities)}')
EditorCamera
                                              reparent to(entity)
                                              get_position(relative_to=scene)
<u>Space</u>
<u>WindowPanel</u>
                                              set position(value, relative to=scene)
Node
                                              add_script(class_instance)
```

```
combine(analyze=False, auto destroy=True, ignore=[])
                                              flip faces()
                                              look_at(target, axis='forward')
<u>light</u> dark
                                              look_at_2d(target, axis='z')
                                              has ancestor(possible ancestor)
Entity
                                              animate(name, value, duration=.1, delay=0, curve=curve.in_expo,
                                              loop=False, resolution=None, interrupt='kill', time step=None,
<u>Text</u>
                                              auto destroy=True)
Button
                                              animate_position(value, duration=.1, **kwargs)
<u>mouse</u>
                                              animate rotation(value, duration=.1, **kwargs)
raycaster
                                              animate_scale(value, duration=.1, **kwargs)
string_utilities
                                              animate_{e}(value, duration=.1, delay=0, **kwargs)
                                              shake(duration=.2, magnitude=1, speed=.05, direction=(1,1))
ursinastuff
                                              animate_color(value, duration=.1, interrupt='finish', **kwargs)
<u>curve</u>
                                              fade_out(value=0, duration=.5, **kwargs)
fade_in(value=1, duration=.5, **kwargs)
texture importer
scene
                                              blink(value=color.clear, duration=.1, delay=0,
<u>text</u>
<u>window</u>
                                              curve=curve.in_expo_boomerang, interrupt='finish', **kwargs)
                                              intersects(traverse target=scene, ignore=(), debug=False)
<u>ursinamath</u>
<u>camera</u>
                                                e = Entity(model='quad', color=color.orange, position=(0,0,1), scale=1.5, rotation=(0,0,45), texture='brick')
shader
<u>main</u>
color
<u>input_handler</u>
                                                '''example of inheriting Entity'''
                                                class Player(Entity):
mesh importer
                                                     def __init__(self, **kwargs):
<u>duplicate</u>
                                                         super().__init__()
<u>build</u>
<u>application</u>
                                                          self.model='cube
                                                         self.color = color.red
sequence
                                                         self.scale y = 2
Vec3
                                                          for key, value in kwargs.items():
Empty
<u>LoopingList</u>
                                                              setattr(self, key, value)
CubicBezier
                                                     def input(self, key):
<u>MeshModes</u>
                                                          if key == 'space':
<u>Mesh</u>
                                                              self.animate x(2, duration=1)
<u>Shader</u>
<u>Audio</u>
Ursina
                                                     def update(self):
                                                          self.x += held_keys['d'] * time.dt * 10
Color
                                                          self.x -= held_keys['a'] * time.dt * 10
Vec4
HitInfo
                                                player = Player(x=-1)
<u>Collider</u>
BoxCollider
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
                                            Text(Entity)
Vec2
<u>Texture</u>
                                              ursina/text
<u>Light</u>
<u>DirectionalLight</u>
                                              Text(text='', start tag=start tag, end tag=end tag, ignore=True,
PointLight
                                              **kwargs)
<u>AmbientLight</u>
SpotLight
                                              size = Text.size
<u>Trigger</u>
                                              parent = camera.ui
<u>FastMesh</u>
                                              text nodes = list()
<u>Func</u>
                                              images = list()
Sequence
                                              origin = (-.5, .5)
                                              font = Text.default font
FileButton
                                              resolution = Text.default_resolution
<u>FileBrowser</u>
                                              line height = 1
VideoRecorder
                                              use_tags = True
start_tag = start_tag
<u>VideoRecorderUI</u>
Cursor
                                              end tag = end tag
<u>Draggable</u>
                                              text_colors = {'default' : color.text_color}
<u>Tooltip</u>
                                              tag = Text.start_tag+'default'+Text.end_tag
SynthGUI
                                              current color = self.text colors['default']
<u>MemoryCounter</u>
                                              scale_override = 1
Panel
                                              appear sequence = None # gets created when calling appear()
Tilemap
                                              text
<u>Sprite</u>
                                              color
                                                                      # sets the default color.
                                              width
                                                                      # gets the width of the widest line.
<u>DropdownMenuButton</u>
                                                                      # gets the height of the text
                                              heiaht
<u>DropdownMenu</u>
                                              lines
<u>Animation</u>
                                              wordwrap
                                                                      # set this to make the text wrap after a certain
FrameAnimation3d
                                              number of characters.
<u>FirstPersonController</u>
                                              background
EditorCamera
<u>Space</u>
                                              create text section(text, tag='', x=0, y=0)
<u>WindowPanel</u>
                                              align()
Node
```

```
create background(padding=size*2, radius=size,
                                             color=ursina.color.black66)
                                             appear(speed=.025, delay=0)
<u>light</u> dark
                                             get width(string, font=None)
                                               from ursina import *
Entity
<u>Text</u>
                                                 app = Ursina()
                                                 descr = dedent('''
Button
                                                      Rainstorm
<u>mouse</u>
                                                      Summon a rain storm to deal 5 water
<u>raycaster</u>
string_utilities
                                                      damage to everyone, test including yourself.
                                                      ursinastuff
                                               2134 2134 1234 1234 1234 1234
<u>curve</u>
texture importer
                                                      Lasts for 4 rounds.''').strip()
scene
                                                 Text.default resolution = 1080 * Text.size
<u>text</u>
<u>window</u>
                                                 test = Text(<del>text</del>=descr, wordwrap=30)
<u>ursinamath</u>
<u>camera</u>
shader
<u>main</u>
                                                 def input(key):
color
input_handler
                                                      if key == 'a':
                                                          print('a')
mesh importer
duplicate
                                                          test.text = '<default><image:file icon>
                                               <red><image:file_icon> test '
<u>build</u>
<u>application</u>
                                                          print('by', test.text)
sequence
                                                 window.fps_counter.enabled = False
Vec3
                                                 print('....', Text.get width('yolo'))
Empty
                                                 app.run()
LoopingList
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
                                           Button(Entity)
<u>Shader</u>
Audio
                                             ursina/prefabs/button
Ursina
Color
                                             Button(text='', radius=.1, **kwargs)
Vec4
<u>HitInfo</u>
                                             parent = camera.ui
<u>Collider</u>
                                             collider = 'box
<u>BoxCollider</u>
                                             disabled = False
<u>SphereCollider</u>
                                             color = Button.color
<u>MeshCollider</u>
                                             text entity = None
<u>Keys</u>
                                             highlight_color = self.color.tint(.2)
Vec2
                                             pressed color = self.color.tint(-.2)
<u>Texture</u>
                                             highlight scale = 1
                                                                       # multiplier
<u>Light</u>
                                             pressed scale = 1
                                                                      # multiplier
<u>DirectionalLight</u>
                                             \overline{\text{original}} scale = self.scale
PointLight
                                             icon = None
<u>AmbientLight</u>
                                             text
SpotLight
                                             text_origin
<u>Trigger</u>
                                             text color
<u>FastMesh</u>
Func
                                             input(key)
<u>Sequence</u>
                                             on mouse enter()
                                             on mouse exit()
FileButton
                                             on_click()
<u>FileBrowser</u>
                                             fit to text(radius=.1)
VideoRecorder
<u>VideoRecorderUI</u>
                                               b = Button(text='hello world!', color=color.azure, icon='sword',
<u>Cursor</u>
                                               scale=.25, text origin=(-.5,0))
Draggable
                                               b.on click = application.quit # assign a function to the button.
<u>Tooltip</u>
                                               b.tooltip = Tooltip('exit')
SynthGUI
<u>MemoryCounter</u>
Panel
Tilemap
<u>Sprite</u>
                                           mouse
                                             ursina/mouse
<u>DropdownMenuButton</u>
<u>DropdownMenu</u>
                                             enabled = False
<u>Animation</u>
                                             visible = True
FrameAnimation3d
                                             locked = False
<u>FirstPersonController</u>
                                             position = Vec3(0,0,0)
EditorCamera
                                             delta = Vec3(0,0,0)
<u>Space</u>
                                             prev x = 0
<u>WindowPanel</u>
                                             prev_y = 0
Node
```

	start_x = 0 start_y = 0	
	<pre>velocity = Vec3(0,</pre>	0.0)
<u>light</u> <u>dark</u>	prev_click_time =	
	<pre>prev_click_pos = N</pre>	
<u>Entity</u> Text	<pre>double_click_dista double click movem</pre>	
Button		lone # returns the closest hovered entity with a
mouse	collider.	,
<u>raycaster</u>	left = False	
string utilities	right = False	
<pre>string_utilities ursinastuff</pre>	<pre>middle = False delta drag = Vec3(</pre>	0.0.0)
curve	update_step = 1	-1-1-1
<u>texture_importer</u>		scene # set this to None to disable collision with
scene toxt	<pre>scene, which might raycast = True</pre>	be a good idea if you have lots of colliders.
<u>text</u> window	collision = None	
<u>ursinamath</u>	collisions = list(	)
camera	enabled = True	
shader	X	
main color	y normal	# returns the normal of the polygon, in local
<u>input handler</u>	space.	" recurred the normal of the potygon, in total
mesh importer	world_normal	# returns the normal of the polygon, in world
<u>duplicate</u>	space.	
<u>build</u> <u>application</u>	point world point	<pre># returns the point hit, in local space # returns the point hit, in world space</pre>
sequence	wor tu_point	# Teturns the point hit, in worth space
	input(key)	
<u>Vec3</u>	update()	
Empty LoopingList	<pre>find_collision() unhover everything</pre>	not hit()
CubicBezier	uillovei_evei ytiiiilig	
MeshModes	Button(parent=sc	ene, text='a')
Mesh	1.5 1.1 ()	
<u>Shader</u> Audio	<pre>def update():     print(mouse)</pre>	position, mouse.point)
Ursina	print(mouse.	position, mouse.point/
Color	Cursor()	
Vec4	<pre>mouse.visible =</pre>	False
<u>HitInfo</u> Collider		
BoxCollider		
<u>SphereCollider</u>		
	raycaster	
<u>Keys</u>	<u>ursina/raycaster</u>	
<u>Vec2</u> Texture		
<u>Light</u>	distance(a, b)	
DirectionalLight		rection=(0,0,1), distance=inf,
PointLight AmbientLight	traverse_target=sc	ene, ignore=list(), debug=False)
SpotLight SpotLight		rection=(0,0,1), distance=9999, thickness=(1,1),
Trigger	<pre>raverse_target=sc raycast, but with</pre>	ene, ignore=list(), debug=False)  # similar to
FastMesh	raycast, but with	with and neight
Func Sequence	111	
<u>Sequence</u>		*origin*, in *direction*, with length *distance*
<u>FileButton</u>	and returns	ning information about what it hit. This ray will
FileBrowser		s with a collider.
<u>VideoRecorder</u> VideoRecorderUI	•	
Cursor		averse_target* to only be able to hit a specific
Draggable		hildren/descendants. nore* list to ignore certain entities.
Tooltip		True will draw the line on screen.
SynthGUI MemoryCounter	3	
Panel	Example where we	only move if a wall is not hit:
<u>Tilemap</u>		
<u>Sprite</u>		
<u>Sky</u> DropdownMenuButton	class Player( <mark>Ent</mark>	ity):
<u>DropdownMenu</u>	المعالين عمل	olf).
<u>Animation</u>	def update(s self.dir	etion = Vec3(
FrameAnimation3d		.forward * (held keys['w'] - held keys['s'])
<u>FirstPersonController</u>	+ se	lf.right * (held_keys['d'] - held_keys['a'])
EditorCamera Space		rmalized() # get the direction we're trying to
<u>WindowPanel</u>	walk in.	
<u>Node</u>		

origin = self.world position + (self.up\*.5) # the ray

```
should start slightly up from the ground so we can walk up slopes
                                                   or walk over small objects.
<u>light</u> dark
                                                             hit info = raycast(origin , self.direction, ignore=(self,),
                                                   distance=.5, debug=False)
Entity
                                                             if not hit_info.hit:
                                                                  \operatorname{self.position} += \operatorname{self.direction} * 5 * \operatorname{time.dt}
<u>Text</u>
Button
                                                   Player(model='cube', origin_y=-.5, color=color.orange)
<u>mouse</u>
                                                   wall_left = Entity(model='cube', collider='box', scale_y=3,
raycaster
                                                   origin_y=-.5, color=color.azure, x=-4)
string_utilities
                                                   wall_right = duplicate(wall_left, x=4)
ursinastuff
                                                   camera.y = 2
<u>curve</u>
texture importer
scene
<u>text</u>
<u>window</u>
                                               string utilities
\underline{\text{ursinamath}}
                                                 ursina/string utilities
<u>camera</u>
shader
<u>main</u>
                                                 camel to snake(value)
color
                                                 snake_to_camel(value)
<u>input_handler</u>
                                                 multireplace(string, replacements, ignore case=False)
mesh importer
                                                 printvar(var)
<u>duplicate</u>
                                                 print_info(str, *args)
<u>build</u>
                                                 print warning(str, *args)
<u>application</u>
sequence
                                                   print(camel_to_snake('CamelToSnake'))
                                                   print(snake_to_camel('snake_to_camel'))
printvar('test')
Vec3
<u>Empty</u>
LoopingList
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
                                               ursinastuff
<u>Shader</u>
                                                 ursinastuff
<u>Audio</u>
Ursina
Color
                                                 invoke(function, *args, **kwargs)
<u>Vec4</u>
                                                 destroy(entity, delay=0)
<u>HitInfo</u>
                                                 find sequence(name, file types, folders) # find frame 0, frame 1,
<u>Collider</u>
                                                 frame 2 and so on
<u>BoxCollider</u>
                                                 import_all_classes(path=application.asset_folder, debug=False)
<u>SphereCollider</u>
                                                 print on screen(text, position=(0,0), origin=(-.5,.5), scale=1,
<u>MeshCollider</u>
                                                 duration=1)
<u>Keys</u>
Vec2
                                                   def test func(item, x=None, y=None):
<u>Texture</u>
                                                        print(item, x, y)
Light
<u>DirectionalLight</u>
                                                   test func('test')
PointLight
                                                   invoke(test_func, 'test', delay=.1)
invoke(test_func, 'testl', 1, 2, delay=.2)
invoke(test_func, 'test2', x=1, y=2, delay=.3)
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
                                                   def input(key):
Func
                                                        if key == 'space':
<u>Sequence</u>
                                                             print on screen('debug message', position=(0,0), origin=
                                                   (0,0), scale=2)
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
Cursor
                                               curve
<u>Draggable</u>
<u>Tooltip</u>
                                                 ursina/curve
SynthGUI
<u>MemoryCounter</u>
Panel
                                                 linear(t)
Tilemap
                                                 in sine(t)
<u>Sprite</u>
                                                 out sine(t)
                                                 in_out_sine(t)
<u>DropdownMenuButton</u>
                                                 in quad(t)
<u>DropdownMenu</u>
                                                 out quad(t)
<u>Animation</u>
                                                 in_out_quad(t)
FrameAnimation3d
                                                 in cubic(t)
<u>FirstPersonController</u>
                                                 out cubic(t)
EditorCamera
                                                 in_out_cubic(t)
<u>Space</u>
                                                 in quart(t)
<u>WindowPanel</u>
                                                 out_quart(t)
Node
```

```
in out quart(t)
                                                  in quint(t)
                                                 out_quint(t)
<u>light</u> dark
                                                  in_out_quint(t)
                                                  in_expo(t)
                                                 out_expo(t)
Entity
                                                  in out expo(t)
<u>Text</u>
                                                 in_circ(t)
out_circ(t)
Button
<u>mouse</u>
raycaster
                                                  in out circ(t)
                                                  in_back(t, magnitude=1.70158)
                                                 out_back(t, magnitude=1.70158)
in_out_back(t, magnitude=1.70158)
string_utilities
ursinastuff
                                                  in_elastic(t, magnitude=.7)
<u>curve</u>
                                                 out_elastic(t, magnitude=.7)
in_out_elastic(t, magnitude=0.65)
texture importer
scene
                                                  out bounce(t)
<u>text</u>
<u>window</u>
                                                  in \overline{b}ounce(t)
                                                 in out bounce(t)
<u>ursinamath</u>
                                                  {e} boomerang(t)
<u>camera</u>
shader
                                                    '''Draws a sheet with every curve and its name'''
<u>main</u>
color
                                                    camera.orthographic = True
input_handler
mesh importer
                                                    camera.fov = 16
duplicate
                                                    camera.position = (9, 6)
                                                   window.color = color.black
<u>build</u>
application
sequence
                                                    for e in dir(curve):
                                                         try:
Vec3
                                                             item = getattr(curve, e)
print(item.__name__, ':', item(.75))
Empty
<u>LoopingList</u>
<u>CubicBezier</u>
                                                              curve renderer = \overline{Entity}(
                                                                   model=Mesh(vertices=[Vec3(i / 31, item(i / 31), 0) for
<u>MeshModes</u>
                                                    i in range(32)], mode='line', thickness=2),
<u>Mesh</u>
                                                              color=color.light_gray)
row = floor(i / 8)
<u>Shader</u>
<u>Audio</u>
Ursina
                                                              curve renderer.x = (i \% 8) * 2.5
                                                              curve\_renderer.y = row * 1.75
Color
Vec4
                                                              label = Text(parent=curve_renderer, text=item.__name__,
                                                   scale=8, color=color.gray, y=-.1)
<u>HitInfo</u>
<u>Collider</u>
                                                              i += 1
<u>BoxCollider</u>
                                                         except:
<u>SphereCollider</u>
                                                              pass
<u>MeshCollider</u>
                                                   c = CubicBezier(0, .5, 1, .5)
print('----', c.calculate(.23))
<u>Keys</u>
Vec2
<u>Texture</u>
<u>Light</u>
                                                    window.exit_button.visible = False
<u>DirectionalLight</u>
                                                   window.fps_counter.enabled = False
PointLight
<u>AmbientLight</u>
                                                   These are used by Entity when animating, like this:
SpotLight
                                                    e = Entity()
<u>Trigger</u>
<u>FastMesh</u>
                                                   e.animate_y(1, curve=curve.in_expo)
Func
<u>Sequence</u>
                                                    e2 = Entitv(x=1.5)
                                                    e2.animate y(1, curve=curve.CubicBezier(0,.7,1,.3))
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
Draggable
                                                texture importer
<u>Tooltip</u>
                                                  ursina/texture_importer
SynthGUI
<u>MemoryCounter</u>
                                                  file_types = ('.tif', '.jpg', '.jpeg', '.png', '.gif')
Panel
                                                  textureless = False
Tilemap
<u>Sprite</u>
                                                  load texture(name, path=None)
                                                 compress textures(name='')
DropdownMenuButton
<u>DropdownMenu</u>
                                                    Entity(model='quad', texture='white_cube')
<u>Animation</u>
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
                                                scene
<u>WindowPanel</u>
                                                  <u>ursina/scene</u>
Node
```

```
render = None
                                             world = None
<u>light</u> dark
                                              camera = None
                                             ui camera = None
                                              entities = []
<u>Entity</u>
                                             hidden = NodePath('hidden')
<u>Text</u>
                                              reflection_map_name = 'reflection_map_3'
Button
mouse
                                              fog color
                                              fog density
raycaster
string_utilities
                                              set_up()
                                             clear()
ursinastuff
<u>curve</u>
                                               e = Entity(model='plane', color=color.black, scale=100)
texture importer
                                               EditorCamera()
scene
<u>text</u>
                                               s = Sky()
<u>window</u>
                                               def input(key):
<u>ursinamath</u>
                                                    if key == 'l':
<u>camera</u>
                                                         for e in scene.entities:
shader
<u>main</u>
                                                             print(e.name)
color
<u>input_handler</u>
                                                    if key == 'd':
mesh importer
                                                         scene.clear()
<u>duplicate</u>
<u>build</u>
                                               scene.fog\_density = .1
                                                                                    # sets exponential density
                                               scene.fog_density = (50, 200) # sets linear density start and end
application
sequence
Vec3
Empty
                                            text
<u>LoopingList</u>
CubicBezier
                                             ursina/text
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
                                              get_width(string, font=None)
<u>Audio</u>
Ursina
                                               from ursina import *
Color
                                                  app = Ursina()
<u>Vec4</u>
                                                  descr = dedent('''
<u>HitInfo</u>
                                                      Rainstorm
<u>Collider</u>
                                                      Summon a rain storm to deal 5 water
<u>BoxCollider</u>
<u>SphereCollider</u>
                                                       damage to everyone, test including yourself.
<u>MeshCollider</u>
                                                       <u>Keys</u>
                                               2134 2134 1234 1234 1234 1234
Vec2
                                                      Lasts for 4 rounds.''').strip()
<u>Texture</u>
<u>Light</u>
                                                  Text.default resolution = 1080 * Text.size
<u>DirectionalLight</u>
                                                  test = Text(text=descr, wordwrap=30)
PointLight
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
                                                  def input(key):
Func
                                                      if key == 'a':
<u>Sequence</u>
                                                           print('a')
                                                           test.text = '<default><image:file icon>
FileButton
                                               <red><image:file_icon> test '
<u>FileBrowser</u>
                                                           print('by', test.text)
VideoRecorder
<u>VideoRecorderUI</u>
                                                  window.fps_counter.enabled = False
<u>Cursor</u>
                                                  print('....', Text.get_width('yolo'))
<u>Draggable</u>
                                                  app.run()
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
<u>Panel</u>
<u>Tilemap</u>
                                           window
<u>Sprite</u>
                                             ursina/window
<u>DropdownMenuButton</u>
                                              vsync = True
                                                              # can't be set during play
<u>DropdownMenu</u>
                                              show ursina splash = False
<u>Animation</u>
                                              title = application.asset_folder.name
FrameAnimation3d
                                              fullscreen size = Vec2(*self.screen resolution)
<u>FirstPersonController</u>
                                             windowed_size = self.fullscreen_size / 1.25
EditorCamera
                                             windowed_position = None  # gets set when entering fullscreen so
<u>Space</u>
                                             position will be correct when going back to windowed mode
<u>WindowPanel</u>
                                              forced_aspect_ratio = None # example: window.forced_aspect_ratio
Node
```

```
always on top = False
                                                 top = \overline{Vec2}(0, .5)
                                                 bottom = Vec2(0, -.5)
<u>light</u> dark
                                                 center = Vec2(0, 0)
                                                 color = color.dark_gray
Entity
                                                 render modes = ('default', 'wireframe', 'colliders', 'normals')
<u>Text</u>
                                                 render_mode = 'default'
Button
                                                 editor ui = None
<u>mouse</u>
                                                 left
raycaster
                                                 right
string_utilities
                                                 top_left
                                                 top right
ursinastuff
                                                 bottom_left
<u>curve</u>
texture importer
                                                 bottom right
                                                 position
scene
<u>text</u>
                                                 icon
<u>window</u>
<u>ursinamath</u>
                                                 late init()
                                                 center_on_screen()
<u>camera</u>
                                                 make_editor_gui()
                                                                          # called by main after setting up camera and
shader
                                                 application.development mode
<u>main</u>
                                                 update aspect ratio()
color
                                                 next render mode()
input_handler
mesh importer
duplicate
                                                   window.title = 'ursina'
<u>build</u>
application
                                                   camera.orthographic = True
                                                   camera.fov = 2
sequence
Vec3
Empty
LoopingList
                                               ursinamath
CubicBezier
<u>MeshModes</u>
                                                 <u>ursinamath</u>
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
                                                 distance(a, b)
Ursina
                                                 distance_2d(a, b)
Color
                                                 distance xz(a, b)
Vec4
                                                 lerp(a, \overline{b}, \dot{t})
<u>HitInfo</u>
                                                 inverselerp(a, b, t)
<u>Collider</u>
                                                 slerp(q1, q2, t)
clamp(value, floor, ceiling)
<u>BoxCollider</u>
<u>SphereCollider</u>
                                                 round_to_closest(value, step=0)
<u>MeshCollider</u>
                                                 rotate_point_2d(point, origin, deg)
chunk_list(l, chunk_size)
<u>Keys</u>
Vec2
                                                 size list()
<u>Texture</u>
                                                 sum(\overline{l})
<u>Light</u>
<u>DirectionalLight</u>
                                                   e1 = Entity(position = (0,0,0))
PointLight
                                                   e^2 = Entity(position = (0,1,1))
<u>AmbientLight</u>
                                                   distance(e1, e2)
SpotLight
                                                   distance_xz(e1, e2.position)
<u>Trigger</u>
<u>FastMesh</u>
                                                   between color = lerp(color.lime, color.magenta, .5)
Func
                                                   print(between color)
<u>Sequence</u>
                                                   print(lerp((0,0), (0,1), .5))
                                                   print(lerp(Vec2(0,0), Vec2(0,1), .5))
print(lerp([0,0], [0,1], .5))
FileButton
<u>FileBrowser</u>
VideoRecorder
                                                   print(round(Vec3(.38, .1351, 353.26), 2))
<u>VideoRecorderUI</u>
<u>Cursor</u>
Draggable
                                                   print(p, 'rotated ->', rotate point 2d(p, (0,0), 90))
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
<u>Panel</u>
Tilemap
                                               camera
<u>Sprite</u>
                                                 ursina/camera
DropdownMenuButton
                                                 parent = scene
<u>DropdownMenu</u>
                                                 name = 'camera'
<u>Animation</u>
                                                 eternal = True
FrameAnimation3d
                                                 ui_size = 40
<u>FirstPersonController</u>
                                                 ui = None
EditorCamera
                                                 fov = 40
<u>Space</u>
                                                 orthographic = False
<u>WindowPanel</u>
                                                 clip_plane_near
Node
```

size = self.windowed size

