ASSEMBLE3

AI Game Builder Guide

How an AI agent turns a prompt into a finished, tested **.a3p** game for assemble3.com/editor - without reading the engine codebase.

Engine core	GDevelop 5.6.282 (libGD, WebAssembly) - Assemble3 fork
Project format	GDevelop project JSON, encoded as A3P1 (.a3p)
Toolchain	Python 3 + Pillow, Node 22, libGD, GDJS runtime, Playwright/Chromium
Verified with	5 games built, validated, exported and play-tested headless (one built from this guide alone)
Document date	2026-10-24

The complete toolkit (a3lib.py, validate.js, export.js, playtest.js, a3p.py, setup.sh and the sample games) is printed in the appendices and also embedded in this PDF as file attachments.

Contents

1. Read this first **5**

1.1 The pipeline in one table . . . 5
1.2 Rules that prevent most failures . . . 5
1.3 Quick start: a complete game in 25 lines . . . 6
1.4 What a finished delivery looks like . . . 6

2. What Assemble3 is **7**

2.1 Consequences for game building . . . 7
2.2 Where games end up . . . 7
2.3 Legal/content cautions . . . 7

3. Environment and toolchain setup **8**

3.1 Requirements . . . 8
3.2 libGD (the engine core) . . . 8
3.3 GDJS runtime (needed for export and play-tests only) . . . 8
3.4 What not to do . . . 8
3.5 Directory layout used in this guide . . . 9

4. Project file format **10**

4.1 Top level . . . 10
4.2 properties . . . 10
4.3 Resources . . . 10
4.4 Objects . . . 11
4.5 Object groups ("families") . . . 12
4.6 Layouts (scenes) . . . 12

5. Variables **13**

5.1 Using variables in expressions and parameters . . . 13
5.2 Save and load a whole game state in two actions . . . 13

6. The event system **14**

6.1 Event types . . . 14
6.2 Execution and picking rules (verified) . . . 14
6.3 Forces and movement . . . 15
6.4 Timers . . . 15
6.5 Several scenes: levels, menus, global variables and shared logic . . . 15

7. Writing instructions (serialization rules) **16**

7.1 Parameter types and how to write them . . . 16
7.2 Capability behaviors (modern, visible instructions) . . . 16
7.3 Visible vs hidden ("legacy") instructions . . . 17

8. Expressions **18**

9. Cookbook: verified patterns **19**

10. .a3p encoding and delivery **24**

10.1 Delivery checklist . . . 24
10.2 Changing an existing game . . . 24

11. Making the assets **25**

11.1 Smooth vector-style art (Pillow) . . . 25

11.2 Pixel art . . . 25

11.3 Sounds (numpy -> WAV) . . . 25

11.4 Fonts . . . 26

12. Behaviors **27**

13. Designing a game from a prompt **28**

13.1 Procedure . . . 28

13.2 Event budget reference (real games) . . . 28

14. Validation with libGD **29**

15. Export and headless play-testing **30**

15.1 Export . . . 30

15.2 Play-test harness . . . 30

15.3 What to test . . . 31

15.4 Reaching runtime internals in page.evaluate . . . 31

16. Gotchas and troubleshooting **32**

16.1 Things the toolchain cannot check . . . 32

17. Final checklist **33**

Appendix A - a3lib.py (builder library) **34**

Appendix B - Tools **41**

tools/setup.sh - downloads libGD, checks the GDJS runtime and Playwright . . . 41

tools/validate.js - validation + code generation + libGD round trip . . . 41

tools/export.js - HTML5 export into a folder, exposes window.__game . . . 43

tools/playtest.js - headless play-test harness . . . 44

tools/a3p.py - .a3p encoder/decoder CLI . . . 46

tools/meta.js - instruction metadata lookup by regex . . . 47

Appendix C - Sample games (complete, tested) **48**

samples/sample_clicker.py . . . 48

samples/test_clicker.js . . . 49

samples/sample_platformer.py . . . 49

samples/test_platformer.js . . . 50

samples/sapo/art.py . . . 51

samples/sapo/build.py . . . 53

samples/sapo/test.js . . . 57

Running the samples . . . 59

Appendix D - Keyboard key names **60**

Appendix E - Instruction reference (generated from libGD) **61**

Events and control flow - BuiltinCommonInstructions . . . 61

Scene - BuiltinScene . . . 61

Objects - BuiltinObject . . . 62

Sprite - Sprite . . . 72

Text object - TextObject . . . 73
Variables - BuiltinVariables . . . 75
Mouse and touch - BuiltinMouse . . . 77
Keyboard - BuiltinKeyboard . . . 79
Timers and time - BuiltinTime . . . 79
Mathematical tools - BuiltinMathematicalTools . . . 80
Conversion - BuiltinCommonConversions . . . 82
Text manipulation - BuiltinStringInstructions . . . 82
Sounds and music - BuiltinAudio . . . 83
Storage - BuiltinFile . . . 85
Layers and cameras - BuiltinCamera . . . 85
Game window and resolution - BuiltinWindow . . . 87
Objects with animations - AnimatableCapability . . . 88
Scalable objects - ScalableCapability . . . 89
Objects with opacity - OpacityCapability . . . 89
Resizable objects - ResizableCapability . . . 89
Flippable objects - FlippableCapability . . . 89
Objects containing a text - TextContainerCapability . . . 90
Objects with effects - EffectCapability . . . 90
Platform behavior - PlatformBehavior . . . 90
Top-down movement - TopDownMovementBehavior . . . 92
Draggable Behavior - DraggableBehavior . . . 94
Destroy Outside Screen Behavior - DestroyOutsideBehavior . . . 94
Anchor - AnchorBehavior . . . 95
Tiled Sprite Object - TiledSpriteObject . . . 95
Panel Sprite (9-patch) Object - PanelSpriteObject . . . 95
Shape painter - PrimitiveDrawing . . . 96
Particle system - ParticleSystem . . . 99
Text entry object - TextEntryObject . . . 102
Pathfinding behavior - PathfindingBehavior . . . 102
Linked objects - LinkedObjects . . . 104
Inventories - Inventory . . . 105
System information - SystemInfo . . . 105
Network - BuiltinNetwork . . . 105
Event functions - BuiltinAdvanced . . . 106
External layouts - BuiltinExternalLayouts . . . 106

1. Read this first

This guide lets an AI agent build a complete game for **Assemble3** (the game editor at `https://assemble3.com/editor/`) from a plain-language prompt, and hand the user a `.a3p` project file that opens in the editor, plays correctly, and is easy for students to read and modify. Everything needed is here: the file format, the event system, the exact serialization rules, a tested Python builder library, a validator that uses the real engine core, an exporter, a headless browser play-test harness, and a reference of every built-in instruction generated from the engine metadata.

You do **not** need to read the engine codebase. When this guide and the code disagree, the code wins - but every rule here was verified by building, validating, exporting and play-testing real games (two shipped games plus the three samples in Appendix C; Sample 3 was built by an agent using only this guide, and what it found missing was added here).

1.1 The pipeline in one table

Step	What you do	Tool / section
1. Design	Turn the prompt into mechanics, progression, screens, objects, variables and an event plan. Keep the event count under the user plan limit (usually 250).	Section 13 (design), 12 (limits)
2. Assets	Draw images and synthesize sounds with Python (Pillow, numpy). Review a contact sheet image.	Section 11
3. Build	Describe the project in Python with a3lib.py and write plain project JSON.	Sections 4-9, App. A
4. Validate	`node tools/validate.js game.json game.rt.json` - checks every instruction, parameter and expression with libGD, generates the game code, writes the libGD round-tripped JSON.	Section 14
5. Export	`node tools/export.js game.rt.json export/` - real HTML5 export with the Assemble3 GDJS runtime.	Section 15
6. Play-test	`node tools/playtest.js export/ scenario.js` - headless Chromium plays the game, checks state, takes screenshots. Look at the screenshots.	Section 15
7. Iterate	Fix logic/visual problems, rebuild, re-run 4-6 until clean.	Section 16
8. Encode	`python3 tools/a3p.py encode game.rt.json Game.a3p` and verify it decodes.	Section 10
9. Deliver	Send the .a3p (and a screenshot) to the user; summarize controls, mechanics, limits.	Section 10

1.2 Rules that prevent most failures

1. Every instruction needs **all** its parameters, in order, including code-only ones written as `""` (e.g. `MouseButtonFromTextReleased` = `["", "\"Left\""]`). Look them up in Appendix E.
2. String parameters are **expressions**: literal text must be quoted inside the string: `"\"Hello\""`. Object names, operators, resource (sound) names, `yes`/`no` are written **without** quotes.
3. Every variable used in an expression must be **declared** (scene, global or object variable) or validation fails.
4. Sub-events run **after** the parent event's actions. Order actions/sub-events accordingly.
5. An action applied to an object runs for **every picked instance**, and expressions that mention that same object are evaluated per instance (so `Enemy.Speed` is each enemy's own value).
6. After `Create`, the new instance is **added** to the picked list; to give several new instances different random values, create them inside a `Repeat` event (one per iteration).
7. Never let a "cleanup" event (fade + delete) target an object that also has a permanent instance - give HUD icons their own object.
8. In Assemble3 the Platformer behavior jumps with the **Up arrow only** (Construct style), not Space; S works as Down.

9. Always validate, export and play-test headless before delivering; read the screenshots yourself.
10. Do not modify the engine or the website repository. Work in a scratch directory; deliver only the .a3p.
11. Keys that change the scene (menus, restart) must use **key released** (`key_released()`): a key still held when the next scene starts triggers that scene's "key pressed" events too (ENTER on an end screen skipped the title).

1.3 Quick start: a complete game in 25 lines

This builds, validates, exports and passes a play-test as-is (tested). The rest of the guide explains each piece.

```
# quickstart.py - smallest complete game: click the box, score goes up. Run next to a3lib.py.
from PIL import Image, ImageDraw
from a3lib import *
import os
os.makedirs('assets', exist_ok=True)
img = Image.new('RGBA', (120, 120)); ImageDraw.Draw(img).rounded_rectangle([4, 4, 115, 115], 20, fill=(0, 200, 1
» 80), outline='white', width=6)
img.save('assets/caixa.png')

p = Project('Clique na Caixa', 960, 540)
p.image('assets/caixa.png')
p.add_object(simple_sprite('Caixa', 'caixa.png'))
p.add_object(text('TextoPontos', 'Pontos: 0', 36, (255, 255, 255)))
lay = p.layout('Jogo', background=(30, 30, 60), layers=['', 'UI'])
lay.instance('Caixa', 480, 270)
lay.instance('TextoPontos', 20, 20, 'UI')
lay.variables = [num('Pontos')]
lay.events = [
    COMMENT('Clique na caixa para ganhar pontos!'),
    E(clicked('Caixa'), [set_var('Pontos', '+', '1'), set_scale('Caixa', '=', '1.2')]),
    E([], [set_scale('Caixa', '=', 'max(1, Caixa.ScaleX() - 2 * TimeDelta())'),
           set_text('TextoPontos', '"Pontos: " + ToString(Pontos)')]),
]
p.save_json('caixa.json')
# then: node tools/validate.js caixa.json caixa.rt.json && python3 tools/a3p.py encode caixa.rt.json CliqueNaCai
» xa.a3p
```

```
python3 quickstart.py
node tools/validate.js caixa.json caixa.rt.json          # OK: no errors, 6 events
node tools/export.js caixa.rt.json export
node tools/playtest.js export test.js                    # PASS lines + screenshots
python3 tools/a3p.py encode caixa.rt.json CliqueNaCaixa.a3p
```

1.4 What a finished delivery looks like

- One `.a3p` file (encoded, see Section 10) - optionally also a screenshot from the play-test.
- Validator output `OK: no errors` and the event total under the plan limit.
- A play-test scenario that passed (state checks + screenshots you looked at).
- A short summary for the user: how it plays, controls, progression, what to tweak (which variables), the event count vs plan limits, and anything you could not verify (e.g. you did not open it in the editor UI itself).

2. What Assemble3 is

Assemble3 is a fork of **GDevelop** (open-source, MIT) whose editor UI was reworked to look and feel like Scirra's Construct 3, aimed at Brazilian schools (students and teachers). The default UI language is Brazilian Portuguese (pt-BR). It runs in the browser at `https://assemble3.com/editor/` and has a Laravel website around it (accounts, plans, cloud saves, classes, a public game gallery).

2.1 Consequences for game building

- **A project is a GDevelop 5.6 project JSON.** Everything about GDevelop's object/event model applies: scenes (called "layouts" in Assemble3), objects, instances, layers, variables, behaviors, events with conditions/actions.
- **Files are `.a3p`.** The editor writes them encoded (`A3P1:` + base64 of zlib-deflated JSON XORed with a fixed key) so students cannot edit them in a text editor. Plain JSON `.a3p`/`.json` files also open. See Section 10.
- **Terminology** in the UI is Construct's: Layout = GDevelop scene, Event sheet = scene events, Object type = object, Family = object group, Instance variable = object variable, Global variable = project variable, Behaviour = behavior.
- **Text in games should be pt-BR** unless the user asks otherwise (the audience is Brazilian students).
- **Event limits per plan.** The editor and the server count every condition and action (sub-events included, comments excluded): guest 100, free 250, premium 500 (a student uses their teacher's plan). A cloud save over the limit is refused. Design for <= 250 unless told otherwise.
- **Objects are project-wide (global) by default** in Assemble3 (Construct-like): put objects in the project `objects` array, not in the layout, so every layout can use them.
- **No GDevelop online services.** Leaderboards, multiplayer, player authentication, asset store, analytics do not work in Assemble3. Use only built-in features and storage (localStorage).
- **Global variables** are shown as green rows at the top of every event sheet. Scene variables are fine too.
- **Assemble3 "functions"** (event-sheet functions) are stored as a special hidden marker condition; do not hand-write them - use plain events (or Groups) instead.

2.2 Where games end up

- The user opens the .a3p in the editor (menu ☰ / ABRIR -> Arquivo...), previews it, edits events, saves locally or to the cloud. On cloud save the editor moves every embedded data URL of 1 KB or more into separately stored files automatically - you do not need to do anything for that.
- Logged-in users can publish to the public gallery from Exportar -> Galeria pública (they pick category/tags in the editor; games are reviewed by an admin). Published games are served from `https://play.assemble3.com/...` inside a sandboxed iframe; `localStorage` works there (shared by all gallery games, so use a unique storage name per game).
- Exported games show an Assemble3 logo; only premium accounts can hide it (editor setting).

2.3 Legal/content cautions

- Do not write "Construct", "Construct 3" or "Scirra" in anything you ship for this product.
- Parodies: avoid real logos, mascots and exact brand names; use clearly altered names (e.g. "KFP" on a striped bucket). Mention trademark risk if the user plans to publish.
- Games are for kids and teens: no gore, no real-world violence against people, nothing sexual; cartoon slapstick is fine (the pigeon-shooting parody was accepted by the user).
- Fonts/assets you embed must be license-compatible (OFL, CC0, CC-BY with credit, or generated by you). Section 11 lists safe bundled fonts.

3. Environment and toolchain setup

3.1 Requirements

Component	Why	How to get it
Python 3 + Pillow (+ numpy)	Build JSON with a3lib.py, draw art, synthesize sounds	`pip install pillow numpy`
Node.js 18+ (22 tested)	Run libGD (validator/exporter) and Playwright	usually preinstalled
libGD.js + libGD.wasm	GDevelop core in WebAssembly: metadata, parsing, code generation, export	`bash tools/setup.sh` downloads them (see 3.2)
GDJS runtime (compiled)	The game engine JavaScript copied into exports	In the Assemble3 repo (3.3)
Playwright + Chromium	Headless play-tests and screenshots	preinstalled in Claude cloud sessions
PHP (optional)	Cross-check encoding with the website's decoder	not required

3.2 libGD (the engine core)

libGD must match the Core the editor uses. Assemble3 pins the upstream GDevelop commit it is synced with; at the time of writing that is `c6d09211037373a3cf315f476ebb45629f14b6eb` (the value of `C3_UPSTREAM_COMMIT` in `editor/newIDE/app/scripts/import-libGD.js` - check it if the repo is available, it changes after upstream merges).

```
U=https://s3.amazonaws.com/gdevelop-gdevelop.js/master/commit/c6d09211037373a3cf315f476ebb45629f14b6eb
curl -fsS -o tools/libGD.js   $U/libGD.js
curl -fsS -o tools/libGD.wasm $U/libGD.wasm
node -e "require('./tools/libGD.js')().then(gd => console.log(gd.VersionWrapper.fullString()))"   # 5.6.282-0
```

NOTE Use `curl` for downloads in sandboxed sessions: it honours the HTTPS proxy. Node scripts that download with their own HTTP client (the editor's `npm start` resource importers) may fail behind the proxy with "Client network socket disconnected"; `NODE_USE_ENV_PROXY=1` helps some of them, axios-based ones need workarounds. You never need those scripts for game building.

3.3 GDJS runtime (needed for export and play-tests only)

The exporter copies the compiled game runtime into the export folder. In the Assemble3 repository it lives at `editor/newIDE/app/resources/GDJS` (folder containing `Runtime/gd.js`). If it is missing, build it (read-only with respect to the repo's tracked files - the output folders are git-ignored):

```
cd editor/GDJS && npm ci                                # GDJS dependencies (esbuild, TypeScript)
cd ../newIDE/app && npm ci --ignore-scripts && npx patch-package   # only if node_modules is missing
cd scripts && node import-GDJS-Runtime.js               # builds + copies runtime to resources/GDJS
```

Set `GDJS_ROOT` (or pass the path as the 3rd argument of export.js) if it lives elsewhere. Using the Assemble3 runtime matters: it contains the fork's runtime changes (e.g. Up-arrow jump), so play-tests match the editor preview.

3.4 What not to do

- Do not edit, build or start the editor/website from the repository tree for a game request (`npm start`, `react-app-rewired build`, `deploy/build-editor.sh`). Users asked for games, not engine changes; automated permission checks may also block it. Everything in this guide works from a scratch directory.
- Do not commit generated games into the repository unless the user asks; deliver the file instead.
- Do not depend on JavaScript-extension features (Tween, Physics 2, 3D, Spine, Tilemap, Bitmap text, Video, Leaderboards...). They are loaded by the editor but **not** by libGD in Node, so the validator/exporter here cannot handle them. Everything in Appendix E is safe.

3.5 Directory layout used in this guide

```
work/
  a3lib.py                # builder library (Appendix A)
  tools/
    setup.sh  libGD.js  libGD.wasm
    validate.js  export.js  playtest.js  a3p.py  meta.js   (Appendix B)
  mygame/
    art.py                # draws assets/*.png, *.wav
    build.py              # uses a3lib -> mygame.json
    test.js               # play-test scenario
    assets/  export/  shots/
```

All toolkit files are embedded as attachments in this PDF (most PDF viewers show a paperclip; with Python: `pypdf.PdfReader(path).attachments`), and printed in Appendices A-C. Extract them instead of retyping.

```
import os
from pypdf import PdfReader
r = PdfReader('Assemble3_AI_Game_Builder_Guide.pdf')
for name, contents in r.attachments.items():          # names like 'tools/validate.js'
    os.makedirs(os.path.dirname(name) or '.', exist_ok=True)
    open(name, 'wb').write(contents[0])
# then: bash tools/setup.sh   (downloads libGD next to the tools)
```

4. Project file format

A project is one JSON object. a3lib.py produces all of it; this section explains every part so you can extend or debug it. Key order does not matter. Unknown/missing keys get defaults when libGD loads the file, which is why we round-trip through libGD before delivering (Section 14).

4.1 Top level

```
{
  "firstLayout": "Jogo",                 // scene that starts the game
  "gdVersion": {"major": 5, "minor": 6, "build": 282, "revision": 0},
  "properties": {...},                   // 4.2
  "resources": {"resources": [...], "resourceFolders": []},   // 4.3
  "objects": [...],                      // project-wide (global) objects - preferred in Assemble3
  "objectsFolderStructure": {"folderName": "__ROOT", "children": [{"objectName": "Nave"}, ...]},
  "objectsGroups": [{"name": "Inimigos", "objects": [{"name": "Morcego"}, {"name": "Fantasma"}]}],
  "variables": [...],                    // global variables
  "layouts": [...],                      // scenes, 4.6
  "externalEvents": [{"name": "Logica", "associatedLayout": "Fase1", "events": [...]}],   // shared sheets, 6.5
  "externalLayouts": [], "eventsFunctionsExtensions": []
}
```

4.2 properties

Key	Meaning / recommended value
name, description, author, version	Game name (shown in the editor and gallery), text, author, "1.0.0".
windowWidth / windowHeight	Game resolution: 1280x720 or 960x540 (landscape) are good defaults; 720x1280 for portrait (also set orientation).
adaptGameResolutionAtRuntime	true. The canvas is scaled to fit the window/iframe.
sizeOnStartupMode	"" (Assemble3 default: keep the game size; letterbox). Other: "adaptWidth", "adaptHeight".
scaleMode	"linear" (smooth) or "nearest" (pixel art - crisp pixels when scaled).
pixelsRounding	true for pixel art (avoids half-pixel blur), else false.
antialiasingMode	"MSAA" or "none" (pixel art).
orientation	"landscape" | "portrait" | "default".
projectUuid	A fresh UUID per game (the gallery uses it to detect new versions of the same game).
packageName	e.g. "com.assemble3.meujogo" (used by mobile exports).
maxFPS / minFPS	60 / 20.
loadingScreen	backgroundColor (int 0xRRGGBB), showProgressBar, minDuration (1.5 s)...
extensionProperties	[] - the editor stores gallery category/tags here (A3Gallery); leave empty.
platforms / currentPlatform	[{"name": "GDevelop JS platform"}] / "GDevelop JS platform".

4.3 Resources

Every image, sound and font file is a resource with a unique `name`; objects and actions reference resources **by name**. Embed the file content as a `data:` URL in `file` - that is exactly what the web editor does for files picked from the device, so the .a3p is self-contained.

```
{"kind": "image", "name": "nave.png", "file": "...", "metadata": "",
 "smoothed": true, "userAdded": true}                    // smoothed:false for pixel art
{"kind": "audio", "name": "pew.wav",  "file": "data:audio/wav;base64,...", "metadata": "",
 "preloadAsSound": true, "preloadAsMusic": false, "preloadInCache": false, "userAdded": true}
{"kind": "font",  "name": "Oxanium-Medium.ttf", "file": "data:font/ttf;base64,...", "metadata": "", "userAdded":
»  true}
```

- Image formats: PNG (preferred, transparency), JPEG (large photographic backgrounds), WebP. Audio: WAV (simple, reliable), OGG, MP3, AAC. Fonts: TTF/OTF/WOFF.

