Text Animation

Not only can the animation system work with graphics and frames, it can also animate text strings.

Think “Ready Set Go” in racing games or animated text in console games (rogue-likes!).

To start this tutorial, let's create a blank project using the init script.

In the config file, remove the default `[Object]` section and everything below it.

Paste in the new CountDown object and animation set:

```
[CountDown]
AnimationSet  = CountDownAnimationSet

[CountDownAnimationSet]
Pivot         = center
Graphic       = @     ; Look in the current section for the Graphic
properties (like Text)
Text          = @     ; Look in the current section for Text properties,
like String. (But no string defined, so starts blank)
StartAnim     = Ready
Ready         = 0
Count         = 0      ;Count animation uses 0, meaning as many frames as
defined in config by convention: Count1, Count2, etc.
Go            = 0
Ready->       = Count
Count->       = Go
Go->          = Go
```

In the code, change the default `Object` creation to our `CountDown` object:

```
orxObject_CreateFromConfig("CountDown");
```

If you're not familiar with the animation system, be sure to have a look at the Animation Walk-through.

To explain the above, we start with an object called `CountDown`. The object is empty and only has an animation set which will perform changes on itself.

In the animation set, there are three animation names defined:

- Ready
- Count
- Go

In addition to defining the animation names, we show what happens when one animation ends and where another begins. `Ready` is first, second is `Count`, and finally `Go` which loops forever.

Now to make the animation frames and timings using numbered convention:

```
[Ready1]
String        = READY!
KeyDuration   = 2

[Count]
KeyDuration   = 0.5

[Count1]
String        = 3

[Count2]
String        = 2

[Count3]
String        = 1

[Go]
KeyDuration   = 0.25

[Go1]
String        = GO

[Go2]
String        = >GO<

[Go3]
String        = >>GO<<

[Go4]
String        = >>>GO<<<

[Go5]
String        = >>GO<<

[Go6]
String        = >GO<
```

To explain the above, `Ready1` is the first frame of `Ready`. It is held for 2 seconds, and shows `READY!` during that time.

Once expired, it moves to `Count1` to `Count3`, using `Count` to get common properties like the duration of each frame.

Finally the `Go` animation plays.

From:
https://www.orx-project.org/wiki/ - **Orx Learning**

Permanent link:
https://www.orx-project.org/wiki/en/tutorials/animation/text_animation?rev=1598877453

Last update: **2025/09/30 17:26 (14 months ago)**