```
clip plane far
                                                aspect ratio
                                                shader
<u>light</u> dark
                                                set up()
Entity
                                                set_shader_input(name, value)
<u>Text</u>
Button
                                                  window.borderless = False
<u>mouse</u>
                                                  camera.orthographic = True
raycaster
string_utilities
                                                  e = Entity()
ursinastuff
                                                  e.model = 'quad'
                                                  e.color = color.random_color()
<u>curve</u>
texture importer
                                                  e.position = (-2, 0, 10)
scene
<u>text</u>
                                                  e = Entity()
<u>window</u>
                                                  e.model = 'quad'
                                                  e.color = color.random_color()
<u>ursinamath</u>
<u>camera</u>
                                                  e.position = (2, 0, 10)
shader
                                                  e = Entity()
<u>main</u>
                                                  e.model = 'quad'
color
<u>input_handler</u>
                                                  e.color = color.random_color()
mesh importer
                                                  e.position = (0, 0, 40)
duplicate
<u>build</u>
                                                  EditorCamera()
application
                                                  from ursina.shaders import camera grayscale shader
<u>sequence</u>
                                                  camera.shader = camera grayscale shader
Vec3
Empty
LoopingList
                                              shader
CubicBezier
<u>MeshModes</u>
                                                ursina/shader
<u>Mesh</u>
<u>Shader</u>
                                                uniform mat4 p3d_ModelViewProjectionMatrix;
<u>Audio</u>
                                                out vec2 uv;
Ursina
                                                void main() {
Color
Vec4
                                                uniform sampler2D tex;
<u>HitInfo</u>
                                                out vec4 color;
<u>Collider</u>
                                                void main() {
<u>BoxCollider</u>
                                                }
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
                                                  from time import perf_counter
Vec2
                                                  t = perf counter()
<u>Texture</u>
                                                  Entity(model='cube', shader=Shader())
<u>Light</u>
                                                  EditorCamera()
<u>DirectionalLight</u>
                                                  print('tttttttttttt', perf_counter() - t)
PointLight
                                                  def input(key):
<u>AmbientLight</u>
                                                       if held_keys['control'] and key == 'r':
SpotLight
                                                            reload_shaders()
<u>Trigger</u>
<u>FastMesh</u>
                                                  def reload shaders():
<u>Func</u>
                                                       for e in scene.entities:
<u>Sequence</u>
                                                            if hasattr(e, '_shader'):
    print('-----', e.shader)
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                              main
Draggable
                                                ursina/main
<u>Tooltip</u>
SynthGUI
                                                time.dt = 0
<u>MemoryCounter</u>
Panel
Tilemap
                                                  app = Ursina()
<u>Sprite</u>
                                                  app.run()
DropdownMenuButton
<u>DropdownMenu</u>
Animation
FrameAnimation3d
                                              color
<u>FirstPersonController</u>
EditorCamera
                                                ursina/color
<u>Space</u>
<u>WindowPanel</u>
                                                color = hsv
Node
```

```
color(0, 0, 1)
                                                 white =
                                                 smoke =
                                                                     color(0, 0, 0.96)
                                                 light_gray =
                                                                    color(0, 0, 0.75)
<u>light</u> dark
                                                 gray =
                                                                     color(0, 0, 0.5)
                                                                    color(0, 0, 0.25)
color(0, 0, 0)
                                                 dark gray =
                                                 black =
<u>Entity</u>
                                                 red =
                                                                     color(0, 1, 1)
<u>Text</u>
                                                                    color(60, 1, 1)
color(90, 1, 1)
                                                 vellow =
Button
mouse
                                                 lime =
                                                 green =
                                                                    color(120, 1, 1)
raycaster
                                                 turquoise =
                                                                    color(150, 1, 1)
string_utilities
                                                 cyan =
                                                                     color(180, 1, 1)
                                                                    color(210, 1, 1)
                                                 azure =
ursinastuff
                                                                    color(240, 1, 1)
                                                 blue =
<u>curve</u>
                                                 violet =
                                                                    color(270, 1, 1)
texture importer
                                                                    color(300, 1, 1)
                                                 magenta =
scene
                                                                    color(330, 1, 1)
<u>text</u>
                                                 pink =
                                                                    rgb(165, 42, 42)
rgb(128, 128, 0)
<u>window</u>
                                                 brown =
                                                 olive =
<u>ursinamath</u>
                                                                    rgb(255, 218, 185)
rgb(255, 215, 0)
rgb(250, 128, 114)
                                                 peach =
<u>camera</u>
                                                 gold =
shader
                                                 salmon =
<u>main</u>
                                                 clear =
                                                                    Color(0, 0, 0, 0)
color
                                                                    Color(1,1,1, 0.10)
Color(1,1,1, 0.33)
                                                 white10 =
<u>input_handler</u>
mesh importer
                                                 white33 =
                                                 white50 =
duplicate
                                                                    Color(1,1,1, 0.50)
                                                                    Color(1,1,1, 0.66)
<u>build</u>
                                                 white66 =
                                                 black10 =
                                                                     Color(0,0,0, 0.10)
application
                                                 black33 =
                                                                    Color(0,0,0, 0.33)
sequence
                                                 black50 =
                                                                     Color(0,0,0, 0.50)
                                                                    Color(0,0,0, 0.66)
Color(0,0,0, 0.90)
Vec3
                                                 black66 =
                                                 black90 =
Empty
                                                 text = smoke
LoopingList
CubicBezier
                                                 light text = smoke
                                                 dark_text = color(0, 0, .1)
<u>MeshModes</u>
                                                 text color = light text
<u>Mesh</u>
                                                 color_names = ('white', 'smoke', 'light_gray', 'gray', 'dark_gray',
<u>Shader</u>
                                                 'black',
<u>Audio</u>
                                                 colors = dict()
Ursina
Color
Vec4
                                                 hsv(h, s, v, a=1)
<u>HitInfo</u>
                                                 rgba(r, g, b, a=255)
                                                 rgb(r, g, b, a=255)
<u>Collider</u>
<u>BoxCollider</u>
                                                 to hsv(color)
                                                 hex(value)
<u>SphereCollider</u>
<u>MeshCollider</u>
                                                 brightness(color)
                                                 inverse(color)
<u>Keys</u>
Vec2
                                                 random_color()
                                                 tint(\overline{color}, amount=.2)
<u>Texture</u>
Light
<u>DirectionalLight</u>
                                                   print(color.brightness(color.blue))
PointLight
                                                   print( 3)
<u>AmbientLight</u>
                                                   p = Entity(x=-2)
SpotLight
                                                   for key in color.colors:
<u>Trigger</u>
<u>FastMesh</u>
                                                        print(key)
Func
                                                        b = Button(parent=p, model=Quad(0), color=color.colors[key],
<u>Sequence</u>
                                                   text=kev)
                                                        b.text entity.scale *= .5
FileButton
                                                   grid layout(p.children, max_x=8)
<u>FileBrowser</u>
VideoRecorder
                                                   for name in ('r', 'g', 'b', 'h', 's', 'v', 'brightness'):
    print(name + ':', getattr(color.random_color(), name))
<u>VideoRecorderUI</u>
Cursor
<u>Draggable</u>
<u>Tooltip</u>
                                                   e = Entity(model='cube', color=color.lime)
SynthGUI
                                                   print(e.color.name)
<u>MemoryCounter</u>
Panel
Tilemap
<u>Sprite</u>
                                               input handler
<u>DropdownMenuButton</u>
                                                 ursina/input handler
<u>DropdownMenu</u>
<u>Animation</u>
                                                 held keys = defaultdict(lambda: 0)
FrameAnimation3d
                                                 rebinds = dict()
<u>FirstPersonController</u>
EditorCamera
                                                 bind(original_key, alternative_key)
<u>Space</u>
                                                 unbind(key)
<u>WindowPanel</u>
                                                 rebind(to key, from key)
Node
```

```
input(key)
                                                input_handler.rebind('z', 'w') # 'z'-key will now be registered as
<u>light</u> dark
                                                 'w'-key
                                                def test():
Entity
                                                    print('----')
<u>Text</u>
                                                def input(key):
Button
<u>mouse</u>
                                                     print(key)
                                                     if key == 'left mouse down':
raycaster
                                                         print('pressed left mouse button')
string_utilities
ursinastuff
                                                     if key == Keys.left mouse down: # same as above, but with
                                                Keys enum.
<u>curve</u>
texture importer
                                                         print('pressed left mouse button')
scene
<u>text</u>
<u>window</u>
                                                def update():
<u>ursinamath</u>
                                                     for key, value in held_keys.items():
                                                         if value != 0:
<u>camera</u>
                                                              print(key, value)
shader
<u>main</u>
color
input_handler
mesh importer
                                            mesh importer
duplicate
<u>build</u>
                                              ursina/mesh importer
<u>application</u>
sequence
                                              blender_scenes = dict()
Vec3
                                              load_model(name, path=application.asset_folder, file_types=('.bam',
'.ursinamesh', '.obj', '.glb', '.gltf', '.blend'),
Empty
LoopingList
                                              use deepcopy=False)
CubicBezier
                                              load blender scene(name, path=application.asset folder, load=True,
<u>MeshModes</u>
                                              reload=False, skip hidden=True, models only=False)
<u>Mesh</u>
                                              get_blender(blend_file) # try to get a matching blender version in
<u>Shader</u>
                                              case we have multiple blender version installed
<u>Audio</u>
                                              compress models(path=None,
Ursina
                                              outpath=application.compressed_models_folder, name='*')
Color
                                              obj to ursinamesh()
<u>Vec4</u>
                                              compress models fast(model name=None, write to disk=False)
<u>HitInfo</u>
                                              ursina_mesh_to_obj(mesh, name=''
<u>Collider</u>
                                              out path=application.compressed models folder, max decimals=3)
<u>BoxCollider</u>
                                              compress_internal()
<u>SphereCollider</u>
<u>MeshCollider</u>
                                                application.asset folder = Path(r'''C:\Users\Petter\Downloads''')
<u>Keys</u>
                                                t = perf_counter()
Vec2
                                                Entity(model='untitled')
<u>Texture</u>
                                                print('----', perf counter() - t)
Light
<u>DirectionalLight</u>
<u>PointLight</u>
<u>AmbientLight</u>
                                                EditorCamera()
SpotLight
                                                Sky(texture='sky_sunset')
<u>Trigger</u>
<u>FastMesh</u>
Func
<u>Sequence</u>
                                            duplicate
FileButton
                                              ursina/duplicate
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
                                              duplicate(entity, copy children=True, **kwargs): # use a for loop
<u>Cursor</u>
                                              instead of duplicate() # use a for loop instead of duplicate() if
Draggable
                                              you can.
<u>Tooltip</u>
SynthGUI
                                                e = Button(parent=scene, scale=1, text='yolo')
<u>MemoryCounter</u>
                                                e^2 = duplicate(e, x=1.25)
Panel
                                                EditorCamera()
Tilemap
<u>Sprite</u>
DropdownMenuButton
<u>DropdownMenu</u>
                                            build
<u>Animation</u>
FrameAnimation3d
                                              ursina/build
<u>FirstPersonController</u>
EditorCamera
                                              python -m ursina.build
<u>Space</u>
<u>WindowPanel</u>
Node
```

open cmd at your project folder and run 'python -m ursina.build' to package your app for windows.

```
<u>light</u> dark
Entity
<u>Text</u>
                                              application
Button
mouse
                                                ursina/application
raycaster
                                                paused = False
string_utilities
                                                time scale = 1
ursinastuff
                                                sequences = list()
<u>curve</u>
                                                trace_entity_definition = False # enable to set
texture importer
                                                entity.line definition
scene
                                                package_folder = Path(_
                                                                            file ).parent
<u>text</u>
                                                blender paths = dict()
<u>window</u>
                                                development mode = True
<u>ursinamath</u>
                                                scenes folder = asset folder / 'scenes/'
<u>camera</u>
                                               scripts_folder = asset_folder / 'scripts/'
fonts_folder = asset_folder / 'fonts/'
shader
<u>main</u>
                                                compressed textures folder = asset folder / 'textures compressed/'
color
                                                compressed_models_folder = asset_folder / 'models_compressed/
input_handler
                                                                             # this will be set once the Ursina() is
                                                base = None
mesh importer
                                                created
duplicate
                                                                             # will be set my main if development mode
                                                hot_reloader = None
<u>build</u>
<u>application</u>
                                                pause()
sequence
                                                resume()
                                                quit()
Vec3
                                                load_settings(path=asset_folder / 'settings.py')
Empty
LoopingList
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
                                              sequence
<u>Shader</u>
                                                ursina/sequence
<u>Audio</u>
Ursina
                                               Wait = float
Color
<u>Vec4</u>
<u>HitInfo</u>
                                                  e = Entity(model='quad')
<u>Collider</u>
                                                  s = Sequence(
<u>BoxCollider</u>
<u>SphereCollider</u>
                                                       Func(print, 'one'),
<u>MeshCollider</u>
                                                       Func(e.fade_out, duration=1),
<u>Keys</u>
Vec2
                                                       Func(print, 'two'),
<u>Texture</u>
                                                       Func(e.fade_in, duration=1),
<u>Light</u>
                                                       loop=True
<u>DirectionalLight</u>
PointLight
<u>AmbientLight</u>
                                                  s.append(
SpotLight
                                                       Func(print, 'appended to sequence')
<u>Trigger</u>
<u>FastMesh</u>
Func
                                                  def input(key):
<u>Sequence</u>
                                                       actions = {'s' : s.start, 'f' : s.finish, 'p' : s.pause, 'r' :
                                                  s.resume}
FileButton
                                                       if key in actions:
<u>FileBrowser</u>
                                                           actions[key]()
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
<u>Draggable</u>
<u>Tooltip</u>
                                              Vec3(PandaVec3)
SynthGUI
<u>MemoryCounter</u>
                                                ursina/vec3
Panel
Tilemap
                                               Х
<u>Sprite</u>
                                               У
                                                Z
<u>DropdownMenuButton</u>
                                                ху
<u>DropdownMenu</u>
                                                ΧZ
<u>Animation</u>
                                                уz
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
                                                  a = Vec3(1,0,0) * 2
                                                  a = Vec3(1,0,1) * Vec3(2,1,2)
<u>Space</u>
<u>WindowPanel</u>
                                                  b = Vec3(1.252352324,0,1)
Node
```

```
b += Vec3(0.1)
```

```
<u>light</u> dark
                                                 Empty()
Entity
<u>Text</u>
                                                   ursinastuff
Button
<u>mouse</u>
                                                   Empty(**kwargs)
raycaster
string_utilities
                                                   invoke(function, *args, **kwargs)
ursinastuff
                                                   destroy(entity, delay=0)
<u>curve</u>
                                                   find_sequence(name, file_types, folders) # find frame_0, frame_1,
texture importer
                                                   frame 2 and so on
scene
                                                   import_all_classes(path=application.asset_folder, debug=False)
<u>text</u>
                                                   print on screen(text, position=(0,0), origin=(-.5,.5), scale=1,
<u>window</u>
                                                   duration=1)
\underline{\text{ursinamath}}
<u>camera</u>
                                                     def test func(item, x=None, y=None):
shader
                                                          print(item, x, y)
<u>main</u>
color
                                                     test_func('test')
                                                     invoke(test_func, 'test', delay=.1)
invoke(test_func, 'testl', 1, 2, delay=.2)
invoke(test_func, 'test2', x=1, y=2, delay=.3)
<u>input_handler</u>
mesh importer
<u>duplicate</u>
<u>build</u>
application
                                                     def input(key):
sequence
                                                          if key == 'space':
                                                               print on screen('debug message', position=(0,0), origin=
Vec3
                                                     (0,0), scale=2)
Empty
LoopingList
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
                                                 LoopingList(list)
<u>Shader</u>
                                                   ursinastuff
<u>Audio</u>
Ursina
Color
Vec4
                                                     def test_func(item, x=None, y=None):
<u>HitInfo</u>
                                                          print(item, x, y)
<u>Collider</u>
<u>BoxCollider</u>
                                                     test func('test')
<u>SphereCollider</u>
                                                     invoke(test_func, 'test', delay=.1)
invoke(test_func, 'testl', 1, 2, delay=.2)
invoke(test_func, 'test2', x=1, y=2, delay=.3)
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
                                                     def input(key):
Light
                                                          if key == 'space':
<u>DirectionalLight</u>
                                                               print on screen('debug message', position=(0,0), origin=
PointLight
                                                     (0,0), scale=2)
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
Func
                                                 CubicBezier
<u>Sequence</u>
                                                   ursina/curve
FileButton
<u>FileBrowser</u>
                                                   CubicBezier(a, b, c, d)
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                                  b = b
Draggable
                                                   c = c
<u>Tooltip</u>
SynthGUI
                                                   cx = 3.0 * a
<u>MemoryCounter</u>
                                                   bx = 3.0 * (c - a) - self.cx
                                                   ax = 1.0 - self.cx - self.bx
Panel
Tilemap
                                                   cy = 3.0 * b
                                                  by = 3.0 * (d - b) - self.cy

ay = 1.0 - self.cy - self.by
<u>Sprite</u>
DropdownMenuButton
<u>DropdownMenu</u>
                                                   sample curve x(t)
<u>Animation</u>
                                                   sample_curve_y(t)
FrameAnimation3d
                                                   sample curve derivative x(t)
<u>FirstPersonController</u>
                                                   calculate(x, epsilon=.0\overline{001})
EditorCamera
                                                   solve_curve_x(t, epsilon=.0001)
<u>Space</u>
<u>WindowPanel</u>
                                                     '''Draws a sheet with every curve and its name'''
Node
```

14 of 55

	<pre>camera.orthographic = True</pre>
	camera.fov = 16
<u>light</u> <u>dark</u>	camera.position = $(9, 6)$
E 111	window.color = color.black
Entity Text	i = 0
Button	for e in dir(curve):
mouse	try:
<u>raycaster</u>	<pre>item = getattr(curve, e)</pre>
	<pre>print(itemname, ':', item(.75))</pre>
<u>string_utilities</u> ursinastuff	<pre>curve_renderer = Entity(     model=Mesh(vertices=[Vec3(i / 31, item(i / 31), 0) for</pre>
<u>curve</u>	i in range(32)], mode='line', thickness=2),
texture importer	color=color.light gray)
scene	row = floor(i / 8)
text	$curve\_renderer.x = (i \% 8) * 2.5$
window what name the	curve_renderer.y = row * 1.75
<u>ursinamath</u> <u>camera</u>	<pre>label = Text(parent=curve_renderer, text=itemname, scale=8, color=color.gray, y=1)</pre>
shader	i += 1
main main	except:
<u>color</u>	pass
<u>input_handler</u>	Cubic Decision (O. F. 1. F.)
<pre>mesh_importer duplicate</pre>	<pre>c = CubicBezier(0, .5, 1, .5) print('', c.calculate(.23))</pre>
build	print(, c.catcatate(.23))
application	<pre>window.exit button.visible = False</pre>
<u>sequence</u>	<pre>window.fps_counter.enabled = False</pre>
Vec3	There are used by Entity when animating like this.
Empty	These are used by Entity when animating, like this:
<u>LoopingList</u>	e = Entity()
CubicBezier	e.animate_y(1, curve=curve.in_expo)
<u>MeshModes</u>	
Mesh	e2 = Entity(x=1.5)
<u>Shader</u> Audio	e2.animate_y(1, curve=curve.CubicBezier(0,.7,1,.3))
Ursina	
Color	
Vec4	
HitInfo	Marala Maralara / Finish
<u>Collider</u> BoxCollider	MeshModes(Enum)
<u>SphereCollider</u>	<u>ursina/fastmesh</u>
MeshCollider	triangle = 'triangle'
<u>Keys</u>	ngon = 'ngon'
<u>Vec2</u>	quad = 'quad'
Texture	line = 'line'
<u>Light</u> DirectionalLight	point = 'point'
PointLight	tristrip = 'tristrip'
AmbientLight	
<u>SpotLight</u>	
Trigger	
<u>FastMesh</u> Func	M 1 / )
Sequence	Mesh()
	<u>ursina/mesh</u>
<u>FileButton</u> FileBrowser	Mesh(vertices=None, triangles=None, colors=None, uvs=None,
VideoRecorder	normals=None, static=True, mode='triangle', thickness=1,
VideoRecorderUI	render_points_in_3d=True)
Cursor	
<u>Draggable</u>	name = 'mesh'
Tooltip	<pre>vertices = vertices triangles = triangles</pre>
<u>SynthGUI</u> MemoryCounter	colors = colors
Panel	uvs = uvs
Tilemap	normals = normals
<u>Sprite</u>	static = static
<u>Sky</u>	<pre>mode = mode thickness = thickness</pre>
<u>DropdownMenuButton</u>	render points in 3d = render points in 3d
<u>DropdownMenu</u> Animation	recipe
FrameAnimation3d	
FirstPersonController	<pre>generate() # call this after setting some of the variables to</pre>
<u>EditorCamera</u>	update it generate normals(smooth=True)
<u>Space</u> WindowPanel	colorize(left=color.white, right=color.blue, down=color.red,
	up=color.green, back=color.white, forward=color.white, smooth=True,
<u>Node</u>	