- Keep the total small: a typical game is 0.2-2 MB. Big backgrounds as JPEG; short sounds 22050 Hz mono 16-bit WAV.
- Resource names: plain ASCII file names (`fundo.png`). The same name is used by sprite frames (`image`), tiled sprites (`texture`), text objects (`font`) and sound actions (`PlaySound` parameter, unquoted).

4.4 Objects

Common fields: `name` (unique, letters/digits/underscore, no spaces - Assemble3 shows them as object types), `type`, `variables` (object/instance variables with default values), `behaviors`, `effects` (use []). Types you can rely on:

type	Use for	a3lib helper
Sprite	Anything with images/animations: characters, buttons, particles, backgrounds	sprite(), simple_sprite()
TextObject::Text	All on-screen text (score, labels, floating "+1")	text()
TiledSpriteObject::TiledSprite	Repeating textures: ground, walls, scrolling backgrounds	tiled_sprite()
PanelSpriteObject::PanelSprite	9-slice panels/buttons that stretch without distortion	panel_sprite()
PrimitiveDrawing::Drawer	Shape painter: draw rectangles/circles/lines from events	shape_painter()
ParticleSystem::ParticleEmitter	Built-in particle emitter (works, but sprite particles are simpler)	(JSON from 4.5 dump)
TextEntryObject::TextEntry	Invisible keyboard text capture	-

Sprite

```
{"type": "Sprite", "name": "Pombo", "updateIfNotVisible": false, "adaptCollisionMaskAutomatically": true,
 "assetStoreId": "", "variables": [{"name": "Vida", "type": "number", "value": 3}], "effects": [], "behaviors":
» [],
 "animations": [                                  // index 0, 1, 2... (SetIndex) or by name (SetName)
   {"name": "Voando", "useMultipleDirections": false,
    "directions": [{"looping": true, "timeBetweenFrames": 0.14, "sprites": [
      {"image": "pombo_1.png", "hasCustomCollisionMask": false, "customCollisionMask": [], "points": [],
       "originPoint": {"name": "origine", "x": 27, "y": 21},     // position/rotation pivot, in image pixels
       "centerPoint": {"name": "centre", "automatic": true, "x": 0, "y": 0}},
      {"image": "pombo_2.png", ...}]}]},
   {"name": "Morto", ...}]}
```

- **Origin**: the instance x/y is where the origin point sits. a3lib defaults to the image centre (good for rotation/scaling and for `SetCenter`); use (0,0) for backgrounds, bars and UI that you position by the top-left.
- **Collision mask**: default = the image bounding box (transparent margins included). Custom polygons: `hasCustomCollisionMask: true` + `customCollisionMask: [[{x,y},...]]` (convex polygons).
- **Custom points** (`points: [{"name": "Cano", "x": 40, "y": 8}]`) are read with `Obj.PointX("Cano")` / `PointY`.
- **Animations** with one frame are fine. A trick for idle blinking without events: 14 copies of the open-eyes frame + 1 blink frame, looping at 0.2 s.

Text

```
{"type": "TextObject::Text", "name": "TextoPontos", "string": "0", "font": "Oxanium-Medium.ttf",
 "characterSize": 32, "textAlignment": "left", "color": {"r": 255, "g": 214, "b": 70},
 "bold": false, "italic": false, "underlined": false, "smoothed": true, "variables": [], "effects": [], "behavio
» rs": [],
 "content": {"text": "0", "font": "Oxanium-Medium.ttf", "characterSize": 32, "color": "255;214;70",
   "textAlignment": "left", "verticalTextAlignment": "top", "lineHeight": 0,
   "isOutlineEnabled": true, "outlineColor": "18;8;48", "outlineThickness": 4,
   "isShadowEnabled": false, "shadowColor": "0;0;0", "shadowOpacity": 127, "shadowAngle": 90,
   "shadowDistance": 4, "shadowBlurRadius": 2, "bold": false, "italic": false, "underlined": false, "smoothed":
» true}}
```

- The same settings appear twice (top level and `content`); keep them identical (a3lib does).
- **Centered/wrapped text**: give the *instance* a custom width (`customSize: true`, `width`) - the text wraps at that width and `textAlignment: "center"` centres it inside. This is how full-width banners are

made.

- Pixel fonts: `smoothed: false` and sizes that are multiples of the font's design size (Kenney Mini: 16/24/32/48).
- Change the text from events with `TextContainerCapability::TextContainerBehavior::SetValue` (visible) or the legacy `TextObject::String`. Colour: `TextObject::ChangeColor` with `"\"r;g;b\""`.

Tiled sprite, panel sprite, shape painter

```
{"type": "TiledSpriteObject::TiledSprite", "name": "Chao", "texture": "chao.png", "width": 24, "height": 24,
 "assetStoreId": "", "variables": [], "effects": [], "behaviors": [{... platform behavior ...}]}
{"type": "PanelSpriteObject::PanelSprite", "name": "Painel", "texture": "painel.png", "width": 200, "height": 80
» ,
 "leftMargin": 8, "topMargin": 8, "rightMargin": 8, "bottomMargin": 8, "tiled": false, ...}
{"type": "PrimitiveDrawing::Drawer", "name": "Pintor", "fillColor": {"r":255,"g":255,"b":255}, "fillOpacity": 25
» 5,
 "outlineColor": {"r":0,"g":0,"b":0}, "outlineSize": 1, "outlineOpacity": 255, "absoluteCoordinates": false,
 "clearBetweenFrames": true, "antialiasing": "none", ...}
```

Tiled/panel sprite instances are sized by the instance `width`/`height` (custom size) - the texture repeats.

4.5 Object groups ("families")

`objectsGroups: [{"name": "Ajudantes", "objects": [{"name": "Drone"}, {"name": "Estacao"}]}]`. A group name can be used anywhere an object name is expected. Object variables used through the group (`Ajudantes.Ang`) must exist in **every** member object.

4.6 Layouts (scenes)

```
{"name": "Jogo", "mangledName": "Jogo", "title": "",
 "r": 12, "v": 8, "b": 40,                // background colour - NOTE: green is "v", not "g"
 "standardSortMethod": true, "stopSoundsOnStartup": true, "disableInputWhenNotFocused": true,
 "uiSettings": {...editor grid/zoom...},
 "layers": [ {"name": "Fundo", ...}, {"name": "", ...}, {"name": "UI", ...} ],   // bottom first; "" = base laye
» r
 "instances": [...], "variables": [...],   // scene variables
 "objects": [], "objectsFolderStructure": {"folderName": "__ROOT", "children": []}, "objectsGroups": [],
 "events": [...], "behaviorsSharedData": []}
```

- **Layers**: the base layer `""` must exist. Use a background layer below it and a `UI` layer above it for the HUD - UI layers do not move with the base camera. Each layer needs one default camera (a3lib's `layer()` adds it).
- **behaviorsSharedData** can stay `[]`; libGD fills in the entries (capabilities, behaviors) when it loads the file.
- The `mangledName` is the name with spaces encoded (`"Game Scene"` -> `"Game_32Scene"`). Prefer names without spaces.

Instances

```
{"name": "Botao", "x": 1060, "y": 140, "layer": "UI", "zOrder": 1, "angle": 0, "depth": 0,
 "customSize": false, "width": 0, "height": 0, "locked": false,
 "persistentUuid": "<uuid4>",                        // unique per instance
 "initialVariables": [{"name": "Id", "type": "number", "value": 2}],   // overrides the object default
 "numberProperties": [], "stringProperties": []}
```

- x/y = position of the object's origin in scene coordinates (top-left of the game is 0,0; y grows downward).
- zOrder sorts instances **within a layer** (higher = in front). Objects created by events get Z-order 0 unless you set it (`ChangePlan`/`SetZOrder`), so put short-lived effects on a higher layer instead.
- Several instances of one object can carry different `initialVariables` (e.g. shop buttons with Id 0..4).

5. Variables

Scope	Declared in	Lifetime	Expression syntax
Global (project)	project `variables`	whole game, survives scene changes	`Score`
Scene	layout `variables`	reset when the scene (re)starts	`Score`
Object (instance)	object `variables` (defaults) + instance `initialVariables`	per instance	`Enemy.Life`
Local	an event's `variables` array	that event and its sub-events	`Temp`

Types: `number`, `string`, `boolean`, `structure` (named children), `array` (indexed children). Serialization:

```
{"name": "Pontos", "type": "number", "value": 0}
{"name": "Nome", "type": "string", "value": "Ana"}
{"name": "Ligado", "type": "boolean", "value": true}
{"name": "Jogo", "type": "structure", "children": [{"name": "Estrelas", "type": "number", "value": 0},
                                                   {"name": "Nivel", "type": "array", "children": [
                                                       {"type": "number", "value": 0}, {"type": "number", "value"
» : 0}]}]}
```

5.1 Using variables in expressions and parameters

- Plain name: `Pontos`. Child: `Jogo.Estrelas`. Array element: `Jogo.Nivel[0]`, with an expression index: `Jogo.Custo[Botao.Id]`, nested: `Limites[Jogo.Patente + 1]`.
- Object variable: `Botao.Id`, `TextoBotao.Id`; array inside an object variable: `Inimigo.Rota[2]`.
- Variable parameters of actions/conditions (`SetNumberVariable`, `NumberVariable`, ...) take the same syntax, including computed indexes: `Jogo.Nivel[Botao.Id]`.
- Strings and numbers are not auto-converted in text: use `ToString(n)` and `ToNumber(s)`.
- Reading an array element that does not exist returns 0 / "" (useful for "next rank" past the end, but handle the max level explicitly).
- **Declare everything.** An undeclared name fails validation ("No variable with this name found") and shows red in the editor. Child variables of a declared structure/array can be created at runtime by actions.

5.2 Save and load a whole game state in two actions

Keep the state to persist in **one structure variable** (e.g. `Jogo`). Save it as JSON text to storage (localStorage), load it back at scene start:

```
E([at_start(), C('GroupExists', q('MeuJogo'), q('jogo'))], [
    A('LireFichierTxt', q('MeuJogo'), q('jogo'), '', 'Salvo'),       # storage text -> scene string variable Sal
» vo
    A('JSONToVariableStructure2', 'Salvo', 'Jogo'),                    # JSON -> structure (replaces it)
]),
E([timer_over('Salvar', 3)], [
    A('EcrireFichierTxt', q('MeuJogo'), q('jogo'), 'ToJSON(Jogo)'),
    reset_timer('Salvar'),
]),
```

- Use a **unique storage name per game** (gallery games share one origin and therefore one localStorage).
- `LireFichierTxt`/`LireFichierExp` write into a **scene** variable (parameter type `scenevar`).
- Reset: `DeleteGroupFichier` then `Scene` (change to the same scene) - see the recipes.
- Values derived from the state (prices, rates) should be recomputed every frame from the saved levels, not saved.

6. The event system

Each layout has an event sheet: a list of events evaluated **top to bottom, once per frame** (60 times per second). An event has conditions and actions; if all conditions are true, its actions run, then its sub-events are evaluated. Conditions about objects also **pick** instances: the actions then apply only to the instances that passed.

6.1 Event types

type	JSON fields	Behaviour
BuiltinCommonInstructions::Standard	conditions, actions, events?, variables?	The normal event. No conditions = runs every frame.
BuiltinCommonInstructions::Else	conditions, actions, events?	Runs when the **previous sibling** did not run. With conditions it is an "else if"; chains work (if / else-if / else verified).
BuiltinCommonInstructions::Repeat	repeatExpression, conditions, actions, events	Runs its body N times (N is an expression). Each iteration starts with fresh picking - ideal for creating N objects with different random values.
BuiltinCommonInstructions::ForEach	object, conditions, actions, events	Runs once per instance of the object; in each iteration only that instance is picked. Its own conditions are tested per instance.
BuiltinCommonInstructions::While	whileConditions, conditions, actions, events, infiniteLoopWarning	Loops while whileConditions are true (same frame - avoid unless needed).
BuiltinCommonInstructions::Group	name, events, colorR/G/B, source, creationTime, parameters	Folder of events in the sheet (named, coloured). No logic - use it to organize for readers.
BuiltinCommonInstructions::Comment	comment, color{r,g,b,textR,textG,textB}	A note for humans. Not counted by the event limit. Write them in pt-BR for students.
BuiltinCommonInstructions::Link	target, include {"includeConfig": 0}	Includes an external events sheet at this place (a3lib `LINK(name)`). See 6.5.

6.2 Execution and picking rules (verified)

1. **Order inside an event**: conditions (in order, each narrowing the picked instances) -> actions (in order) -> sub-events (in order). A variable changed by an action is already changed for the sub-events.
2. **Picking is inherited** by sub-events: an object picked by the parent stays picked (narrowed) in sub-events. A sibling event starts again from "all instances".
3. **Unpicked object = all instances.** If an event never mentions object X in its conditions, actions on X apply to all X instances; expressions like `X.X()` read the **first** instance.
4. **Per-instance expressions**: in an action on X (e.g. `SetCenter` on a group), any expression mentioning X uses the current instance. `Ajudantes.Ang` inside an action on `Ajudantes` is each helper's own angle.
5. **Create**: the new instance becomes picked (added to the list). Two `Create Faisca` actions in one event followed by `AddForceXY Faisca` would push both. Use `REPEAT(n, [Create, AddForceXY random...])`.
6. **Newly created objects** exist immediately for later events in the same frame.
7. **Deleting** removes the instance at once from the current picking.
8. **Else** looks only at whether the previous sibling event **ran** (its conditions passed), not at its sub-events.
9. **Trigger once**: condition `BuiltinCommonInstructions::Once` (no parameters) makes an event run only on the frame its other conditions become true. Mouse "released" conditions are already one-frame events.

10. **Frame-rate independence**: multiply per-frame changes by `TimeDelta()` (seconds since last frame): `SetOpacity - 300 * TimeDelta()` fades in ~0.85 s at any FPS. Forces are already per second.

6.3 Forces and movement

- `AddForceXY(obj, vx, vy, multiplier)`: multiplier `0` = instant force (applies this frame only; add it every frame to keep moving), `1` = permanent (keeps moving until `ClearForces`). Units: pixels per second.
- `AddForceVersPos(obj, x, y, speed, multiplier)`: towards a point (homing items, coins flying to the HUD).
- Permanent forces add up; there is no gravity unless you add it (or use the Platformer behavior).
- Direct positioning: `SetXY` (visible) / legacy `MettreXY`, `SetCenter` (by centre), `SetX`, `SetY`.

6.4 Timers

- Scene timers: `ResetTimer` (starts/restarts), condition `CompareTimer(name >= seconds)`, expression `TimerElapsedTime("name")`. Names are **quoted strings**: `q("Cometa")`.
- Reset each timer at scene start; compare with `>=` and reset it in the same event for repeating spawns.
- Object timers exist too (`ResetObjectTimer`, `CompareObjectTimer`) for per-instance cooldowns.
- Cooldown pattern for hold-to-fire: `MouseButtonPressed` + `CompareTimer("Tiro" >= Cooldown)` -> fire + reset.

6.5 Several scenes: levels, menus, global variables and shared logic

- **Start scene**: `firstLayout`. In a3lib it is the first layout added unless you set `p.start_layout = "Titulo"`.
- **Change scene**: `A("Scene", "", q("Fase2"), "yes")`. The scene name is a string expression, so a scene string variable works: each level declares `Proxima = "Fase2"` / `"Fim"` and the shared logic does `A("Scene", "", "Proxima", "yes")`.
- **Scene variables reset** every time a scene starts. Values that must survive (lives, coins, "won?") go in **global variables**: `p.variables = [num("Vidas", 3), num("Moedas")]`, used with the same syntax (`Vidas`). Reset them on the title screen.
- **Shared logic** - write it once in an external events sheet and include it in every level with a Link event: `p.external_events("Logica", events, "Fase1")` + `lay.events = [LINK("Logica")]` in each level. The associated layout gives the sheet its objects/variables scope (editor + validator). Every layout that links it must declare the same scene variables (build levels with one Python function). Its events count **once** toward the limit.
- **Menus**: trigger scene changes with `key_released("Return")` or `MouseButtonFromTextReleased`, never with "key pressed" (the held key leaks into the next scene).
- A level can be built from a Python function taking the geometry (ground segments, planks, enemies with patrol ranges, coins, flag) - see Sample 3 (Sapo Saltitante, Appendix C): title, 2 levels sharing one logic sheet, end screen.

7. Writing instructions (serialization rules)

An instruction is `{"type": {"value": "<type>"}, "parameters": [...]}`; an inverted condition is `{"type": {"value": "<type>", "inverted": true}, ...}`. The parameter list must contain **every** parameter the metadata declares (Appendix E), in order, as strings.

7.1 Parameter types and how to write them

Parameter type	Write it as	Example
currentScene, objectsContext, conditionInverted, inlineCode (code-only)	always `""`	`""`
object, objectList, objectPtr, objectListOrEmptyIfJustDeclared, objectListOrEmptyWithoutPicking	object or group name, unquoted	`Nave`
expression, number	numeric expression	`PorClique * 2`, `RandomInRange(1, 6)`
string, layer, sceneName, color, objectAnimationName, mouseButton, keyboardKey, identifier (timer names)	string expression - literals quoted	`"\"UI\""`, `"\"255;0;0\""`, `"\"Left\""`, `"\"Cometa\""`
operator (actions)	`= + - * /` (strings: `= +`; booleans: `True False Toggle`)	`+`
relationalOperator (conditions)	`= < > <= >= !=`	`>=`
yesorno	`yes` / `no` (empty = default for optional)	`yes`
trueorfalse	`True` / `False`	`True`
mouse (legacy mouse conditions)	unquoted `Left Right Middle`	`Left`
key (legacy KeyPressed)	unquoted key name (Appendix D)	`Left`, `a`, `Space`
soundfile, musicfile	resource name, unquoted	`pew.wav`
variable, variableOrProperty, variableOrPropertyOrParameter, scenevar, globalvar	variable path, unquoted	`Jogo.Nivel[0]`
objectvar	object variable name (path) without the object	`Vida`
behavior	behavior name on that object (capabilities: Animation, Scale, Opacity, Text, Flippable, Resizable, Effect)	`PlatformerObject`, `Scale`
forceMultiplier	`0` instant, `1` permanent	`1`
objectPointName	string expression, quoted point name, or empty	`"\"Cano\""`

TIP In a3lib, `q("text")` produces the quoted form. So `A("PlaySound", "", "pew.wav", "", 80, 1)` plays a sound, and `A("Create", "", "Faisca", "CursorX()", "CursorY()", q("UI"))` creates an object on the UI layer.

7.2 Capability behaviors (modern, visible instructions)

GDevelop 5.6 moved common object properties to implicit "capability" behaviors that every Sprite/Text has with fixed names. Do **not** add them to `behaviors` - just pass the name as the behavior parameter:

Capability (behavior parameter)	Visible instructions	Older equivalent (hidden but working)
Scale	`ScalableCapability::ScalableBehavior::SetValue` / `::Value` (cond)	`ChangeScale` (Sprite), `TextObject::Scale`
Opacity	`OpacityCapability::OpacityBehavior::SetValue` / `::Value`	`Opacity` (Sprite), `TextObject::Opacity`

Capability (behavior parameter)	Visible instructions	Older equivalent (hidden but working)
Animation	`AnimatableCapability::AnimatableBehavior::SetIndex` / `SetName` / `Index` / `Name` / `HasAnimationEnded`	`ChangeAnimation`, `SetAnimationName`
Text	`TextContainerCapability::TextContainerBehavior::SetValue` / `::Value`	`TextObject::String`
Flippable	`FlippableCapability::FlippableBehavior::FlipX` / `FlipY`	`FlipX`, `FlipY` (Sprite)
Resizable	`ResizableCapability::ResizableBehavior::SetWidth` / `SetHeight`	`ChangeWidth`, `ChangeHeight` (Sprite)

```
A('ScalableCapability::ScalableBehavior::SetValue', 'Nave', 'Scale', '=', '1.12')
A('OpacityCapability::OpacityBehavior::SetValue', 'Faisca', 'Opacity', '-', '350 * TimeDelta()')
A('AnimatableCapability::AnimatableBehavior::SetName', 'Gato', 'Animation', '=', q('Correndo'))
A('TextContainerCapability::TextContainerBehavior::SetValue', 'TextoPontos', 'Text', '=', '"Pontos: " + ToString
» (Pontos)')
```

7.3 Visible vs hidden ("legacy") instructions

Many classic instructions are flagged hidden in 5.6: they still load, run and display in the event sheet (the bundled GDevelop examples use them), but the editor's instruction picker offers the modern equivalents. Prefer the visible ones so students can find the same instruction when they edit; legacy ones are acceptable when there is no visible equivalent. Appendix E marks hidden instructions.

Purpose	Prefer (visible)	Legacy (hidden, works)
At the beginning of the scene	`SceneJustBegins` [""]	`DepartScene` [""]
Cursor/touch on object	`IsCursorOnObject` [obj, "", "", ""]	`SourisSurObjet` [obj, "", "", ""]
Mouse button released / pressed	`MouseButtonFromTextReleased` ["", "\"Left\""] / `...Pressed`	`MouseButtonReleased` ["", "Left"] / `MouseButtonPressed`
Key pressed / released	`KeyFromTextPressed` ["", "\"Left\""] / `KeyFromTextReleased`	`KeyPressed` ["", "Left"]
Position	`SetXY` [obj, op, x, op, y]	`MettreXY`
Hide / show	`Hide` [obj] / `Show` [obj, ""]	`Cache` / `Montre`
Camera centre	`SetCameraCenterX/Y` ["", op, value, "", ""]	`CentreCamera`, `CameraX`

8. Expressions

Expressions are written in GDevelop's expression language - they are what goes into number and string parameters. There is **no ternary/if operator**: branch with events (an event + `ELSE`), or use math tricks (`min(Fritando, 1)` as a 0/1 flag, `clamp`, `max`).

