Sequence Manager XML

Unit extension

```
<extension name="damage" class="UnitDamage" />
```

Include in object file

```
<sequence_manager file="path to your sequence manager" />
```

Example XML:

```
<table>
  <unit>
    <sequence editable_state="true/false" name="'" triggable="true/false">

    </sequence>
  </unit>
</table>
```

Important: Strings need to have `'` at the beginning and end, otherwise they do not work.

Sequences

- `<sequence/>`
  - `name` The name of your sequence. This will show up in the mesh variation dropdown and UnitSequence/UnitSequenceTrigger elements.
  - `editable_state` Makes the sequence show up in the mesh variation dropdown and UnitSequence element. *
  - `triggable` Makes the sequence show up in the UnitSequenceTrigger element *
    - Both editable_state and triggable are optional. Sequences will run and trigger without them, you just have to type in the sequence name manually into the elements.

    - ***That's probably how it's supposed to work, however testing showed that having either of them set to true will make it show in both UnitSequence and UnitSequenceTrigger, as well as the mesh variation dropdown.***
- `once` true / false | Sequence can only run once.
- `<object/>` Used to toggle individual graphic objects.
    - `name` Name of the object. Usually start with "g_".
    - `enabled` true / false
- `<graphic_group/>` Used to toggle graphic groups.
    - `name` Name of the graphic group.
    - `visibility` true / false
- `<body/>` Used to toggle bodies/collisions.
    - `name` Name of your body.
    - `enabled` true / false
- `<decal_mesh/>` Used to toggle decal meshes.
    - `name` Name of your decal mesh. Usually start with "dm_".
    - `enabled` true / false
- `<light/>` Used to toggle lights.
    - `name` Name of your light object.
    - `enabled` true / false
- `<material_config/>` Used to change the material config of your unit.
    - `name` path to the new .material_config you want to apply.
- `<sound/>` Used to play sounds from your unit.
    - `action` play / stop
    - `event` ID of the sound you want to play.
    - `object` Source object the sound is played from.
    - `source` Alternative to `object`, use a soundsource which has been defined in the .unit file.
- `<effect/>` Used to play effects from your unit.
    - `name` Path to the effect you want to play. Example: `name="'effects/particles/explosions/explosion_grenade'"`
    - `parent` object you want to play the effect from. Example: `parent="object( 'smoke' )"`
    - `position` (Not sure what it does exactly, every effect I found had `position="v()"` in it too)
- `<animation_group/>` Plays an animation.
    - `enabled` true / false
    - `name` Name of your animation group.
    - `from` The frame that this animation should start on. Example: `from="0/30` will make the animation play at 30fps.
    - `to` End frame of your animation. Example: `to="64/30`
    - `speed` How fast your animation is playing. 1 is normal speed, can be a negative value to play backwards.
- `<run_sequence/>` Used to run another sequence.
    - `name` The name of the sequence you want to run.

    - `<spawn_unit/>` Spawns a new unit.
        - `name` Path to the unit you want to spawn.
        - `position` Object position, usually an empty, that the unit will be spawned on. Example: `position="object_pos('spawn_doors')"`
        - `rotation` Same as position, but for rotation. Example: `rotation="object_rot('spawn_doors')"`
    - `<interaction/>` Toggle interactions on your unit.
        - `enabled` true / false

        - `start_time` can be used on pretty much everything to add delays.
        - `startup` true / false

Hitboxes

You can take any body from your .object and use it as a hitbox to trigger sequences.

```
<table>
  <unit>
    <body name="''">
      <endurance>
        <run_sequence name="''"/>
      </endurance>
    </body>
  </unit>
</table>
```

- `<body/>` This defines a body from your .object as a hitbox.
    - `name` The name of the body.
    - `<endurance/>` Controls how much damage the hitbox can take before it triggers the sequences inside.
        - `bullet` How many times you need to shoot it.
        - `explosions` How many explosions you need.
        - `melee` How many melee hits you need.
        - `lock` Doors, deposit boxes and basically anything you can use a saw on use this. (For Example: `lock="15"` on deposit boxes.)

Filters and Variables

You can define variables and filter sequences based on the value of a variable.

```
<table>
  <unit>
    <variables>
    <your_variable_name value="#"/>
    </variables>

    <filter name="'your_filter_name'">
      <check value="vars.your_variable_name == #"/>
    </filter>
  </unit>
</table>
```

Add `filter="your_filter_name"` to a sequence to make it only run if the variable and the filter have the same value.

Use `<set_variables your_variable_name="#"/>` to change variable values.

Example:

```
<table>
  <unit>
    <variables>
    <var_loot_type value="0"/>
    </variables>

    <filter name="'loot_money'">
      <check value="vars.var_loot_type == 1"/>
    </filter>

    <filter name="'loot_gold'">
      <check value="vars.var_loot_type == 2"/>
    </filter>

    <sequence editable_state="true" name="'set_loot_money'" triggable="true">
      <set_variables var_loot_type="1"/>
    </sequence>
    <sequence editable_state="true" name="'set_loot_gold'" triggable="true">
      <set_variables var_loot_type="2"/>
    </sequence>

    <sequence editable_state="true" name="'spawn_loot'" triggable="true">
```

```
    <spawn_unit filter="'loot_money'"
name="'units/pd2_dlc1/vehicles/str_vehicle_truck_gensec_transport/spawn_deposit/spawn_money'".../>
    <spawn_unit filter="'loot_gold'"
name="'units/pd2_dlc1/vehicles/str_vehicle_truck_gensec_transport/spawn_deposit/spawn_gold'".../>
  </sequence>

 </unit>
</table>
```

To be continued...

(old notes from rex)

Vector3 form in sequence manager: `v(0, 0, 0)`

MaterialElement

- `<material/>` material sequences affect materials of the functioning unit. `Unique="true"` is required to only affect this unit.
  - `name` is the name of the material you are modifying.
  - `time` set the time of animated materials, Joys mask uses this feature.
  - `state` sets a state argument, Joys mask uses this feature to pause the UV scrolling.
  - `render_template` sets the render template.
  - `glossiness` sets the glossiness value.
  - If you use a custom `key="value"` you can affect material config elements on your own.
    - Example: `<material name="'mat_mat'" il_multiplier="10"/>`
    - This will modify the il_multiplier of an illuminated material.
    - Vector3 values need to be written like so `key="v(1, 2, 3)"`

---

pick random sequence thingy `<run_sequence name="'sequence_'..pick('1','2','3')"`

Revision #19
Created 18 July 2021 03:42:35 by Rex
Updated 15 August 2024 20:58:21 by soosh.exe