```
project uvs(aspect ratio=1, direction='forward')
                                                 clear(regenerate=True)
<u>light</u> dark
                                                 save(name='', path=application.compressed models folder)
                                                   verts = ((0,0,0), (1,0,0), (.5, 1, 0), (-.5,1,0))

tris = (1, 2, 0, 2, 3, 0)

uvs = ((1.0, 0.0), (0.0, 1.0), (0.0, 0.0), (1.0, 1.0))
Entity
<u>Text</u>
Button
                                                   norms = ((0, 0, -1),) * len(verts)
<u>mouse</u>
                                                   colors = (color.red, color.blue, color.lime, color.black)
raycaster
string_utilities
                                                   e = Entity(model=Mesh(vertices=verts, triangles=tris, uvs=uvs,
ursinastuff
                                                   normals=norms, colors=colors), scale=2)
<u>curve</u>
                                                   verts = (\text{Vec3}(0,0,0), \text{Vec3}(0,1,0), \text{Vec3}(1,1,0), \text{Vec3}(2,2,0), \text{Vec3}(0,3,0), \text{Vec3}(-2,3,0))
texture importer
scene
                                                   tris = ((0,1), (3,4,5))
<u>text</u>
<u>window</u>
                                                   lines = Entity(model=Mesh(vertices=verts, triangles=tris,
<u>ursinamath</u>
                                                   mode='line', thickness=4), color=color.cyan, z=-1)
<u>camera</u>
                                                   points = Entity(model=Mesh(vertices=verts, mode='point',
shader
                                                   thickness=.05), color=color.red, z=-1.01)
<u>main</u>
color
input_handler
mesh importer
duplicate
                                               Shader
<u>build</u>
<u>application</u>
                                                 ursina/shader
<u>sequence</u>
                                                 Shader(language=Panda3dShader.SL_GLSL, vertex=default_vertex_shader,
Vec3
                                                 fragment=default_fragment_shader, geometry='', **kwargs)
Empty
<u>LoopingList</u>
                                                 path = Path(_caller.filename)
CubicBezier
                                                 language = language
<u>MeshModes</u>
                                                 vertex = vertex
<u>Mesh</u>
                                                 fragment = fragment
<u>Shader</u>
                                                 geometry = geometry
<u>Audio</u>
                                                 entity = None
Ursina
                                                 default_input = dict()
Color
                                                 compiled = False
Vec4
<u>HitInfo</u>
                                                 compile()
<u>Collider</u>
<u>BoxCollider</u>
                                                   from time import perf_counter
<u>SphereCollider</u>
                                                   t = perf counter()
<u>MeshCollider</u>
                                                   Entity(model='cube', shader=Shader())
<u>Keys</u>
                                                   EditorCamera()
Vec2
                                                   print('tttttttttttt', perf counter() - t)
<u>Texture</u>
                                                   def input(key):
Light
                                                        if held_keys['control'] and key == 'r':
<u>DirectionalLight</u>
                                                             reload shaders()
PointLight
<u>AmbientLight</u>
                                                   def reload_shaders():
SpotLight
                                                        for e in scene.entities:
<u>Trigger</u>
                                                             if hasattr(e, '_shader'):
    print('-----', e.shader)
<u>FastMesh</u>
<u>Func</u>
<u>Sequence</u>
FileButton
<u>FileBrowser</u>
                                               Audio (Entity)
VideoRecorder
                                                 ursina/audio
<u>VideoRecorderUI</u>
Cursor
                                                 Audio(sound_file_name='', autoplay=True, auto_destroy=False,
Draggable
                                                 **kwargs)
<u>Tooltip</u>
SynthGUI
                                                 volume = 1
<u>MemoryCounter</u>
                                                 pitch = 1
Panel
                                                 balance = 0
Tilemap
                                                 loop = False
<u>Sprite</u>
                                                 loops = 1
                                                 autoplay = autoplay
<u>DropdownMenuButton</u>
                                                 auto destroy = auto destroy
<u>DropdownMenu</u>
                                                 clip
<u>Animation</u>
                                                 length
FrameAnimation3d
                                                 status
<u>FirstPersonController</u>
                                                 ready
EditorCamera
                                                 playing
<u>Space</u>
                                                 time
<u>WindowPanel</u>
Node
```

world space=True, strength=1)

```
play(start=0)
                                                  pause()
                                                  resume()
<u>light</u> dark
                                                  stop(destroy=True)
                                                  fade(value, duration=.5, delay=0, curve=curve.in_expo,
                                                  resolution=None, interrupt=True)
Entity
                                                  fade_in(value=1, duration=.5, delay=0, curve=curve.in_expo,
resolution=None, interrupt='finish',)
<u>Text</u>
Button
                                                  fade_out(value=0, duration=.5, delay=0, curve=curve.in_expo,
<u>mouse</u>
                                                  resolution=None, interrupt='finish',)
<u>raycaster</u>
string_utilities
                                                    from ursina import Ursina, printvar
ursinastuff
                                                    a = Audio('life_is_currency', pitch=1, loop=True, autoplay=True)
<u>curve</u>
                                                    print(a.clip)
texture importer
                                                    a.volume=0
scene
                                                    b = Audio(a.clip)
<u>text</u>
<u>window</u>
                                                    def input(key):
<u>ursinamath</u>
                                                         if key == 'f':
<u>camera</u>
shader
                                                              a.fade out(duration=4, curve=curve.linear)
<u>main</u>
                                                    def update():
color
<u>input_handler</u>
                                                         print(a.time)
mesh importer
duplicate
<u>build</u>
application
                                                Ursina()
sequence
                                                  ursina/main
Vec3
Empty
                                                  Ursina(**kwargs): # optional arguments: title, fullscreen, size,
<u>LoopingList</u>
                                                  forced_aspect_ratio, position, vsync, borderless, show_ursina_splash,
render_mode, development_mode, editor_ui_enabled
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
                                                  mouse = mouse
<u>Shader</u>
<u>Audio</u>
                                                  input up(key)
Ursina
                                                  input_hold(key)
Color
                                                  input(key)
Vec4
                                                  keystroke(key)
<u>HitInfo</u>
                                                  run()
<u>Collider</u>
<u>BoxCollider</u>
                                                    app = Ursina()
<u>SphereCollider</u>
                                                    app.run()
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
Light
                                                Color(Vec4)
<u>DirectionalLight</u>
                                                  ursina/color
PointLight
<u>AmbientLight</u>
                                                  Color(self,*p)
SpotLight
<u>Trigger</u>
                                                  name
<u>FastMesh</u>
<u>Func</u>
                                                  r
                                                  g
<u>Sequence</u>
                                                  b
                                                  а
FileButton
                                                  hsv
<u>FileBrowser</u>
                                                  h
VideoRecorder
                                                  s
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                                  brightness
Draggable
<u>Tooltip</u>
                                                  invert()
SynthGUI
                                                  tint(amount)
<u>MemoryCounter</u>
                                                  hsv(h, s, v, a=1)
Panel
                                                  rgba(r, g, b, a=255)
Tilemap
                                                  rgb(r, g, b, a=255)
<u>Sprite</u>
                                                  to hsv(color)
                                                  hex(value)
DropdownMenuButton
                                                  brightness(color)
<u>DropdownMenu</u>
                                                  inverse(color)
<u>Animation</u>
                                                  random color()
FrameAnimation3d
                                                  tint(\overline{color}, amount=.2)
<u>FirstPersonController</u>
EditorCamera
                                                    print(color.brightness(color.blue))
<u>Space</u>
                                                    print( 3)
<u>WindowPanel</u>
Node
```

Node

```
p = Entity(x=-2)
                                                for key in color.colors:
                                                     print(key)
<u>light</u> dark
                                                     b = Button(parent=p, model=Quad(0), color=color.colors[key],
                                                text=key)
<u>Entity</u>
                                                     b.text_entity.scale *= .5
<u>Text</u>
                                                grid layout(p.children, \max x=8)
Button
<u>mouse</u>
                                                for name in ('r', 'g', 'b', 'h', 's', 'v', 'brightness'):
raycaster
                                                     print(name + ':', getattr(color.random_color(), name))
string_utilities
                                                e = Entity(model='cube', color=color.lime)
ursinastuff
                                                print(e.color.name)
<u>curve</u>
texture importer
scene
<u>text</u>
<u>window</u>
                                            Vec4(PandaVec4)
<u>ursinamath</u>
<u>camera</u>
                                              ursina/vec4
shader
<u>main</u>
color
input_handler
                                                a = Vec4(1,0,0,0) * 2
mesh importer
                                                a = Vec4(1,0,1,1) * Vec4(2,1,2,3)
duplicate
                                                b = Vec4(1.252352324,0,1,.2)
<u>build</u>
                                                b += Vec_{4}(0,1)
application
sequence
Vec3
Empty
                                            HitInfo
<u>LoopingList</u>
                                              ursina/hit info
CubicBezier
<u>MeshModes</u>
                                              HitInfo(**kwarqs)
<u>Mesh</u>
<u>Shader</u>
                                              hit = None
<u>Audio</u>
                                              entity = None
Ursina
                                              point = None
Color
                                              world point = None
<u>Vec4</u>
                                              distance = math.inf
<u>HitInfo</u>
                                              normal = None
<u>Collider</u>
                                              world normal = None
<u>BoxCollider</u>
                                              hits = []
<u>SphereCollider</u>
                                              entities = []
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
<u>Light</u>
<u>DirectionalLight</u>
                                            Collider()
PointLight
<u>AmbientLight</u>
                                              ursina/collider
SpotLight
<u>Trigger</u>
                                              Collider()
<u>FastMesh</u>
Func
                                              visible
<u>Sequence</u>
                                              show()
FileButton
                                              hide()
<u>FileBrowser</u>
                                              remove()
VideoRecorder
<u>VideoRecorderUI</u>
                                                e = Entity(model='sphere', x=2)
<u>Cursor</u>
                                                e.collider = 'box'
                                                                                # add BoxCollider based on entity's
<u>Draggable</u>
                                                bounds.
<u>Tooltip</u>
                                                e.collider = 'sphere'
                                                                                # add SphereCollider based on entity's
SynthGUI
                                                bounds.
<u>MemoryCounter</u>
                                                e.collider = 'mesh'
                                                                                # add MeshCollider matching the
Panel
                                                entity's model.
Tilemap
                                                e.collider = 'file name'
                                                                                 # load a model and us it as
<u>Sprite</u>
                                                MeshCollider.
                                                e.collider = e.model
                                                                                 # copy target model/Mesh and use it as
<u>DropdownMenuButton</u>
                                                MeshCollider.
<u>DropdownMenu</u>
<u>Animation</u>
                                                e.collider = BoxCollider(e, center=Vec3(0,0,0), size=Vec3(1,1,1))
FrameAnimation3d
                                                # add BoxCollider at custom positions and size.
<u>FirstPersonController</u>
                                                e.collider = SphereCollider(e, center=Vec3(0,0,0), radius=.75)
EditorCamera
                                                # add SphereCollider at custom positions and size.
<u>Space</u>
                                                e.collider = MeshCollider(e, mesh=e.model, center=Vec3(0,0,0))
<u>WindowPanel</u>
                                                # add MeshCollider with custom shape and center.
```

```
<u>light</u> dark
<u>Entity</u>
<u>Text</u>
Button
<u>mouse</u>
<u>raycaster</u>
string_utilities
ursinastuff
<u>curve</u>
texture importer
scene
<u>text</u>
<u>window</u>
<u>ursinamath</u>
<u>camera</u>
shader
<u>main</u>
color
input_handler
mesh importer
<u>duplicate</u>
<u>build</u>
<u>application</u>
sequence
Vec3
Empty
<u>LoopingList</u>
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
Ursina
Color
<u>Vec4</u>
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
<u>Light</u>
<u>DirectionalLight</u>
PointLight
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
Func
<u>Sequence</u>
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
<u>Draggable</u>
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
Panel
Tilemap
<u>Sprite</u>
DropdownMenuButton
<u>DropdownMenu</u>
<u>Animation</u>
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
<u>WindowPanel</u>
Node
```

```
m = Pipe(base_shape=Circle(6), thicknesses=(1, .5))
   e = Button(parent=scene, model='cube', collider='mesh',
   color=color.red, highlight color=color.yellow)
   EditorCamera()
   def input(key):
       if key == 'c':
           e.collider = None
BoxCollider(Collider)
 ursina/collider
 BoxCollider(entity, center=(0,0,0), size=(1,1,1))
 shape = CollisionBox(Vec3(center[0], center[1], center[2]), size[0],
 size[1], size[2])
 collision node = CollisionNode('CollisionNode')
 node_path = entity.attachNewNode(self.collision_node)
 visible = False
   e = Entity(model='sphere', x=2)
   e.collider = 'box'
                               # add BoxCollider based on entity's
   bounds.
   e.collider = 'sphere'
                               # add SphereCollider based on entity's
   bounds.
   e.collider = 'mesh'
                               # add MeshCollider matching the
   entity's model.
   e.collider = 'file name'
                                # load a model and us it as
   MeshCollider.
   e.collider = e.model
                                # copy target model/Mesh and use it as
   MeshCollider.
   e.collider = BoxCollider(e, center=Vec3(0,0,0), size=Vec3(1,1,1))
   # add BoxCollider at custom positions and size.
   e.collider = SphereCollider(e, center=Vec3(0,0,0), radius=.75)
   # add SphereCollider at custom positions and size.
   e.collider = MeshCollider(e, mesh=e.model, center=Vec3(0,0,0))
   # add MeshCollider with custom shape and center.
   m = Pipe(base_shape=Circle(6), thicknesses=(1, .5))
e = Button(parent=scene, model='cube', collider='mesh',
   color=color.red, highlight color=color.yellow)
   EditorCamera()
   def input(key):
       if key == 'c':
           e.collider = None
SphereCollider(Collider)
 ursina/collider
 SphereCollider(entity, center=(0,0,0), radius=.5)
 shape = CollisionSphere(center[0], center[1], center[2], radius)
 node_path = entity.attachNewNode(CollisionNode('CollisionNode'))
 visi\overline{ble} = False
   e = Entity(model='sphere', x=2)
   e.collider = 'box'
                               # add BoxCollider based on entity's
   bounds.
   e.collider = 'sphere'
                               # add SphereCollider based on entity's
   bounds.
   e.collider = 'mesh'
                               # add MeshCollider matching the
   entity's model.
   e.collider = 'file name'
                                # load a model and us it as
   MeshCollider.
   e.collider = e.model
                                # copy target model/Mesh and use it as
```

MeshCollider.

```
e.collider = BoxCollider(e, center=Vec3(0,0,0), size=Vec3(1,1,1))
                                                # add BoxCollider at custom positions and size.
                                               e.collider = SphereCollider(e, center=Vec3(0,0,0), radius=.75)
                                               # add SphereCollider at custom positions and size.
<u>light</u> dark
                                                e.collider = MeshCollider(e, mesh=e.model, center=Vec3(0,0,0))
Entity
                                                # add MeshCollider with custom shape and center.
<u>Text</u>
                                               m = Pipe(base_shape=Circle(6), thicknesses=(1, .5))
Button
                                               e = Button(parent=scene, model='cube', collider='mesh',
<u>mouse</u>
                                                color=color.red, highlight color=color.yellow)
<u>raycaster</u>
string_utilities
                                                EditorCamera()
ursinastuff
                                               def input(key):
<u>curve</u>
                                                    if key == 'c':
texture importer
                                                         e.collider = None
scene
<u>text</u>
<u>window</u>
<u>ursinamath</u>
<u>camera</u>
                                            MeshCollider(Collider)
shader
<u>main</u>
                                              ursina/collider
color
input_handler
                                              MeshCollider(entity, mesh=None, center=(0,0,0))
mesh importer
<u>duplicate</u>
                                              node_path = entity.attachNewNode(CollisionNode('CollisionNode'))
<u>build</u>
                                              collision_polygons = list()
<u>application</u>
                                              visible = False
sequence
                                              remove()
Vec3
Empty
                                               e = Entity(model='sphere', x=2)
LoopingList
                                                e.collider = 'box'
                                                                                # add BoxCollider based on entity's
CubicBezier
                                               bounds.
<u>MeshModes</u>
                                                e.collider = 'sphere'
                                                                                # add SphereCollider based on entity's
<u>Mesh</u>
                                                bounds.
<u>Shader</u>
                                                e.collider = 'mesh'
                                                                                # add MeshCollider matching the
Audio
                                                entity's model.
Ursina
                                                e.collider = 'file_name'
                                                                                # load a model and us it as
<u>Color</u>
                                               MeshCollider.
<u>Vec4</u>
                                                e.collider = e.model
                                                                                # copy target model/Mesh and use it as
<u>HitInfo</u>
                                               MeshCollider.
<u>Collider</u>
<u>BoxCollider</u>
                                               e.collider = BoxCollider(e, center=Vec3(0,0,0), size=Vec3(1,1,1))
<u>SphereCollider</u>
                                               # add BoxCollider at custom positions and size.
<u>MeshCollider</u>
                                               e.collider = SphereCollider(e, center=Vec3(0,0,0), radius=.75)
<u>Keys</u>
                                                # add SphereCollider at custom positions and size.
Vec2
                                                e.collider = MeshCollider(e, mesh=e.model, center=Vec3(0,0,0))
<u>Texture</u>
                                                # add MeshCollider with custom shape and center.
<u>Light</u>
<u>DirectionalLight</u>
                                               m = Pipe(base shape=Circle(6), thicknesses=(1, .5))
PointLight
                                                e = Button(parent=scene, model='cube', collider='mesh',
<u>AmbientLight</u>
                                                color=color.red, highlight_color=color.yellow)
SpotLight
<u>Trigger</u>
                                               EditorCamera()
<u>FastMesh</u>
Func
                                               def input(key):
<u>Sequence</u>
                                                    if key == 'c':
                                                         e.collider = None
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
Cursor
                                            Keys (Enum)
<u>Draggable</u>
                                              ursina/input handler
<u>Tooltip</u>
SynthGUI
                                              left_mouse_down = 'left mouse down'
left_mouse_up = 'left mouse up'
<u>MemoryCounter</u>
Panel
                                              middle mouse down = 'middle mouse down'
Tilemap
                                              middle mouse up = 'middle mouse up'
<u>Sprite</u>
                                              right mouse down = 'right mouse down'
                                              right mouse up = 'right mouse up'
<u>DropdownMenuButton</u>
                                              double_click = 'double click'
<u>DropdownMenu</u>
                                             scroll_up = 'scroll up'
scroll_down = 'scroll down'
left_arrow = 'left arrow'
<u>Animation</u>
FrameAnimation3d
<u>FirstPersonController</u>
                                              left_arrow_up = 'left arrow up'
EditorCamera
                                              up_arrow = 'up arrow'
<u>Space</u>
                                              up arrow up = 'up arrow up'
<u>WindowPanel</u>
                                              down_arrow = 'down arrow'
Node
```

	down arrow up = 'down arrow up'
	right_arrow = 'right arrow'
	right_arrow_up = 'right arrow up'
<u>light</u> <u>dark</u>	<pre>left_control = 'left control' pight control = 'night control'</pre>
Entity	right_control = 'right control' left shift = 'left shift'
Text	right shift = 'right shift'
Button	left alt = 'left alt'
mouse	right alt = 'right alt'
raycaster	left control up = 'left control up'
	right control up = 'right control up'
<pre>string_utilities</pre>	left_shift_up = 'left shift up'
<u>ursinastuff</u>	right_shift_up = 'right shift up'
<u>curve</u>	left_alt_up = 'left alt up'
<u>texture_importer</u>	right_alt_up = 'right alt up'
scene	page_down = 'page down'
text	page_down_up = 'page down up'
window	page_up = 'page up'
<u>ursinamath</u>	<pre>page_up_up = 'page up up' ontor = 'ontor'</pre>
<u>camera</u> shader	enter = 'enter'
main	<pre>backspace = 'backspace' escape = 'escape'</pre>
color	tab = 'tab'
input handler	gamepad a = 'gamepad a'
mesh importer	gamepad a up = 'gamepad a up'
duplicate	gamepad b = 'gamepad b'
build	gamepad b up = 'gamepad b up'
application	gamepad x = 'gamepad x'
sequence	<pre>gamepad_x_up = 'gamepad x up'</pre>
	<pre>gamepad_y = 'gamepad y'</pre>
<u>Vec3</u>	<pre>gamepad_y_up = 'gamepad y up'</pre>
<u>Empty</u>	<pre>gamepad_left_stick = 'gamepad left stick'</pre>
LoopingList	<pre>gamepad_left_stick_up = 'gamepad left stick up'</pre>
CubicBezier	<pre>gamepad_right_stick = 'gamepad right stick'</pre>
<u>MeshModes</u>	<pre>gamepad_right_stick_up = 'gamepad right stick up'</pre>
<u>Mesh</u> Shader	gamepad_back = 'gamepad back' gamepad back up = 'gamepad back up'
Audio	gamepad_back_up = gamepad_back_up gamepad_start = 'gamepad start'
Ursina	gamepad dpad down = 'gamepad dpad down'
Color	gamepad_apad_down up = 'gamepad dpad down up'
Vec4	gamepad dpad up = 'gamepad dpad up'
HitInfo	gamepad dpad up up = 'gamepad dpad up up'
Collider	gamepad dpad left = 'gamepad dpad left'
BoxCollider	gamepad dpad left up = 'gamepad dpad left up'
SphereCollider	gamepad dpad right = 'gamepad dpad right'
MeshCollider	<pre>gamepad_dpad_right_up = 'gamepad dpad right up'</pre>
<u>Keys</u>	gamepad_left_shoulder = 'gamepad left shoulder'
<u>Vec2</u>	gamepad_left_shoulder_up = 'gamepad left shoulder up'
<u>Texture</u>	gamepad_right_shoulder = 'gamepad right shoulder'
<u>Light</u>	gamepad_right_shoulder_up = 'gamepad right shoulder up'
<u>DirectionalLight</u>	
PointLight	
AmbientLight	
<u>SpotLight</u> <u>Trigger</u>	
FastMesh	Vec2(PandaVec2)
Func	
Sequence	<u>ursina/vec2</u>
<u> </u>	v
<u>FileButton</u>	X
<u>FileBrowser</u>	У
<u>VideoRecorder</u>	
<u>VideoRecorderUI</u>	a = Vec2(1,1)
<u>Cursor</u>	print(a)
<u>Draggable</u>	print(round(a))
Tooltip	
SynthGUI Marrana Caranta a	
MemoryCounter Panel	
<u>Panel</u> Tilemap	<b>-</b>
Sprite	Texture()
Sky	<u>ursina/texture</u>
DropdownMenuButton	
DropdownMenu	Texture(value)
Animation	
FrameAnimation3d	<pre>filtering = Texture.default_filtering # None/'bilinear'/'mipmap</pre>
FirstPersonController	default: 'None'
<u>EditorCamera</u>	name
<u>Space</u>	size
<u>WindowPanel</u>	width height
Node	height

```
pixels
                                             repeat
<u>light</u> dark
                                             get_pixel(x, y)
                                             get pixels(start, end)
<u>Entity</u>
                                             set_pixel(x, y, color)
                                             apply()
<u>Text</u>
                                             save(path)
Button
mouse
<u>raycaster</u>
                                                    The Texture class rarely used manually but usually instantiated
string_utilities
                                                    when assigning a texture to an Entity
                                                    texture = Texture(path / PIL.Image / panda3d.core.Texture)
ursinastuff
<u>curve</u>
texture importer
                                                    A texture file can be a .png, .jpg or .psd.
                                                    If it's a .psd it and no compressed version exists, it will
scene
                                               compress it automatically.
<u>text</u>
<u>window</u>
                                               e = Entity(model='quad', texture='brick')
<u>ursinamath</u>
                                               e.texture.set_pixel(0, 2, color.blue)
<u>camera</u>
                                               e.texture.apply()
shader
<u>main</u>
color
input_handler
mesh importer
<u>duplicate</u>
                                           Light(Entity)
<u>build</u>
                                             ursina/lights
application
sequence
                                             Light(**kwargs)
Vec3
                                             color
Empty
LoopingList
CubicBezier
                                               from ursina.shaders import lit with shadows shader # you have to
<u>MeshModes</u>
                                               apply this shader to enties for them to receive shadows.
<u>Mesh</u>
                                               EditorCamera()
<u>Shader</u>
                                               Entity(model='plane', scale=10, color=color.gray,
<u>Audio</u>
                                               shader=lit with shadows shader)
Ursina
                                               Entity(model='cube', y=1, shader=lit_with_shadows_shader)
Color
                                               pivot = Entity()
<u>Vec4</u>
                                               DirectionalLight(parent=pivot, y=2, z=3, shadows=True)
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
                                           DirectionalLight(Light)
<u>Keys</u>
                                             ursina/lights
Vec2
<u>Texture</u>
                                             DirectionalLight(shadows=True, **kwargs)
<u>Light</u>
<u>DirectionalLight</u>
                                             shadow map resolution = Vec2(1024, 1024)
PointLight
                                             shadows
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
                                               from ursina.shaders import lit with shadows shader # you have to
<u>FastMesh</u>
                                               apply this shader to enties for them to receive shadows.
Func
                                               EditorCamera()
<u>Sequence</u>
                                               Entity(model='plane', scale=10, color=color.gray,
                                               shader=lit with shadows shader)
FileButton
                                               Entity(model='cube', y=1, shader=lit with shadows shader)
<u>FileBrowser</u>
                                               pivot = Entity()
VideoRecorder
                                               DirectionalLight(parent=pivot, y=2, z=3, shadows=True)
<u>VideoRecorderUI</u>
<u>Cursor</u>
<u>Draggable</u>
<u>Tooltip</u>
SynthGUI
                                           PointLight(Light)
<u>MemoryCounter</u>
Panel
                                             ursina/lights
Tilemap
<u>Sprite</u>
                                             PointLight(**kwargs)
<u>DropdownMenuButton</u>
<u>DropdownMenu</u>
<u>Animation</u>
                                               from ursina.shaders import lit_with_shadows_shader # you have to
FrameAnimation3d
                                               apply this shader to enties for them to receive shadows.
<u>FirstPersonController</u>
                                               EditorCamera()
EditorCamera
                                               Entity(model='plane', scale=10, color=color.gray,
<u>Space</u>
                                               shader=lit with shadows shader)
<u>WindowPanel</u>
                                               Entity(model='cube', y=1, shader=lit_with_shadows_shader)
Node
```

<u>Text</u>

mouse

<u>curve</u>

scene

<u>text</u>

<u>main</u> color

<u>build</u>

Vec3 **Empty** 

<u>Mesh</u>

<u>Audio</u>

Color

<u>Vec4</u>

<u>Keys</u>

Vec2

<u>Light</u>

Func

Panel

<u>Space</u>

Node