Category	Examples
Arithmetic	`+ - * /`, parentheses, `mod(a, b)`, unary minus
Math	`abs min max clamp(x, a, b) floor ceil round trunc sqrt pow(a, b) exp log sin cos tan atan2(y, x) ToRad(deg) ToDeg(rad) lerp(a, b, t) sign Pi() roundTo(x, decimals) AngleBetweenPositions(x1,y1,x2,y2) DistanceBetweenPositions(x1,y1,x2,y2) XFromAngleAndDistance(angle, d)`
Random	`Random(max)` (integer 0..max), `RandomInRange(a, b)` (integer), `RandomFloat(max)`, `RandomFloatInRange(a, b)`, `RandomWithStep(a, b, step)`
Time	`TimeDelta()`, `TimeFromStart()`, `TimerElapsedTime("name")`, `Time("hour")`...
Mouse/touch	`CursorX()`, `CursorY()` (optional layer: `CursorX("UI")`), `MouseWheelDelta()`, `TouchX(id)`
Screen/camera	`SceneWindowWidth()`, `SceneWindowHeight()`, `CameraCenterX()`, `CameraCenterY()`
Objects	`Obj.X() Obj.Y() Obj.CenterX() Obj.CenterY() Obj.Width() Obj.Height() Obj.Angle() Obj.ZOrder() Obj.Opacity() Obj.Animation() Obj.ScaleX() Obj.PointX("p") Obj.DistanceToPosition(x, y) Obj.Distance(Other) Obj.VariableName SceneInstancesCount(Obj) PickedInstancesCount(Obj)`
Behaviors	`Gato.PlatformerObject::CurrentSpeed()` style: `Object.BehaviorName::Expression()`
Text	`"a" + "b"`, `ToString(n)`, `ToNumber(s)`, `NewLine()`, `StrLength(s)`, `ToUpperCase(s)`, `ToLowerCase(s)`, `SubStr(s, start, len)`, `StrFind(s, part)`, `StrReplaceAll(s, a, b)`, `ToJSON(var)`
Text object	`TextoPontos.String()` (legacy) / `TextoPontos.Text::Value()`

- Numbers print with all decimals (`ToString(12.5)` -> `12.5`); round for display: `ToString(round(x * 10) / 10)`.
- Angles are in degrees in object instructions (0 = right, 90 = down). `atan2` returns radians -> `ToDeg(...)`.
- Aim at the cursor: `SetAngle(Arma, =, ToDeg(atan2(CursorY() - Arma.Y(), CursorX() - Arma.X())))`.
- A point at distance d along an object's angle: `Obj.X() + cos(ToRad(Obj.Angle())) * d`.
- Use `NewLine()` for multi-line texts; Text objects render `\n` from it.

9. Cookbook: verified patterns

Every snippet uses a3lib (Appendix A) and was exercised in a play-tested game. `C`/`A`/`E`/`ELSE`/`REPEAT`/`FOREACH`/`GROUP`/`COMMENT`/`q` and the shortcuts (`at_start`, `clicked`, `set_var`, `set_text`, `create`…) are defined there.

Click / tap on an object

```
E(clicked('Nave'), [
    set_var('Pontos', '+', 'PorClique'),
    set_scale('Nave', '=', '1.12'),            # squash for feedback
    play_sound('clique.wav', 60, 'RandomFloatInRange(0.9, 1.25)'),
]),
E([], [set_scale('Nave', '=', 'max(1, Nave.ScaleX() - 1.5 * TimeDelta())')]),   # ease back every frame
```

Floating "+N" text that rises and fades

```
# on the event that scores:
create('TextoMais', 'CursorX() - 20', 'CursorY() - 50', 'UI'),
set_text('TextoMais', '"+" + ToString(PorClique)'),
# every frame:
E([], [A('SetXY', 'TextoMais', '+', '0', '-', '70 * TimeDelta()'),
       set_opacity('TextoMais', '-', '280 * TimeDelta()')]),
E([opacity_is('TextoMais', '<=', '0')], [delete('TextoMais')]),
```

Burst of particles with random directions

```
REPEAT(6, [
    create('Faisca', 'CursorX()', 'CursorY()', 'UI'),
    A('AddForceXY', 'Faisca', 'RandomInRange(-260, 260)', 'RandomInRange(-320, 120)', '1'),
    set_scale('Faisca', '=', 'RandomFloatInRange(0.4, 0.9)'),
]),
# plus fade + delete like the floating text (and A('Rotate', 'Faisca', '200', ''))
```

Spawn an enemy from a random side

```
E([timer_over('Pombo', 'Intervalo'), compare('SceneInstancesCount(Pombo)', '<', 'MaxPombos')], [
    reset_timer('Pombo'),
    create('Pombo', '-40', 'RandomInRange(50, 290)'),
    A('SetNumberObjectVariable', 'Pombo', 'Vel', '=', 'RandomInRange(80, 150)'),
], [                                           # sub-events: the new pigeon is still picked
    E([compare('Random(1)', '=', '1')], [
        A('SetXY', 'Pombo', '=', '1000', '+', '0'),
        A('FlippableCapability::FlippableBehavior::FlipX', 'Pombo', 'Flippable', 'yes'),
        A('SetNumberObjectVariable', 'Pombo', 'Vel', '*', '-1')]),
    E([compare('Random(99)', '<', '7')], [set_animation('Pombo', '2'),          # rare golden variant
        A('SetNumberObjectVariable', 'Pombo', 'Valor', '=', '5')]),
    E([], [A('AddForceXY', 'Pombo', 'Pombo.Vel', '0', '1')]),
]),
E([C('PosX', 'Pombo', '<', '-80')], [delete('Pombo')]),
E([C('PosX', 'Pombo', '>', '1040')], [delete('Pombo')]),
```

Hold to fire with a cooldown, area hit test

```
E([C('MouseButtonFromTextPressed', '', q('Left')), compare('CursorY()', '<', '430'),   # not over the shop
   timer_over('Tiro', 'Cooldowns[Jogo.Nivel[0]]')], [
    reset_timer('Tiro'),
    create('Impacto', 'CursorX()', 'CursorY()'),
], [
    FOREACH('Pombo', conditions=[
        C('NumberObjectVariable', 'Pombo', 'Morto', '=', '0'),
        compare('Pombo.DistanceToPosition(CursorX(), CursorY())', '<=', 'Raios[Jogo.Nivel[0]] + 12'),
    ], actions=[
        A('SetNumberObjectVariable', 'Pombo', 'Morto', '=', '1'),
        A('ClearForces', 'Pombo'), A('AddForceXY', 'Pombo', '0', '300', '1'),
        set_var('Jogo.Abatidos', '+', '1'),
    ]),
]),
```

Aim a weapon at the cursor and flip it

```
E([], [A('SetAngle', 'Arma', '=', 'ToDeg(atan2(CursorY() - Arma.Y(), CursorX() - Arma.X()))')]),
E([C('CursorX', '', '<', 'Arma.X()', '', '')], [A('FlippableCapability::FlippableBehavior::FlipY', 'Arma', 'Flip
» pable', 'yes')]),
ELSE([A('FlippableCapability::FlippableBehavior::FlipY', 'Arma', 'Flippable', 'no')]),
```

Objects orbiting a centre (fake 3D)

```
E([], [
    A('SetNumberObjectVariable', 'Ajudantes', 'Ang', '+', 'Ajudantes.Vel * TimeDelta()'),
    A('SetCenter', 'Ajudantes', '=', 'Nave.X() + cos(ToRad(Ajudantes.Ang)) * Ajudantes.Raio',
      '=', 'Nave.Y() + sin(ToRad(Ajudantes.Ang)) * Ajudantes.Raio * 0.45'),
    A('ChangePlan', 'Ajudantes', '=', 'round(sin(ToRad(Ajudantes.Ang)) * 5)'),   # behind/in front of z=0
    set_scale('Ajudantes', '=', '0.85 + sin(ToRad(Ajudantes.Ang)) * 0.15'),
]),
```

Keep N visual helpers in sync with a level (works after loading too)

```
E([compare('SceneInstancesCount(Drone)', '<', 'min(Jogo.Nivel[1], 8)')], [
    create('Drone', 'Nave.X()', 'Nave.Y()'),
    A('SetNumberObjectVariable', 'Drone', 'Ang', '=', 'RandomInRange(0, 359)'),
]),      # creates one per frame until the count matches
```

Shop with arrays, else-if chain and max level

```
E(clicked('Botao'), [], [
    E([compare('Jogo.Nivel[Botao.Id]', '>=', 'NivelMax[Botao.Id]')], [play_sound('negado.wav')]),
    ELSE(conditions=[compare('Jogo.Dinheiro', '>=', 'Jogo.Custo[Botao.Id]')], actions=[
        set_var('Jogo.Dinheiro', '-', 'Jogo.Custo[Botao.Id]'),
        set_var('Jogo.Nivel[Botao.Id]', '+', '1'),
        set_var('Jogo.Custo[Botao.Id]', '=', 'ceil(Jogo.Custo[Botao.Id] * Mult[Botao.Id])'),
        play_sound('compra.wav')]),
    ELSE([play_sound('negado.wav'),
          create('TextoMais', 'CursorX() - 60', 'CursorY() - 60', 'UI'),
          set_text('TextoMais', '"Faltam R$ " + ToString(Jogo.Custo[Botao.Id] - Jogo.Dinheiro) + "!"')]),
]),
# derived values every frame:
E([], [set_var('Preco', '=', '3 + Jogo.Nivel[1] * 2'), set_var('TempoFritura', '=', '3 * pow(0.85, Jogo.Nivel[2]
» )')]),
```

Dim what you cannot afford (per instance) + labels from arrays

```
FOREACH('Botao', sub_events=[
    E([compare('Jogo.Dinheiro', '>=', 'Jogo.Custo[Botao.Id]')], [set_opacity('Botao', '=', '255')]),
    ELSE([set_opacity('Botao', '=', '120')]),
]),
E([], [set_text('TextoBotao', 'Nomes[TextoBotao.Id] + NewLine() + "R$ " + ToString(Jogo.Custo[TextoBotao.Id])')]
» ),
# ^ one action, evaluated per TextoBotao instance (each has its own Id)
```

Hover tooltip with Else

```
E([C('IsCursorOnObject', 'Botao', '', '', '')], [
    set_scale('Botao', '=', '1.04'), A('Show', 'CaixaDica', ''),
    set_text('TextoDica', 'Descricoes[Botao.Id]')]),
ELSE([A('Hide', 'CaixaDica')]),
```

Progress bar

```
# Barra: sprite with origin (0,0), 1x1 or small image; width = max * ratio
E([], [A('ResizableCapability::ResizableBehavior::SetWidth', 'Barra', 'Resizable', '=',
         '326 * clamp((Total - Limites[Patente]) / (Limites[Patente + 1] - Limites[Patente]), 0, 1)')]),
```

Production loop with a timed batch (fryer)

```
E([var_is('Fritando', '=', '0'), var_is('Jogo.Carne', '>=', '1')], [
    set_var('Fritando', '=', 'min(Jogo.Carne, Lote)'), set_var('Jogo.Carne', '-', 'Fritando'),
    reset_timer('Fritura')]),
E([var_is('Fritando', '>', '0'), timer_over('Fritura', 'TempoFritura')], [
    set_var('Jogo.Dinheiro', '+', 'Fritando * Preco')], [
    REPEAT('min(Fritando, 6)', [create('Moeda', 'Fritadeira.X()', 'Fritadeira.Y() - 30', 'UI')]),
    E([], [set_var('Fritando', '=', '0')]),      # after the Repeat used it
]),
```

Random "event" with a random instance

```
E([timer_over('Coco', 3), C('PickRandomInstance', '', 'Pombo'),
   C('NumberObjectVariable', 'Pombo', 'Morto', '=', '0')], [
    reset_timer('Coco'), create('Coco', 'Pombo.X()', 'Pombo.Y() + 15'),
    A('AddForceXY', 'Coco', '0', '230', '1')]),
E([C('CollisionNP', 'Coco', 'Balde', '', '', '')], [delete('Coco')]),
```

Two-click reset (confirmation)

```
E(clicked('BotaoReset'), [], [
    E([var_is('Confirmar', '=', '0')], [set_var('Confirmar', '=', '1'),
        set_text('TextoReset', q('Tem certeza?')), reset_timer('Confirmar')]),
    ELSE([A('DeleteGroupFichier', q('MeuJogo'), q('jogo')), A('Scene', '', q('Jogo'), 'yes')]),
]),
E([var_is('Confirmar', '=', '1'), timer_over('Confirmar', 3)], [set_var('Confirmar', '=', '0'),
    set_text('TextoReset', q('Recomeçar'))]),
# put this group LAST so the autosave earlier in the frame cannot rewrite the save after deletion
```

Rank / level up with celebration

```
E([var_is('Jogo.Patente', '<', '5'), compare('Jogo.Total', '>=', 'Limites[Jogo.Patente + 1]')], [
    set_var('Jogo.Patente', '+', '1'), play_sound('patente.wav', 80),
    set_text('TextoBanner', '"NOVA PATENTE!" + NewLine() + Patentes[Jogo.Patente]'),
    set_opacity('TextoBanner', '=', '255')], [
    REPEAT(30, [create('Faisca', 'Nave.X()', 'Nave.Y()', 'UI'),
                A('AddForceXY', 'Faisca', 'RandomInRange(-420, 420)', 'RandomInRange(-420, 420)', '1')])]),
E([], [set_animation('Nave', 'Jogo.Patente')]),   # visuals follow the saved state every frame
```

Platformer: animation, flip, camera, fall = restart

```
E([C('PlatformBehavior::PlatformerObjectBehavior::IsMovingEvenALittle', 'Gato', 'PlatformerObject')],
  [A('AnimatableCapability::AnimatableBehavior::SetName', 'Gato', 'Animation', '=', q('Correndo'))]),
ELSE([A('AnimatableCapability::AnimatableBehavior::SetName', 'Gato', 'Animation', '=', q('Parado'))]),
E([key_pressed('Left')], [A('FlippableCapability::FlippableBehavior::FlipX', 'Gato', 'Flippable', 'yes')]),
E([key_pressed('Right')], [A('FlippableCapability::FlippableBehavior::FlipX', 'Gato', 'Flippable', 'no')]),
E([], [A('SetCameraCenterX', '', '=', 'max(480, Gato.X())', '', '')]),
E([C('PosY', 'Gato', '>', '700')], [A('Scene', '', q('Fase'), 'yes')]),
```

Collect items

```
E([C('CollisionNP', 'Gato', 'Moeda', '', '', '')], [set_var('Moedas', '+', '1'), delete('Moeda')]),
# for per-item values use FOREACH('CarneCrua', conditions=[C('CollisionNP', 'CarneCrua', 'Fritadeira', '', '', '
» ')], ...)
```

Countdown and game over

```
E([var_is('Fim', '=', '0')], [set_var('Tempo', '=', 'max(0, 30 - TimerElapsedTime("Jogo"))'),
    set_text('TextoTempo', 'ToString(ceil(Tempo))')]),
E([var_is('Fim', '=', '0'), var_is('Tempo', '<=', '0')], [set_var('Fim', '=', '1'), A('Hide', 'Estrela'),
    set_text('TextoFim', '"FIM DE JOGO!" + NewLine() + "Aperte R para jogar de novo"')]),
E([var_is('Fim', '=', '1'), key_pressed('r')], [A('Scene', '', q('Jogo'), 'yes')]),
```

Platformer animation chain (else-if)

```
E([C('PlatformBehavior::IsJumping', 'Sapo', 'PlatformerObject')],
  [A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Pulando'))]),
ELSE(conditions=[C('PlatformBehavior::IsFalling', 'Sapo', 'PlatformerObject')],
     actions=[A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Pulando'))]),
ELSE(conditions=[C('PlatformBehavior::PlatformerObjectBehavior::IsMovingEvenALittle', 'Sapo', 'PlatformerObject'
» )],
     actions=[A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Correndo'))]),
ELSE([A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Parado'))]),
E([C('PlatformBehavior::IsJumping', 'Sapo', 'PlatformerObject'), C('BuiltinCommonInstructions::Once')],
  [play_sound('pulo.wav', 50)]),              # jump sound once per jump
```

Patrol between per-instance limits (enemies, moving platforms)

```
# each instance gets initialVariables Min/Max; conditions compare each instance with its own values
E([C('NumberObjectVariable', 'Besouro', 'Morto', '=', '0')],
  [A('SetX', 'Besouro', '+', 'Besouro.Dir * Besouro.Vel * TimeDelta()')]),
E([C('PosX', 'Besouro', '<', 'Besouro.Min')], [A('SetNumberObjectVariable', 'Besouro', 'Dir', '=', '1')]),
E([C('PosX', 'Besouro', '>', 'Besouro.Max')], [A('SetNumberObjectVariable', 'Besouro', 'Dir', '=', '-1')]),
# a sprite with platform() moved like this carries a PlatformerObject standing on it (verified)
```

Stomp an enemy, or get hurt from the side

```
E([C('CollisionNP', 'Sapo', 'Besouro', '', '', ''), C('NumberObjectVariable', 'Besouro', 'Morto', '=', '0'),
   C('PlatformBehavior::IsFalling', 'Sapo', 'PlatformerObject')], [
    A('SetNumberObjectVariable', 'Besouro', 'Morto', '=', '1'),
    A('AnimatableCapability::AnimatableBehavior::SetName', 'Besouro', 'Animation', '=', q('Esmagado')),
    A('PlatformBehavior::SetCanJump', 'Sapo', 'PlatformerObject'),        # allow a jump in mid-air...
    A('PlatformBehavior::SimulateJumpKey', 'Sapo', 'PlatformerObject')]),   # ...and bounce
E([C('CollisionNP', 'Sapo', 'Besouro', '', '', ''), C('NumberObjectVariable', 'Besouro', 'Morto', '=', '0'),
   var_is('Invencivel', '<=', '0')], [set_var('Vidas', '-', '1'), set_var('Invencivel', '=', '2')]),
```

Invincibility blink after damage

```
E([var_is('Invencivel', '>', '0')], [set_var('Invencivel', '-', 'TimeDelta()'),
    set_opacity('Sapo', '=', '255 - 180 * mod(round(TimeFromStart() * 12), 2)')]),
ELSE([set_opacity('Sapo', '=', '255')]),
```

Pit: lose a life and respawn

```
E([at_start()], [set_var('InicioX', '=', 'Sapo.X()'), set_var('InicioY', '=', 'Sapo.Y()')]),
E([C('PosY', 'Sapo', '>', '640')], [set_var('Vidas', '-', '1'), A('SetXY', 'Sapo', '=', 'InicioX', '=', 'InicioY
» ')]),
E([var_is('Vidas', '<=', '0')], [A('Scene', '', q('Fim'), 'yes')]),
```

Hearts HUD with a tiled sprite

```
# Coracoes = tiled_sprite('Coracoes', 'coracao.png', 21, 21) on the UI layer; one heart = 21 px
E([], [A('ResizableCapability::ResizableBehavior::SetWidth', 'Coracoes', 'Resizable', '=', 'max(0, Vidas) * 21')
» ]),
```

Camera clamped to the level + parallax layer

```
E([], [A('SetCameraCenterX', '', '=', 'clamp(Sapo.X(), 480, LarguraFase - 480)', '', ''),
       A('SetCameraCenterY', '', '=', '270', '', ''),
       A('SetCameraCenterX', '', '=', '480 + (CameraCenterX() - 480) * 0.4', q('Montanhas'), '')]),
# layers bottom->top: 'Ceu' (fixed), 'Montanhas' (slow), '' (game), 'UI' (HUD)
```

Level complete -> next scene from a variable

```
E([C('CollisionNP', 'Sapo', 'Bandeira', '', '', ''), var_is('Completa', '=', '0')], [
    set_var('Completa', '=', '1'), set_text('TextoBanner', q('Fase completa!')),
    A('PlatformBehavior::IgnoreDefaultControls', 'Sapo', 'PlatformerObject', 'yes'),   # freeze the player
    reset_timer('Fim')]),
E([var_is('Completa', '=', '1'), timer_over('Fim', 2.5)], [A('Scene', '', 'Proxima', 'yes')]),   # Proxima: scen
» e string var
```

Menu keys: released, not pressed

```
E([key_released('Return')], [A('Scene', '', q('Fase1'), 'yes')]),
E([C('MouseButtonFromTextReleased', '', q('Left'))], [A('Scene', '', q('Fase1'), 'yes')]),
```

Hide a custom crosshair over UI

```
E([], [A('SetXY', 'Mira', '=', 'CursorX()', '=', 'CursorY()')]),
E([compare('CursorY()', '<', '430')], [A('Show', 'Mira', '')]),
ELSE([A('Hide', 'Mira')]),
```

10. .a3p encoding and delivery

The editor saves every project as: `"A3P1:" + base64( zlib_deflate(utf8 JSON) XOR key )`, key = the ASCII bytes of `Assemble3/a3p#7f3c9e21b54d80a6` repeated. "zlib deflate" is the RFC 1950 zlib stream (Python `zlib.compress`, PHP `gzcompress`, browser `CompressionStream("deflate")`). It is obfuscation, not security. Plain JSON also opens, but deliver encoded files - that is what users get from the editor.

```
import base64, zlib
KEY = b'Assemble3/a3p#7f3c9e21b54d80a6'

def encode_a3p(json_text):
    raw = bytearray(zlib.compress(json_text.encode('utf-8'), 6))
    for i in range(len(raw)): raw[i] ^= KEY[i % len(KEY)]
    return 'A3P1:' + base64.b64encode(bytes(raw)).decode()

def decode_a3p(text):
    if not text.startswith('A3P1:'): return text
    raw = bytearray(base64.b64decode(text[5:]))
    for i in range(len(raw)): raw[i] ^= KEY[i % len(KEY)]
    return zlib.decompress(bytes(raw)).decode('utf-8')
```

Verified: files encoded this way decode byte-identically with the editor's decoder (browser `DecompressionStream`) and the website's PHP decoder, and vice versa.

10.1 Delivery checklist

1. Encode the **libGD round-tripped** JSON (`game.rt.json` from validate.js): it is exactly what the editor would write.
2. `python3 tools/a3p.py decode Game.a3p check.json` and `cmp check.json game.rt.json` (identical).
3. `node tools/validate.js Game.a3p` (validate.js reads .a3p directly) -> `OK: no errors`.
4. Name the file after the game, ASCII, no spaces: `ClickerEspacial.a3p`, `KentuckyFriedPombo.a3p`.
5. Size: typical 0.2-2 MB. The server limit per project is 50 MB and accounts have a 2 GB storage quota.
6. Send the file to the user (in Claude Code: the SendUserFile tool) together with one screenshot from the play-test.
7. Tell the user: open it at assemble3.com/editor with ABRIR -> Arquivo... (or drag it in), then Preview.

10.2 Changing an existing game

- **You have the build script** (you made the game): change the Python, rebuild, re-run validate/export/play-test, re-encode. Keep `projectUuid` stable if the game was already published (pass it in by editing `to_json()["properties"]["projectUuid"]` before saving) so the gallery treats it as a new version.
- **You only have the .a3p** (the user edited it in the editor): decode it (`a3p.py decode`), load the JSON in Python, modify it surgically (append events to `layouts[i]["events"]`, add objects/resources/variables with the a3lib helpers, which return plain dicts), keep everything else (user changes!), validate, re-encode.
- Objects the user created in the editor may live in the project `objects` (default) or in the layout `objects`. Look in both before adding one with the same name.
- Resources edited in the cloud may be `a3file:<sha256>:<mime>` references instead of data URLs (cloud saves move files out). A file downloaded with "Baixar uma cópia"/local save contains data URLs again; ask the user for a local copy if you need to export/play-test a cloud project.

11. Making the assets

You will usually generate every asset yourself with Python - no downloads, no license questions. Draw at 4x and downsample with LANCZOS for smooth art; draw tiny and upscale with NEAREST for pixel art. Always render a **contact sheet** (all assets composited on the background) and look at it before building.

11.1 Smooth vector-style art (Pillow)

```
from PIL import Image, ImageDraw, ImageFilter
SS = 4                                  # supersampling factor
def canvas(w, h): return Image.new('RGBA', (w * SS, h * SS), (0, 0, 0, 0))
def P(pts): return [(x * SS, y * SS) for x, y in pts]
img = canvas(240, 240); d = ImageDraw.Draw(img)
d.polygon(P([(120, 22), (142, 70), (146, 150), (94, 150), (98, 70)]), fill=(235, 238, 245, 255),
          outline=(60, 60, 80, 255), width=SS * 3)      # body with outline
glow = Image.new('RGBA', img.size); ImageDraw.Draw(glow).ellipse(P([(60, 60), (180, 180)]), fill=(255, 200, 60,
» 150))
img = Image.alpha_composite(glow.filter(ImageFilter.GaussianBlur(18 * SS)), img)   # soft glow behind
img.resize((240, 240), Image.LANCZOS).save('assets/nave.png')
```

- Leave transparent margins for glows/flames, but remember the default collision mask is the full image box.
- Backgrounds: numpy gradients + blurred colour blobs + random 1-px stars; save JPEG (q 85) if large.
- Buttons: rounded rectangles with a 2-3 px outline in the accent colour, a darker shadow offset by 3 px, an icon at the left and empty space at the right for a Text object.
- Readability for kids: thick dark outlines, saturated colours, big shapes, high contrast with the background.

11.2 Pixel art

```
PAL = {'k': (34, 28, 40), 'g': (150, 156, 172), 'w': (255, 255, 255), 'o': (255, 120, 30), 'y': (250, 200, 60)}
PIGEON = ['......kk..........',
          '.....kllk.........',
          '...  (one string per row, one char per pixel, "." = transparent)']
def from_ascii(rows):
    img = Image.new('RGBA', (max(map(len, rows)), len(rows)), (0, 0, 0, 0))
    for y, row in enumerate(rows):
        for x, ch in enumerate(row):
            if ch in PAL: img.putpixel((x, y), PAL[ch] + (255,))
    return img
def up(img, f=3): return img.resize((img.width * f, img.height * f), Image.NEAREST)
up(from_ascii(PIGEON)).save('assets/pombo_1.png')
```

- Pick one pixel scale for the whole game (x3 at 960x540 worked well) and pre-scale every image by it.
- Project: `Project(..., pixel_art=True)` -> scaleMode nearest, pixelsRounding, no antialiasing, resources `smoothed: false`. Text objects: `smoothed=False` and a pixel font.
- Backgrounds: draw at 320x180 (dithered sky bands, simple buildings with lit windows) and upscale x3.
- Two-frame animations (wings up/down) loop at 0.1-0.15 s; "dead" frames can be the flipped image with X eyes.

11.3 Sounds (numpy -> WAV)

```
import numpy as np, wave
RATE = 22050
def wav(name, s):
    s = np.clip(s, -1, 1)
    with wave.open(name, 'wb') as w:
        w.setnchannels(1); w.setsampwidth(2); w.setframerate(RATE); w.writeframes((s * 30000).astype('<i2').toby
» tes())
def sweep(freqs, dur, kind='square', vol=0.3):
    n = int(RATE * dur); f = np.interp(np.arange(n), np.linspace(0, n, len(freqs)), freqs)
    ph = np.cumsum(f) / RATE
    return (np.where(ph % 1 < 0.5, vol, -vol) if kind == 'square' else np.sin(2 * np.pi * ph) * vol)
def env(n, k): return np.exp(-np.arange(n) / RATE * k)          # exponential decay
s = sweep([1500, 500], 0.09, 'sine', 0.5); wav('assets/clique.wav', s * env(len(s), 30))     # click/pew
coin = np.concatenate([sweep([988], 0.07), sweep([1319], 0.22)]); wav('assets/moeda.wav', coin)  # coin
noise = np.random.default_rng(1).uniform(-1, 1, int(RATE * 0.2)); wav('assets/tiro.wav', noise * env(len(noise),
»  20))
```

- Typical set: click/hit, purchase (arpeggio), denied (low buzz), reward/coin, level-up fanfare, explosion (low noise), ambient loop (optional, keep short).
- Vary pitch per play with the `PlaySound` pitch parameter (`RandomFloatInRange(0.9, 1.2)`) so repeated sounds are not tiring. Keep volumes 40-80.

11.4 Fonts

The Assemble3 repo bundles fonts with its examples (`editor/newIDE/app/public/examples/*/assets/`). These cover Portuguese accents (ã ç é ô ...) and are license-safe to embed:

Font file (example folder)	Style	License	Missing glyphs
Oxanium-Medium.ttf (game-feel-demo)	sci-fi, clean	OFL	★ → (avoid symbols)
Kenney_Mini.ttf (keyboard-practice)	pixel font (sizes 16/24/32)	Kenney (CC0/CC-BY-SA meta)	-
Bungee-Regular.ttf (babyfoot-3d)	bold display	OFL	★
LuckiestGuy.ttf (pairs)	cartoon	Apache 2.0	-
Ranchers-Regular.ttf (object-slicer)	comic	OFL	-
PTSans-Bold.ttf (space-asteroids)	neutral sans	OFL	-
Kenney_Future_Narrow.ttf (minesweeper)	futuristic	Kenney	-

Check glyph coverage before using a font: `fontTools.ttLib.TTFont(path).getBestCmap()` must contain every character of your texts. Draw stars/hearts/arrows as sprites, not characters. If no font is set, the runtime uses a default sans-serif.

12. Behaviors

Behaviors give objects ready-made logic. Add them in the object's `behaviors` array (name = how events refer to it). The layout's `behaviorsSharedData` is filled by libGD automatically. Defaults below were dumped from libGD 5.6.282 (a3lib has helpers with these defaults and keyword overrides).

Behavior type (helper)	Default name	Key properties (defaults)
PlatformBehavior::PlatformerObjectBehavior (platformer_object)	PlatformerObject	gravity 1000, maxFallingSpeed 700, maxSpeed 250, acceleration/deceleration 1500, jumpSpeed 600, jumpSustainTime 0.2, ignoreDefaultControls false, canGrabPlatforms false, slopeMaxAngle 60
PlatformBehavior::PlatformBehavior (platform)	Platform	platformType NormalPlatform | Jumpthru | Ladder, canBeGrabbed
TopDownMovementBehavior::TopDownMovementBehavior (top_down)	TopDownMovement	maxSpeed 200, acceleration 400, deceleration 800, allowDiagonals, rotateObject true, viewpoint TopDown
DraggableBehavior::Draggable (draggable)	Draggable	checkCollisionMask true
DestroyOutsideBehavior::DestroyOutside (destroy_outside)	DestroyOutside	extraBorder 200
AnchorBehavior::AnchorBehavior (anchor)	Anchor	left/top/right/bottomEdgeAnchor (0 none, 1 left/top, 2 right/bottom, 3 proportional)
PathfindingBehavior::PathfindingBehavior	Pathfinding	cellWidth/Height 20, maxSpeed 200, allowDiagonals
PathfindingBehavior::PathfindingObstacleBehavior	PathfindingObstacle	impassable true, cost 2

IMPORTANT Assemble3 runtime controls for PlatformerObject (default controls on): Left/Right arrows move, **Up arrow jumps** (Space does NOT), Down or **S** drops through jump-thru platforms / climbs down ladders. TopDownMovement uses the four arrows. Tell users the controls in a Comment and in an on-screen hint.

```
p.add_object(sprite('Gato', [anim('Parado', ['gato.png']), anim('Correndo', ['gato.png', 'gato_corre.png'], loop
» =True)],
                    behaviors=[platformer_object(maxSpeed=280, jumpSpeed=650)]))
p.add_object(tiled_sprite('Chao', 'chao.png', 24, 24, behaviors=[platform()]))
# events: C('PlatformBehavior::IsOnFloor', 'Gato', 'PlatformerObject'), A('PlatformBehavior::SimulateJumpKey', '
» Gato', 'PlatformerObject')
```

Behavior conditions/actions/expressions are listed in Appendix E under "Behavior: ...". Their second parameter is always the behavior name. Avoid JavaScript-extension behaviors (Tween, Physics 2, etc.) - see 3.4.

13. Designing a game from a prompt

13.1 Procedure

1. **Extract requirements**: theme, core verb (click, shoot, jump, drag, avoid), win/lose or endless, progression (upgrades, levels, ranks), audience (kids/teens -> simple controls), language (pt-BR default), style (pixel / smooth), platform (mouse + touch: prefer click/tap mechanics; keyboard games need on-screen hints).
2. **Choose resolution and layout**: 1280x720 or 960x540 landscape. Sketch screen zones: play area, HUD (top), shop (right or bottom panel), hints, buttons. Keep shop clicks from triggering game actions (check `CursorY()`).
3. **List objects** (with animations, origins, layers) and **variables**: one `Jogo` structure for persistent state; arrays for item tables (names, descriptions, costs, multipliers, max levels); scene numbers for derived values; object variables for per-instance data (Id, speed, state flags).
4. **Plan event groups**: Início (setup/load), core mechanic, production/economy, shop, progression, effects cleanup, HUD, save, reset. Budget conditions+actions per group; total <= 250 (free plan) unless told otherwise.
5. **Balance**: early reward within 10-20 seconds (first upgrade affordable after ~15 clicks), costs growing x1.15-1.6 per level, a big multiplier upgrade late, a rare random bonus (golden variant, comet) for surprise, visible goals (next rank and its threshold), a final "max" state with a message.
6. **Juice** (cheap, high impact): squash on click, floating "+N", particles, sounds with pitch variation, tinted/dimmed buttons, tooltips, banners on level up, things orbiting/flying to the HUD, idle animation.
7. **Readable events**: Groups with pt-BR names, one Comment per idea explaining *why*, Portuguese variable names (Pontos, Dinheiro, Nivel), no clever tricks without a comment. Students will open the event sheet.
8. **Persistence**: autosave every 3 s + load at start + two-click reset. Unique storage name per game.

13.2 Event budget reference (real games)

Game	Features	Conditions + actions
Clicker Espacial	click, 5 upgrades, orbiting helpers, 6 ranks w/ ship change, comet bonus, save/reset	130
Kentucky Fried Pombo	hold-to-shoot, 5 guns, spawns, golden variant, poop gag, fryer pipeline, 5 upgrades, tooltips, coins to HUD, save/reset	201
Caça-Estrelas (sample)	30-s click challenge, sparkles, best score saved, restart	33
Pulo do Gato (sample)	platformer, coins, camera, animations, fall restart	15
Sapo Saltitante (sample 3)	title + 2 levels sharing an external sheet + end screen; stomp/patrol enemies, spikes, pits, jump-thru, moving platforms, lives HUD, parallax, best score	124

14. Validation with libGD

`tools/validate.js` loads the project into the real engine core (the same libGD the editor uses), checks every instruction and expression, generates each layout's JavaScript exactly like a preview/export, and writes the libGD-serialized project (round trip). It accepts `.json` or `.a3p`.

```
$ node tools/validate.js mygame.json mygame.rt.json
layout "Jogo": 130 conditions+actions, 63334 bytes of generated code
TOTAL events counted by the Assemble3 limit: 130 (guest 100 / free 250 / premium 500)
OK: no errors
```

With external events it also prints `external events "Logica" (scope: Fase1): N conditions+actions` and checks that every Link target exists; the layouts that link a sheet show 0 own events but their generated code includes it.

Messages from libGD on stderr like `Platform "GDevelop JS platform" is unknown` and `Child previewLayout not found` are harmless. Example of real errors it reports:

```
ERROR Jogo/#6 - SceneJustBegins: has 0 parameters, needs 1
ERROR Jogo/#6 > SetXY #0 (object) - unknown object "Inimigo"
ERROR Jogo/#6 > SetNumberVariable #2 (number) - [number] "1 +* 2": More than one term was found...
ERROR Jogo/#6 > PlatformBehavior::SimulateJumpKey #1 (behavior) - object "Estrela" has no behavior "PlatformerObject"
ERROR Jogo/#6 - unknown action "MudarCor"
ERROR Jogo/#6 > SetNumberVariable #0 (variableOrProperty) - [variable] "Vidas": No variable with this name found.
FAILED: 6 error(s)
```

Error	Cause / fix
unknown condition/action "X"	Wrong type string. Search Appendix E or `node tools/meta.js "<regex>"` (prints matching instructions with parameters).
has N parameters, needs M	Missing code-only `""` parameters or optional ones; copy the parameter list from Appendix E.
unknown object	Typo or object not added; groups must be declared in objectsGroups.
No variable with this name found	Declare the variable (scene/global/object) or fix the path.
You must enter a number or a valid expression call	Bad expression: quotes in a number field, a missing operator, wrong function name.
object has no behavior	Behavior not in the object's behaviors array, or wrong name. Capabilities (Scale, Opacity...) are implicit and not checked.
libGD kept N of M conditions+actions	Malformed event JSON (e.g. wrong key names) dropped on load.

NOTE The validator cannot know your intent: a string parameter fed with a number variable is accepted, a sound name with a typo is reported only for resource-typed parameters, and logic errors only show in play-tests.

Discover instructions from the command line with meta.js (regex on type or full name):

```
$ node tools/meta.js "^PlaySound$"
ACT BuiltinAudio :: PlaySound -- Play a sound
   0:currentScene(code)[]""
   1:soundfile[]"Audio file (or audio resource name)"
   2:yesorno?[]"Repeat the sound"=no
   3:expression?[]"Volume"=100
   4:expression?[]"Pitch (speed)"=1
```

15. Export and headless play-testing

15.1 Export

```
$ node tools/export.js mygame.rt.json mygame/export       # optional 3rd arg: GDJS root
exported to mygame/export
```

This runs `gd.Exporter.exportWholePixiProject` with an in-memory file system adapter: data-URL resources are written as files, the game code is generated, and the Assemble3 runtime is copied. The exporter patches `index.html` so the running game is reachable as `window.__game` (a `gdjs.RuntimeGame`).

15.2 Play-test harness

`tools/playtest.js <exportDir> <scenario.js>` starts a small static server, launches headless Chromium (SwiftShader WebGL), runs your scenario, prints PASS/FAIL lines, saves screenshots, and fails on any JavaScript error. Scenario = a module exporting `async t => {...}`:

t.<method>	Purpose
boot(width, height)	Open the game at that viewport (use the game resolution) and wait for the first scene.
reload()	Reload the page (tests saving/loading).
variable(path) / setVariable(path, v)	Read/write scene variables: "Pontos", "Jogo.Estrelas", "Jogo.Nivel.0".
objects(name)	Array of {x, y, cx, cy, w, h, angle, visible, layer, z, animation, opacity, text}.
objectVariable(name, i, var)	Number value of an instance variable.
count(name), sceneName()	Instance count; current scene.
click(x, y) / clickObject(name, i)	Click at scene coordinates (camera and canvas scaling handled).
hold(x, y, ms)	Press and hold the mouse (hold-to-fire).
key(name, ms)	Press a key: Playwright names "ArrowUp", "ArrowLeft", "Space", "r"...
shot(name)	Screenshot to shots/<name>.png - open it and look.
check(label, cond, details)	Assertion line (PASS/FAIL).
forceRandom(value, ms)	Make Math.random return value for ms (forces rare random branches).
clearStorage()	Clear localStorage (fresh start; saves persist between runs otherwise).
globalVariable(path) / setGlobalVariable(path, v)	Same for global (project) variables.
setPosition(name, i, x, y)	Teleport an instance (test stomps, pits, flags without playing the level).
cameraX(layer)	Camera centre X of a layer ("" = base) - camera follow and parallax checks.
page, sleep(ms)	Raw Playwright page; delay.

```
// Scenario for sample_clicker.py (Caça-Estrelas)
module.exports = async t => {
  await t.boot(960, 540);
  await t.clearStorage();
  await t.shot('clicker-1-start');
  t.check('starts with 0 points', (await t.variable('Pontos')) === 0);
  for (let i = 0; i < 5; i++) await t.clickObject('Estrela');   // the star moves after each click
  t.check('5 clicks = 5 points', (await t.variable('Pontos')) === 5, await t.variable('Pontos'));
  t.check('sparkles were created', (await t.count('Brilho')) > 0, await t.count('Brilho'));
  await t.shot('clicker-2-playing');
  await t.setVariable('Tempo', 0);                                  // skip the 30 s wait...
  await t.page.evaluate(() => window.__game.getSceneStack().getCurrentScene().getTimeManager().getTimer('Jogo').
» setTime(31000));
  await t.sleep(300);
  t.check('game over after time is up', (await t.variable('Fim')) === 1);
  t.check('best score saved', (await t.variable('Recorde')) === 5);
  await t.shot('clicker-3-game-over');
  await t.key('r'); await t.sleep(500);
  t.check('R restarts the scene', (await t.variable('Pontos')) === 0 && (await t.variable('Recorde')) === 5);
};
```

```
$ node tools/playtest.js samples/export_caca_estrelas samples/test_clicker.js
PASS starts with 0 points
PASS 5 clicks = 5 points -> 5
PASS sparkles were created -> 30
PASS game over after time is up
PASS best score saved
PASS R restarts the scene
PASS no JavaScript errors in the game -> []
PLAYTEST PASSED | screenshots in samples/shots
```

15.3 What to test

- Start state (variables, texts, which buttons are dimmed) + a start screenshot.
- The core verb: click/hold/press and assert the score/state changed by the expected amount.
- Every upgrade/shop item: give money with setVariable, click each button, assert levels/costs, the max-level path and the "not enough money" path (money unchanged).
- Timed systems: wait (or manipulate timers: `scene.getTimeManager().getTimer("Jogo").setTime(31000)` in page.evaluate; values are in **milliseconds**) and assert outcomes.
- Rare branches: `forceRandom(0.02, 2600)` then assert the special object appeared and gives its reward.
- Progression: set totals high, assert rank/animation/banner; set to max, assert the "max" message.
- Persistence: wait > autosave period, `reload()`, assert state restored; then the reset path.
- Visual review: open 4-8 screenshots (start, mid-game, shop hover, special event, max state) and check overlaps, clipping (art cut at canvas edges), unreadable text, misplaced UI, effects that never disappear.

> **NOTE** Headless Chromium has no audio device; sound actions run silently but must not throw. `GPU stall due to ReadPixels` and `Unable to find texture for resource ""` warnings are harmless.

15.4 Reaching runtime internals in page.evaluate

```
const s = window.__game.getSceneStack().getCurrentScene();
s.getVariables().get('Jogo').getChild('Nivel').getChildAt(0).getAsNumber();
s.getObjects('Pombo').map(o => [o.getX(), o.getY(), o.getAnimation(), o.getVariables().get('Morto').getAsNumber(
» )]);
s.getObjects('TextoInfo')[0].getText();
s.getLayer('').getCameraX();  s.getTimeManager().getTimer('Fritura').getTime();   // ms
window.__game.getGameResolutionWidth();
```

16. Gotchas and troubleshooting

Symptom	Cause	Fix
A HUD icon disappears after a moment	A fade/delete event for particles of the same object also faded the permanent instance	Separate object for the HUD icon
Several created objects all move the same	Create accumulates picked instances	Create inside REPEAT
Action uses an old value in a sub-event	Actions run before sub-events	Reorder; compute first
Text never centres	No instance width	customSize + width on the instance, align center
Jump does not work in the platformer	Assemble3 jumps with Up arrow only	Tell users "seta para cima"
Clicking the shop also fires/scores	Game action ignores where the cursor is	`compare("CursorY()", "<", top_of_shop)`
Objects created by events appear behind others	New instances get z-order 0	Create them on a UI layer or set z
Scaled crosshair/pixel art looks blocky and huge	Scaling pixel art by non-integers	Keep fixed scale, show size another way
Game loses progress	Autosave after reset or not loading	Load at start; reset group last; unique storage name
Text shows boxes	Glyph missing in the font	Check cmap; draw symbols as sprites
Art clipped at the edges	Drawing outside the canvas	Bigger canvas with offset; contact sheet review
Expression shows red in editor	Undeclared variable / wrong syntax	validate.js reports it
Object variable via group fails	Variable missing in one member	Declare in every member object
"Else" never runs	Else attaches to the previous **sibling** event	Place it right after the event
Game too hard/long to test	Real-time waits	setVariable / timer setTime / forceRandom in the scenario
ENTER on a menu skips the next screen too	A held key triggers "pressed" in the new scene	Use key released for scene-changing keys
Linked logic misbehaves in level 2 only	Level 2 lacks a scene variable the shared sheet uses	Build all levels with one function
Download script fails behind proxy	Node HTTP clients ignore the proxy	Use curl
Building the editor is blocked	Not needed for games	Use libGD + GDJS runtime only

16.1 Things the toolchain cannot check

- How the editor UI displays the events (it uses the same libGD, so valid JSON displays; but take care that comments and names read well).
- JavaScript-extension features (not loaded in Node).
- Audio quality (listen is impossible headless) - keep sounds simple and short.
- Mobile/touch specifics: mouse conditions also fire on touch (mouse emulation), but hover tooltips do not exist on touch screens - never hide essential information behind hover.

17. Final checklist

1. Prompt requirements all implemented (re-read the prompt).
2. pt-BR texts (or the requested language); no "Construct"/"Scirra"; no real brands/logos in parodies.
3. Comments + Groups in the event sheet explain the game for students.
4. validate.js: `OK: no errors`; event total within the user's plan (<= 250 by default) - tell the user the number.
5. Exported and play-tested: all checks PASS, no JavaScript errors; screenshots reviewed and look right.
6. Persistence and reset tested (if the game saves).
7. Encoded from the round-tripped JSON; decode check identical; file named after the game.
8. Delivered the .a3p + a screenshot; summary includes controls, mechanics, tweakable variables, plan-limit note, and what was not verified (e.g. not opened in the editor UI).
9. Repository untouched (`git status` clean) unless the user asked otherwise.