```
pivot = Entity()
                                               DirectionalLight(parent=pivot, y=2, z=3, shadows=True)
<u>light</u> dark
<u>Entity</u>
                                           AmbientLight(Light)
Button
                                             ursina/lights
raycaster
                                             AmbientLight(**kwargs)
string_utilities
ursinastuff
                                               from ursina.shaders import lit_with_shadows_shader # you have to
texture importer
                                               apply this shader to enties for them to receive shadows.
                                               EditorCamera()
                                               Entity(model='plane', scale=10, color=color.gray,
<u>window</u>
                                               shader=lit with shadows shader)
<u>ursinamath</u>
                                               Entity(model='cube', y=1, shader=lit with shadows shader)
<u>camera</u>
                                               pivot = Entity()
shader
                                               DirectionalLight(parent=pivot, y=2, z=3, shadows=True)
input_handler
mesh importer
duplicate
                                           SpotLight(Light)
                                             ursina/lights
<u>application</u>
sequence
                                             SpotLight(**kwargs)
LoopingList
                                              from \ ursina.shaders \ import \ lit\_with\_shadows\_shader \ \textit{\# you have to}
CubicBezier
                                               apply this shader to enties for them to receive shadows.
<u>MeshModes</u>
                                               EditorCamera()
                                               Entity(model='plane', scale=10, color=color.gray,
<u>Shader</u>
                                               shader=lit with shadows shader)
                                              Entity(model='cube', y=1, shader=lit_with_shadows_shader)
Ursina
                                               pivot = Entity()
                                               DirectionalLight(parent=pivot, y=2, z=3, shadows=True)
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
<u>SphereCollider</u>
                                           Trigger(Entity)
<u>MeshCollider</u>
                                             ursina/trigger
<u>Texture</u>
                                             Trigger(**kwargs)
<u>DirectionalLight</u>
                                             trigger_targets = None
PointLight
                                             radius = .5
<u>AmbientLight</u>
                                             triggerers = list()
SpotLight
                                             update_rate = 4
<u>Trigger</u>
<u>FastMesh</u>
                                             update()
<u>Sequence</u>
                                               player = Entity(model='cube', color=color.azure, scale=.05)
                                               def update():
FileButton
                                                   player.x += held_keys['d'] * time.dt * 2
<u>FileBrowser</u>
                                                   player.x -= held keys['a'] * time.dt * 2
VideoRecorder
<u>VideoRecorderUI</u>
                                               t = Trigger(trigger targets=(player,), x=1, model='sphere',
<u>Cursor</u>
                                              color=color(0,1,1,.5))
<u>Draggable</u>
                                              t.on_trigger_enter = Func(print, 'enter')
                                              t.on_trigger_exit = Func(print, 'exit')
t.on_trigger_stay = Func(print, 'stay')
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
Tilemap
<u>Sprite</u>
                                           FastMesh()
DropdownMenuButton
                                             ursina/fastmesh
<u>DropdownMenu</u>
<u>Animation</u>
                                             FastMesh(vertices=None, triangles=None, colors=None, uvs=None,
FrameAnimation3d
                                             normals=None, static=True, mode='triangle', thickness=1,
<u>FirstPersonController</u>
                                             render_points_in_3d=True)
EditorCamera
                                             name = 'mesh'
<u>WindowPanel</u>
                                             vertices = vertices
```

```
triangles = triangles
                                                colors = colors
                                                uvs = uvs
<u>light</u> dark
                                                normals = normals
                                                static = static
<u>Entity</u>
                                                mode = mode
                                                thickness = thickness
<u>Text</u>
                                                render points in 3d = render points in 3d
Button
mouse
raycaster
                                                                # call this after setting some of the variables to
                                                update it
string_utilities
                                                  from ursina import *
ursinastuff
<u>curve</u>
                                                       app = Ursina()
texture importer
scene
<u>text</u>
                                                   \text{verts} = ((0.5,\ 0.5,\ 0.0),\ (-0.5,\ 0.5,\ 0.0),\ (-0.5,\ -0.5,\ 0.0),\ (0.5,\ -0.5,\ 0.0),\ (0.5,\ 0.5,\ 0.0),\ (-0.5,\ -0.5,\ 0.0)) 
<u>window</u>
<u>ursinamath</u>
                                                       sphere_model = load_model('sphere')
<u>camera</u>
                                                       verts = sphere model.vertices
shader
                                                       norms =
<u>main</u>
                                                                  sphere model.normals
color
input_handler
                                                       flat_verts = []
                                                       [flat verts.extend(v) for v in verts]
mesh importer
                                                       flat_verts = array.array("f", flat_verts)#.tobytes()
duplicate
<u>build</u>
                                                       flat norms = []
application
                                                       [flat norms.extend(v) for v in norms]
sequence
                                                       flat norms = array.array("f", flat norms)#.tobytes()
Vec3
Empty
                                                       from time import perf_counter
LoopingList
CubicBezier
                                                       t = perf_counter()
<u>MeshModes</u>
                                                       fm = FastMesh(vertices=flat verts)
<u>Mesh</u>
                                                       print('-----fast mesh:', perf_counter() - t)
<u>Shader</u>
<u>Audio</u>
Ursina
                                                       t = perf counter()
                                                       m = Mesh(vertices=flat verts, fast=True)
Color
                                                       print('----old mesh:', perf_counter() - t)
<u>Vec4</u>
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
                                                       Entity(model=copy(fm), x=0)
<u>SphereCollider</u>
<u>MeshCollider</u>
                                                       EditorCamera()
                                                       app.run()
<u>Keys</u>
Vec2
<u>Texture</u>
<u>Light</u>
<u>DirectionalLight</u>
                                              Func()
PointLight
<u>AmbientLight</u>
                                                ursina/sequence
SpotLight
<u>Trigger</u>
                                                Func(func, *args, **kwargs)
<u>FastMesh</u>
Func
                                                func = func
<u>Sequence</u>
                                                args = args
                                                kwargs = kwargs
FileButton
                                                delay = 0
<u>FileBrowser</u>
                                                finished = False
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                                  e = Entity(model='quad')
<u>Draggable</u>
                                                  s = Sequence(
<u>Tooltip</u>
SynthGUI
                                                       Func(print, 'one'),
<u>MemoryCounter</u>
                                                       Func(e.fade_out, duration=1),
Panel
Tilemap
                                                       Func(print, 'two'),
<u>Sprite</u>
                                                       Func(e.fade_in, duration=1),
<u>Sky</u>
                                                       loop=True
<u>DropdownMenuButton</u>
<u>DropdownMenu</u>
Animation
                                                  s.append(
FrameAnimation3d
                                                       Func(print, 'appended to sequence')
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
                                                  def input(key):
<u>WindowPanel</u>
                                                       actions = {'s' : s.start, 'f' : s.finish, 'p' : s.pause, 'r' :
Node
```

```
s.resume}
                                                      if key in actions:
                                                           actions[key]()
<u>light</u> dark
Entity
<u>Text</u>
<u>Button</u>
                                              Sequence()
<u>mouse</u>
                                               ursina/sequence
raycaster
                                               Sequence(*args, **kwargs)
string_utilities
ursinastuff
                                               args = list(args)
<u>curve</u>
                                               t = 0
texture importer
                                               time step = Sequence.default time step
scene
                                               duration = 0
<u>text</u>
                                               funcs = []
<u>window</u>
                                               paused = True
<u>ursinamath</u>
                                               loop = False
<u>camera</u>
                                               auto destroy = True
shader
                                               finished
<u>main</u>
color
                                               generate()
input_handler
                                               append(arg)
mesh importer
                                               extend(list)
duplicate
                                               start()
<u>build</u>
                                               pause()
application
                                               resume()
sequence
                                               finish()
                                               kill()
Vec3
                                               update()
Empty
LoopingList
                                                 e = Entity(model='quad')
CubicBezier
                                                 s = Sequence(
<u>MeshModes</u>
                                                      1,
<u>Mesh</u>
                                                      Func(print, 'one'),
<u>Shader</u>
                                                      Func(e.fade_out, duration=1),
Audio
Ursina
                                                      Func(print, 'two'),
Color
                                                      Func(e.fade in, duration=1),
Vec4
                                                      loop=True
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
                                                 s.append(
<u>SphereCollider</u>
                                                      Func(print, 'appended to sequence')
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
                                                      actions = {'s' : s.start, 'f' : s.finish, 'p' : s.pause, 'r' :
<u>Light</u>
                                                 s.resume}
<u>DirectionalLight</u>
                                                      if key in actions:
PointLight
                                                           actions[key]()
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
<u>Func</u>
                                              FileButton(Button)
<u>Sequence</u>
                                               ursina/prefabs/file browser
FileButton
                                               FileButton(load menu, **kwargs)
<u>FileBrowser</u>
VideoRecorder
                                               load_menu = load_menu
<u>VideoRecorderUI</u>
                                               selected
<u>Cursor</u>
Draggable
                                               on click()
<u>Tooltip</u>
                                               on double click()
SynthGUI
<u>MemoryCounter</u>
                                                 fb = FileBrowser(file types=('.*'), enabled=True)
Panel
Tilemap
                                                 def on_submit(paths):
<u>Sprite</u>
                                                      print('----', paths)
                                                      for p in paths:
DropdownMenuButton
                                                           print('---', p)
<u>DropdownMenu</u>
Animation
                                                 fb.on submit = on submit
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
<u>WindowPanel</u>
Node
```

Node

## FileBrowser(Entity)

ursina/prefabs/file\_browser

```
<u>light</u> dark
                                                FileBrowser(**kwarqs)
Entity
                                                file_types = ['.*', ]
start_path = Path('.').resolve()
<u>Text</u>
Button
                                                return_files = True
<u>mouse</u>
                                                return folders = False
raycaster
                                                selection limit = 1
                                                \max buttons = 24
string_utilities
                                                title_bar = Button(parent=self, scale=(.9,.035), text='<gray>Open',
ursinastuff
                                                color=color.dark_gray, collision=False)
<u>curve</u>
                                                address bar = Button(parent=self, scale=(.8,.035), text='//',
texture importer
                                                text_origin=(-.5,0), y=-.05, highlight_color=color.black)
scene
                                                folder_up_button = Button(parent=self, scale=(.035,.035),
<u>text</u>
                                                texture='arrow_down', rotation_z=180, position=(-.42,-.05), color=color.white, highlight_color=color.azure,
<u>window</u>
<u>ursinamath</u>
                                                on click=self.folder up)
<u>camera</u>
                                                button parent = Entity(parent=self)
shader
                                                back_panel = Entity(parent=self, model='quad', collider='box',
<u>main</u>
                                                origin y=.5, scale=(.9,(self.max buttons*.025)+.19), color=color. 32,
color
                                                z = .1)
<u>input_handler</u>
                                                bg = Button(parent=self, z=1, scale=(999,999), color=color.black66,
mesh importer
                                                highlight_color=color.black66, pressed_color=color.black66) cancel_button = Button(parent=self, scale=(.875*.24, .05),
duplicate
<u>build</u>
                                                y=(-self.max buttons*.025)-.15, origin x=-.5, x=-.875/2,
<u>application</u>
                                                text='Cancel', on_click=self.close)
open_button = Button(parent=self, scale=(.875*.74, .05),
<u>sequence</u>
                                                y=(-self.max_buttons*.025)-.15, origin_x=.5, x=.875/2, text='Open',
Vec3
                                                color=color.dark_gray, on_click=self.open)
Empty
                                                cancel button 2 = Button(parent=self.title bar, model=Circle())
<u>LoopingList</u>
                                                world_scale=self.title_bar.world_scale_y*.75, origin_x=.5, x=.495,
z=-.1, text='<gray>x', on_click=self.close)
CubicBezier
<u>MeshModes</u>
                                                can scroll up indicator = Entity(parent=self, model='quad',
<u>Mesh</u>
                                                texture='arrow_down', rotation_z=180, scale=(.05,.05), y=-.0765,
<u>Shader</u>
                                                z=-.1, color=color.dark gray, enabled=False,
Audio
                                                add to scene entities=False)
Ursina
                                                can_scroll_down_indicator = Entity(parent=self, model='quad',
Color
                                                texture='arrow_down', scale=(.05,.05),
y=(-self.max_buttons*.025)-.104, z=-.1, color=color.dark_gray,
Vec4
HitInfo
                                                enabled=False, add_to_scene_entities=False)
<u>Collider</u>
                                                scroll
BoxCollider
                                                path
<u>SphereCollider</u>
                                                selection
MeshCollider
<u>Keys</u>
                                                input(key)
Vec2
                                                on enable()
<u>Texture</u>
                                                close()
Light
                                                folder up()
<u>DirectionalLight</u>
                                                open(path=None)
PointLight
<u>AmbientLight</u>
                                                  fb = FileBrowser(file types=('.*'), enabled=True)
SpotLight
<u>Trigger</u>
                                                  def on_submit(paths):
<u>FastMesh</u>
                                                       print('----
                                                                          , paths)
<u>Func</u>
                                                       for p in paths:
Sequence
                                                            print('---', p)
FileButton
                                                  fb.on submit = on submit
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
Cursor
Draggable
                                              VideoRecorder(Entity)
<u>Tooltip</u>
SynthGUI
                                                ursina/prefabs/video recorder
<u>MemoryCounter</u>
Panel
                                                VideoRecorder(duration=5, name='untitled video', **kwargs)
Tilemap
<u>Sprite</u>
                                                recording = False
                                                file path = Path(application.asset folder) / 'video temp'
DropdownMenuButton
                                                i = \overline{0}
<u>DropdownMenu</u>
                                                duration = duration
<u>Animation</u>
                                                fps = 30
FrameAnimation3d
                                                video name = name
<u>FirstPersonController</u>
EditorCamera
                                                max frames = int(self.duration * self.fps)
<u>Space</u>
                                                frames = []
<u>WindowPanel</u>
```

```
start recording()
                                             stop recording()
                                             update()
<u>light</u> dark
                                             convert_to_gif()
<u>Entity</u>
                                               window.size *= .5
                                               from ursina.prefabs.first person controller import
<u>Text</u>
                                               FirstPersonController
Button
                                               from ursina.shaders import lit with shadows shader
mouse
                                               random.seed(0)
raycaster
                                               Entity.default_shader = lit_with_shadows_shader
string_utilities
ursinastuff
                                               ground = Entity(model='plane', collider='box', scale=64,
                                               texture='grass', texture_scale=(4,4))
<u>curve</u>
texture importer
                                               editor camera = EditorCamera(enabled=False, ignore paused=True)
scene
                                               player = FirstPersonController(model='cube', z=-10,
<u>text</u>
<u>window</u>
                                               color=color.orange, origin_y=-.5, speed=8)
<u>ursinamath</u>
                                               player.collider = BoxCollider(player, Vec3(0,1,0), Vec3(1,2,1))
<u>camera</u>
                                               gun = Entity(model='cube', parent=camera, position=(.5,-.25,.25),
shader
                                               scale=(.3,.2,1), origin z=-.5, color=color.red, on cooldown=False)
<u>main</u>
color
                                               shootables_parent = Entity()
input_handler
mesh importer
                                               mouse.traverse target = shootables parent
duplicate
<u>build</u>
                                               for i in range(16):
                                                    Entity(model='cube', origin y=-.5, scale=2, texture='brick',
<u>application</u>
                                               texture scale=(1,2),
sequence
                                                         x=random.uniform(-8,8),
Vec3
                                                         z=random.uniform(-8,8) + 8,
                                                        collider='box',
<u>Empty</u>
<u>LoopingList</u>
                                                         scale_y = random.uniform(2,3),
CubicBezier
                                                         color=color.hsv(0, 0, random.uniform(.9, 1))
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
                                               sun = DirectionalLight()
Ursina
                                               sun.look at(Vec3(1,-1,-1))
Color
                                               Sky()
Vec4
<u>HitInfo</u>
                                               vr = VideoRecorder(duration=2)
<u>Collider</u>
                                               def input(key):
<u>BoxCollider</u>
                                                    if key == '5':
                                                        vr.start_recording()
<u>SphereCollider</u>
<u>MeshCollider</u>
                                                    if key == '6':
<u>Keys</u>
                                                        vr.stop_recording()
Vec2
<u>Texture</u>
Light
<u>DirectionalLight</u>
                                            VideoRecorderUI(WindowPanel)
PointLight
<u>AmbientLight</u>
                                             ursina/prefabs/video recorder
SpotLight
<u>Trigger</u>
                                             VideoRecorderUI(**kwargs)
<u>FastMesh</u>
Func
                                             duration_label = Text('duration:')
duration_field = InputField(default_value='5')
<u>Sequence</u>
                                             fps_label = Text('fps:')
FileButton
                                             fps_field = InputField(default_value='30')
<u>FileBrowser</u>
                                             name label = Text('name:')
VideoRecorder
                                             name field = InputField(default value='untitled video')
<u>VideoRecorderUI</u>
                                             start_button = Button(text='Start Recording [Shift+F12]',
<u>Cursor</u>
                                             color=color.azure, on click=self.start recording)
<u>Draggable</u>
                                             y = .5
<u>Tooltip</u>
                                             visible = False
SynthGUI
<u>MemoryCounter</u>
                                             input(key)
Panel
                                             start_recording()
Tilemap
<u>Sprite</u>
                                               window.size *= .5
                                               from ursina.prefabs.first person_controller import
<u>DropdownMenuButton</u>
                                               FirstPersonController
<u>DropdownMenu</u>
                                               from ursina.shaders import lit_with_shadows_shader
<u>Animation</u>
                                               random.seed(0)
FrameAnimation3d
                                               Entity.default_shader = lit_with_shadows_shader
<u>FirstPersonController</u>
EditorCamera
                                               ground = Entity(model='plane', collider='box', scale=64,
<u>Space</u>
                                               texture='grass', texture scale=(4,4))
<u>WindowPanel</u>
Node
```

```
editor camera = EditorCamera(enabled=False, ignore_paused=True)
                                                player = FirstPersonController(model='cube', z=-10,
                                                color=color.orange, origin_y=-.5, speed=8)
<u>light</u> dark
                                                player.collider = BoxCollider(player, Vec3(0,1,0), Vec3(1,2,1))
Entity
                                                gun = Entity(model='cube', parent=camera, position=(.5,-.25,.25),
                                                scale=(.3,.2,1), origin z=-.5, color=color.red, on cooldown=False)
<u>Text</u>
Button
                                                shootables parent = Entity()
<u>mouse</u>
                                                mouse.traverse target = shootables parent
raycaster
string_utilities
                                                for i in range(16):
                                                    Entity(model='cube', origin y=-.5, scale=2, texture='brick',
ursinastuff
                                                texture_scale=(1,2),
<u>curve</u>
texture importer
                                                         x=random.uniform(-8,8),
                                                         z=random.uniform(-8,8) + 8,
scene
                                                         collider='box',
<u>text</u>
<u>window</u>
                                                         scale y = random.uniform(2,3),
                                                         color=color.hsv(0, 0, random.uniform(.9, 1))
<u>ursinamath</u>
<u>camera</u>
shader
<u>main</u>
                                                sun = DirectionalLight()
color
                                                sun.look at(Vec3(1,-1,-1))
input_handler
mesh importer
                                                Sky()
duplicate
<u>build</u>
                                                vr = VideoRecorder(duration=2)
<u>application</u>
                                                def input(key):
                                                    if key == '5':
sequence
                                                         vr.start_recording()
Vec3
                                                    if key == '6':
                                                         vr.stop_recording()
Empty
<u>LoopingList</u>
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
                                            Cursor(Entity)
<u>Shader</u>
<u>Audio</u>
                                              ursina/prefabs/cursor
Ursina
Color
                                              Cursor(**kwargs)
Vec4
<u>HitInfo</u>
                                              parent = camera.ui
<u>Collider</u>
                                              texture = 'cursor
<u>BoxCollider</u>
                                              model = 'quad'
<u>SphereCollider</u>
                                              color = color.light_gray
<u>MeshCollider</u>
                                              render queue = 1
<u>Keys</u>
Vec2
                                              update()
<u>Texture</u>
Light
                                                Button('button').fit to text()
<u>DirectionalLight</u>
PointLight
                                                camera.orthographic = True
<u>AmbientLight</u>
                                                camera.fov = 100
SpotLight
                                               e = Entity(model='cube')
<u>Trigger</u>
                                               mouse. mouse watcher.setGeometry(e.model.node())
<u>FastMesh</u>
                                                mouse.visible = False
Func
<u>Sequence</u>
FileButton
<u>FileBrowser</u>
                                            Draggable(<u>Button</u>)
VideoRecorder
                                              ursina/prefabs/draggable
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                              Draggable(**kwargs)
<u>Draggable</u>
<u>Tooltip</u>
                                              require key = None
SynthGUI
                                              dragging = False
<u>MemoryCounter</u>
                                              delta drag = 0
Panel
                                              start_pos = self.world_position
Tilemap
                                              start_offset = (0,0,0)
<u>Sprite</u>
                                              step = (0,0,0)
                                              plane direction = (0,0,1)
<u>DropdownMenuButton</u>
                                              lock = Vec3(0,0,0)
                                                                        # set to 1 to lock movement on any of x, y and
<u>DropdownMenu</u>
                                              z axes
<u>Animation</u>
                                              min_x, self.min_y, self.min_z = -inf, -inf, -inf
FrameAnimation3d
                                              \max x, self.\max y, self.\max z = \inf, inf, inf
<u>FirstPersonController</u>
EditorCamera
                                              input(key)
<u>Space</u>
                                              start dragging()
<u>WindowPanel</u>
                                              stop_dragging()
Node
```