Appendix A - a3lib.py (builder library)

Copy this file next to your build script (also attached to the PDF as `a3lib.py`). Lines longer than the page are wrapped with a leading » on the continuation - use the attached file for exact code.

```
"""a3lib - build Assemble3 (.a3p) / GDevelop 5.6 projects from Python.

Usage sketch:
    from a3lib import *
    p = Project('Meu Jogo', 1280, 720)
    p.image('assets/nave.png')                      # resource name = file name
    p.add_object(sprite('Nave', [anim('', ['nave.png'])], assets_dir='assets'))
    lay = p.layout('Jogo')                          # first layout = start scene
    lay.instance('Nave', 640, 360)
    lay.events = [E([C('SceneJustBegins', '')], [A('SetXY', 'Nave', '=', '100', '=', '100')])]
    p.save_json('game.json'); p.save_a3p('game.a3p')

Multi-level games: put the shared logic in p.external_events('Logica', events, 'Fase1') and add LINK('Logica')
to each level layout; use project-wide p.variables for values that survive scene changes (lives, score);
p.start_layout = 'Titulo' picks the scene that starts the game.
"""
import base64, json, os, uuid, zlib

A3P_KEY = b'Assemble3/a3p#7f3c9e21b54d80a6'
MIME = {'png': 'image/png', 'jpg': 'image/jpeg', 'jpeg': 'image/jpeg', 'gif': 'image/gif', 'webp': 'image/webp',
        'wav': 'audio/wav', 'mp3': 'audio/mpeg', 'ogg': 'audio/ogg', 'aac': 'audio/aac',
        'ttf': 'font/ttf', 'otf': 'font/otf', 'woff': 'font/woff', 'woff2': 'font/woff2', 'json': 'application/json'}


def uid():
    return str(uuid.uuid4())


def data_url(path):
    ext = path.rsplit('.', 1)[1].lower()
    with open(path, 'rb') as f:
        return 'data:%s;base64,%s' % (MIME[ext], base64.b64encode(f.read()).decode())


# ---------------------------------------------------------------- .a3p encoding
def encode_a3p(json_text):
    """'A3P1:' + base64(zlib(JSON) XOR key) - identical to the editor and the site."""
    raw = bytearray(zlib.compress(json_text.encode('utf-8'), 6))
    for i in range(len(raw)):
        raw[i] ^= A3P_KEY[i % len(A3P_KEY)]
    return 'A3P1:' + base64.b64encode(bytes(raw)).decode()


def decode_a3p(text):
    if not text.startswith('A3P1:'):
        return text  # plain JSON is accepted too
    raw = bytearray(base64.b64decode(text[5:]))
    for i in range(len(raw)):
        raw[i] ^= A3P_KEY[i % len(A3P_KEY)]
    return zlib.decompress(bytes(raw)).decode('utf-8')


# ---------------------------------------------------------------- variables
def num(name, v=0):
    return {'name': name, 'type': 'number', 'value': v}


def text_var(name, v=''):
    return {'name': name, 'type': 'string', 'value': v}


def bool_var(name, v=False):
    return {'name': name, 'type': 'boolean', 'value': v}


def array(name, values):
    def child(v):
        if isinstance(v, bool):
            return {'type': 'boolean', 'value': v}
        if isinstance(v, str):
```

```
            return {'type': 'string', 'value': v}
        if isinstance(v, dict):
            return v  # already a variable (structure/array), without name
        return {'type': 'number', 'value': v}
    return {'name': name, 'type': 'array', 'children': [child(v) for v in values]}


def structure(name, children):
    """children: list of named variables (num(), text_var(), array(), structure()...)."""
    return {'name': name, 'type': 'structure', 'children': list(children)}


# ------------------------------------------------------------------ objects
def _image_size(assets_dir, fn):
    from PIL import Image
    return Image.open(os.path.join(assets_dir, fn)).size


def frame(image, origin=None, assets_dir='assets', points=(), collision=None):
    """One sprite frame. origin=None -> image centre. points: [(name, x, y)].
    collision: list of polygons [[(x,y),...]] or None for the automatic bounding box."""
    w, h = _image_size(assets_dir, image)
    ox, oy = origin if origin is not None else (w // 2, h // 2)
    return {'hasCustomCollisionMask': collision is not None, 'image': image,
            'points': [{'name': n, 'x': x, 'y': y} for n, x, y in points],
            'originPoint': {'name': 'origine', 'x': ox, 'y': oy},
            'centerPoint': {'automatic': True, 'name': 'centre', 'x': 0, 'y': 0},
            'customCollisionMask': [[{'x': x, 'y': y} for x, y in poly] for poly in (collision or [])]}


def anim(name, frames, loop=False, seconds_per_frame=0.08, origin=None, assets_dir='assets'):
    """frames: list of image names or frame() dicts."""
    fr = [f if isinstance(f, dict) else frame(f, origin, assets_dir) for f in frames]
    return {'name': name, 'useMultipleDirections': False,
            'directions': [{'looping': loop, 'timeBetweenFrames': seconds_per_frame, 'sprites': fr}]}


def sprite(name, animations, variables=(), behaviors=(), effects=()):
    return {'adaptCollisionMaskAutomatically': True, 'assetStoreId': '', 'name': name, 'type': 'Sprite',
            'updateIfNotVisible': False, 'variables': list(variables), 'effects': list(effects),
            'behaviors': list(behaviors), 'animations': animations}


def simple_sprite(name, image, variables=(), origin=None, behaviors=(), assets_dir='assets'):
    return sprite(name, [anim('', [frame(image, origin, assets_dir)])], variables, behaviors)


def text(name, string, size=20, color=(0, 0, 0), font='', align='left', outline=None, outline_thickness=2,
         shadow=False, bold=False, italic=False, smoothed=True, variables=(), behaviors=()):
    """font = font resource name ('' = default). color/outline = (r, g, b). align: left|center|right."""
    rgb = '%d;%d;%d' % tuple(color)
    return {'assetStoreId': '', 'bold': bold, 'italic': italic, 'name': name, 'smoothed': smoothed,
            'type': 'TextObject::Text', 'underlined': False, 'variables': list(variables), 'effects': [],
            'behaviors': list(behaviors), 'string': string, 'font': font, 'textAlignment': align,
            'characterSize': size, 'color': dict(zip('rgb', color)),
            'content': {'bold': bold, 'isOutlineEnabled': outline is not None, 'isShadowEnabled': shadow,
                        'italic': italic, 'outlineColor': '%d;%d;%d' % tuple(outline or (0, 0, 0)),
                        'outlineThickness': outline_thickness, 'shadowAngle': 90, 'shadowBlurRadius': 2,
                        'shadowColor': '0;0;0', 'shadowDistance': 4, 'shadowOpacity': 127, 'smoothed': smoothed,
                        'underlined': False, 'text': string, 'font': font, 'textAlignment': align,
                        'verticalTextAlignment': 'top', 'characterSize': size, 'lineHeight': 0, 'color': rgb}}


def tiled_sprite(name, texture, width=32, height=32, variables=(), behaviors=()):
    return {'assetStoreId': '', 'height': height, 'name': name, 'texture': texture,
            'type': 'TiledSpriteObject::TiledSprite', 'width': width, 'variables': list(variables),
            'effects': [], 'behaviors': list(behaviors)}


def panel_sprite(name, texture, width, height, margins=(8, 8, 8, 8), tiled=False, variables=(), behaviors=()):
```

```
    left, top, right, bottom = margins
    return {'assetStoreId': '', 'bottomMargin': bottom, 'height': height, 'leftMargin': left, 'name': name,
            'rightMargin': right, 'texture': texture, 'tiled': tiled, 'topMargin': top,
            'type': 'PanelSpriteObject::PanelSprite', 'width': width, 'variables': list(variables),
            'effects': [], 'behaviors': list(behaviors)}


def shape_painter(name, fill=(255, 255, 255), outline=(0, 0, 0), outline_size=1, absolute=False, clear=True):
    return {'assetStoreId': '', 'name': name, 'type': 'PrimitiveDrawing::Drawer', 'variables': [], 'effects': [],
            'behaviors': [], 'fillOpacity': 255, 'outlineSize': outline_size, 'outlineOpacity': 255,
            'absoluteCoordinates': absolute, 'clearBetweenFrames': clear, 'antialiasing': 'none',
            'fillColor': dict(zip('rgb', fill)), 'outlineColor': dict(zip('rgb', outline))}


# ------------------------------------------------------------------- behaviors (defaults dumped from libGD 5.6.282)
def platformer_object(name='PlatformerObject', **kw):
    b = {'acceleration': 1500, 'canGoDownFromJumpthru': True, 'canGrabPlatforms': False,
         'canGrabWithoutMoving': True, 'deceleration': 1500, 'gravity': 1000, 'ignoreDefaultControls': False,
         'jumpSpeed': 600, 'jumpSustainTime': 0.2, 'ladderClimbingSpeed': 150, 'maxFallingSpeed': 700,
         'maxSpeed': 250, 'name': name, 'slopeMaxAngle': 60, 'type': 'PlatformBehavior::PlatformerObjectBehavior',
         'useLegacyTrajectory': False, 'useRepeatedJump': False, 'xGrabTolerance': 10, 'yGrabOffset': 0}
    b.update(kw)
    return b


def platform(name='Platform', platform_type='NormalPlatform', **kw):
    """platform_type: NormalPlatform | Jumpthru | Ladder"""
    b = {'canBeGrabbed': True, 'name': name, 'platformType': platform_type,
         'type': 'PlatformBehavior::PlatformBehavior', 'yGrabOffset': 0}
    b.update(kw)
    return b


def top_down(name='TopDownMovement', **kw):
    b = {'acceleration': 400, 'allowDiagonals': True, 'angleOffset': 0, 'angularMaxSpeed': 180,
         'customIsometryAngle': 30, 'deceleration': 800, 'ignoreDefaultControls': False, 'maxSpeed': 200,
         'movementAngleOffset': 0, 'name': name, 'rotateObject': True,
         'type': 'TopDownMovementBehavior::TopDownMovementBehavior', 'useLegacyTurnBack': False,
         'viewpoint': 'TopDown'}
    b.update(kw)
    return b


def draggable(name='Draggable'):
    return {'checkCollisionMask': True, 'name': name, 'type': 'DraggableBehavior::Draggable'}


def destroy_outside(name='DestroyOutside', extra_border=200):
    return {'extraBorder': extra_border, 'name': name, 'type': 'DestroyOutsideBehavior::DestroyOutside',
            'unseenGraceDistance': 10000}


def anchor(name='Anchor', left=0, top=0, right=0, bottom=0):
    """0 = no anchor, 1 = window left/top edge, 2 = window right/bottom edge, 3 = proportional."""
    return {'bottomEdgeAnchor': bottom, 'leftEdgeAnchor': left, 'name': name, 'relativeToOriginalWindowSize': True,
            'rightEdgeAnchor': right, 'topEdgeAnchor': top, 'type': 'AnchorBehavior::AnchorBehavior',
            'useLegacyBottomAndRightAnchors': False}


# ------------------------------------------------------------------- events
def C(type_, *params, inverted=False):
    """Condition. params = EVERY parameter of the instruction, code-only ones as ''."""
    t = {'inverted': True, 'value': type_} if inverted else {'value': type_}
    return {'type': t, 'parameters': [str(p) for p in params]}


def A(type_, *params):
    """Action. params = EVERY parameter of the instruction, code-only ones as ''."""
    return {'type': {'value': type_}, 'parameters': [str(p) for p in params]}
```

```
def E(conditions=(), actions=(), sub_events=None, local_variables=None):
    e = {'type': 'BuiltinCommonInstructions::Standard', 'conditions': list(conditions), 'actions': list(actions)}
    if sub_events:
        e['events'] = list(sub_events)
    if local_variables:
        e['variables'] = list(local_variables)
    return e


def ELSE(actions=(), conditions=(), sub_events=None):
    """Runs when the previous sibling event did NOT run. With conditions it is an 'else if'."""
    e = {'type': 'BuiltinCommonInstructions::Else', 'conditions': list(conditions), 'actions': list(actions)}
    if sub_events:
        e['events'] = list(sub_events)
    return e


def REPEAT(count_expression, actions=(), conditions=(), sub_events=()):
    return {'type': 'BuiltinCommonInstructions::Repeat', 'repeatExpression': str(count_expression),
            'conditions': list(conditions), 'actions': list(actions), 'events': list(sub_events)}


def FOREACH(object_name, conditions=(), actions=(), sub_events=()):
    return {'type': 'BuiltinCommonInstructions::ForEach', 'object': object_name,
            'conditions': list(conditions), 'actions': list(actions), 'events': list(sub_events)}


def WHILE(while_conditions, conditions=(), actions=(), sub_events=()):
    return {'type': 'BuiltinCommonInstructions::While', 'infiniteLoopWarning': True,
            'whileConditions': list(while_conditions), 'conditions': list(conditions),
            'actions': list(actions), 'events': list(sub_events)}


def LINK(external_events_name):
    """Includes the events of an external events sheet here (they run as if written at this place)."""
    return {'type': 'BuiltinCommonInstructions::Link', 'include': {'includeConfig': 0}, 'target': external_events_name}


def COMMENT(text_, background=(255, 230, 109), color=(0, 0, 0)):
    return {'type': 'BuiltinCommonInstructions::Comment',
            'color': {'r': background[0], 'g': background[1], 'b': background[2],
                      'textR': color[0], 'textG': color[1], 'textB': color[2]}, 'comment': text_}


def GROUP(name, events, color=(74, 144, 226)):
    return {'type': 'BuiltinCommonInstructions::Group', 'name': name, 'source': '', 'creationTime': 0,
            'colorR': color[0], 'colorG': color[1], 'colorB': color[2], 'parameters': [], 'events': list(events)}


def q(s):
    """Quote a Python string as an event string literal: q('Olá') -> '"Olá"'."""
    return json.dumps(s, ensure_ascii=False)


# frequently used shortcuts (modern, visible instructions)
def at_start():
    return C('SceneJustBegins', '')


def clicked(obj, button='Left'):
    """Mouse button released while the cursor/touch is on obj (touch works through mouse emulation)."""
    return [C('MouseButtonFromTextReleased', '', q(button)), C('IsCursorOnObject', obj, '', '', '')]


def key_pressed(key):
    return C('KeyFromTextPressed', '', q(key))


def key_released(key):
```

```
    """Use for menu/scene-changing keys: a key still held when a new scene starts would trigger its 'pressed' events."""
    return C('KeyFromTextReleased', '', q(key))


def compare(a, op, b):
    return C('BuiltinCommonInstructions::CompareNumbers', a, op, b)


def var_is(var, op, value):
    return C('NumberVariable', var, op, value)


def set_var(var, op, value):
    return A('SetNumberVariable', var, op, value)


def set_text(obj, string_expression):
    return A('TextContainerCapability::TextContainerBehavior::SetValue', obj, 'Text', '=', string_expression)


def set_scale(obj, op, value):
    return A('ScalableCapability::ScalableBehavior::SetValue', obj, 'Scale', op, value)


def set_opacity(obj, op, value):
    return A('OpacityCapability::OpacityBehavior::SetValue', obj, 'Opacity', op, value)


def opacity_is(obj, op, value):
    return C('OpacityCapability::OpacityBehavior::Value', obj, 'Opacity', op, value)


def set_animation(obj, index_expression):
    return A('AnimatableCapability::AnimatableBehavior::SetIndex', obj, 'Animation', '=', index_expression)


def create(obj, x, y, layer=''):
    return A('Create', '', obj, x, y, q(layer))


def delete(obj):
    return A('Delete', obj, '')


def play_sound(resource, volume=100, pitch=1, loop=False):
    return A('PlaySound', '', resource, 'yes' if loop else '', volume, pitch)


def reset_timer(name):
    return A('ResetTimer', '', q(name))


def timer_over(name, seconds):
    return C('CompareTimer', '', q(name), '>=', seconds)


def count_events(events):
    """What the Assemble3 event limit counts: every condition and action, sub-events included."""
    n = 0
    for e in events:
        n += len(e.get('conditions', [])) + len(e.get('actions', [])) + len(e.get('whileConditions', []))
        n += count_events(e.get('events', []))
    return n


# ---------------------------------------------------------------------- layout / project
def camera():
    return {'defaultSize': True, 'defaultViewport': True, 'height': 0, 'viewportBottom': 1, 'viewportLeft': 0,
            'viewportRight': 1, 'viewportTop': 0, 'width': 0}
```

```
def layer(name, visible=True):
    return {'ambientLightColorB': 200, 'ambientLightColorG': 200, 'ambientLightColorR': 200,
            'camera2DPlaneMaxDrawingDistance': 5000, 'camera3DFarPlaneDistance': 10000,
            'camera3DFieldOfView': 45, 'camera3DNearPlaneDistance': 3, 'cameraType': '',
            'followBaseLayerCamera': False, 'isLightingLayer': False, 'isLocked': False, 'name': name,
            'renderingType': '', 'visibility': visible, 'cameras': [camera()], 'effects': []}


class Layout:
    def __init__(self, name, background=(50, 50, 50), layers=('',)):
        self.name = name
        self.background = background
        self.layers = list(layers)   # bottom first; '' is the base layer and must exist
        self.instances = []
        self.variables = []          # scene variables
        self.events = []
        self.objects = []            # objects local to this scene (prefer project-wide ones)

    def instance(self, obj, x, y, layer='', z=0, width=None, height=None, angle=0, variables=()):
        """width/height -> custom size (also the wrapping width of a Text object)."""
        i = {'angle': angle, 'customSize': width is not None, 'depth': 0, 'height': height or 0, 'layer': layer,
             'locked': False, 'name': obj, 'persistentUuid': uid(), 'width': width or 0, 'x': x, 'y': y,
             'zOrder': z, 'numberProperties': [], 'stringProperties': [], 'initialVariables': list(variables)}
        self.instances.append(i)
        return i

    def to_json(self):
        r, g, b = self.background
        return {'b': b, 'disableInputWhenNotFocused': True, 'mangledName': self.name.replace(' ', '_32'),
                'name': self.name, 'r': r, 'v': g, 'standardSortMethod': True, 'stopSoundsOnStartup': True,
                'title': '', 'uiSettings': {
                    'grid': False, 'gridType': 'rectangular', 'gridWidth': 32, 'gridHeight': 32, 'gridDepth': 32,
                    'gridOffsetX': 0, 'gridOffsetY': 0, 'gridOffsetZ': 0, 'gridColor': 10401023, 'gridAlpha': 0.8,
                    'snap': False, 'zoomFactor': 1, 'windowMask': False, 'selectedLayer': '',
                    'gameEditorMode': 'instances-editor'},
                'objectsGroups': [], 'variables': self.variables, 'instances': self.instances,
                'objects': self.objects,
                'objectsFolderStructure': {'folderName': '__ROOT',
                                           'children': [{'objectName': o['name']} for o in self.objects]},
                'events': self.events, 'layers': [layer(n) for n in self.layers], 'behaviorsSharedData': []}


class Project:
    def __init__(self, name, width=1280, height=720, pixel_art=False, author='', description='', version='1.0.0',
                 package='com.assemble3.game', orientation='landscape', loading_color=0x000000):
        self.name, self.width, self.height = name, width, height
        self.pixel_art = pixel_art
        self.author, self.description, self.version, self.package = author, description, version, package
        self.orientation, self.loading_color = orientation, loading_color
        self.resources, self.objects, self.groups, self.variables = [], [], [], []
        self.layouts = []
        self.external = []           # external events sheets: shared logic included in layouts with LINK()
        self.start_layout = None     # name of the scene that starts the game (default: the first layout added)
        self._names = set()

    # resources (embedded as data: URLs, like the web editor does)
    def _res(self, kind, path, name, extra):
        name = name or os.path.basename(path)
        assert name not in self._names, 'duplicate resource ' + name
        self._names.add(name)
        r = {'file': data_url(path), 'kind': kind, 'metadata': '', 'name': name, 'userAdded': True}
        r.update(extra)
        self.resources.append(r)
        return name

    def image(self, path, name=None):
        return self._res('image', path, name, {'smoothed': not self.pixel_art})

    def audio(self, path, name=None):
```

```
        return self._res('audio', path, name, {'preloadAsMusic': False, 'preloadAsSound': True, 'preloadInCache': False})

    def music(self, path, name=None):
        return self._res('audio', path, name, {'preloadAsMusic': True, 'preloadAsSound': False, 'preloadInCache': False})

    def font(self, path, name=None):
        return self._res('font', path, name, {})

    def images_from(self, directory, extensions=('.png', '.jpg', '.jpeg', '.webp', '.gif')):
        for f in sorted(os.listdir(directory)):
            if f.lower().endswith(extensions):
                self.image(os.path.join(directory, f))

    def sounds_from(self, directory, extensions=('.wav', '.mp3', '.ogg', '.aac')):
        for f in sorted(os.listdir(directory)):
            if f.lower().endswith(extensions):
                self.audio(os.path.join(directory, f))

    def add_object(self, obj):
        self.objects.append(obj)
        return obj['name']

    def group(self, name, object_names):
        self.groups.append({'name': name, 'objects': [{'name': n} for n in object_names]})

    def layout(self, name, **kw):
        lay = Layout(name, **kw)
        self.layouts.append(lay)
        return lay

    def external_events(self, name, events, associated_layout):
        """Shared event sheet. associated_layout = a layout whose objects/variables it uses (for the editor and the
        validator). Include it in each layout with LINK(name). Its events count once toward the event limit."""
        self.external.append({'name': name, 'associatedLayout': associated_layout, 'events': list(events)})
        return name

    def to_json(self):
        assert self.layouts, 'add at least one layout'
        props = {
            'adaptGameResolutionAtRuntime': True, 'antialiasingMode': 'none' if self.pixel_art else 'MSAA',
            'antialisingEnabledOnMobile': False, 'folderProject': False, 'orientation': self.orientation,
            'packageName': self.package, 'pixelsRounding': self.pixel_art, 'projectUuid': uid(),
            'scaleMode': 'nearest' if self.pixel_art else 'linear', 'sizeOnStartupMode': '', 'templateSlug': '',
            'version': self.version, 'name': self.name, 'description': self.description, 'author': self.author,
            'windowWidth': self.width, 'windowHeight': self.height, 'latestCompilationDirectory': '',
            'maxFPS': 60, 'minFPS': 20, 'verticalSync': False, 'platformSpecificAssets': {},
            'loadingScreen': {'backgroundColor': self.loading_color, 'backgroundFadeInDuration': 0.2,
                              'backgroundImageResourceName': '', 'gdevelopLogoStyle': 'light',
                              'logoAndProgressFadeInDuration': 0.2, 'logoAndProgressLogoFadeInDelay': 0,
                              'minDuration': 1.5, 'progressBarColor': 16777215, 'progressBarHeight': 20,
                              'progressBarMaxWidth': 200, 'progressBarMinWidth': 40, 'progressBarWidthPercent': 30,
                              'showGDevelopSplash': True, 'showProgressBar': True},
            'watermark': {'placement': 'bottom-left', 'showWatermark': True},
            'authorIds': [], 'authorUsernames': [], 'categories': [], 'playableDevices': [],
            'extensionProperties': [], 'platforms': [{'name': 'GDevelop JS platform'}],
            'currentPlatform': 'GDevelop JS platform'}
        return {
            'firstLayout': self.start_layout or self.layouts[0].name,
            'gdVersion': {'build': 282, 'major': 5, 'minor': 6, 'revision': 0},
            'properties': props,
            'resources': {'resources': self.resources, 'resourceFolders': []},
            'objects': self.objects,
            'objectsFolderStructure': {'folderName': '__ROOT', 'children': [{'objectName': o['name']} for o in self.objects]},
            'objectsGroups': self.groups,
            'variables': self.variables,
            'layouts': [l.to_json() for l in self.layouts],
            'externalEvents': self.external, 'eventsFunctionsExtensions': [], 'externalLayouts': [],
        }

    def event_count(self):
        return sum(count_events(l.events) for l in self.layouts) + sum(count_events(x['events']) for x in self.external)

    def save_json(self, path):
        with open(path, 'w', encoding='utf-8') as f:
            json.dump(self.to_json(), f, ensure_ascii=False)
        return path

    def save_a3p(self, path, json_text=None):
        """Pass json_text to encode a specific JSON (e.g. the libGD round-tripped one)."""
        if json_text is None:
            json_text = json.dumps(self.to_json(), ensure_ascii=False)
        with open(path, 'w', encoding='ascii') as f:
            f.write(encode_a3p(json_text))
        return path
```

Appendix B - Tools

tools/setup.sh - downloads libGD, checks the GDJS runtime and Playwright

```
#!/usr/bin/env bash
# One-time setup of the a3kit tools. Run from anywhere: bash tools/setup.sh
set -e
cd "$(dirname "$0")"
# libGD (GDevelop core compiled to WebAssembly) built for the upstream commit Assemble3 is synced with.
# (The exact hash is C3_UPSTREAM_COMMIT in editor/newIDE/app/scripts/import-libGD.js.)
COMMIT=${LIBGD_COMMIT:-c6d09211037373a3cf315f476ebb45629f14b6eb}
for f in libGD.js libGD.wasm; do
  [ -s "$f" ] || curl -fsS -o "$f" "https://s3.amazonaws.com/gdevelop-gdevelop.js/master/commit/$COMMIT/$f"
done
node -e "require('./libGD.js')().then(gd => console.log('libGD', gd.VersionWrapper.fullString()))" 2>/dev/null | grep libGD
# Compiled GDJS game runtime (needed only for export/play-testing).
GDJS=${GDJS_ROOT:-/home/user/Assemble3/editor/newIDE/app/resources/GDJS}
if [ -f "$GDJS/Runtime/gd.js" ]; then echo "GDJS runtime: $GDJS"; else
  echo "GDJS runtime missing at $GDJS. In the Assemble3 repo: (cd editor/GDJS && npm ci) && (cd editor/newIDE/app/scripts && nod
» e import-GDJS-Runtime.js)"; fi
# Playwright (headless Chromium) for play-tests.
node -e "require('playwright')" 2>/dev/null && echo "playwright: OK" || \
  (ls /usr/local/lib/node_modules_global/playwright >/dev/null 2>&1 && echo "playwright: /usr/local/lib/node_modules_global/play
» wright") || \
  echo "playwright missing: npm i -g playwright (and a Chromium build)"
```

tools/validate.js - validation + code generation + libGD round trip

```
// Validate a GDevelop/Assemble3 project JSON with libGD and re-serialize it.
// Usage: node validate.js <project.json|.a3p> [roundtrip-out.json]
// Checks: every condition/action type exists, parameter counts, objects/behaviors exist,
// every expression/variable parameter parses with no error, then generates each layout's
// JavaScript code like the preview/export does. Exit code 1 if any error.
const fs = require('fs');
const path = require('path');
const zlib = require('zlib');
const [, , inPath, outPath] = process.argv;
const KEY = Buffer.from('Assemble3/a3p#7f3c9e21b54d80a6');
const readProject = p => {
  let text = fs.readFileSync(p, 'utf8');
  if (text.startsWith('A3P1:')) {
    const raw = Buffer.from(text.slice(5), 'base64');
    for (let i = 0; i < raw.length; i++) raw[i] ^= KEY[i % KEY.length];
    text = zlib.inflateSync(raw).toString('utf8');
  }
  return JSON.parse(text);
};
const OPERATORS = ['=', '+', '-', '*', '/', '<', '>', '<=', '>=', '!=', 'True', 'False', 'Toggle'];
require(path.join(__dirname, 'libGD.js'))().then(gd => {
  let errors = 0;
  const report = (where, msg) => { errors++; console.log('ERROR', where, '-', msg); };
  try {
    const json = readProject(inPath);
    const project = gd.ProjectHelper.createNewGDJSProject();
    const el = gd.Serializer.fromJSObject(json);
    project.unserializeFrom(el);
    el.delete();
    const platform = gd.JsPlatform.get();
    const rm = project.getResourcesManager();
    const count = evs => evs.reduce((n, e) => n + (e.conditions || []).length + (e.actions || []).length + (e.whileConditions ||
»  []).length + count(e.events || []), 0);
    let total = 0;
    const sheets = json.layouts.map(l => ({ name: l.name, layoutName: l.name, events: l.events, isLayout: true }))
      .concat((json.externalEvents || []).map(x => ({ name: x.name, layoutName: x.associatedLayout, events: x.events, isLayout:
» false })));
    sheets.forEach(layoutJson => {
      if (!project.hasLayoutNamed(layoutJson.layoutName)) { report(layoutJson.name, `associated layout "${layoutJson.layoutName}
» " does not exist`); return; }
      const layout = project.getLayout(layoutJson.layoutName);
      const psc = gd.ProjectScopedContainers.makeNewProjectScopedContainersForProjectAndLayout(project, layout);
      const getObject = name => layout.getObjects().hasObjectNamed(name) ? layout.getObjects().getObject(name)
        : project.getObjects().hasObjectNamed(name) ? project.getObjects().getObject(name) : null;
      const isGroup = name => layout.getObjects().getObjectGroups().has(name) || project.getObjects().getObjectGroups().has(name
» );
      const validateExpr = (where, type, expr, objectName, extra) => {
        const parser = new gd.ExpressionParser2();
        const node = parser.parseExpression(expr).get();
        const v = new gd.ExpressionValidator(platform, psc, type, objectName || '', extra || '');
        node.visit(v);
        const errs = v.getAllErrors();
        for (let i = 0; i < errs.size(); i++) report(where, `[${type}] ${JSON.stringify(expr)}: ${errs.at(i).getMessage()}`);
        v.delete(); parser.delete();
      };
      const checkInstr = (instr, isCond, where) => {
        const type = instr.type.value;
        const md = isCond ? gd.MetadataProvider.getConditionMetadata(platform, type) : gd.MetadataProvider.getActionMetadata(pla
» tform, type);
        if (gd.MetadataProvider.isBadInstructionMetadata(md)) return report(where, `unknown ${isCond ? 'condition' : 'action'} "
» ${type}"`);
        const n = md.getParametersCount();
        if (instr.parameters.length !== n) report(where, `${type}: has ${instr.parameters.length} parameters, needs ${n}`);
        let objectName = '';
        for (let i = 0; i < n; i++) {
          const pm = md.getParameter(i), pt = pm.getType();
          const value = instr.parameters[i] === undefined ? '' : instr.parameters[i];
          const w = `${where} > ${type} #${i} (${pt})`;
          if (pm.isCodeOnly()) continue;
          if (gd.ParameterMetadata.isObject(pt)) {
```

```
            objectName = value;
            if (!getObject(value) && !isGroup(value)) report(w, `unknown object "${value}"`);
            continue;
          }
          if (pt === 'behavior') {
            const o = getObject(objectName);
            const capability = /Capability::/.test(pm.getExtraInfo());
            if (o && !capability && !o.hasBehaviorNamed(value)) report(w, `object "${objectName}" has no behavior "${value}"`);
            continue;
          }
          if (value === '' && pm.isOptional()) continue;
          if (gd.ParameterMetadata.isExpression('number', pt)) validateExpr(w, 'number', value, objectName, pm.getExtraInfo());
          else if (gd.ParameterMetadata.isExpression('string', pt)) validateExpr(w, 'string', value, objectName, pm.getExtraInfo
» ());
          else if (gd.ParameterMetadata.isExpression('variable', pt)) validateExpr(w, pt === 'objectvar' ? 'objectvar' : 'variab
» le', value, objectName, pm.getExtraInfo());
          else if (pt === 'operator' || pt === 'relationalOperator') { if (!OPERATORS.includes(value)) report(w, `bad operator "
» ${value}"`); }
          else if (pt === 'yesorno') { if (!['yes', 'no', ''].includes(value)) report(w, `use "yes"/"no", got "${value}"`); }
          else if (pt === 'trueorfalse') { if (!['True', 'False', 'true', 'false', ''].includes(value)) report(w, `use "True"/"F
» alse", got "${value}"`); }
          else if (pt === 'mouse') { if (!['Left', 'Right', 'Middle', 'Back', 'Forward'].includes(value)) report(w, `bad mouse b
» utton "${value}"`); }
          else if (['soundfile', 'musicfile', 'imageResource', 'fontResource', 'audioResource'].includes(pt)) { if (value && !rm
» .hasResource(value.replace(/^"|"$/g, ''))) report(w, `missing resource "${value}"`); }
        }
      };
      const walk = (events, trail) => events.forEach((e, idx) => {
        const where = `${trail}/${e.name || '#' + (idx + 1)}`;
        (e.whileConditions || []).forEach(c => checkInstr(c, true, where));
        (e.conditions || []).forEach(c => checkInstr(c, true, where));
        (e.actions || []).forEach(a => checkInstr(a, false, where));
        if (e.type === 'BuiltinCommonInstructions::Link' && !(json.externalEvents || []).some(x => x.name === e.target)) report(
» where, `link to unknown external events "${e.target}"`);
        if (e.type === 'BuiltinCommonInstructions::Repeat') validateExpr(where + ' repeat', 'number', e.repeatExpression, '', ''
» );
        if (e.type === 'BuiltinCommonInstructions::ForEach' && !getObject(e.object) && !isGroup(e.object)) report(where, `ForEac
» h on unknown object "${e.object}"`);
        if (e.events) walk(e.events, where);
      });
      walk(layoutJson.events, layoutJson.name);
      const c0 = count(layoutJson.events);
      total += c0;
      if (!layoutJson.isLayout) { console.log(`external events "${layoutJson.name}" (scope: ${layoutJson.layoutName}): ${c0} con
» ditions+actions`); return; }
      // Code generation, as the preview/export does (linked external events are included here).
      const codeGen = new gd.LayoutCodeGenerator(project);
      const includes = new gd.SetString();
      const code = codeGen.generateLayoutCompleteCode(layout, includes, true);
      console.log(`layout "${layoutJson.name}": ${c0} conditions+actions, ${code.length} bytes of generated code`);
    });
    const out = new gd.SerializerElement();
    project.serializeTo(out);
    const rt = gd.Serializer.toJSON(out);
    const rtJson = JSON.parse(rt);
    const rtCount = rtJson.layouts.reduce((n, l) => n + count(l.events), 0) + (rtJson.externalEvents || []).reduce((n, x) => n +
»  count(x.events), 0);
    if (rtCount !== total) report('round trip', `libGD kept ${rtCount} of ${total} conditions+actions (something was dropped)`);
    if (outPath) fs.writeFileSync(outPath, rt);
    console.log(`TOTAL events counted by the Assemble3 limit: ${total} (guest 100 / free 250 / premium 500)`);
    console.log(errors ? `FAILED: ${errors} error(s)` : 'OK: no errors');
  } catch (e) { errors++; console.log('EXCEPTION', e && (e.stack || e.message) || e); }
  process.exit(errors ? 1 : 0);
});
```

tools/export.js - HTML5 export into a folder, exposes window.__game

```
// HTML5 export with libGD + a compiled GDJS runtime (read-only).
// Usage: node export.js <project.json|.a3p> <outDir> [gdjsRoot]
// gdjsRoot = folder containing Runtime/ (in the Assemble3 repo: editor/newIDE/app/resources/GDJS).
// The exported index.html is patched so tests can reach the game as window.__game.
const fs = require('fs');
const path = require('path');
const crypto = require('crypto');
const zlib = require('zlib');
const [, , inPath, outDir, gdjsArg] = process.argv;
const gdjsRoot = gdjsArg || process.env.GDJS_ROOT || '/home/user/Assemble3/editor/newIDE/app/resources/GDJS';
const KEY = Buffer.from('Assemble3/a3p#7f3c9e21b54d80a6');
let text = fs.readFileSync(inPath, 'utf8');
if (text.startsWith('A3P1:')) {
  const raw = Buffer.from(text.slice(5), 'base64');
  for (let i = 0; i < raw.length; i++) raw[i] ^= KEY[i % KEY.length];
  text = zlib.inflateSync(raw).toString('utf8');
}
const isData = s => s.startsWith('data:');
const EXT = { 'image/png': '.png', 'image/jpeg': '.jpg', 'image/gif': '.gif', 'image/webp': '.webp', 'audio/wav': '.wav',
  'audio/mpeg': '.mp3', 'audio/ogg': '.ogg', 'audio/aac': '.aac', 'font/ttf': '.ttf', 'font/otf': '.otf', 'font/woff': '.woff',
  'font/woff2': '.woff2', 'application/json': '.json' };
const extOf = s => EXT[s.slice(5, s.indexOf(';'))] || '.bin';
require(path.join(__dirname, 'libGD.js'))().then(gd => {
  try {
    const project = gd.ProjectHelper.createNewGDJSProject();
    const el = gd.Serializer.fromJSObject(JSON.parse(text));
    project.unserializeFrom(el);
    const impl = {
      mkDir: p => { fs.mkdirSync(p, { recursive: true }); return true; },
      dirExists: p => fs.existsSync(p),
      clearDir: p => { fs.rmSync(p, { recursive: true, force: true }); fs.mkdirSync(p, { recursive: true }); },
      getTempDir: () => path.join(outDir, '..', '.a3tmp'),
      fileNameFrom: p => isData(p) ? 'r' + crypto.createHash('sha1').update(p).digest('hex').slice(0, 12) + extOf(p) : path.base
» name(p),
      dirNameFrom: p => path.dirname(p),
      makeAbsolute: (f, base) => isData(f) ? f : path.resolve(base, f),
      makeRelative: (f, base) => isData(f) ? f : path.relative(base, f),
      isAbsolute: p => isData(p) || p.length === 0 || p.startsWith('/'),
      copyFile: (src, dest) => {
        fs.mkdirSync(path.dirname(dest), { recursive: true });
        if (isData(src)) fs.writeFileSync(dest, Buffer.from(src.slice(src.indexOf(',') + 1), 'base64'));
        else fs.copyFileSync(src, dest);
        return true;
      },
      writeToFile: (f, c) => { fs.mkdirSync(path.dirname(f), { recursive: true }); fs.writeFileSync(f, c); return true; },
      readFile: f => { try { return fs.readFileSync(f, 'utf8'); } catch (e) { return ''; } },
      readDir: (p, ext) => {
        const out = new gd.VectorString();
        if (fs.existsSync(p)) fs.readdirSync(p).forEach(f => { if (!ext || f.toUpperCase().endsWith(ext.toUpperCase())) out.push
» _back(p + '/' + f); });
        return out;
      },
      fileExists: f => { try { return fs.statSync(f).isFile(); } catch (e) { return false; } },
    };
    const fileSystem = Object.assign(new gd.AbstractFileSystemJS(), impl);
    const exporter = new gd.Exporter(fileSystem, gdjsRoot);
    const options = new gd.ExportOptions(project, outDir);
    const ok = exporter.exportWholePixiProject(options);
    if (!ok) { console.log('EXPORT FAILED:', exporter.getLastError()); process.exit(1); }
    const index = path.join(outDir, 'index.html');
    fs.writeFileSync(index, fs.readFileSync(index, 'utf8').replace('var game = new gdjs.RuntimeGame', 'var game = window.__game
» = new gdjs.RuntimeGame'));
    console.log('exported to', outDir);
  } catch (e) { console.log('EXCEPTION', e && (e.stack || e.message) || e); process.exit(1); }
});
```

tools/playtest.js - headless play-test harness

```
// Headless play-test harness for an exported game (see export.js).
// Usage: node playtest.js <exportDir> <scenario.js> [shotsDir]
// The scenario module exports: async function (t) { ... } where t is the toolkit below.
const path = require('path');
const fs = require('fs');
const http = require('http');
let chromium;
try { ({ chromium } = require('playwright')); } catch (e) { ({ chromium } = require('/usr/local/lib/node_modules_global/playwrig
» ht')); }
const [, , exportDir, scenarioPath, shotsArg] = process.argv;
const shots = shotsArg || path.join(path.dirname(path.resolve(scenarioPath)), 'shots');
fs.mkdirSync(shots, { recursive: true });
const MIME = { '.html': 'text/html', '.js': 'text/javascript', '.json': 'application/json', '.png': 'image/png', '.jpg': 'image/
» jpeg',
  '.wav': 'audio/wav', '.mp3': 'audio/mpeg', '.ogg': 'audio/ogg', '.ttf': 'font/ttf', '.otf': 'font/otf', '.webmanifest': 'appli
» cation/manifest+json' };
const sleep = ms => new Promise(r => setTimeout(r, ms));

(async () => {
  // tiny static server (no dependency)
  const server = http.createServer((req, res) => {
    const file = path.join(exportDir, decodeURIComponent(req.url.split('?')[0]).replace(/\/$/, '/index.html'));
    fs.readFile(file, (err, data) => {
      if (err) { res.writeHead(404); return res.end(); }
      res.writeHead(200, { 'Content-Type': MIME[path.extname(file)] || 'application/octet-stream', 'Cache-Control': 'no-store' }
» );
      res.end(data);
    });
  }).listen(0);
  const url = `http://127.0.0.1:${server.address().port}/index.html`;
  const browser = await chromium.launch({ args: ['--autoplay-policy=no-user-gesture-required', '--use-gl=swiftshader', '--enable
» -unsafe-swiftshader'] });
  const page = await browser.newPage({ viewport: { width: 1280, height: 720 } });
  const errors = [];
  page.on('console', m => { if (m.type() === 'error') errors.push('console.error: ' + m.text()); });
  page.on('pageerror', e => errors.push('pageerror: ' + e.message));
  let failures = 0;
  const t = {
    page, sleep,
    // Load (or reload) the game and wait until the first scene runs. Viewport = game size by default.
    async boot(width, height) {
      if (width) await page.setViewportSize({ width, height });
      await page.goto(url);
      await page.waitForFunction(() => window.__game && window.__game.getSceneStack().getCurrentScene(), null, { timeout: 30000
» });
      await sleep(2200); // loading screen minimum duration is 1.5 s
    },
    async reload() { await page.reload(); await page.waitForFunction(() => window.__game && window.__game.getSceneStack().getCur
» rentScene()); await sleep(2200); },
    // Read a scene variable: path like 'Pontos', 'Jogo.Estrelas', 'Jogo.Nivel.0' (number index for arrays)
    variable: p => page.evaluate(p => {
      const parts = p.split('.');
      let v = window.__game.getSceneStack().getCurrentScene().getVariables().get(parts[0]);
      for (const k of parts.slice(1)) v = /^\d+$/.test(k) ? v.getChildAt(+k) : v.getChild(k);
      const t = v.getType();
      return t === 'string' ? v.getAsString() : t === 'boolean' ? v.getAsBoolean() : t === 'number' ? v.getAsNumber() : JSON.par
» se(JSON.stringify(v.toJSObject ? v.toJSObject() : null));
    }, p),
    setVariable: (p, value) => page.evaluate(([p, value]) => {
      const parts = p.split('.');
      let v = window.__game.getSceneStack().getCurrentScene().getVariables().get(parts[0]);
      for (const k of parts.slice(1)) v = /^\d+$/.test(k) ? v.getChildAt(+k) : v.getChild(k);
      typeof value === 'string' ? v.setString(value) : typeof value === 'boolean' ? v.setBoolean(value) : v.setNumber(value);
    }, [p, value]),
    // Global (project) variables: same path syntax
    globalVariable: p => page.evaluate(p => {
      const parts = p.split('.');
      let v = window.__game.getVariables().get(parts[0]);
      for (const k of parts.slice(1)) v = /^\d+$/.test(k) ? v.getChildAt(+k) : v.getChild(k);
      return v.getType() === 'string' ? v.getAsString() : v.getType() === 'boolean' ? v.getAsBoolean() : v.getAsNumber();
```

```
    }, p),
    setGlobalVariable: (p, value) => page.evaluate(([p, value]) => {
      const parts = p.split('.');
      let v = window.__game.getVariables().get(parts[0]);
      for (const k of parts.slice(1)) v = /^\d+$/.test(k) ? v.getChildAt(+k) : v.getChild(k);
      typeof value === 'string' ? v.setString(value) : typeof value === 'boolean' ? v.setBoolean(value) : v.setNumber(value);
    }, [p, value]),
    // Teleport instance #index of an object (x/y = its origin position in scene coordinates)
    setPosition: (name, index, x, y) => page.evaluate(([n, i, x, y]) => {
      const o = window.__game.getSceneStack().getCurrentScene().getObjects(n)[i]; o.setX(x); o.setY(y);
    }, [name, index, x, y]),
    cameraX: (layer = '') => page.evaluate(l => window.__game.getSceneStack().getCurrentScene().getLayer(l).getCameraX(), layer)
» ,
    // Summary of all live instances of an object
    objects: name => page.evaluate(name => window.__game.getSceneStack().getCurrentScene().getObjects(name).map(o => ({
      x: o.getX(), y: o.getY(), cx: o.getCenterXInScene(), cy: o.getCenterYInScene(), w: o.getWidth(), h: o.getHeight(),
      angle: o.getAngle(), visible: !o.isHidden(), layer: o.getLayer(), z: o.getZOrder(),
      animation: o.getAnimationName ? o.getAnimationName() : undefined, opacity: o.getOpacity ? o.getOpacity() : undefined,
      text: o.getText ? o.getText() : undefined,
    })), name),
    // Number value of an object variable on instance #index
    objectVariable: (name, index, varName) => page.evaluate(([n, i, v]) =>
      window.__game.getSceneStack().getCurrentScene().getObjects(n)[i].getVariables().get(v).getAsNumber(), [name, index, varNam
» e]),
    count: name => page.evaluate(name => window.__game.getSceneStack().getCurrentScene().getObjects(name).length, name),
    sceneName: () => page.evaluate(() => window.__game.getSceneStack().getCurrentScene().getName()),
    // Scene coordinates -> page pixels (handles a canvas scaled to the window)
    async toPage(x, y) {
      return page.evaluate(([x, y]) => {
        const g = window.__game, c = g.getRenderer().getCanvas(), r = c.getBoundingClientRect();
        const cam = g.getSceneStack().getCurrentScene().getLayer('');
        const sx = x - (cam.getCameraX() - g.getGameResolutionWidth() / 2), sy = y - (cam.getCameraY() - g.getGameResolutionHeig
» ht() / 2);
        return { x: r.left + sx * r.width / g.getGameResolutionWidth(), y: r.top + sy * r.height / g.getGameResolutionHeight() }
» ;
      }, [x, y]);
    },
    async click(x, y) { const p = await t.toPage(x, y); await page.mouse.click(p.x, p.y); await sleep(60); },
    async clickObject(name, index = 0) { const o = (await t.objects(name))[index]; if (!o) throw new Error('no instance of ' + n
» ame); await t.click(o.cx, o.cy); },
    async hold(x, y, ms) { const p = await t.toPage(x, y); await page.mouse.move(p.x, p.y); await page.mouse.down(); await sleep
» (ms); await page.mouse.up(); },
    async key(key, ms = 80) { await page.keyboard.down(key); await sleep(ms); await page.keyboard.up(key); },
    async shot(name) { await page.screenshot({ path: path.join(shots, name + '.png') }); },
    check(label, condition, details) {
      if (!condition) failures++;
      console.log(`${condition ? 'PASS' : 'FAIL'} ${label}${details !== undefined ? ' -> ' + JSON.stringify(details) : ''}`);
    },
    // Make Math.random return a fixed value for ms milliseconds (to force rare random branches)
    async forceRandom(value, ms) { await page.evaluate(v => { window.__rnd = window.__rnd || Math.random; Math.random = () => v;
»  }, value); await sleep(ms); await page.evaluate(() => { Math.random = window.__rnd; }); },
    clearStorage: () => page.evaluate(() => localStorage.clear()),
  };
  try {
    await require(path.resolve(scenarioPath))(t);
  } catch (e) { failures++; console.log('FAIL scenario threw:', e.stack || e.message); }
  const real = errors.filter(e => !/GL Driver|GPU stall/.test(e));
  t.check('no JavaScript errors in the game', real.length === 0, real.slice(0, 10));
  console.log(failures ? `PLAYTEST FAILED (${failures})` : 'PLAYTEST PASSED', '| screenshots in', shots);
  await browser.close(); server.close();
  process.exit(failures ? 1 : 0);
})();
```