```
Entity(model='plane', scale=8, texture='white_cube', texture_scale=
<u>light</u> dark
                                                (8,8))
                                               draggable button = Draggable(scale=.1, text='drag me',
<u>Entity</u>
                                               position=(-.5, 0))
                                               world space draggable = Draggable(parent=scene, model='cube',
<u>Text</u>
                                               color=color.azure, plane_direction=(0,1,0), lock=(1,0,0))
Button
mouse
                                               EditorCamera(rotation=(30,10,0))
raycaster
                                               world_space_draggable.drop = Func(print, 'dropped cube')
string_utilities
ursinastuff
<u>curve</u>
texture importer
                                            Tooltip(<u>Text</u>)
scene
text
                                              ursina/prefabs/tooltip
<u>window</u>
<u>ursinamath</u>
                                             Tooltip(text='', background_color=color.black66, **kwargs)
<u>camera</u>
shader
                                             original scale = self.scale
<u>main</u>
color
                                             update()
input_handler
mesh importer
                                               app = Ursina()
duplicate
<u>build</u>
                                                    tooltip test = Tooltip(
<u>application</u>
                                                    '<scale:1.5><pink>' + 'Rainstorm' + '<scale:1> \n \n' +
sequence
                                                '''Summon a <blue>rain
                                                storm <default>to deal 5 <blue>water
Vec3
                                               damage <default>to <red>everyone, <default>including
<u>Empty</u>
                                               <orange>yourself. <default>
<u>LoopingList</u>
                                               Lasts for 4 rounds.'''.replace('\n', ''),
CubicBezier
                                                        background color=color.red
<u>MeshModes</u>
                                               )
<u>Mesh</u>
<u>Shader</u>
                                                    tooltip test.enabled = True
<u>Audio</u>
                                                    app.run()
Ursina
Color
<u>Vec4</u>
<u>HitInfo</u>
<u>Collider</u>
                                            SynthGUI(<u>Entity</u>)
<u>BoxCollider</u>
                                              ursina/prefabs/ursfx
<u>SphereCollider</u>
<u>MeshCollider</u>
                                             SynthGUI(**kwargs)
<u>Keys</u>
Vec2
                                             wave panel = Entity(parent=self, scale=.35, x=-0)
<u>Texture</u>
                                             waveform = Entity(parent=self.wave panel, scale y=.75)
<u>Light</u>
                                             waveform_bg = Entity(parent=self.waveform, model='quad', origin=
<u>DirectionalLight</u>
                                              (-.5,-.5), z=.01, color=color.black66)
PointLight
                                             volume slider = Slider(parent=self.wave panel, x=-.05, vertical=True,
<u>AmbientLight</u>
                                              scale=1.95, min=.05, max=1, default=.75, step=.01,
SpotLight
                                              on_value_changed=self.play)
<u>Trigger</u>
                                             wave selector = ButtonGroup(('sine', 'triangle', 'square', 'noise'),
<u>FastMesh</u>
                                             parent=self.wave_panel, scale=.11, y=-.075)
Func
                                              pitch_slider = Slider(parent=self.wave_panel, y=-.25, scale=1.95,
<u>Sequence</u>
                                             min=-36, max=36, default=0, step=1, on value changed=self.play,
                                              text='pitch')
FileButton
                                              pitch change slider = Slider(parent=self.wave panel, y=-.325,
<u>FileBrowser</u>
                                             scale=1.95, min=-12, max=12, default=0, step=1,
on_value_changed=self.play, text='pitch change')
VideoRecorder
<u>VideoRecorderUI</u>
                                             speed slider = Slider(parent=self.wave panel, scale=1.95, min=.5,
<u>Cursor</u>
                                             max=4, default=1, step=.1, on_value_changed=self.play, y=-.4,
text='speed')
<u>Draggable</u>
<u>Tooltip</u>
                                              coin button = Button(text='coin', parent=self.wave panel, scale=(.25,
SynthGUI
                                              .125), origin=(-.5,.5), y=-.45, on_click=coin_sound)
<u>MemoryCounter</u>
                                              knobs = [Draggable(parent=self.waveform, scale=.05, model='circle',
Panel
                                              color=color.light_gray, highlight_color=color.azure,
Tilemap
                                              position=default_positions[i], i=i, min_y=0, max_y=1) for i in
<u>Sprite</u>
                                              line = Entity(parent=self.waveform, model=Mesh(vertices=[Vec3(0,0,0),
<u>DropdownMenuButton</u>
                                              Vec3(1,0,0)], mode='line', thickness=3), z=.01,
<u>DropdownMenu</u>
                                              color=color.light gray)
<u>Animation</u>
                                              bg = Entity(parent=self.wave panel, model='wireframe quad', origin=
FrameAnimation3d
                                              (-.5,-.5), z=.02, color=color.black, scale x=1)
<u>FirstPersonController</u>
                                             play_button = Button(text='>', parent=self.wave_panel,
model='circle', scale=(.125, .125), color=color.azure, origin=
EditorCamera
<u>Space</u>
                                              (-.5,-.5), position=(-.075,1.025), on_click=self.play)
<u>WindowPanel</u>
                                              copy_button = Button(text='copy', parent=self.wave_panel, scale=(.25,
Node
```

update()

```
<u>light</u> dark
Entity
<u>Text</u>
Button
<u>mouse</u>
raycaster
string_utilities
ursinastuff
                                                     coin_sound()
<u>curve</u>
texture importer
                                                     drag(this knob=knob)
                                                     drop(this knob=knob)
scene
<u>text</u>
                                                     update()
<u>window</u>
                                                     input(key)
<u>ursinamath</u>
                                                     copy_code()
                                                     paste_code(code="")
<u>camera</u>
                                                     draw()
shader
<u>main</u>
                                                     play()
color
                                                       app = Ursina()
input_handler
mesh importer
duplicate
<u>build</u>
<u>application</u>
sequence
Vec3
Empty
<u>LoopingList</u>
CubicBezier
<u>MeshModes</u>
                                                             _name__
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
                                                            app.run()
Ursina
Color
Vec4
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
Vec2
                                                     parent = camera.ui
<u>Texture</u>
Light
                                                     origin = (0.5, -0.5)
<u>DirectionalLight</u>
PointLight
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
                                                     update()
<u>FastMesh</u>
<u>Func</u>
                                                       MemoryCounter()
<u>Sequence</u>
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
Draggable
                                                   Panel (Entity)
<u>Tooltip</u>
                                                     ursina/prefabs/panel
SynthGUI
<u>MemoryCounter</u>
                                                     Panel(**kwargs)
Panel
Tilemap
                                                     parent = camera.ui
<u>Sprite</u>
                                                     model = Quad()
                                                     color = Button.color
DropdownMenuButton
<u>DropdownMenu</u>
<u>Animation</u>
                                                       p = Panel()
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
<u>WindowPanel</u>
Node
```

```
.125), color=color.dark_gray, origin=(-.5,-.5), position=
  (-.075+.125+.025, 1.025), on click=self.copy code)
 paste_button = Button(text='paste', parent=self.wave_panel,
scale=(.25, .125), color=color.dark_gray, origin=(-.5,-.5), position=
  (self.copy_button.x + self.copy_button.scale_x + .025, 1.025),
 on_click=self.paste_code)
  code_text = Text('', parent=self.wave_panel, scale=2, position=
  (self.paste button.x+self.paste button.scale x+.025,1.11))
 background panel = Entity(model=Quad(radius=.025),
 parent=self.wave_panel, color=color.black66, z=1, origin=(-.5,-.5),
  scale=(1.125,1.75+.025), position=(-.1,-.6))
   gui = SynthGUI(enabled=False)
   def toggle_gui_input(key):
    if key == 'f3':
            gui.enabled = not gui.enabled
   Entity(input=toggle gui input)
               _== '__main__':
        Sprite('shore', z=10, ppu=64, color=color.gray)
        gui.enabled = True
MemoryCounter(<u>Text</u>)
  ursina/prefabs/memory_counter
 MemoryCounter(**kwargs)
  position = window.bottom right - Vec2(.025,0)
 process = psutil.Process(os.getpid())
  text = 'eofiwjeofiwejf'
   Displays the amount of memory used in the bottom right corner
```

Node

## ursina/prefabs/tilemap <u>light</u> dark Tilemap(tilemap='', tileset='', tileset size=(8,8), \*\*kwargs) **Entity** grid = [[self.tilemap.get pixel(x,y) for y in <u>Text</u> range(self.tilemap.height)] for x in range(self.tilemap.width)] Button tileset = tileset <u>mouse</u> tileset size = tileset size raycaster model = Mesh() texture = tileset string\_utilities colliders = list() ursinastuff auto\_render = False <u>curve</u> outline = Entity(parent=self, model=Quad(segments=0, mode='line', texture importer thickness=1), color=color.cyan, z=.01, origin=(-.5,-.5), scene enabled=self.edit mode) <u>text</u> uv dict = { <u>window</u> '111111111' : [(4,1), (5,1), (6,1), (7,1)], single\_block\_coordinates = [(4,0), (5,0), (6,0), (7,0)] variation\_chance = [0,0,0,0,1,1,1,2,2,3] # fill $\underline{\text{ursinamath}}$ <u>camera</u> shader $uv_margin = .002$ <u>main</u> color update() <u>input\_handler</u> draw\_temp(position) mesh importer input(key) duplicate render() <u>build</u> save() <u>application</u> sequence EditorCamera() tilemap = Tilemap('tilemap test level', tileset='test tileset', Vec3 tileset\_size=(8,4), parent=scene) **Empty** camera.orthographic = True <u>LoopingList</u> camera.position = tilemap.tilemap.size / 2 CubicBezier camera.fov = tilemap.tilemap.height <u>MeshModes</u> <u>Mesh</u> Text('press tab to toggle edit mode', origin=(.5,0), position= <u>Shader</u> (-.55,.4))<u>Audio</u> Ursina Color Vec4 <u>HitInfo</u> Sprite(Entity) <u>Collider</u> <u>BoxCollider</u> ursina/prefabs/sprite <u>SphereCollider</u> <u>MeshCollider</u> Sprite(texture=None, ppu=ppu, \*\*kwarqs) <u>Keys</u> Vec2 model = 'quad' <u>Texture</u> texture = texture **Light** ppu = ppuaspect ratio = self.texture.width / self.texture.height <u>DirectionalLight</u> PointLight scale\_x = self.scale\_y \* self.aspect\_ratio <u>AmbientLight</u> **SpotLight** <u>Trigger</u> camera.orthographic = True <u>FastMesh</u> camera.fov = 1<u>Func</u> Sprite.ppu = 16<u>Sequence</u> Texture.default\_filtering = None s = Sprite('brick', filtering=False) **FileButton** <u>FileBrowser</u> VideoRecorder <u>VideoRecorderUI</u> Cursor Sky(Entity) Draggable ursina/prefabs/sky <u>Tooltip</u> **SynthGUI** Sky(\*\*kwargs) <u>MemoryCounter</u> Panel Tilemap update() <u>Sprite</u> **DropdownMenuButton** <u>DropdownMenu</u> <u>Animation</u> DropdownMenuButton(Button) FrameAnimation3d ursina/prefabs/dropdown menu <u>FirstPersonController</u> **EditorCamera** DropdownMenuButton(text='', \*\*kwargs) <u>Space</u> <u>WindowPanel</u>

Tilemap(GridEditor)

31 of 55 4/21/22, 09:53

```
from ursina.prefabs.dropdown menu import DropdownMenu,
                                                  DropdownMenuButton
<u>light</u> dark
                                                  DropdownMenu('File', buttons=(
    DropdownMenuButton('New'),
Entity
                                                       DropdownMenuButton('Open'),
<u>Text</u>
                                                       DropdownMenu('Reopen Project', buttons=(
Button
                                                            DropdownMenuButton('Project 1'),
<u>mouse</u>
                                                            DropdownMenuButton('Project 2'),
raycaster
                                                            )),
string_utilities
                                                       DropdownMenuButton('Save'),
                                                       DropdownMenu('Options', buttons=(
    DropdownMenuButton('Option a'),
ursinastuff
<u>curve</u>
texture importer
                                                            DropdownMenuButton('Option b'),
scene
                                                       DropdownMenuButton('Exit'),
<u>text</u>
<u>window</u>
<u>ursinamath</u>
<u>camera</u>
shader
<u>main</u>
                                              DropdownMenu(DropdownMenuButton)
color
<u>input_handler</u>
                                                ursina/prefabs/dropdown menu
mesh importer
duplicate
                                                DropdownMenu(text='', buttons=list(), **kwargs)
<u>build</u>
application
                                                position = window.top left
sequence
                                                buttons = buttons
                                                arrow symbol = Text(world parent=self, text='>', origin=(.5,.5),
Vec3
                                                position=(.95, 0), color=color.gray)
Empty
<u>LoopingList</u>
                                                open()
<u>CubicBezier</u>
                                                close()
<u>MeshModes</u>
                                                on mouse enter()
<u>Mesh</u>
                                                input(key)
<u>Shader</u>
                                                update()
<u>Audio</u>
Ursina
                                                  from ursina.prefabs.dropdown menu import DropdownMenu,
Color
                                                  DropdownMenuButton
Vec4
<u>HitInfo</u>
                                                  DropdownMenu('File', buttons=(
<u>Collider</u>
                                                       DropdownMenuButton('New'),
DropdownMenuButton('Open'),
<u>BoxCollider</u>
<u>SphereCollider</u>
                                                       DropdownMenu('Reopen Project', buttons=(
<u>MeshCollider</u>
                                                            DropdownMenuButton('Project 1'),
DropdownMenuButton('Project 2'),
<u>Keys</u>
Vec2
<u>Texture</u>
                                                       DropdownMenuButton('Save'),
<u>Light</u>
                                                       DropdownMenu('Options', buttons=(
<u>DirectionalLight</u>
                                                            DropdownMenuButton('Option a'),
PointLight
                                                            DropdownMenuButton('Option b'),
<u>AmbientLight</u>
SpotLight
                                                       DropdownMenuButton('Exit'),
<u>Trigger</u>
<u>FastMesh</u>
<u>Func</u>
Sequence
FileButton
                                              Animation(Sprite)
<u>FileBrowser</u>
                                                ursina/prefabs/animation
VideoRecorder
<u>VideoRecorderUI</u>
                                                Animation(name, fps=12, loop=True, autoplay=True, frame_times=None,
Cursor
                                                **kwargs)
<u>Draggable</u>
<u>Tooltip</u>
                                                sequence = Sequence(loop=loop, auto destroy=False)
SynthGUI
                                                frame times = frame times
<u>MemoryCounter</u>
                                                is playing = False
Panel
                                                autoplay = autoplay
Tilemap
                                                                         # get the duration of the animation. you can't
                                                duration
<u>Sprite</u>
                                                set it. to do so, change the fps instead.
<u>DropdownMenuButton</u>
                                                start()
DropdownMenu
                                                pause()
<u>Animation</u>
                                                resume()
FrameAnimation3d
                                                finish()
<u>FirstPersonController</u>
EditorCamera
                                                  app = Ursina()
<u>Space</u>
<u>WindowPanel</u>
                                                       111
Node
```

Loads an image sequence as a frame animation.

```
So if you have some frames named image 000.png, image 001.png,
                                               image 002.png and so on,
<u>light</u> dark
                                                    you can load it like this: Animation('image')
<u>Entity</u>
                                                    You can also load a .gif by including the file type:
                                               Animation('image.gif')
<u>Text</u>
Button
mouse
raycaster
                                                    a = Animation('ursina wink')
                                                    destroy(a)
string_utilities
                                                    app.run()
ursinastuff
<u>curve</u>
texture importer
scene
<u>text</u>
<u>window</u>
                                           FrameAnimation3d(<u>Entity</u>)
<u>ursinamath</u>
                                             ursina/prefabs/frame animation 3d
<u>camera</u>
shader
                                             FrameAnimation3d(name, fps=12, loop=True, autoplay=True,
<u>main</u>
                                             frame_times=None, **kwargs)
color
input_handler
                                             frames = [Entity(parent=self, model=e.stem, enabled=False,
mesh importer
                                             add to scene entities=False) for e in model names]
duplicate
                                             sequence = Sequence(loop=loop, auto_destroy=False)
<u>build</u>
                                             is playing = False
<u>application</u>
                                             autoplay = autoplay
sequence
                                             duration
Vec3
                                             start()
Empty
                                             pause()
LoopingList
                                             resume()
CubicBezier
                                             finish()
<u>MeshModes</u>
<u>Mesh</u>
                                               application.asset_folder = application.asset_folder.parent.parent /
<u>Shader</u>
                                                'samples
<u>Audio</u>
Ursina
Color
                                               Loads an obj sequence as a frame animation.
<u>Vec4</u>
                                               So if you have some frames named run cycle 000.obj,
<u>HitInfo</u>
                                               run_cycle_001.obj, run_cycle_002.obj and so on,
<u>Collider</u>
                                               you can load it like this: FrameAnimation3d('run cycle ')
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
                                               FrameAnimation3d('blob animation ')
<u>Keys</u>
Vec2
<u>Texture</u>
<u>Light</u>
<u>DirectionalLight</u>
                                           FirstPersonController(Entity)
PointLight
                                             ursina/prefabs/first person controller
<u>AmbientLight</u>
SpotLight
                                             FirstPersonController(**kwargs)
<u>Trigger</u>
<u>FastMesh</u>
                                             cursor = Entity(parent=camera.ui, model='quad', color=color.pink,
Func
                                             scale=.008, rotation z=45)
<u>Sequence</u>
                                             speed = 5
                                             height = 2
FileButton
                                             camera pivot = Entity(parent=self, y=self.height)
<u>FileBrowser</u>
                                             mouse sensitivity = Vec2(40, 40)
VideoRecorder
                                             gravity = 1
<u>VideoRecorderUI</u>
                                             grounded = False
<u>Cursor</u>
                                             jump height = 2
Draggable
                                             jump_up_duration = .5
<u>Tooltip</u>
                                             fall after = .35 # will interrupt jump up
SynthGUI
                                             jumping = False
<u>MemoryCounter</u>
                                             air time = 0
Panel
Tilemap
                                             update()
<u>Sprite</u>
                                             input(key)
                                             jump()
DropdownMenuButton
                                             start fall()
<u>DropdownMenu</u>
                                             land()
<u>Animation</u>
                                             on enable()
FrameAnimation3d
                                             on disable()
<u>FirstPersonController</u>
EditorCamera
                                               from ursina.prefabs.first_person_controller import
<u>Space</u>
                                               FirstPersonController
<u>WindowPanel</u>
                                               ground = Entity(model='plane', scale=(100,1,100),
Node
```

```
<u>light</u> dark
<u>Entity</u>
<u>Text</u>
Button
<u>mouse</u>
raycaster
string_utilities
ursinastuff
<u>curve</u>
texture importer
scene
<u>text</u>
<u>window</u>
<u>ursinamath</u>
<u>camera</u>
shader
<u>main</u>
color
input_handler
mesh importer
<u>duplicate</u>
<u>build</u>
<u>application</u>
sequence
Vec3
<u>Empty</u>
<u>LoopingList</u>
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
Ursina
Color
Vec4
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
Light
<u>DirectionalLight</u>
PointLight
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
Func
<u>Sequence</u>
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
Cursor
<u>Draggable</u>
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
Panel
Tilemap
<u>Sprite</u>
<u>DropdownMenuButton</u>
<u>DropdownMenu</u>
<u>Animation</u>
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
<u>WindowPanel</u>
Node
```

```
color=color.yellow.tint(-.2), texture='white cube', texture scale=
   (100,100), collider='box')
   e = Entity(model='cube', scale=(1,5,10), x=2, y=.01, rotation_y=45,
collider='box', texture='white_cube')
   e.texture_scale = (e.scale_z, e.scale_y)
   e = Entity(model='cube', scale=(1,5,10), x=-2, y=.01,
collider='box', texture='white_cube')
   e.texture scale = (e.scale z, e.scale y)
   player = FirstPersonController(y=2, origin y=-.5)
   player.gun = None
   gun = Button(parent=scene, model='cube', color=color.blue,
   origin_y=-.5, position=(3,0,3), collider='box')
   gun.on click = Sequence(Func(setattr, gun, 'parent', camera),
   Func(setattr, player, 'gun', gun))
   gun_2 = duplicate(gun, z=7, x=8)
   slope = Entity(model='cube', collider='box', position=(0,0,8),
scale=6, rotation=(45,0,0), texture='brick', texture_scale=(8,8))
slope = Entity(model='cube', collider='box', position=(5,0,10),
scale=6, rotation=(80,0,0), texture='brick', texture_scale=(8,8))
   hookshot target = Button(parent=scene, model='cube',
   color=color.brown, position=(4,5,5))
   hookshot_target.on_click = Func(player.animate_position,
   hookshot target.position, duration=.5, curve=curve.linear)
   def input(key):
        if key == 'left mouse down' and player.gun:
             gun.blink(color.orange)
             bullet = Entity(parent=gun, model='cube', scale=.1,
   color=color.black)
             bullet.world_parent = scene
             bullet.animate position(bullet.position+
   (bullet.forward*50), curve=curve.linear, duration=1)
             destroy(bullet, delay=1)
EditorCamera(Entity)
 ursina/prefabs/editor_camera
 EditorCamera(**kwargs)
 gizmo = Entity(parent=self, model='sphere', color=color.orange,
 scale=.025, add_to_scene_entities=False, enabled=False)
 rotation_speed = 2\overline{00}
 pan speed = Vec2(5, 5)
 move_speed = 10
 zoom\_speed = 1.25
 zoom smoothing = 8
 rotate around mouse hit = False
 smoothing helper = Entity(add to scene entities=False)
 rotation smoothing = 0
 start position = self.position
 perspective_fov = camera.fov
 orthographic_fov = camera.fov
 on destroy = self.on disable
 hotkeys = {'toggle_orthographic':'shift+p', 'focus':'f',
'reset_center':'shift+f'}
 on_enable()
 on_disable()
 on destroy()
 input(key)
 update()
   app = Ursina(vsync=False)
```

sky = Sky()

e.model.colorize()

Simple camera for debugging.

color=color.white, collider='box')

Hold right click and move the mouse to rotate around point.

e = Entity(model=load\_model('cube', use\_deepcopy=True),

```
from ursina.prefabs.first person controller import
                                                 FirstPersonController
<u>light</u> dark
                                                 ground = Entity(model='plane', scale=32, texture='white_cube',
Entity
                                                 texture_scale=(32,32), collider='box')
                                                 box = Entity(model='cube', collider='box', texture='white_cube',
scale=(10,2,2), position=(2,1,5), color=color.light_gray)
<u>Text</u>
Button
                                                 ec = EditorCamera(rotation smoothing=10, enabled=1, rotation=
<u>mouse</u>
raycaster
string_utilities
                                                 rotation_info = Text(position=window.top_left)
ursinastuff
                                                 def update():
<u>curve</u>
texture importer
                                                      rotation info.text = str(int(ec.rotation y)) + '\n' +
                                                 str(int(ec.rotation x))
scene
<u>text</u>
<u>window</u>
<u>ursinamath</u>
                                                 def input(key):
                                                      if key == 'tab':
                                                                             # press tab to toggle edit/play mode
<u>camera</u>
                                                           ec.enabled = not ec.enabled
shader
                                                           player.enabled = not player.enabled
<u>main</u>
color
input_handler
mesh importer
duplicate
<u>build</u>
                                             Space()
<u>application</u>
                                               ursina/prefabs/window_panel
sequence
                                               Space(height=1)
Vec3
Empty
                                               height = height
LoopingList
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
                                                 WindowPanel is an easy way to create UI. It will automatically
<u>Shader</u>
                                                 layout the content.
<u>Audio</u>
Ursina
                                                 wp = WindowPanel(
Color
                                                      title='Custom Window',
<u>Vec4</u>
                                                      content=(
<u>HitInfo</u>
                                                           Text('Name:'),
<u>Collider</u>
                                                           InputField(name='name field'),
<u>BoxCollider</u>
                                                           Button(text='Submit', color=color.azure),
<u>SphereCollider</u>
                                                           Slider(),
<u>MeshCollider</u>
                                                           Slider(),
<u>Keys</u>
Vec2
                                                           popup=True,
<u>Texture</u>
                                                           enabled=False
<u>Light</u>
                                                      )
<u>DirectionalLight</u>
PointLight
                                                 def input(key):
<u>AmbientLight</u>
                                                      if key == 'space':
SpotLight
                                                          wp.enabled = True
<u>Trigger</u>
<u>FastMesh</u>
Func
<u>Sequence</u>
                                             WindowPanel(Draggable)
FileButton
                                               ursina/prefabs/window panel
<u>FileBrowser</u>
VideoRecorder
                                               WindowPanel(title='', content=[], **kwargs)
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                               content = content
Draggable
                                               text = title
<u>Tooltip</u>
                                               popup = False
SynthGUI
                                               panel = Entity(parent=self, model='quad', origin=(0,.5), z=.1,
<u>MemoryCounter</u>
                                               color=self.color.tint(.1), collider='box')
Panel
Tilemap
                                               layout()
<u>Sprite</u>
                                               on enable()
                                               close()
DropdownMenuButton
<u>DropdownMenu</u>
<u>Animation</u>
                                                 WindowPanel is an easy way to create UI. It will automatically
FrameAnimation3d
                                                 layout the content.
<u>FirstPersonController</u>
EditorCamera
                                                 wp = WindowPanel(
<u>Space</u>
                                                      title='Custom Window',
<u>WindowPanel</u>
                                                      content=(
Node
```

35 of 55 4/21/22, 09:53

```
Text('Name:'),
                                                            InputField(name='name field'),
                                                            Button(text='Submit', color=color.azure),
<u>light</u> dark
                                                            Slider(),
                                                           Slider(),
<u>Entity</u>
                                                           popup=True,
<u>Text</u>
                                                            enabled=False
Button
                                                       )
mouse
raycaster
                                                  def input(key):
string_utilities
                                                      if key == 'space':
                                                           wp.enabled = True
ursinastuff
<u>curve</u>
texture importer
scene
<u>text</u>
<u>window</u>
                                              Node
<u>ursinamath</u>
                                                ursina/prefabs/conversation
<u>camera</u>
shader
                                                Node(title='', content=[], **kwargs)
<u>main</u>
color
input_handler
mesh importer
                                                  app = Ursina()
<u>duplicate</u>
                                                       conversation = Conversation()
<u>build</u>
                                                       bar mission solved = False
<u>application</u>
                                                       convo = ''
sequence
                                                  I'm looking for my sister. Can you help me find her, please? I haven't seen her in days! Who know what could've happened!?
Vec3
                                                  I'm worried. Will you help me?
Empty
                                                       ^st Yes, of course. This can be a dangerous city.
<u>LoopingList</u>
                                                            Oh no! Do you think something happened to her?
CubicBezier
                                                           What should I do?!
<u>MeshModes</u>
                                                                 * She's probably fine. She can handle herself.
<u>Mesh</u>
                                                                     You're right. I'm still worried though.
<u>Shader</u>
                                                                         * Don't worry, I'll look for her.
<u>Audio</u>
                                                                * Maybe. (chaos += 1)
Ursina
                                                       Help me look for her, please! *runs off*
* I'm sorry, but I don't have time right now.
Color
<u>Vec4</u>
                                                           A true friend wouldn't say that. (evil += 1)
<u>HitInfo</u>
                                                       * I know where she is! (if bar_mission_solved)
<u>Collider</u>
                                                           Really? Where?
<u>BoxCollider</u>
                                                                * I saw her on a ship by the docks, it looked like they
<u>SphereCollider</u>
                                                  were ready to set off.
<u>MeshCollider</u>
                                                                     Thank you! *runs off*
<u>Keys</u>
Vec2
                                                       conversation.start conversation(convo)
<u>Texture</u>
<u>Light</u>
                                                       window.size = window.fullscreen size * .5
<u>DirectionalLight</u>
                                                       Sprite('shore', z=1)
PointLight
                                                       app.run()
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
Func
                                              Conversation(Entity)
Sequence
                                                ursina/prefabs/conversation
FileButton
                                                Conversation(**kwargs)
<u>FileBrowser</u>
VideoRecorder
                                                question = Button(
<u>VideoRecorderUI</u>
                                                         parent=self,
<u>Cursor</u>
                                                         text_origin=(-.5,0),
<u>Draggable</u>
                                                         scale_x=1,
<u>Tooltip</u>
                                                         scale y=.1,
SynthGUI
                                                         model=Quad(radius=.5, aspect=1/.1),
<u>MemoryCounter</u>
                                                          text='What do you want\nWhat do you want?'
Panel
<u>Tilemap</u>
                                                more_indicator = Entity(parent=self.question, model=Circle(3),
<u>Sprite</u>
                                               position=(.45,-.4,-.1), rotation_z=180, color=color.azure,
world_scale=.5, z=-1, enabled=False)
<u>DropdownMenuButton</u>
                                                spacing = 4 * .02
<u>DropdownMenu</u>
                                                wordwrap = 65
<u>Animation</u>
                                                button_model = Quad(radius=.5, aspect=1/.075)
FrameAnimation3d
                                                answer 0 = Button(parent=self, text='answer 0', y=self.question.y-
<u>FirstPersonController</u>
                                                self.spacing-.025, scale=(1,.075), text_origin=(-.5,0),
EditorCamera
                                                model=copy(self.button_model))
<u>Space</u>
                                                answer_1 = Button(parent=self, text='answer_1', y=self.answer_0.y-
<u>WindowPanel</u>
                                                self.spacing, scale=(1,.075), text_origin=(-.5,0),
Node
```

```
answer 2 = Button(parent=self, text='answer 2', y=self.answer 1.y-
                                              self.spacing, scale=(1,.075), text_origin=(-.5,0),
                                              model=copy(self.button_model))
<u>light</u> dark
                                              buttons = (self.answer_0, self.answer_1, self.answer_2)
                                              question_appear_sequence = None
<u>Entity</u>
                                              button appear sequence = None
<u>Text</u>
                                              started = False
Button
<u>mouse</u>
raycaster
                                              toggle()
                                              ask(node, question_part=0)
string_utilities
                                              on_click(node=child)
                                              input(key)
ursinastuff
<u>curve</u>
                                              next()
                                              start conversation(conversation)
texture importer
                                              parse_conversation(convo)
scene
<u>text</u>
<u>window</u>
                                                app = Ursina()
<u>ursinamath</u>
                                                    conversation = Conversation()
                                                    bar_mission_solved = False
<u>camera</u>
shader
                                               \ensuremath{\text{I'm}} looking for my sister. Can you help me find her, please? 
 \ensuremath{\text{I}}
<u>main</u>
                                                haven't seen her in days! Who know what could've happened!?
color
                                               I'm worried. Will you help me?
input_handler
mesh importer
                                                     'Yes, of course. This can be a dangerous city.
                                                         Oh no! Do you think something happened to her?
duplicate
<u>build</u>
                                                         What should I do?!
                                                              * She's probably fine. She can handle herself.
<u>application</u>
                                                                  You're right. I'm still worried though.
sequence
                                                                      * Don't worry, I'll look for her.
Vec3
                                                              * Maybe. (chaos += 1)
                                                                  Help me look for her, please! *runs off*
<u>Empty</u>
                                                    * I'm sorry, but I don't have time right now.
<u>LoopingList</u>
CubicBezier
                                                         A true friend wouldn't say that. (evil += 1)
                                                    * I know where she is! (if bar_mission_solved)
<u>MeshModes</u>
                                                         Really? Where?
<u>Mesh</u>
                                                                I saw her on a ship by the docks, it looked like they
<u>Shader</u>
                                                were ready to set off.
<u>Audio</u>
Ursina
                                                                  Thank you! *runs off*
Color
                                                    conversation.start_conversation(convo)
<u>Vec4</u>
<u>HitInfo</u>
                                                    window.size = window.fullscreen size * .5
<u>Collider</u>
<u>BoxCollider</u>
                                                    Sprite('shore', z=1)
                                                    app.run()
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
                                            PlatformerController2d(Entity)
<u>Light</u>
<u>DirectionalLight</u>
                                              ursina/prefabs/platformer controller 2d
PointLight
<u>AmbientLight</u>
                                              PlatformerController2d(**kwargs)
SpotLight
<u>Trigger</u>
                                              model = 'cube'
<u>FastMesh</u>
                                              origin y = -.5
Func
                                              scale_y = 2
<u>Sequence</u>
                                              color = color.orange
                                              collider = 'box'
FileButton
                                              animator = Animator({'idle' : None, 'walk' : None, 'jump' : None})
<u>FileBrowser</u>
                                              walk speed = 8
VideoRecorder
                                              walking = False
<u>VideoRecorderUI</u>
                                              velocity = 0 # the walk diection is stored here. -1 for left and 1
<u>Cursor</u>
                                              for right.
<u>Draggable</u>
                                              jump\_height = 4
<u>Tooltip</u>
                                              jump\_duration = .5
SynthGUI
                                              jumping = False
<u>MemoryCounter</u>
                                              max_jumps = 1
Panel
                                              jumps left = self.max jumps
Tilemap
                                              gravity = 1
<u>Sprite</u>
                                              grounded = True
                                              air time = 0
                                                              # this increase while we're falling and used when
<u>DropdownMenuButton</u>
                                              calculating the distance we fall so we fall faster and faster instead
<u>DropdownMenu</u>
<u>Animation</u>
                                                                              \ensuremath{\mbox{\#}} by default, it will collide with
                                              traverse_target = scene
FrameAnimation3d
                                              everything except itself. you can change this to change the boxcast
<u>FirstPersonController</u>
                                              traverse target.
EditorCamera
                                              gravity = 0
<u>Space</u>
<u>WindowPanel</u>
                                              update()
Node
```

model=copy(self.button model))

```
input(key)
                                               jump()
                                               start fall()
<u>light</u> dark
                                               land()
Entity
                                                 camera.orthographic = True
                                                 camera.fov = 10
<u>Text</u>
Button
                                                 ground = Entity(model='cube', color=color.white33, origin_y=.5,
scale=(20, 10, 1), collider='box')
<u>mouse</u>
raycaster
                                                 wall = Entity(model='cube', color=color.azure, origin=(-.5,.5),
string_utilities
                                                 scale=(5,10), x=10, y=.5, collider='box')
ursinastuff
                                                 wall 2 = Entity(model='cube', color=color.white33, origin=(-.5,.5),
                                                 scale=(5,10), x=10, y=5, collider='box')
<u>curve</u>
                                                 ceiling = Entity(model='cube', color=color.white33, origin_y=-.5,
scale=(1, 1, 1), y=1, collider='box')
texture importer
scene
<u>text</u>
<u>window</u>
                                                 def input(key):
                                                     if key == 'c':
<u>ursinamath</u>
                                                          wall.collision = not wall.collision
<u>camera</u>
                                                          print(wall.collision)
shader
<u>main</u>
color
                                                 player controller = PlatformerController2d(scale y=2,
input_handler
mesh importer
                                                 jump height=4, x=3)
                                                 camera.add script(SmoothFollow(target=player controller, offset=
duplicate
<u>build</u>
                                                 [0,1,-30], speed=4))
<u>application</u>
                                                 EditorCamera()
sequence
Vec3
Empty
<u>LoopingList</u>
CubicBezier
                                             Animator()
<u>MeshModes</u>
                                               ursina/prefabs/animator
<u>Mesh</u>
<u>Shader</u>
                                               Animator(animations=None, start_state='')
<u>Audio</u>
Ursina
                                               animations = animations
                                                                               # dict
Color
                                               start state = start state
<u>Vec4</u>
                                               state = start state
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
                                                 anim = Animation('ursina_wink', loop=True, autoplay=False)
<u>SphereCollider</u>
                                                 a = Animator(
<u>MeshCollider</u>
                                                     animations = {
<u>Keys</u>
                                                           'lol' : Entity(model='cube', color=color.red),
Vec2
                                                           'yo' : Entity(model='cube', color=color.green, x=1),
<u>Texture</u>
                                                           'help' : anim,
Light
                                                     }
<u>DirectionalLight</u>
PointLight
                                                 a.state = 'yo'
<u>AmbientLight</u>
SpotLight
                                                 Text('press <red>1<default>, <green>2<default> or
<u>Trigger</u>
                                                 <violet>3<default> to toggle different animator states', origin=
<u>FastMesh</u>
                                                 (0, -.5), y=-.4
Func
<u>Sequence</u>
                                                 def input(key):
                                                     if key == '1':
FileButton
                                                          a.state = 'lol'
<u>FileBrowser</u>
                                                     if key == '2':
VideoRecorder
                                                     a.state = 'yo'
if key == '3':
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                                          a.state = 'help'
Draggable
                                                          print(anim.enabled)
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
Panel
Tilemap
                                             ButtonGroup(Entity)
<u>Sprite</u>
                                               ursina/prefabs/button_group
DropdownMenuButton
                                               ButtonGroup(options=None, default='', min_selection=1,
<u>DropdownMenu</u>
                                               max selection=1, **kwarqs)
<u>Animation</u>
FrameAnimation3d
                                               deselected color = Button.color
<u>FirstPersonController</u>
                                               selected color = color.azure
EditorCamera
                                               min selection = min_selection
<u>Space</u>
                                               max selection = max(min selection, max selection)
<u>WindowPanel</u>
                                               buttons = list()
Node
```

```
selected = list()
                                               options = options
                                               parent = camera.ui
<u>light</u> dark
                                               scale = Text.size * 2
<u>Entity</u>
                                               layout()
<u>Text</u>
                                               input(key)
Button
mouse
                                               select(b)
raycaster
                                               on value changed()
                                                 gender_selection = ButtonGroup(('man', 'woman', 'other'))
on_off_switch = ButtonGroup(('off', 'on'), min_selection=1, y=-.1,
string_utilities
ursinastuff
                                                 default='on', selected_color=color.red)
<u>curve</u>
texture importer
                                                 def on value changed():
scene
                                                      print('set gender:', gender_selection.value)
<u>text</u>
<u>window</u>
                                                 gender_selection.on_value_changed = on_value_changed
<u>ursinamath</u>
                                                 def on_value_changed():
<u>camera</u>
                                                      print('turn:', on_off_switch.value)
shader
<u>main</u>
                                                 on_off_switch.on_value_changed = on_value_changed
color
input_handler
                                                 window.color = color. 32
mesh importer
duplicate
<u>build</u>
application
                                             FileBrowserSave(FileBrowser)
sequence
                                               ursina/prefabs/file browser save
Vec3
Empty
                                               FileBrowserSave(**kwargs)
LoopingList
CubicBezier
                                               save button = self.open button
<u>MeshModes</u>
                                               file name field = InputField(
<u>Mesh</u>
                                                        parent
                                               file_type = '' # to save as
<u>Shader</u>
<u>Audio</u>
Ursina
                                               last_saved_file = None
                                                                              # gets set when you save a file
Color
                                               overwrite prompt = WindowPanel(
<u>Vec4</u>
                                                         content=(
<u>HitInfo</u>
                                                             Text('0verwrite?'),
<u>Collider</u>
                                                             Button('Yes', color=color.azure, on click=self.save),
<u>BoxCollider</u>
                                                             Button('Cancel')
<u>SphereCollider</u>
                                                         ),
<u>MeshCollider</u>
                                                         z=-1,
<u>Keys</u>
                                                        popup=True,
Vec2
                                                         enabled=False)
<u>Texture</u>
<u>Light</u>
                                               save()
<u>DirectionalLight</u>
PointLight
                                                 from ursina.prefabs.file_browser_save import FileBrowserSave
<u>AmbientLight</u>
SpotLight
                                                 wp = FileBrowserSave(file_type = '.oto')
<u>Trigger</u>
<u>FastMesh</u>
Func
                                                 import json
<u>Sequence</u>
                                                 save data = {'level': 4, 'name':'Link'}
                                                 wp.data = json.dumps(save_data)
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                             TextField(<u>Entity</u>)
<u>Draggable</u>
                                               ursina/prefabs/text field
<u>Tooltip</u>
SynthGUI
                                               TextField(**kwargs)
<u>MemoryCounter</u>
Panel
                                               font = 'VeraMono.ttf'
Tilemap
                                               line height = 1
<u>Sprite</u>
                                               \max \overline{lines} = 99999
                                               character limit = None
<u>DropdownMenuButton</u>
                                               text entity = Text(
<u>DropdownMenu</u>
                                                        parent
<u>Animation</u>
                                               line numbers = Text(
FrameAnimation3d
                                                        parent
<u>FirstPersonController</u>
                                               character_width = Text.get_width('a', font=self.font)
EditorCamera
                                               cursor_parent = Entity(parent=self, scale=(self.character_width,
<u>Space</u>
                                               -1*Text.size))
<u>WindowPanel</u>
                                               cursor = Entity(parent=self.cursor_parent, model='cube',
Node
```

	color=color.cyan, origin=(5,5), scale=(.1, 1, 0), visible=False)	
	<pre>bg = Entity(parent=self.cursor_parent, model='cube',</pre>	
<u>light</u> <u>dark</u>	<pre>color=color.dark_gray, origin=(5,5), z=0.005, scale=(120, 20, 0.001), collider='box', visible=False)</pre>	
Entity	selection = None	
Text	<pre>selection_parent = Entity(parent=self.cursor_parent, scale=(1,1,0)) register mouse input = False</pre>	
Button	world space mouse = False	
<u>mouse</u>	<pre>triple_click_delay = 0.3</pre>	
<u>raycaster</u>	scroll_enabled = False	
string_utilities	<pre>scroll_size = (0,0) scroll position = (0,0)</pre>	
ursinastuff	scroll delay = 0.07	
curve	highlight color = color.col	.or(120,1,1,.1)
<pre>texture_importer</pre>	text = ''	
scene	replacements = dict()	
<u>text</u> window	on_undo = list() on redo = list()	
ursinamath	shortcuts = {	
camera	'newline':	('enter', 'enter hold'),
<u>shader</u>	'erase':	('backspace', 'backspace hold'),
<u>main</u> color	'erase_word': hold'),	('ctrl+backspace', 'ctrl+backspace
<u>input handler</u>	'delete line':	('ctrl+shift+k',),
mesh importer	'duplicate line':	('ctrl+d',),
<u>duplicate</u>	'undo':	('ctrl+z', 'ctrl+z hold'),
<u>build</u>	'redo':	('ctrl+y', 'ctrl+y hold', 'ctrl+shift+z',
<u>application</u> sequence	'ctrl+shift+z hold'), # 'save':	('ctrl+s',),
<u>sequence</u>	# 'save . # 'save as':	('ctrl+shift+s',),
Vec3	'indent':	('tab',),
<u>Empty</u>	'dedent':	('shift+tab',),
<u>LoopingList</u>	'move_line_down':	('ctrl+down arrow', 'ctrl+down arrow
<u>CubicBezier</u> MeshModes	hold'), 'move line up':	('ctrl+up arrow', 'ctrl+up arrow hold'),
Mesh	'scroll up':	('scroll up',),
Shader	'scroll down':	('scroll down',),
<u>Audio</u>	'cut':	('ctrl+x',),
<u>Ursina</u>	'copy':	('ctrl+c',),
<u>Color</u> Vec4	'paste': 'select all':	('ctrl+v',), ('ctrl+a',),
HitInfo	# 'toggle comment':	
Collider	# 'find':	('ctrl+f',),
BoxCollider	active = (not self.register	mouse_input)
<u>SphereCollider</u> MeshCollider	blink cursor()	
<u>Keys</u>	add text(s, move cursor=True)	
Vec2	move_line(a, b)	
<u>Texture</u>	erase()	
Light Directionallight	<pre>delete_selected()</pre>	
<u>DirectionalLight</u> <u>PointLight</u>	<pre>get_selected() mousePosUnclamped()</pre>	
<u>AmbientLight</u>	mousePos()	
<u>SpotLight</u>	clampMouseScrollOrigin()	
Trigger	input(key)	
<u>FastMesh</u> Func	<pre>keystroke(key) render()</pre>	
Sequence	update()	
<u></u>	resetScrollWait(field)	
FileButton	on_destroy()	
<u>FileBrowser</u>	select_all()	
<u>VideoRecorder</u> VideoRecorderUI	<pre>draw_selection()</pre>	
Cursor	window.x = $200$	
<u>Draggable</u>		
Tooltip	window.color = color.color( $0$ , $0$ , .1)	
<u>SynthGUI</u> MemoryCounter	Button.color = color20 window.color = color. 25	
Panel	window.cotor = cotor23	
<u>Tilemap</u>	<pre>Text.default_font = 'consola.ttf'</pre>	
<u>Sprite</u>	Text.default_resolution = 16*2	
Sky DrondownMenuRutton	<pre>te = TextField(max_lines=300, scale=1, register_mouse_input = True) to tout = dedont();</pre>	
<u>DropdownMenuButton</u> DropdownMenu	<pre>te.text = dedent(''' Lorem ipsum dolor sit amet, consectetur adipiscing elit.</pre>	
Animation	Aliquam sapien tellus, venenatis sit amet ante et, malesuada	
FrameAnimation3d	porta risus.	
<u>FirstPersonController</u>		iverra urna at, maximus eros. Sed dictum
<u>EditorCamera</u> Space	faucibus purus,	imentum in. Mauris iaculis arcu nec justo
<u>Space</u> WindowPanel	rutrum euismod.	Imentam In. Hautis lacutts after het justo
Node		tor, congue id erat sit amet, sollicitudin

```
facilisis velit.'''
                                                    )[1:]
                                                te.render()
<u>light</u> dark
<u>Entity</u>
<u>Text</u>
                                            HealthBar(Button)
Button
mouse
                                              ursina/prefabs/health bar
raycaster
                                              HealthBar(max value=100, show text=True, show lines=False, **kwargs)
string_utilities
ursinastuff
                                              bar = Entity(parent=self, \ model='quad', \ origin=self.origin, \ z=-.01,
<u>curve</u>
                                              color=color.red.tint(-.2), ignore=True)
texture importer
                                              animation duration = .1
scene
                                              lines = Entity(parent=self.bar, origin=self.origin, y=-1,
<u>text</u>
                                              color=color.black33, ignore=True)
<u>window</u>
                                              roundness = .25
<u>ursinamath</u>
                                              max value = max value
<u>camera</u>
                                              clamp = True
shader
                                              show_lines = show_lines
<u>main</u>
                                              show text = show text
color
                                              scale_x = self.scale_x # update rounded corners
input_handler
                                              scale y = self.scale y # update background's rounded corners
mesh importer
                                              value = self.max value
duplicate
<u>build</u>
<u>application</u>
                                                health bar 1 = HealthBar(bar color=color.lime.tint(-.25),
sequence
                                                roundness=.5, value=50)
Vec3
                                                def input(key):
    if key == '+' or key == '+ hold':
Empty
<u>LoopingList</u>
                                                         health bar 1.value += 10
CubicBezier
                                                    if key == '-' or key == '- hold':
<u>MeshModes</u>
                                                         health bar 1.value -= 10
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
Ursina
Color
                                            ContentTypes
Vec4
                                              ursina/prefabs/input_field
<u>HitInfo</u>
<u>Collider</u>
                                              ContentTypes(max_value=100, show_text=True, show_lines=False,
<u>BoxCollider</u>
                                              **kwargs)
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
Vec2
                                                background = Entity(model='quad', texture='pixelscape_combo',
<u>Texture</u>
                                                parent=camera.ui, scale=(camera.aspect_ratio,1), color=color.white)
<u>Light</u>
                                                gradient = Entity(model='quad', texture='vertical_gradient',
<u>DirectionalLight</u>
                                                parent=camera.ui, scale=(camera.aspect_ratio,1),
PointLight
                                                color=color.hsv(240,.6,.1,.75))
<u>AmbientLight</u>
SpotLight
                                                username_field = InputField(y=-.12, limit_content_to='0123456789')
password_field = InputField(y=-.18, hide_content=True)
<u>Trigger</u>
<u>FastMesh</u>
                                                username_field.next_field = password_field
Func
<u>Sequence</u>
                                                def submit():
                                                    print('ursername:', username_field.text)
FileButton
                                                    print('password:', password_field.text)
<u>FileBrowser</u>
VideoRecorder
                                                Button('Login', scale=.1, color=color.cyan.tint(-.4), y=-.26,
<u>VideoRecorderUI</u>
                                                on click=submit).fit to text()
<u>Cursor</u>
<u>Draggable</u>
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
                                            InputField(Button)
Panel
Tilemap
                                              ursina/prefabs/input field
<u>Sprite</u>
                                              InputField(default_value='', label='', max_lines=1,
<u>DropdownMenuButton</u>
                                              character_limit=24, **kwargs)
<u>DropdownMenu</u>
<u>Animation</u>
                                              default_value = default_value
FrameAnimation3d
                                              limit content to = None
<u>FirstPersonController</u>
                                              hide_content = False  # if set to True, will display content as '*'.
EditorCamera
                                              can also be set to character instead of True.
<u>Space</u>
                                              next field = None
<u>WindowPanel</u>
                                              text_field = TextField(world_parent
Node
```