tools/a3p.py - .a3p encoder/decoder CLI

```
#!/usr/bin/env python3
"""Encode/decode Assemble3 project files.
  python3 a3p.py encode project.json game.a3p
  python3 a3p.py decode game.a3p project.json
"""
import os, sys
sys.path.insert(0, os.path.join(os.path.dirname(os.path.abspath(__file__)), '..'))
from a3lib import encode_a3p, decode_a3p

if __name__ == '__main__':
    mode, src, dst = sys.argv[1:4]
    data = open(src, encoding='utf-8').read()
    out = encode_a3p(data) if mode == 'encode' else decode_a3p(data)
    open(dst, 'w', encoding='utf-8').write(out)
    print(mode, src, '->', dst, len(out), 'chars')
```

tools/meta.js - instruction metadata lookup by regex

```
// Usage: node meta.js <regex on type or fullname>
const re = new RegExp(process.argv[2], 'i');
require('./libGD.js')().then(gd => {
  const exts = gd.JsPlatform.get().getAllPlatformExtensions();
  const dumpMap = (kind, map, ext) => {
    const keys = map.keys();
    for (let i = 0; i < keys.size(); i++) {
      const t = keys.at(i);
      const m = map.get(t);
      const full = m.getFullName ? m.getFullName() : '';
      if (!re.test(t) && !re.test(full)) continue;
      const ps = [];
      for (let j = 0; j < m.getParametersCount(); j++) {
        const p = m.getParameter(j);
        ps.push(
          `${j}:${p.getType()}${p.isCodeOnly() ? '(code)' : ''}${p.isOptional() ? '?' : ''}[${p.getExtraInfo()}]"${p.getDescript
» ion()}"${p.getDefaultValue() ? '=' + p.getDefaultValue() : ''}`
        );
      }
      console.log(`${kind} ${ext} :: ${t} -- ${full}\n   ${ps.join('\n   ')}`);
    }
  };
  for (let i = 0; i < exts.size(); i++) {
    const e = exts.at(i);
    const n = e.getName();
    dumpMap('COND', e.getAllConditions(), n);
    dumpMap('ACT', e.getAllActions(), n);
    dumpMap('EXP', e.getAllExpressions(), n);
    dumpMap('SEXP', e.getAllStrExpressions(), n);
    const objs = e.getExtensionObjectsTypes();
    for (let k = 0; k < objs.size(); k++) {
      const o = objs.at(k);
      dumpMap('OCOND', e.getAllConditionsForObject(o), n + '/' + o);
      dumpMap('OACT', e.getAllActionsForObject(o), n + '/' + o);
      dumpMap('OEXP', e.getAllExpressionsForObject(o), n + '/' + o);
      dumpMap('OSEXP', e.getAllStrExpressionsForObject(o), n + '/' + o);
    }
    const objs0 = [''];
    for (const o of objs0) {
      dumpMap('BASEOCOND', e.getAllConditionsForObject(o), n);
      dumpMap('BASEOACT', e.getAllActionsForObject(o), n);
      dumpMap('BASEOEXP', e.getAllExpressionsForObject(o), n);
    }
  }
});
```

Appendix C - Sample games (complete, tested)

All three samples were built, validated (no errors), exported and play-tested with the scenarios below (all PASS). Sample 3 (samples/sapo: art.py draws the pixel art and sounds, build.py builds the game, test.js has 34 checks) is a complete multi-level platformer.

samples/sample_clicker.py

```
"""Sample 1 - "Caça-Estrelas": click the star before time runs out (30 s). Shows: art generation,
resources, sprite/text objects, instances, scene variables, timers, click detection, Else, Repeat,
floating text, game over + restart, save of the best score in storage."""
import math, os, sys
from PIL import Image, ImageDraw

HERE = os.path.dirname(os.path.abspath(__file__))
sys.path.insert(0, os.path.join(HERE, '..'))
from a3lib import *

ASSETS = os.path.join(HERE, 'assets_clicker')
os.makedirs(ASSETS, exist_ok=True)


def star_png(path, size, color):
    S = 4  # draw 4x bigger then downsample = anti-aliasing
    img = Image.new('RGBA', (size * S, size * S), (0, 0, 0, 0))
    c, r1, r2 = size * S / 2, size * S * 0.48, size * S * 0.2
    pts = [(c + (r1 if i % 2 == 0 else r2) * math.cos(math.radians(-90 + i * 36)),
            c + (r1 if i % 2 == 0 else r2) * math.sin(math.radians(-90 + i * 36))) for i in range(10)]
    ImageDraw.Draw(img).polygon(pts, fill=color + (255,), outline=(255, 255, 255, 255), width=S * 2)
    img.resize((size, size), Image.LANCZOS).save(path)


star_png(os.path.join(ASSETS, 'estrela.png'), 96, (255, 200, 40))
star_png(os.path.join(ASSETS, 'brilho.png'), 24, (255, 240, 150))

p = Project('Caça-Estrelas', 960, 540, description='Clique na estrela antes do tempo acabar!')
p.images_from(ASSETS)
p.add_object(simple_sprite('Estrela', 'estrela.png', assets_dir=ASSETS))
p.add_object(simple_sprite('Brilho', 'brilho.png', assets_dir=ASSETS))
p.add_object(text('TextoPontos', 'Pontos: 0', 32, (255, 255, 255), outline=(20, 20, 60)))
p.add_object(text('TextoTempo', '30', 32, (255, 220, 80), align='right', outline=(20, 20, 60)))
p.add_object(text('TextoFim', '', 40, (255, 255, 255), align='center', outline=(20, 20, 60)))

lay = p.layout('Jogo', background=(20, 24, 60), layers=['', 'UI'])
lay.instance('Estrela', 480, 270)
lay.instance('TextoPontos', 20, 16, 'UI')
lay.instance('TextoTempo', 740, 16, 'UI', width=200, height=40)
lay.instance('TextoFim', 0, 200, 'UI', width=960, height=140)
lay.variables = [num('Pontos'), num('Recorde'), num('Tempo', 30), num('Fim', 0), text_var('Salvo')]

lay.events = [
    COMMENT('Caça-Estrelas: clique na estrela o mais rápido que puder!'),
    E([at_start()], [reset_timer('Jogo'), set_text('TextoFim', '""')]),
    E([at_start(), C('GroupExists', q('CacaEstrelas'), q('recorde'))],
      [A('LireFichierExp', q('CacaEstrelas'), q('recorde'), '', 'Recorde')]),

    GROUP('Clique', [
        E([var_is('Fim', '=', '0')] + clicked('Estrela'), [
            set_var('Pontos', '+', '1'),
            set_scale('Estrela', '=', '1.25'),
            A('SetCenter', 'Estrela', '=', 'RandomInRange(80, 880)', '=', 'RandomInRange(100, 480)'),
        ], [
            REPEAT(6, [
                create('Brilho', 'CursorX()', 'CursorY()'),
                A('AddForceXY', 'Brilho', 'RandomInRange(-200, 200)', 'RandomInRange(-200, 200)', '1'),
            ]),
        ]),
        E([], [set_scale('Estrela', '=', 'max(1, Estrela.ScaleX() - 2 * TimeDelta())'),
               set_opacity('Brilho', '-', '400 * TimeDelta()')]),
        E([opacity_is('Brilho', '<=', '0')], [delete('Brilho')]),
    ]),

    GROUP('Tempo e fim de jogo', [
        E([var_is('Fim', '=', '0')], [
            set_var('Tempo', '=', 'max(0, 30 - TimerElapsedTime("Jogo"))'),
            set_text('TextoPontos', '"Pontos: " + ToString(Pontos) + "   Recorde: " + ToString(Recorde)'),
            set_text('TextoTempo', 'ToString(ceil(Tempo))'),
        ]),
```

```
        E([var_is('Fim', '=', '0'), var_is('Tempo', '<=', '0')], [
            set_var('Fim', '=', '1'),
            A('Hide', 'Estrela'),
            set_text('TextoFim', '"FIM DE JOGO!" + NewLine() + "Você fez " + ToString(Pontos) + " pontos"'
                                 ' + NewLine() + "Aperte R para jogar de novo"'),
        ], [
            E([compare('Pontos', '>', 'Recorde')], [
                set_var('Recorde', '=', 'Pontos'),
                A('EcrireFichierExp', q('CacaEstrelas'), q('recorde'), 'Recorde'),
            ]),
        ]),
        E([var_is('Fim', '=', '1'), key_pressed('r')], [A('Scene', '', q('Jogo'), 'yes')]),
    ]),
]

out = sys.argv[1] if len(sys.argv) > 1 else os.path.join(HERE, 'caca_estrelas')
p.save_json(out + '.json')
print('events:', p.event_count(), '->', out + '.json')
```

samples/test_clicker.js

```
// Scenario for sample_clicker.py (Caça-Estrelas)
module.exports = async t => {
  await t.boot(960, 540);
  await t.clearStorage();
  await t.shot('clicker-1-start');
  t.check('starts with 0 points', (await t.variable('Pontos')) === 0);
  for (let i = 0; i < 5; i++) await t.clickObject('Estrela');   // the star moves after each click
  t.check('5 clicks = 5 points', (await t.variable('Pontos')) === 5, await t.variable('Pontos'));
  t.check('sparkles were created', (await t.count('Brilho')) > 0, await t.count('Brilho'));
  await t.shot('clicker-2-playing');
  await t.setVariable('Tempo', 0);                                  // skip the 30 s wait...
  await t.page.evaluate(() => window.__game.getSceneStack().getCurrentScene().getTimeManager().getTimer('Jogo').setTime(31000));
  await t.sleep(300);
  t.check('game over after time is up', (await t.variable('Fim')) === 1);
  t.check('best score saved', (await t.variable('Recorde')) === 5);
  await t.shot('clicker-3-game-over');
  await t.key('r'); await t.sleep(500);
  t.check('R restarts the scene', (await t.variable('Pontos')) === 0 && (await t.variable('Recorde')) === 5);
};
```

samples/sample_platformer.py

```
"""Sample 2 - "Pulo do Gato": a tiny pixel-art platformer. Shows: pixel art (nearest scaling),
behaviors (PlatformerObject + Platform), tiled sprites, animations by name, flipping,
camera following the player, collecting coins, falling = restart."""
import os, sys
from PIL import Image, ImageDraw

HERE = os.path.dirname(os.path.abspath(__file__))
sys.path.insert(0, os.path.join(HERE, '..'))
from a3lib import *

ASSETS = os.path.join(HERE, 'assets_platformer')
os.makedirs(ASSETS, exist_ok=True)
X = 3  # pixel art scale
PAL = {'k': (30, 25, 40), 'o': (240, 150, 50), 'w': (255, 255, 255), 'p': (255, 150, 170), 'y': (250, 210, 60),
       'g': (80, 180, 80), 'G': (50, 130, 60), 'b': (140, 90, 50), 'B': (110, 65, 35)}


def ascii_png(rows, path):
    img = Image.new('RGBA', (len(rows[0]), len(rows)), (0, 0, 0, 0))
    for y, row in enumerate(rows):
        for x, ch in enumerate(row):
            if ch in PAL:
                img.putpixel((x, y), PAL[ch] + (255,))
    img.resize((img.width * X, img.height * X), Image.NEAREST).save(path)


CAT = ['.k......k.', 'kok....kok', 'koookkoook', 'kowoooowok', 'kokoookokk', 'kooopoooko', '.kooooook.',
       '.kooooook', '.kokkkkok.', '.kk....kk.']
CAT_RUN = ['.k......k.', 'kok....kok', 'koookkoook', 'kowoooowok', 'kokoookokk', 'kooopoooko', '.kooooook.',
           '.kooooook', '.kokkkko..', 'kk.....kk.']
ascii_png(CAT, os.path.join(ASSETS, 'gato.png'))
ascii_png(CAT_RUN, os.path.join(ASSETS, 'gato_corre.png'))
ascii_png(['.kkk.', 'kyyyk', 'kywyk', 'kyyyk', '.kkk.'], os.path.join(ASSETS, 'moeda.png'))
ascii_png(['gggggggg', 'gGgggGgg', 'bbbbbbbb', 'bBbbbbBb', 'bbbbBbbb', 'bbBbbbbb', 'bbbbbbBb', 'bBbbbbbb'],
          os.path.join(ASSETS, 'chao.png'))

p = Project('Pulo do Gato', 960, 540, pixel_art=True)
p.images_from(ASSETS)
p.add_object(sprite('Gato', [anim('Parado', ['gato.png'], assets_dir=ASSETS),
                             anim('Correndo', ['gato.png', 'gato_corre.png'], loop=True, seconds_per_frame=0.12,
                                  assets_dir=ASSETS)],
                    behaviors=[platformer_object(maxSpeed=280, jumpSpeed=650)]))
p.add_object(tiled_sprite('Chao', 'chao.png', 24, 24, behaviors=[platform()]))
p.add_object(simple_sprite('Moeda', 'moeda.png', assets_dir=ASSETS))
p.add_object(text('TextoMoedas', 'Moedas: 0', 24, (255, 255, 255), outline=(30, 25, 40), smoothed=False))

lay = p.layout('Fase', background=(120, 190, 255), layers=['', 'UI'])
lay.instance('Gato', 100, 300)
for x, y, w in ((0, 456, 720), (800, 400, 288), (1160, 330, 240), (1480, 456, 720)):
    lay.instance('Chao', x, y, width=w, height=96)
for x, y in ((300, 430), (360, 430), (900, 330), (1250, 220), (1700, 430), (1760, 430)):
    lay.instance('Moeda', x, y)
lay.instance('TextoMoedas', 16, 12, 'UI')
lay.variables = [num('Moedas')]

lay.events = [
    COMMENT('Setas para andar, seta para cima para pular. Pegue todas as moedas!'),
    E([C('PlatformBehavior::PlatformerObjectBehavior::IsMovingEvenALittle', 'Gato', 'PlatformerObject')],
      [A('AnimatableCapability::AnimatableBehavior::SetName', 'Gato', 'Animation', '=', q('Correndo'))]),
    ELSE([A('AnimatableCapability::AnimatableBehavior::SetName', 'Gato', 'Animation', '=', q('Parado'))]),
    E([key_pressed('Left')], [A('FlippableCapability::FlippableBehavior::FlipX', 'Gato', 'Flippable', 'yes')]),
    E([key_pressed('Right')], [A('FlippableCapability::FlippableBehavior::FlipX', 'Gato', 'Flippable', 'no')]),
    E([C('CollisionNP', 'Gato', 'Moeda', '', '', '')], [set_var('Moedas', '+', '1'), delete('Moeda')]),
    E([], [
        A('SetCameraCenterX', '', '=', 'max(480, Gato.X())', '', ''),
        A('SetCameraCenterY', '', '=', '270', '', ''),
        set_text('TextoMoedas', '"Moedas: " + ToString(Moedas) + " / 6"'),
    ]),
    E([C('PosY', 'Gato', '>', '700')], [A('Scene', '', q('Fase'), 'yes')]),
]
```

```
out = sys.argv[1] if len(sys.argv) > 1 else os.path.join(HERE, 'pulo_do_gato')
p.save_json(out + '.json')
print('events:', p.event_count(), '->', out + '.json')
```

samples/test_platformer.js

```
// Scenario for sample_platformer.py (Pulo do Gato)
module.exports = async t => {
  await t.boot(960, 540);
  await t.sleep(800);                                          // let the cat land
  const start = (await t.objects('Gato'))[0];
  await t.shot('platformer-1-start');
  await t.key('ArrowRight', 1200);
  const moved = (await t.objects('Gato'))[0];
  t.check('right arrow moves the cat', moved.x > start.x + 100, { from: start.x, to: moved.x });
  t.check('collected the first coins', (await t.variable('Moedas')) >= 2, await t.variable('Moedas'));
  await t.page.keyboard.down('ArrowRight'); await t.key('ArrowUp', 300);
  await t.sleep(150); await t.shot('platformer-2-jump');
  const air = (await t.objects('Gato'))[0];
  await t.page.keyboard.up('ArrowRight');
  t.check('up arrow jumps (cat above the floor)', air.y < start.y - 20, { y: air.y, floorY: start.y });
  t.check('running animation plays', ['Correndo', 'Parado'].includes(air.animation), air.animation);
  t.check('camera follows the cat', await t.page.evaluate(() => window.__game.getSceneStack().getCurrentScene().getLayer('').get
» CameraX()) >= 480);
};
```