```
active = False
                                              text
                                              text color
<u>light</u> dark
                                              render()
<u>Entity</u>
                                              input(key)
<u>Text</u>
                                               background = Entity(model='quad', texture='pixelscape_combo',
Button
                                               parent=camera.ui, scale=(camera.aspect ratio,1), color=color.white)
mouse
                                               gradient = Entity(model='quad', texture='vertical gradient',
raycaster
                                               parent=camera.ui, scale=(camera.aspect_ratio,1),
string_utilities
                                               color=color.hsv(240,.6,.1,.75))
ursinastuff
                                               username_field = InputField(y=-.12, limit_content_to='0123456789')
password_field = InputField(y=-.18, hide_content=True)
<u>curve</u>
texture importer
                                               username_field.next_field = password_field
scene
<u>text</u>
<u>window</u>
                                               def submit():
                                                    print('ursername:', username_field.text)
<u>ursinamath</u>
                                                    print('password:', password_field.text)
<u>camera</u>
shader
                                               Button('Login', scale=.1, color=color.cyan.tint(-.4), y=-.26,
<u>main</u>
                                               on click=submit).fit to text()
color
input_handler
mesh importer
duplicate
<u>build</u>
                                           TrailRenderer(<u>Entity</u>)
<u>application</u>
sequence
                                              ursina/prefabs/trail_renderer
Vec3
                                              TrailRenderer(thickness=10, color=color.white, end_color=color.clear,
Empty
                                             length=6, **kwargs)
LoopingList
CubicBezier
                                              renderer = Entity(
<u>MeshModes</u>
                                                       model
<u>Mesh</u>
                                             update\_step = .025
<u>Shader</u>
<u>Audio</u>
                                              update()
Ursina
                                             on_destroy()
Color
Vec4
                                               window.color = color.black
<u>HitInfo</u>
                                               mouse.visible = False
<u>Collider</u>
                                               player = Entity()
<u>BoxCollider</u>
                                               player.graphics = Entity(parent=player, scale=.1, model='circle')
<u>SphereCollider</u>
                                               trail_renderer = TrailRenderer(parent=player, thickness=100,
<u>MeshCollider</u>
                                               color=color.yellow, length=6)
<u>Keys</u>
Vec2
                                               pivot = Entity(parent=player)
<u>Texture</u>
                                               trail renderer = TrailRenderer(parent=pivot, x=.1, thickness=20,
Light
                                               color=color.orange)
<u>DirectionalLight</u>
                                               trail renderer = TrailRenderer(parent=pivot, y=1, thickness=20,
PointLight
                                               color=color.orange)
<u>AmbientLight</u>
                                               trail_renderer = TrailRenderer(parent=pivot, thickness=2,
SpotLight
                                               color=color.orange, alpha=.5, position=(.4,.8))
<u>Trigger</u>
                                               trail_renderer = TrailRenderer(parent=pivot, thickness=2,
<u>FastMesh</u>
                                               color=color.orange, alpha=.5, position=(-.5,.7))
Func
<u>Sequence</u>
                                               def update():
                                                    player.position = lerp(player.position, mouse.position*10,
FileButton
                                               time.dt*4)
<u>FileBrowser</u>
VideoRecorder
                                                    if pivot:
<u>VideoRecorderUI</u>
                                                         pivot.rotation z -= 3
Cursor
                                                         pivot.rotation x -= 2
Draggable
<u>Tooltip</u>
                                               def input(key):
SynthGUI
                                                    if key == 'space':
<u>MemoryCounter</u>
                                                         destroy(pivot)
Panel
Tilemap
<u>Sprite</u>
DropdownMenuButton
                                            Slider(<u>Entity</u>)
<u>DropdownMenu</u>
                                              ursina/prefabs/slider
<u>Animation</u>
FrameAnimation3d
                                              Slider(min=0, max=1, default=None, height=Text.size, text='',
<u>FirstPersonController</u>
                                              dynamic=False, **kwargs)
EditorCamera
<u>Space</u>
                                             parent = camera.ui
<u>WindowPanel</u>
                                              vertical = False
Node
```

```
min = min
                                            max = max
                                            default = default
<u>light</u> dark
                                            step = 0
                                            height = height
<u>Entity</u>
                                            on value changed = None
                                                                         # set this to a function you want to be
                                            called when the slider changes
<u>Text</u>
                                            setattr = None
                                                                          # set this to (object, 'attrname') to set
Button
                                            that value when the slider changes
mouse
                                            label = Text(parent=self, origin=(0.5, 0), x=-0.025, text=text)
raycaster
                                            bg = Entity(parent=self, model=Quad(scale=(.525, height),
string_utilities
                                            radius=Text.size/2, segments=3),
                                                     origin_x=-0.25, collider='box', color=color.black66)
ursinastuff
                                            knob = Draggable(parent=self, min_x=0, max_x=.5, min_y=0, max_y=.5,
<u>curve</u>
                                            step=self.step,
texture importer
                                                    model=Quad(radius=Text.size/2, scale=(Text.size, height)),
scene
                                            <u>text</u>
<u>window</u>
                                            value = self.default
<u>ursinamath</u>
                                                                   # if set to True, will call on_value_changed()
                                            dynamic = dynamic
<u>camera</u>
                                            while dragging. if set to False, will only call on_value_changed()
shader
<u>main</u>
                                            after dragging.
color
                                            bg_click()
input_handler
mesh importer
                                            drop()
duplicate
                                            update()
<u>build</u>
                                            slide()
<u>application</u>
                                              box = Entity(model='cube', origin_y=-.5, scale=1,
sequence
                                              color=color.orange)
Vec3
                                              def scale box():
Empty
                                                  box.scale_y = slider.value
LoopingList
CubicBezier
                                                  print(thin_slider.value)
<u>MeshModes</u>
                                              slider = Slider(0, 20, default=10, height=Text.size*3, y=-.4,
<u>Mesh</u>
                                              step=1, on value changed=scale box, vertical=True)
<u>Shader</u>
<u>Audio</u>
                                              thin slider = ThinSlider(text='height', dynamic=True,
Ursina
Color
                                              on value changed=scale box)
Vec4
<u>HitInfo</u>
                                              thin slider.label.origin = (0,0)
<u>Collider</u>
                                              thin slider.label.position = (.25, -.1)
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
                                          ThinSlider(Slider)
Vec2
<u>Texture</u>
                                            ursina/prefabs/slider
<u>Light</u>
<u>DirectionalLight</u>
                                            ThinSlider(*args, **kwargs)
PointLight
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
                                             box = Entity(model='cube', origin y=-.5, scale=1,
<u>FastMesh</u>
                                              color=color.orange)
Func
<u>Sequence</u>
                                              def scale box():
                                                  box.scale_y = slider.value
FileButton
                                                  print(thin_slider.value)
<u>FileBrowser</u>
VideoRecorder
                                              slider = Slider(0, 20, default=10, height=Text.size*3, y=-.4,
<u>VideoRecorderUI</u>
                                              step=1, on value changed=scale box, vertical=True)
Cursor
<u>Draggable</u>
                                              thin slider = ThinSlider(text='height', dynamic=True,
<u>Tooltip</u>
                                              on_value_changed=scale_box)
SynthGUI
<u>MemoryCounter</u>
                                              thin_slider.label.origin = (0,0)
Panel
                                              thin slider.label.position = (.25, -.1)
Tilemap
<u>Sprite</u>
DropdownMenuButton
<u>DropdownMenu</u>
                                          RadialMenu(Entity)
<u>Animation</u>
                                            ursina/prefabs/radial menu
FrameAnimation3d
<u>FirstPersonController</u>
                                            RadialMenu(buttons=list(), **kwargs)
EditorCamera
<u>Space</u>
                                            parent = camera.ui
<u>WindowPanel</u>
                                            buttons = buttons
Node
```

```
<u>light</u> dark
<u>Entity</u>
<u>Text</u>
Button
mouse
raycaster
string_utilities
ursinastuff
<u>curve</u>
texture importer
scene
text
<u>window</u>
<u>ursinamath</u>
<u>camera</u>
shader
<u>main</u>
color
input_handler
mesh importer
duplicate
<u>build</u>
<u>application</u>
sequence
Vec3
<u>Empty</u>
LoopingList
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
Ursina
Color
Vec4
<u>HitInfo</u>
<u>Collider</u>
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
Vec2
<u>Texture</u>
<u>Light</u>
<u>DirectionalLight</u>
PointLight
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
Func
<u>Sequence</u>
FileButton
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
<u>Draggable</u>
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
Panel
Tilemap
<u>Sprite</u>
<u>DropdownMenuButton</u>
<u>DropdownMenu</u>
<u>Animation</u>
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
<u>WindowPanel</u>
Node
```

```
open at cursor = True
 open duration = .25
 bg = Panel(
          parent=self,
          model='quad',
          z = 99.
          scale=999,
          collider='box',
          color=color.color(0,0,0,.1),
          enabled=False)
 z = -99
 scale = .075
 on_enable()
 input(key)
   rm = RadialMenu(
       buttons = (
            RadialMenuButton(text='1'),
            RadialMenuButton(text='2'),
            RadialMenuButton(text='3'),
            RadialMenuButton(text='4'),
            RadialMenuButton(text='5', scale=.5),
RadialMenuButton(text='6', color=color.red),
       enabled = False
   RadialMenuButton(text='6', color=color.red,x =-.5, scale=.06),
   def enable radial menu():
        rm.enabled = True
   cube = Button(parent=scene, model='cube', color=color.orange,
   highlight_color=color.azure, on_click=enable_radial_menu)
   EditorCamera()
RadialMenuButton(<u>Button</u>)
 ursina/prefabs/radial_menu
 RadialMenuButton(**kwargs)
   rm = RadialMenu(
       buttons = (
            RadialMenuButton(text='1'),
            RadialMenuButton(text='2'),
            RadialMenuButton(text='3'),
            RadialMenuButton(text='4'),
            RadialMenuButton(text='5', scale=.5),
RadialMenuButton(text='6', color=color.red),
       enabled = False
   RadialMenuButton(text='6', color=color.red,x =-.5, scale=.06),
   def enable radial menu():
       rm.enabled = True
   cube = Button(parent=scene, model='cube', color=color.orange,
   highlight_color=color.azure, on_click=enable_radial_menu)
   EditorCamera()
ButtonList(<u>Entity</u>)
 ursina/prefabs/button list
 ButtonList(button_dict, button_height=1.1, fit_height=True, width=.5,
 font=Text.default_font, **kwargs)
 fit height = fit height
 button height = button height
 text_entity = Text(parent=self, font=font, origin=(-.5,.5),
text='empty', world_scale=20, z=-.1, x=.01,
 line height=button height)
 button_height = self.text_entity.height
 button_dict = button_dict
 highlight = Entity(parent=self, model='quad', color=color.white33,
```

scale=(1,self.button\_height), origin=(-.5,.5), z=-.01,

```
add to scene entities=False)
                                                    selection marker = Entity(parent=self, model='quad',
                                                    color=color.azure, scale=(1,self.button_height), origin=(-.5,.5),
<u>light</u> dark
                                                    z=-.02, enabled=False, add to scene entities=False)
<u>Entity</u>
                                                    input(key)
                                                    update()
<u>Text</u>
                                                    on disable()
Button
mouse
                                                      default = Func(print, 'not yet implemented')
raycaster
string_utilities
                                                      def test(a=1, b=2):
                                                           print('----:', a, b)
ursinastuff
<u>curve</u>
                                                      button_dict = {
texture importer
                                                            'one':
                                                                           None.
scene
                                                           'two':
                                                                           default,
<u>text</u>
                                                                           Func(test, 3, 4),
Func(test, b=3, a=4),
<u>window</u>
                                                            'tree':
                                                           'four' :
<u>ursinamath</u>
<u>camera</u>
                                                      for i in range(6, 20):
   button_dict[f'button {i}'] = Func(print, i)
shader
<u>main</u>
color
                                                      sound_effects = {}
input_handler
                                                      current_sound = None
mesh importer
duplicate
                                                      model_names = ['cube', 'sphere', 'plane', 'test', 'test_2',
  'test_3', 'test_4', 'test_5', 'another_test','player_idle']
model_dict = {name : Func(print, name) for name in model_names}
<u>build</u>
application
sequence
Vec3
                                                      bl = ButtonList(model dict, font='VeraMono.ttf')
Empty
LoopingList
CubicBezier
<u>MeshModes</u>
                                                  SpriteSheetAnimation(Entity)
<u>Mesh</u>
<u>Shader</u>
                                                    ursina/prefabs/sprite_sheet_animation
<u>Audio</u>
Ursina
                                                    SpriteSheetAnimation(texture, animations, tileset_size=[4,1], fps=12,
model='quad', autoplay=True, **kwargs)
Color
<u>Vec4</u>
<u>HitInfo</u>
                                                    animations = animations # should be a dict
<u>Collider</u>
<u>BoxCollider</u>
                                                    play_animation(animation_name)
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
                                                      (0,0) is in bottom left
Vec2
<u>Texture</u>
                                                      from ursina import Ursina
<u>Light</u>
                                                      player_graphics = SpriteSheetAnimation('sprite_sheet',
<u>DirectionalLight</u>
                                                      tileset_size=(4,4), fps=6, animations={
                                                           'idle': ((0,0), (0,0)),

'walk_up': ((0,0), (3,0)),

'walk_right': ((0,1), (3,1)),

'walk_left': ((0,2), (3,2)),

'walk_down': ((0,3), (3,3)),
PointLight
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
Func
<u>Sequence</u>
                                                      def input(key):
FileButton
                                                           if key == 'w':
                                                           player_graphics.play_animation('walk_up')
elif key == 's':
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
                                                                player_graphics.play_animation('walk_down')
<u>Cursor</u>
                                                           elif key == 'd':
<u>Draggable</u>
                                                                player_graphics.play_animation('walk_right')
<u>Tooltip</u>
                                                           elif key == 'a':
SynthGUI
                                                                 player_graphics.play_animation('walk_left')
<u>MemoryCounter</u>
Panel
Tilemap
<u>Sprite</u>
                                                      Entity(model='quad', texture='sprite sheet', x=-1)
DropdownMenuButton
<u>DropdownMenu</u>
Animation
FrameAnimation3d
                                                  DebugMenu(<u>Draggable</u>)
<u>FirstPersonController</u>
                                                    ursina/prefabs/debug_menu
EditorCamera
<u>Space</u>
                                                    DebugMenu(target, **kwargs)
<u>WindowPanel</u>
Node
```

```
target = target
                                             scale = (.2, .025)
                                             text = '<orange>' + target.__class__.__name_
<u>light</u> dark
                                             draw functions()
<u>Entity</u>
                                               DebugMenu(Audio('night sky'))
<u>Text</u>
Button
mouse
raycaster
string_utilities
                                           GridEditor(Entity)
ursinastuff
                                             ursina/prefabs/grid_editor
<u>curve</u>
texture importer
                                             GridEditor(size=(32,32), palette=(' ', '#', '|', 'o'), **kwargs)
scene
<u>text</u>
                                             w, self.h = int(size[0]), int(size[1])
<u>window</u>
                                             brush size = 1
<u>ursinamath</u>
                                             auto render = True
<u>camera</u>
                                             cursor = Entity(parent=self, model=Quad(segments=0, mode='line'),
shader
                                             origin=(-.5,-.5), scale=(1/self.w, 1/self.h),
<u>main</u>
                                             color=color.color(0,1,1,.5), z=-.1)
color
                                             selected char = palette[1]
input_handler
                                             palette = palette
mesh importer
                                             prev draw = None
duplicate
                                             start_pos = (0,0)
<u>build</u>
                                             outline = Entity(parent=self, model=Quad(segments=0, mode='line',
<u>application</u>
                                             thickness=1), color=color.cyan, z=.01, origin=(-.5,-.5))
sequence
                                             undo_cache = list()
                                             undo_index = 0
Vec3
                                             help_text = Text(
Empty
                                                      text=dedent('''
LoopingList
                                                          left mouse:
                                                                           draw
CubicBezier
                                                          control(hold): draw lines
<u>MeshModes</u>
                                                          alt(hold):
                                                                           select character
<u>Mesh</u>
                                                          ctrl + z:
                                                                           undo
<u>Shader</u>
                                                          ctrl + y:
                                                                            redo
                                                      '''),
<u>Audio</u>
Ursina
                                                      position=window.top_left,
Color
                                                      scale=.75
<u>Vec4</u>
<u>HitInfo</u>
                                             edit mode = True
<u>Collider</u>
<u>BoxCollider</u>
                                             update()
<u>SphereCollider</u>
                                             draw(x, y)
<u>MeshCollider</u>
                                             input(key)
<u>Keys</u>
                                             record_undo()
Vec2
                                             floodfill(matrix, x, y, first=True)
<u>Texture</u>
<u>Light</u>
<u>DirectionalLight</u>
                                               pixel editor example, it's basically a drawing tool.
PointLight
                                               can be useful for level editors and such
<u>AmbientLight</u>
                                               here we create a new texture, but can also give it an existing
SpotLight
                                               texture to modify.
<u>Trigger</u>
<u>FastMesh</u>
                                               from PIL import Image
Func
                                               t = Texture(Image.new(mode='RGBA', size=(32,32), color=(0,0,0,1)))
<u>Sequence</u>
                                               from ursina.prefabs.grid editor import PixelEditor
                                               PixelEditor(texture=load_texture('brick'))
FileButton
<u>FileBrowser</u>
VideoRecorder
                                               same as the pixel editor, but with text.
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                               from ursina.prefabs.grid editor import ASCIIEditor
<u>Draggable</u>
                                               ASCIIEditor()
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
Panel
Tilemap
                                           PixelEditor(GridEditor)
<u>Sprite</u>
                                             ursina/prefabs/grid_editor
DropdownMenuButton
                                             PixelEditor(texture, palette=(color.black, color.white,
<u>DropdownMenu</u>
                                             color.light_gray, color.gray, color.red, color.orange, color.yellow,
<u>Animation</u>
                                             color.lime, color.green, color.turquoise, color.cyan, color.azure,
FrameAnimation3d
                                             color.blue, color.violet, color.magenta, color.pink), **kwargs)
<u>FirstPersonController</u>
EditorCamera
                                             grid = [[texture.get_pixel(x,y) for y in range(texture.height)] for x
<u>Space</u>
                                             in range(texture.width)]
<u>WindowPanel</u>
Node
```

```
save()
<u>light</u> dark
                                                   pixel editor example, it's basically a drawing tool.
                                                   can be useful for level editors and such
<u>Entity</u>
                                                  here we create a new texture, but can also give it an existing
<u>Text</u>
                                                   texture to modify.
Button
<u>mouse</u>
                                                  from PIL import Image
raycaster
                                                   t = Texture(Image.new(mode='RGBA', size=(32,32), color=(0,0,0,1)))
string_utilities
                                                   from ursina.prefabs.grid_editor import PixelEditor
                                                   PixelEditor(texture=load texture('brick'))
ursinastuff
<u>curve</u>
texture importer
                                                   same as the pixel editor, but with text.
scene
<u>text</u>
<u>window</u>
                                                   from ursina.prefabs.grid editor import ASCIIEditor
                                                  ASCIIEditor()
<u>ursinamath</u>
<u>camera</u>
shader
<u>main</u>
color
                                              ASCIIEditor(GridEditor)
<u>input_handler</u>
mesh importer
                                                 ursina/prefabs/grid editor
duplicate
                                                ASCIIEditor(size=(61,28), palette=(' ', '#', '|', 'A', '/', '\\', 'o', '_', '-', 'i', 'M', '.'), font='VeraMono.ttf', color=color.black, line_height=1.1, **kwargs)
<u>build</u>
<u>application</u>
sequence
Vec3
                                                 text entity = Text(parent=self.parent, text=text, x=self.x,
Empty
                                                 line_height=line_height, font=font)
<u>LoopingList</u>
                                                 scale = (self.text entity.width, self.text entity.height)
CubicBezier
<u>MeshModes</u>
                                                 render()
<u>Mesh</u>
                                                input(key)
<u>Shader</u>
<u>Audio</u>
Ursina
                                                  pixel editor example, it's basically a drawing tool. can be useful for level editors and such
Color
<u>Vec4</u>
                                                  here we create a new texture, but can also give it an existing
<u>HitInfo</u>
                                                   texture to modify.
<u>Collider</u>
<u>BoxCollider</u>
                                                   from PIL import Image
<u>SphereCollider</u>
                                                   t = Texture(Image.new(mode='RGBA', size=(32,32), color=(0,0,0,1)))
<u>MeshCollider</u>
                                                   from ursina.prefabs.grid_editor import PixelEditor
<u>Keys</u>
                                                  PixelEditor(texture=load_texture('brick'))
Vec2
<u>Texture</u>
Light
                                                   same as the pixel editor, but with text.
<u>DirectionalLight</u>
PointLight
                                                   from ursina.prefabs.grid_editor import ASCIIEditor
<u>AmbientLight</u>
                                                   ASCIIEditor()
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
Func
<u>Sequence</u>
                                              HotReloader(Entity)
                                                 ursina/prefabs/hot_reloader
FileButton
<u>FileBrowser</u>
                                                HotReloader(path=__file__, **kwargs)
VideoRecorder
<u>VideoRecorderUI</u>
                                                path = path
<u>Cursor</u>
                                                 path = Path(self.path)
Draggable
                                                 hotreload = False # toggle with f9
<u>Tooltip</u>
                                                hotkeys = {
SynthGUI
                                                          'ctrl+r' : self.reload_code,
<u>MemoryCounter</u>
                                                                     : self.reload_code,
: self.reload_textures,
                                                           'f5'
Panel
Tilemap
                                                           'f7'
                                                                     : self.reload_models,
<u>Sprite</u>
                                                                     : self.reload_shaders,: self.toggle_hotreloading,
                                                           'f8'
                                                           'f9'
DropdownMenuButton
<u>DropdownMenu</u>
<u>Animation</u>
                                                 input(key)
FrameAnimation3d
                                                 update()
<u>FirstPersonController</u>
                                                 get_source_code()
EditorCamera
                                                 toggle_hotreloading()
<u>Space</u>
                                                 reload_code(reset_camera=True)
<u>WindowPanel</u>
                                                 reload_textures()
Node
```

render()

```
reload models()
                                              reload shaders()
<u>light</u> dark
                                                application.hot reloader.path =
                                                application.asset_folder.parent.parent / 'samples' /
<u>Entity</u>
                                                'platformer.py'
<u>Text</u>
Button
                                                By default you can press F5 to reload the starting script, F6 to
mouse
raycaster
                                                reimport textures and F7 to reload models.
string_utilities
ursinastuff
<u>curve</u>
texture importer
                                            ExitButton(Button)
scene
<u>text</u>
                                              ursina/prefabs/exit_button
<u>window</u>
<u>ursinamath</u>
                                              ExitButton(**kwargs)
<u>camera</u>
shader
<u>main</u>
                                              on click()
color
                                              input(key)
input_handler
mesh importer
duplicate
                                                This is the button in the upper right corner.
<u>build</u>
                                                You can click on it or press Shift+Q to close the program.
<u>application</u>
                                                To disable it, set window.exit_button.enabled to False
sequence
Vec3
Empty
LoopingList
CubicBezier
                                            combine
<u>MeshModes</u>
                                              ursina/scripts/combine
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
                                              combine(combine_parent, analyze=False, auto_destroy=True, ignore=[])
Ursina
Color
                                                p = Entity()
<u>Vec4</u>
                                                e1 = Entity(parent=p, model='sphere', y=1.5, color=color.pink)
<u>HitInfo</u>
                                                e2 = Entity(parent=p, model='cube', color=color.yellow, x=1,
<u>Collider</u>
<u>BoxCollider</u>
                                                e3 = Entity(parent=e2, model='cube', color=color.yellow, y=2,
<u>SphereCollider</u>
                                                scale=.5)
<u>MeshCollider</u>
<u>Keys</u>
                                                def input(key):
Vec2
                                                     if key == 'space':
<u>Texture</u>
                                                         from time import perf_counter
<u>Light</u>
                                                         t = perf counter()
<u>DirectionalLight</u>
                                                         p.combine()
PointLight
                                                         print('combined in:', perf counter() - t)
<u>AmbientLight</u>
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
                                                EditorCamera()
Func
<u>Sequence</u>
FileButton
<u>FileBrowser</u>
                                            SmoothFollow()
VideoRecorder
<u>VideoRecorderUI</u>
                                              ursina/scripts/smooth follow
<u>Cursor</u>
<u>Draggable</u>
                                              SmoothFollow(target=None, offset=(0,0,0), speed=8, rotation_speed=0,
<u>Tooltip</u>
                                              rotation_offset=(0,0,0))
SynthGUI
<u>MemoryCounter</u>
                                              target = target
Panel
                                              offset = offset
Tilemap
                                              speed = speed
<u>Sprite</u>
                                              rotation speed = rotation speed
                                              rotation\_offset = rotation\_offset
<u>DropdownMenuButton</u>
<u>DropdownMenu</u>
                                              update()
Animation
FrameAnimation3d
                                                player = Entity(model='cube', color=color.orange)
<u>FirstPersonController</u>
EditorCamera
                                                def update():
<u>Space</u>
                                                     player.x += held keys['d'] * .1
<u>WindowPanel</u>
                                                     player.x -= held_keys['a'] * .1
Node
```

```
e = Entity(model='cube')
                                                sf = e.add_script(SmoothFollow(target=player, offset=(0,2,0)))
<u>light</u> dark
                                                def input(key):
<u>Entity</u>
                                                     global sf
                                                     if key == '1' and sf in e.scripts:
<u>Text</u>
                                                         e.scripts.remove(sf)
Button
mouse
                                                EditorCamera()
raycaster
string_utilities
ursinastuff
<u>curve</u>
                                            PositionLimiter()
texture importer
scene
                                              ursina/scripts/position limiter
<u>text</u>
<u>window</u>
                                              PositionLimiter(min_x=-math.inf, max_x=math.inf, min_y=-math.inf,
<u>ursinamath</u>
                                              max_y=math.inf, min_z=-math.inf, max_z=math.inf)
<u>camera</u>
shader
                                              min x = min x
<u>main</u>
                                              \max x = \max x
color
                                              min_y = min_y
input_handler
                                              \max y = \max y
mesh importer
                                              min z = min z
duplicate
                                              max_z = max_z
<u>build</u>
application
                                              update()
sequence
Vec3
Empty
LoopingList
                                            generate normals
CubicBezier
                                              ursina/scripts/generate normals
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
                                              normalize v3(arr)
<u>Audio</u>
                                              generate_normals(vertices, triangles=None, smooth=True)
Ursina
Color
                                                vertices = (
Vec4
                                                     (-0.0, -0.5, 0.0), (0.1, -0.48, -0.073), (-0.038, -0.48,
<u>HitInfo</u>
                                                -0.11),
<u>Collider</u>
                                                     (0.361804, -0.22, -0.26), (0.3, -0.32, -0.22), (0.40, -0.25,
<u>BoxCollider</u>
                                                -0.14),
<u>SphereCollider</u>
                                                     (-0.0, -0.5, 0.0), (-0.038, -0.48, -0.11), (-0.03, -0.48,
<u>MeshCollider</u>
                                                -0.11)
<u>Keys</u>
Vec2
                                                norms = generate normals(vertices)
<u>Texture</u>
<u>Light</u>
<u>DirectionalLight</u>
PointLight
<u>AmbientLight</u>
                                            NoclipMode
SpotLight
<u>Trigger</u>
                                              ursina/scripts/noclip mode
<u>FastMesh</u>
Func
                                              NoclipMode(speed=10, require_key='shift')
<u>Sequence</u>
                                              speed = speed
FileButton
                                              require_key = require_key
<u>FileBrowser</u>
                                              ignore paused = True
VideoRecorder
<u>VideoRecorderUI</u>
                                              input(key)
<u>Cursor</u>
                                              update()
<u>Draggable</u>
<u>Tooltip</u>
                                                player = Entity(model='cube', color=color.orange)
SynthGUI
                                                Entity(model='plane', scale=10)
<u>MemoryCounter</u>
                                                EditorCamera()
Panel
Tilemap
<u>Sprite</u>
                                                player.add script(NoclipMode2d())
<u>Sky</u>
DropdownMenuButton
<u>DropdownMenu</u>
Animation
FrameAnimation3d
                                            NoclipMode2d
<u>FirstPersonController</u>
                                              ursina/scripts/noclip_mode
EditorCamera
<u>Space</u>
                                              NoclipMode2d(speed=10, require_key='shift')
<u>WindowPanel</u>
Node
```

```
require key = require key
                                               ignore_paused = True
<u>light</u> dark
                                               input(key)
Entity
                                               update()
<u>Text</u>
                                                player = Entity(model='cube', color=color.orange)
Button
                                                 Entity(model='plane', scale=10)
<u>mouse</u>
                                                 EditorCamera()
raycaster
string_utilities
ursinastuff
                                                player.add script(NoclipMode2d())
<u>curve</u>
texture importer
scene
<u>text</u>
<u>window</u>
                                             chunk mesh
\underline{\text{ursinamath}}
                                               ursina/scripts/chunk mesh
<u>camera</u>
shader
                                               app = Ursina()
<u>main</u>
                                               t = time.time()
color
                                               application.asset_folder = application.asset_folder.parent.parent
input_handler
                                               terrain = Entity(model=Terrain('heightmap_1', skip=8),
mesh importer
                                               texture='grass', texture scale=(3,3), scale=256)
<u>duplicate</u>
                                               grid = [[None for z in range(8)] for x in range(8)] # make 2d array
<u>build</u>
                                               of entities
<u>application</u>
                                              x  slices = 8
sequence
                                               z_slices = 8
                                               terrain.model.generated vertices = [v+Vec3(.5,0.5)] for v in
Vec3
                                               terrain.model.generated vertices]
Empty
                                               EditorCamera()
LoopingList
                                               app.run()
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
<u>Audio</u>
Ursina
                                             project uvs
Color
                                               ursina/scripts/project_uvs
<u>Vec4</u>
<u>HitInfo</u>
<u>Collider</u>
                                               project uvs(model, aspect ratio=1, direction='forward',
<u>BoxCollider</u>
                                               regenerate=False)
<u>SphereCollider</u>
<u>MeshCollider</u>
                                                 e = Entity(model='sphere', texture='ursina_logo')
<u>Keys</u>
                                                 project_uvs(e.model)
Vec2
                                                 EditorCamera()
<u>Texture</u>
Light
<u>DirectionalLight</u>
PointLight
<u>AmbientLight</u>
                                             terraincast
SpotLight
<u>Trigger</u>
                                               ursina/scripts/terraincast
<u>FastMesh</u>
Func
<u>Sequence</u>
                                               terraincast(world position, terrain entity, height values=None)
                                               uses x and z to return y on terrain.
FileButton
                                                terrain_entity = Entity(model=Terrain('heightmap_1', skip=8),
scale=(40, 5, 20), texture='heightmap_1')
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
                                                 player = Entity(model='sphere', color=color.azure, scale=.2,
<u>Cursor</u>
                                                 origin y=-.5)
Draggable
<u>Tooltip</u>
SynthGUI
                                                hv = terrain entity.model.height values
<u>MemoryCounter</u>
Panel
                                                 def update():
Tilemap
                                                     direction = Vec3(held keys['d'] - held keys['a'], 0,
<u>Sprite</u>
                                                 held keys['w'] - held keys['s']).normalized()
                                                     player.position += direction * time.dt * 4
DropdownMenuButton
<u>DropdownMenu</u>
                                                     player.y = terraincast(player.world position, terrain entity,
<u>Animation</u>
                                                hv)
FrameAnimation3d
<u>FirstPersonController</u>
                                                EditorCamera()
EditorCamera
                                                 Sky()
<u>Space</u>
<u>WindowPanel</u>
Node
```

speed = speed

## colorize ursina/scripts/colorize <u>light</u> dark **Entity** get world normals(model) <u>Text</u> colorize(model, left=color.white, right=color.blue, down=color.red, Button up=color.green, back=color.white, forward=color.white, smooth=True, mouse world\_space=True, strength=1) raycaster import random string\_utilities for i in range(10): ursinastuff e = Entity(model=load model('sphere', <u>curve</u> path=application.internal models compressed folder, texture importer use\_deepcopy=True)) scene e.position = <u>text</u> (random.uniform(-3,3), random.uniform(-3,3), random.uniform(-3,3)) <u>window</u> e.rotation = <u>ursinamath</u> (random.uniform(0,360), random.uniform(0,360), random.uniform(0,360))<u>camera</u> e.scale = random.uniform(1.3) shader e.model.colorize(smooth=False, world space=True, strength=.5) <u>main</u> color input\_handler Sky(color=color.gray) mesh importer EditorCamera() duplicate <u>build</u> <u>application</u> sequence grid layout Vec3 ursina/scripts/grid layout **Empty LoopingList** CubicBezier <u>MeshModes</u> grid layout(l, max x=8, max y=8, spacing=(0,0,0), origin=(-.5,.5,0), <u>Mesh</u> offset=(0,0,0)<u>Shader</u> <u>Audio</u> center = Entity(model='quad', scale=.1, color=color.red) Ursina p = Entity() Color for i in range(4\*5): <u>Vec4</u> b = Button(parent=p, model='quad', scale=.5, scale x=1, <u>HitInfo</u> text=str(i), color=color.tint(color.random color(),-.6)) <u>Collider</u> b.text\_entity.world\_scale = 1 <u>BoxCollider</u> t = time.time() <u>SphereCollider</u> grid layout(p.children, max x=7, max y=10, origin=(0, .5)) <u>MeshCollider</u> center = Entity(parent=camera.ui, model=Circle(), scale=.005, <u>Keys</u> color=color.lime) Vec2 EditorCamera() <u>Texture</u> print(time.time() - t) <u>Light</u> <u>DirectionalLight</u> **PointLight** <u>AmbientLight</u> **SpotLight** Scrollable() <u>Trigger</u> ursina/scripts/scrollable <u>FastMesh</u> Func Scrollable(\*\*kwargs) <u>Sequence</u> max = inf**FileButton** min = -inf<u>FileBrowser</u> $scroll_speed = .05$ VideoRecorder scroll\_smoothing = 16 <u>VideoRecorderUI</u> axis = 'y'<u>Cursor</u> target\_value = None <u>Draggable</u> <u>Tooltip</u> update() **SynthGUI** input(key) <u>MemoryCounter</u> Panel Tilemap This will make target entity move up or down when you hover the <u>Sprite</u> entity/its children while scrolling the scroll wheel. **DropdownMenuButton** <u>DropdownMenu</u> <u>Animation</u> p = Button(model='quad', scale=(.4, .8), collider='box') FrameAnimation3d for i in range(8): <u>FirstPersonController</u> Button(parent=p , scale\_y=.05, text=f'giopwjoigjwr{i}', **EditorCamera** origin y=.5, y=.5-(i\*.05)) <u>Space</u> <u>WindowPanel</u> p.add script(Scrollable()) Node

Node

```
<u>light</u> dark
                                               merge vertices
Entity
                                                 ursina/scripts/merge_vertices
<u>Text</u>
Button
<u>mouse</u>
                                                 distance(a, b)
raycaster
                                                 merge overlapping vertices(vertices, triangles=None, max distance=.1)
string_utilities
                                                   verts = ((0,0,0), (1,0,0), (1,1,0), (0,0,0), (1,1,0), (0,1,0))
ursinastuff
                                                   tris = (0,1,2,3,4,5)
<u>curve</u>
texture importer
                                                   new verts, new tris = merge overlapping vertices(verts, tris)
scene
                                                   print('verts:', (verts), (new_verts))
print('tris:', (tris), (new_tris))
<u>text</u>
window
\underline{\text{ursinamath}}
<u>camera</u>
                                                   e = Entity(model=Mesh(new_verts, new_tris, mode='triangle'))
shader
                                                   EditorCamera()
<u>main</u>
color
input_handler
mesh importer
duplicate
                                               models
<u>build</u>
                                                  'line'
<u>application</u>
                                                  ' guad '
sequence
                                                  'wireframe cube'
                                                  'plane'
Vec3
                                                  'circle'
Empty
                                                  'diamond'
<u>LoopingList</u>
                                                  'wireframe_quad'
CubicBezier
                                                  'sphere'
<u>MeshModes</u>
                                                  'cube'
<u>Mesh</u>
                                                  'icosphere'
<u>Shader</u>
                                                  'cube uv top'
Audio
                                                  'arrow'
Ursina
                                                  'sky_dome'
Color
                                                  'scale_gizmo'
Vec4
<u>HitInfo</u>
<u>Collider</u>
                                                   e = Entity(model='quad')
<u>BoxCollider</u>
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
Vec2
                                               textures
<u>Texture</u>
Light
                                                  'cursor'
<u>DirectionalLight</u>
                                                  'noise'
PointLight
                                                  'grass'
<u>AmbientLight</u>
                                                  'vignette'
                                                 'arrow_right'
'test_tileset'
SpotLight
<u>Trigger</u>
<u>FastMesh</u>
                                                  'tilemap test level'
<u>Func</u>
                                                  'shore'
<u>Sequence</u>
                                                  'file icon'
                                                  'sky_sunset'
FileButton
                                                  'radial_gradient'
<u>FileBrowser</u>
                                                  'circle
VideoRecorder
                                                  'perlin_noise'
<u>VideoRecorderUI</u>
                                                  'brick'
<u>Cursor</u>
                                                  'ursina wink 0001'
                                                  'grass_tintable'
Draggable
<u>Tooltip</u>
                                                  'circle_outlined'
SynthGUI
                                                  'ursina_logo'
<u>MemoryCounter</u>
                                                  'arrow_down
Panel
                                                  'reflection_map_3'
Tilemap
                                                  'cog'
<u>Sprite</u>
                                                  'vertical gradient'
                                                  'white_cube'
DropdownMenuButton
                                                  'horizontal gradient'
DropdownMenu
                                                  'folder'
Animation
                                                  'ursina_wink_0000'
FrameAnimation3d
                                                  'rainbow'
<u>FirstPersonController</u>
                                                  'heightmap 1'
EditorCamera
                                                  'sky_default'
<u>Space</u>
<u>WindowPanel</u>
```

```
e = Entity(model='cube', texture='brick')
```

```
<u>light</u> dark
                                             shaders
<u>Entity</u>
<u>Text</u>
                                               basic lighting shader
Button
                                               colored_lights_shader
mouse
                                               fresnel shader
raycaster
                                              projector_shader
                                               instancing_shader
string_utilities
                                              texture blend shader
ursinastuff
                                              matcap_shader
<u>curve</u>
                                               triplanar_shader
texture importer
                                              unlit shader
scene
                                              geom shader
<u>text</u>
                                              normals shader
<u>window</u>
                                              transition shader
<u>ursinamath</u>
                                              noise fog shader
<u>camera</u>
                                               lit_with_shadows_shader
shader
                                              fxaa
<u>main</u>
                                              camera empty
color
                                              ssao
<u>input_handler</u>
                                              camera outline shader
mesh importer
                                              pixelation shader
duplicate
                                              camera\_contrast
<u>build</u>
                                               camera vertical blur
application
                                              camera_grayscale
sequence
Vec3
                                                from ursina.shaders import normals shader
Empty
                                                e = Entity(shader=normals_shader)
LoopingList
CubicBezier
<u>MeshModes</u>
<u>Mesh</u>
<u>Shader</u>
                                             Pipe(Mesh)
<u>Audio</u>
                                               ursina/models/procedural/pipe
Ursina
Color
                                              \label{eq:pipe} Pipe(base\_shape=Quad, origin=(0,0), path=((0,0,0),(0,1,0)),\\
<u>Vec4</u>
                                               thicknesses=((1,1),), look_at=True, cap_ends=True, mode='triangle',
<u>HitInfo</u>
                                               **kwargs)
<u>Collider</u>
<u>BoxCollider</u>
                                              base shape = base shape
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
                                                path = (Vec3(0,0,0), Vec3(0,1,0), Vec3(0,3,0), Vec3(0,4,0),
Vec2
                                                Vec3(2,5,0))
<u>Texture</u>
                                                thicknesses = ((1,1), (.5,.5), (.75,.75), (.5,.5), (1,1))
<u>Light</u>
                                                e = Entity(model=Pipe(path=path, thicknesses=thicknesses))
<u>DirectionalLight</u>
                                                e.model.colorize()
PointLight
<u>AmbientLight</u>
                                                EditorCamera()
SpotLight
                                                origin = Entity(model='cube', color=color.magenta)
<u>Trigger</u>
                                                origin.scale *= .25
<u>FastMesh</u>
Func
<u>Sequence</u>
FileButton
                                             Plane(Mesh)
<u>FileBrowser</u>
VideoRecorder
                                               ursina/models/procedural/plane
<u>VideoRecorderUI</u>
<u>Cursor</u>
                                              Plane(subdivisions=(1,1), mode='triangle', **kwargs)
<u>Draggable</u>
<u>Tooltip</u>
                                              vertices, self.triangles = list(), list()
SynthGUI
                                              uvs = list()
<u>MemoryCounter</u>
Panel
Tilemap
                                                front = Entity(model=Plane(subdivisions=(3,6)), texture='brick',
<u>Sprite</u>
                                                rotation x=-90)
<u>Sky</u>
DropdownMenuButton
                                                 ed = EditorCamera()
<u>DropdownMenu</u>
                                                Entity(model='cube', color=color.green, scale=.05)
Animation
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
                                             Circle(Mesh)
<u>WindowPanel</u>
                                               ursina/models/procedural/circle
Node
```

Node

```
Circle(resolution=16, radius=.5, rotate=True, mode='ngon', **kwargs)
<u>light</u> dark
                                             vertices = list()
Entity
                                               e = Entity(model=Circle(8, mode='line', thickness=10),
<u>Text</u>
                                               color=color(60,1,1,.3))
Button
mouse
                                               print(e.model.recipe)
                                               origin = Entity(model='quad', color=color.orange, scale=(.05, .05))
<u>raycaster</u>
                                               ed = EditorCamera(rotation_speed = 200, panning_speed=200)
string_utilities
ursinastuff
<u>curve</u>
texture importer
                                           QuadMesh (Mesh)
scene
text
                                             ursina/models/procedural/quad
window
<u>ursinamath</u>
                                             QuadMesh(radius=.1, segments=8, aspect=1, scale=(1,1), mode='ngon',
<u>camera</u>
                                             thickness=1)
shader
<u>main</u>
                                             vertices = [Vec3(0,0,0), Vec3(1,0,0), Vec3(1,1,0), Vec3(0,1,0)]
color
                                             radius = radius
input_handler
                                             mode = mode
mesh importer
                                             thickness = thickness
duplicate
                                             uvs = list()
<u>build</u>
                                             vertices = [(v[0]-offset[0], v[1]-offset[1], v[2]-offset[2]) for v in
application
                                             self.vertices]
sequence
Vec3
                                               from time import perf_counter
Empty
                                               t = perf_counter()
LoopingList
                                               for i in range(100):
CubicBezier
                                                   Entity(model=Quad(scale=(3,1), thickness=3, segments=3,
<u>MeshModes</u>
                                               mode='line'), color = color.color(0,1,1,.7))
<u>Mesh</u>
                                               print('----', (perf_counter() - t))
<u>Shader</u>
<u>Audio</u>
                                               origin = Entity(model='quad', color=color.orange, scale=(.05, .05))
Ursina
Color
                                               Entity(model=Quad(0), texture='shore', x=-1)
<u>Vec4</u>
<u>HitInfo</u>
                                               camera.z = -5
<u>Collider</u>
BoxCollider
<u>SphereCollider</u>
<u>MeshCollider</u>
<u>Keys</u>
                                           Cone (Mesh)
Vec2
                                             ursina/models/procedural/cone
<u>Texture</u>
<u>Light</u>
                                             Cone(resolution=4, radius=.5, height=1, add_bottom=True,
<u>DirectionalLight</u>
                                             mode='triangle', **kwargs)
<u>PointLight</u>
<u>AmbientLight</u>
SpotLight
Trigger
                                               from ursina import Ursina, Entity, color, EditorCamera
<u>FastMesh</u>
                                               e = Entity(model=Cone(3), texture='brick')
Func
<u>Sequence</u>
                                               origin = Entity(model='quad', color=color.orange, scale=(.05, .05))
FileButton
                                               ed = EditorCamera()
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
<u>Cursor</u>
<u>Draggable</u>
                                           Grid(Mesh)
<u>Tooltip</u>
<u>SynthGUI</u>
                                             ursina/models/procedural/grid
<u>MemoryCounter</u>
Panel
                                             Grid(width, height, mode='line', thickness=1, **kwargs)
Tilemap
<u>Sprite</u>
                                             width = width
<u>Sky</u>
                                             height = height
DropdownMenuButton
<u>DropdownMenu</u>
Animation
                                               Entity(model=Grid(2, 6))
FrameAnimation3d
<u>FirstPersonController</u>
<u>EditorCamera</u>
<u>Space</u>
<u>WindowPanel</u>
                                           Cylinder(Pipe)
```

## ursina/models/procedural/cylinder

```
Cylinder(resolution=8, radius=.5, start=0, height=1, direction=
<u>light</u> dark
                                              (0,1,0), mode='triangle', **kwargs)
Entity
<u>Text</u>
                                                Entity(model=Cylinder(6, start=-.5), color=color.color(60,1,1,.3))
Button
                                               origin = Entity(model='quad', color=color.orange, scale=(5, .05))
<u>mouse</u>
                                                ed = EditorCamera(rotation speed = 200, panning speed=200)
raycaster
string_utilities
ursinastuff
<u>curve</u>
                                            Terrain(Mesh)
texture importer
scene
                                              ursina/models/procedural/terrain
<u>text</u>
<u>window</u>
                                              Terrain(heightmap='', height values=None, skip=1, **kwargs)
\underline{\text{ursinamath}}
<u>camera</u>
                                              width = len(self.height values)
shader
                                              depth = len(self.height_values[0])
<u>main</u>
                                              aspect ratio = self.width / self.depth
color
<u>input_handler</u>
                                              generate()
mesh importer
duplicate
                                                '''Terrain using an RGB texture as input'''
<u>build</u>
                                                terrain from heightmap texture =
<u>application</u>
                                                Entity(model=Terrain('heightmap 1', skip=8), scale=(40,5,20),
sequence
                                                texture='heightmap_1')
Vec3
Empty
                                               \ensuremath{\text{I'm}} just getting the height values from the previous terrain as an
LoopingList
                                                example, but you can provide your own.
CubicBezier
                                                It should be a list of lists, where each value is between 0 and
<u>MeshModes</u>
                                                255.
<u>Mesh</u>
<u>Shader</u>
                                               hv = terrain_from_heightmap_texture.model.height_values.tolist()
Audio
                                                terrain from list = Entity(model=Terrain(height values=hv), scale=
Ursina
                                                (40,5,20), texture='heightmap_1', x=40)
Color
Vec4
                                                def input(key):
<u>HitInfo</u>
                                                    if key == 'space': # randomize the terrain
<u>Collider</u>
                                                         terrain from list.model.height values =
<u>BoxCollider</u>
                                                [[random.uniform(0,255) for a in column] for column in
<u>SphereCollider</u>
                                                terrain_from_list.model.height_values]
<u>MeshCollider</u>
                                                         terrain from list.model.generate()
<u>Keys</u>
Vec2
                                                EditorCamera()
<u>Texture</u>
                                                Sky()
<u>Light</u>
<u>DirectionalLight</u>
                                                player = Entity(model='sphere', color=color.azure, scale=.2,
PointLight
                                                origin v=-.5)
<u>AmbientLight</u>
                                               hv = terrain_from_list.model.height_values
SpotLight
<u>Trigger</u>
                                               def update():
<u>FastMesh</u>
                                                    direction = Vec3(held keys['d'] - held keys['a'], 0,
Func
                                               held keys['w'] - held_keys['s']).normalized()
Sequence
                                                    player.position += direction * time.dt * 8
                                                    player.y = terraincast(player.world_position,
FileButton
                                                terrain_from_list, hv)
<u>FileBrowser</u>
VideoRecorder
<u>VideoRecorderUI</u>
Cursor
<u>Draggable</u>
<u>Tooltip</u>
SynthGUI
<u>MemoryCounter</u>
Panel
Tilemap
<u>Sprite</u>
<u>DropdownMenuButton</u>
<u>DropdownMenu</u>
<u>Animation</u>
FrameAnimation3d
<u>FirstPersonController</u>
EditorCamera
<u>Space</u>
<u>WindowPanel</u>
Node
```