samples/sapo/art.py

```
"""Pixel art + sounds for "Sapo Saltitante" (guide section 11.2 / 11.3)."""
import math, os, random, wave
import numpy as np
from PIL import Image, ImageDraw

OUT = os.path.join(os.path.dirname(os.path.abspath(__file__)), 'assets')
os.makedirs(OUT, exist_ok=True)
X = 3
random.seed(3)
PAL = {'k': (28, 30, 38), 'g': (90, 200, 80), 'G': (50, 140, 60), 'l': (170, 235, 120), 'w': (255, 255, 255),
       'r': (220, 60, 70), 'p': (255, 150, 170), 'y': (250, 210, 60), 'Y': (200, 150, 30), 'o': (255, 150, 40),
       'b': (140, 90, 50), 'B': (100, 62, 35), 'd': (170, 120, 70), 's': (190, 195, 205), 'S': (120, 125, 140),
       'v': (130, 70, 170), 'V': (90, 40, 130), 'c': (110, 230, 255)}


def from_ascii(rows, pal=PAL):
    img = Image.new('RGBA', (max(map(len, rows)), len(rows)), (0, 0, 0, 0))
    for y, row in enumerate(rows):
        for x, ch in enumerate(row):
            if ch in pal:
                img.putpixel((x, y), pal[ch] + (255,))
    return img


def up(img, f=X):
    return img.resize((img.width * f, img.height * f), Image.NEAREST)


def save(img, name):
    up(img).save(os.path.join(OUT, name))


FROG = ['..kkk....kkk..', '.kwwwk..kwwwk.', '.kwkwkkkkwkwk.', '.kgwwggggwwgk.', 'kgggggggggggk',
        'kgkgggggggkgk', 'kggkkkkkkkkggk', 'kgggggggggggggk', '.kllllllllllk.', '.kglllllllgk.',
        'kgk.kgggggk.kgk', 'kk...kkkkk...kk']
FROG_RUN = ['..kkk....kkk..', '.kwwwk..kwwwk.', '.kwkwkkkkwkwk.', '.kgwwggggwwgk.', 'kgggggggggggk',
            'kgkgggggggkgk', 'kggkkkkkkkkggk', 'kgggggggggggggk', '.kllllllllllk.', '.kglllllllgk.',
            '.kgkgggggkgk..', '..kk.kkkk.kk..']
FROG_JUMP = ['..kkk....kkk..', '.kwwwk..kwwwk.', '.kwkwkkkkwkwk.', '.kgwwggggwwgk.', 'kgggggggggggk',
             'kgkgggggggkgk', 'kggggkkkkggggk', 'kgggggggggggggk', '.kllllllllllk.', 'kglllllllllgk',
             'kgk........kgk', 'kk..........kk']
BEETLE = ['...kkkkkk...', '..kvvvvvvk..', '.kvVvvvvVvk.', 'kvvvvvvvvvvk', 'kvVvvvvvVvk', 'kkwkkkkkkwkk',
          '.kkkkkkkkkk.', '.k.k.k.k.k..']
BEETLE2 = ['...kkkkkk...', '..kvvvvvvk..', '.kvVvvvvVvk.', 'kvvvvvvvvvvk', 'kvVvvvvvVvk', 'kkwkkkkkkwkk',
           '.kkkkkkkkkk.', '..k.k.k.k.k.']
BEETLE_FLAT = ['............', '............', '............', '............', '...kkkkkk...', '.kvvVvvVvvk.',
               'kkxkkkkkkxkk', '.kkkkkkkkkk.']
COIN = [['.kkk.', 'kyyYk', 'kywYk', 'kywYk', 'kyyYk', 'kyyYk', '.kkk.'],
        ['.kk.', 'kyYk', 'kwYk', 'kwYk', 'kyYk', 'kyYk', '.kk.'],
        ['.k.', 'kYk', 'kYk', 'kYk', 'kYk', 'kYk', '.k.'],
        ['.kk.', 'kYyk', 'kYwk', 'kYwk', 'kYyk', 'kYyk', '.kk.']]
HEART = ['.kk.kk.', 'krrkrrk', 'krwrrrk', 'krrrrrk', '.krrrk.', '..krk..', '...k...']
FLAG1 = ['kk........', 'kSkkkkkk..', 'kSrrrrrrk.', 'kSrrwwrrrk', 'kSrrrrrrk.', 'kSkkkkkk..', 'kS........',
         'kS........', 'kS........', 'kS........', 'kS........', 'kS........', 'kS........', 'kS........',
         'kS........', 'kS........', 'kS........', 'kS........', 'kS........', 'kk........', 'kkk.......']
FLAG2 = ['kk........', 'kSkkkkk...', 'kSrrrrrkk.', 'kSrrwwrrrk', 'kSrrrrrrrk', 'kSkkkkkkk.', 'kS........'] + FLAG1[7:]
SPIKES = ['...k....', '..ksk...', '..ksk...', '.ksSsk..', '.ksSsk..', 'ksSSSsk.', 'kkkkkkkk']
GROUND = ['lglglglg', 'gggggggg', 'GgGgGgGg'] + ['bdbbbdbb', 'bbbBbbbb', 'bBbbbbdb', 'bbbbBbbb', 'bbdbbbbB', 'bbbbbbbb', 'bBbbdb
» bb'] * 4 + ['bbbbbbbb']
PLANK = ['dddddddd', 'bbbbbbbb', 'kBkkkkBk']
STAR = ['..y..', '.yyy.', 'yywyy', '.yyy.', '..y..']


def moving_platform():
    img = Image.new('RGBA', (32, 7), (0, 0, 0, 0))
    d = ImageDraw.Draw(img)
    d.rectangle([0, 0, 31, 6], fill=PAL['k'])
    d.rectangle([1, 1, 30, 4], fill=PAL['s'])
    d.rectangle([1, 4, 30, 5], fill=PAL['S'])
    for x in range(3, 30, 6):
```

```
        d.point((x, 2), fill=PAL['S'])
    return img


def sky():
    W, H = 320, 180
    img = Image.new('RGB', (W, H))
    px = img.load()
    bands = [(110, 190, 255), (130, 200, 255), (150, 212, 255), (172, 224, 255), (196, 236, 255)]
    for y in range(H):
        b = min(len(bands) - 1, y // 36)
        for x in range(W):
            px[x, y] = bands[b + 1] if (y % 36 >= 33 and b + 1 < len(bands) and (x + y) % 2 == 0) else bands[b]
    d = ImageDraw.Draw(img)
    d.ellipse([250, 18, 280, 48], fill=(255, 240, 150))
    for cx, cy in ((50, 40), (150, 25), (220, 70)):
        for dx, dy, r in ((0, 0, 9), (10, -4, 11), (21, 0, 9)):
            d.ellipse([cx + dx - r, cy + dy - r // 2, cx + dx + r, cy + dy + r // 2], fill=(255, 255, 255))
    up(img).save(os.path.join(OUT, 'ceu.png'))


def hills():
    W, H = 160, 70
    img = Image.new('RGBA', (W, H), (0, 0, 0, 0))
    px = img.load()
    for x in range(W):
        h1 = 40 + 18 * math.sin(x / W * 2 * math.pi) + 6 * math.sin(x / W * 6 * math.pi)
        for y in range(H):
            if y > H - h1:
                px[x, y] = (80, 170, 110, 255) if y > H - h1 + 2 else (60, 140, 90, 255)
    save(img, 'montanhas.png')


RATE = 22050


def wav(name, s):
    s = np.clip(s, -1, 1)
    with wave.open(os.path.join(OUT, name), 'wb') as w:
        w.setnchannels(1); w.setsampwidth(2); w.setframerate(RATE); w.writeframes((s * 30000).astype('<i2').tobytes())


def sweep(freqs, dur, vol=0.25, duty=0.5):
    n = int(RATE * dur)
    f = np.interp(np.arange(n), np.linspace(0, n, len(freqs)), freqs)
    ph = np.cumsum(f) / RATE
    return np.where(ph % 1 < duty, vol, -vol)


def env(n, k):
    return np.exp(-np.arange(n) / RATE * k)


def sounds():
    s = sweep([300, 700], 0.15); wav('pulo.wav', s * env(len(s), 10))
    a = sweep([988], 0.06, duty=0.25); b = sweep([1319], 0.18, duty=0.25)
    wav('moeda.wav', np.concatenate([a, b * env(len(b), 12)]))
    s = sweep([500, 120], 0.15); wav('pisao.wav', s * env(len(s), 12))
    s = sweep([400, 300, 200, 100], 0.4, 0.28); wav('dano.wav', s * env(len(s), 5))
    wav('vitoria.wav', np.concatenate([sweep([f], d, 0.22, 0.25) * env(int(RATE * d), 3 / d)
                                       for f, d in ((523, .1), (659, .1), (784, .1), (1047, .25), (784, .1), (1047, .4))]))


if __name__ == '__main__':
    save(from_ascii(FROG), 'sapo.png'); save(from_ascii(FROG_RUN), 'sapo_corre.png'); save(from_ascii(FROG_JUMP), 'sapo_pulo.png
» ')
    save(from_ascii(BEETLE), 'besouro_1.png'); save(from_ascii(BEETLE2), 'besouro_2.png')
    save(from_ascii(BEETLE_FLAT, dict(PAL, x=PAL['k'])), 'besouro_esmagado.png')
    for i, c in enumerate(COIN):
        save(from_ascii(c), 'moeda_%d.png' % i)
```

```
    save(from_ascii(HEART), 'coracao.png')
    save(from_ascii(FLAG1), 'bandeira_1.png'); save(from_ascii(FLAG2), 'bandeira_2.png')
    save(from_ascii(SPIKES), 'espinhos.png'); save(from_ascii(GROUND), 'chao.png'); save(from_ascii(PLANK), 'tabua.png')
    save(from_ascii(STAR), 'estrela.png'); save(moving_platform(), 'plataforma.png')
    sky(); hills(); sounds()
    for f in sorted(os.listdir(OUT)):
        print(f, Image.open(os.path.join(OUT, f)).size if f.endswith('.png') else '')
```

samples/sapo/build.py

```
"""Sapo Saltitante - 2-level pixel platformer built only with the AI Game Builder Guide + a3kit."""
import os, sys
HERE = os.path.dirname(os.path.abspath(__file__))
sys.path.insert(0, os.path.dirname(HERE)); sys.path.insert(0, os.path.dirname(os.path.dirname(HERE)))  # a3lib.py
from a3lib import *

A_DIR = os.path.join(HERE, 'assets')
FONT = '/home/user/Assemble3/editor/newIDE/app/public/examples/keyboard-practice/assets/Kenney_Mini.ttf'

p = Project('Sapo Saltitante', 960, 540, pixel_art=True, author='Assemble3',
            description='Ajude o sapo a pular pelas fases, pegar moedas e chegar na bandeira!',
            package='com.assemble3.saposaltitante', loading_color=0x6ebeff)
p.images_from(A_DIR)
p.sounds_from(A_DIR)
p.font(FONT)
F = 'Kenney_Mini.ttf'
INK = (28, 30, 38)

# ---------------------------------------------------------------- objects (project-wide)
def a(name, images, loop=False, spf=0.1):
    return anim(name, images, loop=loop, seconds_per_frame=spf, assets_dir=A_DIR)

p.add_object(sprite('Sapo', [a('Parado', ['sapo.png']), a('Correndo', ['sapo.png', 'sapo_corre.png'], True, 0.1),
                             a('Pulando', ['sapo_pulo.png'])],
                    behaviors=[platformer_object(maxSpeed=300, jumpSpeed=720, gravity=1400, maxFallingSpeed=900)]))
p.add_object(sprite('Besouro', [a('Andando', ['besouro_1.png', 'besouro_2.png'], True, 0.15), a('Esmagado', ['besouro_esmagado.p
» ng'])],
                    variables=[num('Dir', -1), num('Vel', 60), num('Min'), num('Max'), num('Morto')]))
p.add_object(sprite('Moeda', [a('', ['moeda_0.png', 'moeda_1.png', 'moeda_2.png', 'moeda_3.png'], True, 0.12)]))
p.add_object(sprite('Bandeira', [a('', ['bandeira_1.png', 'bandeira_2.png'], True, 0.3)]))
p.add_object(simple_sprite('Estrela', 'estrela.png', assets_dir=A_DIR))
p.add_object(simple_sprite('IconeMoeda', 'moeda_0.png', assets_dir=A_DIR))
p.add_object(simple_sprite('PlataformaMovel', 'plataforma.png', assets_dir=A_DIR,
                           variables=[num('Dir', 1), num('Vel', 80), num('Min'), num('Max')], behaviors=[platform()]))
p.add_object(tiled_sprite('Chao', 'chao.png', 24, 96, behaviors=[platform()]))
p.add_object(tiled_sprite('Tabua', 'tabua.png', 24, 9, behaviors=[platform(platform_type='Jumpthru')]))
p.add_object(tiled_sprite('Espinhos', 'espinhos.png', 24, 21))
p.add_object(tiled_sprite('Coracoes', 'coracao.png', 21, 21))
p.add_object(tiled_sprite('Ceu', 'ceu.png', 960, 540))
p.add_object(tiled_sprite('Montanhas', 'montanhas.png', 480, 210))
p.add_object(text('TextoMoedas', 'x 0', 32, (255, 230, 90), font=F, outline=INK, smoothed=False))
p.add_object(text('TextoFase', 'Fase 1', 24, (255, 255, 255), font=F, align='right', outline=INK, smoothed=False))
p.add_object(text('TextoBanner', '', 48, (255, 255, 255), font=F, align='center', outline=INK, outline_thickness=4, smoothed=Fal
» se))
p.add_object(text('TextoTitulo', 'SAPO SALTITANTE', 64, (140, 230, 100), font=F, align='center', outline=INK, outline_thickness=
» 5, smoothed=False))
p.add_object(text('TextoInfo', '', 24, (255, 255, 255), font=F, align='center', outline=INK, smoothed=False))
p.add_object(text('TextoPiscando', 'Aperte ENTER ou clique para começar', 24, (255, 230, 90), font=F, align='center', outline=IN
» K, smoothed=False))

# global variables (survive scene changes)
p.variables = [num('Moedas'), num('Vidas', 3), num('Venceu')]


def backdrop(lay, width):
    lay.instance('Ceu', 0, 0, 'Ceu', width=960, height=540)
    lay.instance('Montanhas', 0, 270, 'Montanhas', width=width + 960, height=210)


# ---------------------------------------------------------------- levels
def level(name, number, width, next_scene, ground, planks, movers, spikes, beetles, coins, flag_x):
    lay = p.layout(name, background=(110, 190, 255), layers=['Ceu', 'Montanhas', '', 'UI'])
    backdrop(lay, width)
    lay.instance('Sapo', 80, 420, z=10)
    for x0, x1 in ground:
        lay.instance('Chao', x0, 468, width=x1 - x0, height=96)
    for x, y, w in planks:
        lay.instance('Tabua', x, y, width=w, height=9)
    for x, y, lo, hi in movers:
        lay.instance('PlataformaMovel', x, y, variables=[num('Min', lo), num('Max', hi)])
```

```
    for x, w in spikes:
        lay.instance('Espinhos', x, 447, width=w, height=21)
    for x, lo, hi in beetles:
        lay.instance('Besouro', x, 456, z=5, variables=[num('Min', lo), num('Max', hi)])
    for x, y in coins:
        lay.instance('Moeda', x, y)
    lay.instance('Bandeira', flag_x, 437)
    lay.instance('Coracoes', 16, 14, 'UI', width=63, height=21)
    lay.instance('IconeMoeda', 30, 64, 'UI')
    lay.instance('TextoMoedas', 48, 48, 'UI')
    lay.instance('TextoFase', 744, 14, 'UI', width=200, height=30)
    lay.instance('TextoBanner', 0, 150, 'UI', width=960, height=120)
    lay.variables = [num('Fase', number), text_var('Proxima', next_scene), num('LarguraFase', width), num('InicioX'),
                     num('InicioY'), num('Invencivel'), num('Completa')]
    lay.events = [COMMENT('Fase %d: a lógica do jogo fica na folha de eventos externa "Logica", incluída aqui.' % number),
                  LINK('Logica')]
    return lay


row = lambda x0, n, y, step=48: [(x0 + i * step, y) for i in range(n)]
level('Fase1', 1, 2880, 'Fase2',
      ground=[(0, 900), (1020, 1700), (1820, 2880)],
      planks=[(480, 360, 144), (1240, 340, 168), (2040, 350, 144)],
      movers=[(1660, 420, 1640, 1880)],
      spikes=[(1430, 72)],
      beetles=[(720, 600, 860), (1150, 1100, 1380), (2300, 2150, 2550)],
      coins=row(300, 3, 430) + row(500, 3, 320) + row(1265, 3, 300) + row(1700, 3, 370) + row(2060, 3, 310) + row(2600, 3, 430),
      flag_x=2790)
level('Fase2', 2, 3360, 'Fim',
      ground=[(0, 700), (860, 1400), (1700, 2200), (2560, 3360)],
      planks=[(380, 350, 120), (1050, 330, 144), (1500, 300, 120), (2280, 330, 144)],
      movers=[(1440, 430, 1410, 1660), (2240, 420, 2210, 2500)],
      spikes=[(560, 72), (1200, 48), (1900, 96), (2800, 72)],
      beetles=[(400, 250, 520), (1000, 880, 1180), (2000, 1720, 2150), (2700, 2600, 2780), (3050, 2900, 3200)],
      coins=row(390, 3, 310, 40) + row(1060, 3, 290) + row(1505, 3, 260, 40) + row(1500, 3, 380) + row(2290, 3, 290) + row(3000,
»  4, 430),
      flag_x=3270)

# ------------------------------------------------------------------ shared logic (external events)
hurt = [set_var('Vidas', '-', '1'), set_var('Invencivel', '=', '2'), play_sound('dano.wav', 70)]
logic = [
    COMMENT('Lógica das fases (usada pela Fase1 e pela Fase2). Setas para andar, seta para cima para pular. '
            'Pule em cima dos besouros, desvie dos espinhos e chegue na bandeira!'),
    GROUP('Início da fase', [
        E([at_start()], [
            set_var('InicioX', '=', 'Sapo.X()'), set_var('InicioY', '=', 'Sapo.Y()'),
            set_text('TextoBanner', '"Fase " + ToString(Fase)'), set_opacity('TextoBanner', '=', '255'),
        ]),
    ]),
    GROUP('Animações do sapo', [
        E([C('PlatformBehavior::IsJumping', 'Sapo', 'PlatformerObject')],
          [A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Pulando'))]),
        ELSE(conditions=[C('PlatformBehavior::IsFalling', 'Sapo', 'PlatformerObject')],
             actions=[A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Pulando'))]),
        ELSE(conditions=[C('PlatformBehavior::PlatformerObjectBehavior::IsMovingEvenALittle', 'Sapo', 'PlatformerObject')],
             actions=[A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Correndo'))]),
        ELSE([A('AnimatableCapability::AnimatableBehavior::SetName', 'Sapo', 'Animation', '=', q('Parado'))]),
        E([key_pressed('Left')], [A('FlippableCapability::FlippableBehavior::FlipX', 'Sapo', 'Flippable', 'yes')]),
        E([key_pressed('Right')], [A('FlippableCapability::FlippableBehavior::FlipX', 'Sapo', 'Flippable', 'no')]),
        E([C('PlatformBehavior::IsJumping', 'Sapo', 'PlatformerObject'), C('BuiltinCommonInstructions::Once')],
          [play_sound('pulo.wav', 50)]),
    ]),
    GROUP('Câmera', [
        COMMENT('A câmera segue o sapo sem sair da fase; as montanhas andam mais devagar (efeito de profundidade).'),
        E([], [
            A('SetCameraCenterX', '', '=', 'clamp(Sapo.X(), 480, LarguraFase - 480)', '', ''),
            A('SetCameraCenterY', '', '=', '270', '', ''),
            A('SetCameraCenterX', '', '=', '480 + (CameraCenterX() - 480) * 0.4', q('Montanhas'), ''),
        ]),
    ]),
```

```
    GROUP('Moedas', [
        E([C('CollisionNP', 'Sapo', 'Moeda', '', '', '')], [
            set_var('Moedas', '+', '1'), play_sound('moeda.wav', 60),
            create('Estrela', 'Moeda.X()', 'Moeda.Y()'), delete('Moeda'),
        ]),
        E([], [A('SetXY', 'Estrela', '+', '0', '-', '60 * TimeDelta()'), set_opacity('Estrela', '-', '400 * TimeDelta()')]),
        E([opacity_is('Estrela', '<=', '0')], [delete('Estrela')]),
    ]),
    GROUP('Besouros e plataformas móveis', [
        COMMENT('Cada besouro e cada plataforma anda entre os seus Min e Max (variáveis de instância).'),
        E([C('NumberObjectVariable', 'Besouro', 'Morto', '=', '0')],
          [A('SetX', 'Besouro', '+', 'Besouro.Dir * Besouro.Vel * TimeDelta()')]),
        E([C('PosX', 'Besouro', '<', 'Besouro.Min')], [A('SetNumberObjectVariable', 'Besouro', 'Dir', '=', '1'),
                                                       A('FlippableCapability::FlippableBehavior::FlipX', 'Besouro', 'Flippable',
» 'yes')]),
        E([C('PosX', 'Besouro', '>', 'Besouro.Max')], [A('SetNumberObjectVariable', 'Besouro', 'Dir', '=', '-1'),
                                                       A('FlippableCapability::FlippableBehavior::FlipX', 'Besouro', 'Flippable',
» 'no')]),
        E([], [A('SetX', 'PlataformaMovel', '+', 'PlataformaMovel.Dir * PlataformaMovel.Vel * TimeDelta()')]),
        E([C('PosX', 'PlataformaMovel', '<', 'PlataformaMovel.Min')], [A('SetNumberObjectVariable', 'PlataformaMovel', 'Dir', '=
» ', '1')]),
        E([C('PosX', 'PlataformaMovel', '>', 'PlataformaMovel.Max')], [A('SetNumberObjectVariable', 'PlataformaMovel', 'Dir', '=
» ', '-1')]),
        COMMENT('Caiu em cima do besouro: ele é esmagado e o sapo quica.'),
        E([C('CollisionNP', 'Sapo', 'Besouro', '', '', ''), C('NumberObjectVariable', 'Besouro', 'Morto', '=', '0'),
           C('PlatformBehavior::IsFalling', 'Sapo', 'PlatformerObject')], [
            A('SetNumberObjectVariable', 'Besouro', 'Morto', '=', '1'),
            A('AnimatableCapability::AnimatableBehavior::SetName', 'Besouro', 'Animation', '=', q('Esmagado')),
            play_sound('pisao.wav', 70),
            A('PlatformBehavior::SetCanJump', 'Sapo', 'PlatformerObject'),
            A('PlatformBehavior::SimulateJumpKey', 'Sapo', 'PlatformerObject'),
        ]),
        COMMENT('Encostou num besouro vivo (sem cair em cima): perde uma vida.'),
        E([C('CollisionNP', 'Sapo', 'Besouro', '', '', ''), C('NumberObjectVariable', 'Besouro', 'Morto', '=', '0'),
           var_is('Invencivel', '<=', '0')], hurt),
        E([C('NumberObjectVariable', 'Besouro', 'Morto', '=', '1')], [set_opacity('Besouro', '-', '200 * TimeDelta()')]),
        E([C('NumberObjectVariable', 'Besouro', 'Morto', '=', '1'), opacity_is('Besouro', '<=', '0')], [delete('Besouro')]),
    ]),
    GROUP('Perigos e vidas', [
        E([C('CollisionNP', 'Sapo', 'Espinhos', '', '', ''), var_is('Invencivel', '<=', '0')], hurt),
        COMMENT('Caiu num buraco: perde uma vida e volta para o começo da fase.'),
        E([C('PosY', 'Sapo', '>', '640')], hurt + [A('SetXY', 'Sapo', '=', 'InicioX', '=', 'InicioY')]),
        COMMENT('Depois de levar dano, o sapo pisca e fica invencível por 2 segundos.'),
        E([var_is('Invencivel', '>', '0')], [set_var('Invencivel', '-', 'TimeDelta()'),
                                             set_opacity('Sapo', '=', '255 - 180 * mod(round(TimeFromStart() * 12), 2)')]),
        ELSE([set_opacity('Sapo', '=', '255')]),
        E([var_is('Vidas', '<=', '0')], [set_var('Venceu', '=', '0'), A('Scene', '', q('Fim'), 'yes')]),
    ]),
    GROUP('Bandeira', [
        E([C('CollisionNP', 'Sapo', 'Bandeira', '', '', ''), var_is('Completa', '=', '0')], [
            set_var('Completa', '=', '1'), set_var('Venceu', '=', '1'), play_sound('vitoria.wav', 80),
            set_text('TextoBanner', '"Fase completa!"'), set_opacity('TextoBanner', '=', '255'),
            A('PlatformBehavior::IgnoreDefaultControls', 'Sapo', 'PlatformerObject', 'yes'),
            reset_timer('Fim'),
        ]),
        E([var_is('Completa', '=', '1'), timer_over('Fim', 2.5)], [A('Scene', '', 'Proxima', 'yes')]),
    ]),
    GROUP('Placar', [
        E([], [
            A('ResizableCapability::ResizableBehavior::SetWidth', 'Coracoes', 'Resizable', '=', 'max(0, Vidas) * 21'),
            set_text('TextoMoedas', '"x " + ToString(Moedas)'),
            set_text('TextoFase', '"Fase " + ToString(Fase)'),
        ]),
        E([var_is('Completa', '=', '0')], [set_opacity('TextoBanner', '-', '90 * TimeDelta()')]),
    ]),
]
p.external_events('Logica', logic, 'Fase1')

# ------------------------------------------------------------------ title + end screens
tit = p.layout('Titulo', background=(110, 190, 255), layers=['Ceu', 'Montanhas', '', 'UI'])
```

```
backdrop(tit, 960)
tit.instance('Chao', 0, 468, width=960, height=96)
tit.instance('Sapo', 480, 400)
tit.instance('TextoTitulo', 0, 110, 'UI', width=960, height=80)
tit.instance('TextoInfo', 0, 220, 'UI', width=960, height=80)
tit.instance('TextoPiscando', 0, 330, 'UI', width=960, height=40)
tit.events = [
    COMMENT('Tela de título. Você já pode testar os pulos do sapo aqui!'),
    E([at_start()], [set_var('Moedas', '=', '0'), set_var('Vidas', '=', '3'), set_var('Venceu', '=', '0'),
                     set_text('TextoInfo', '"Setas: andar     Seta para cima: pular" + NewLine() + "Pule em cima dos besouros!"'
» )]),
    E([], [set_opacity('TextoPiscando', '=', '155 + 100 * sin(TimeFromStart() * 4)')]),
    E([C('KeyFromTextReleased', '', q('Return'))], [A('Scene', '', q('Fase1'), 'yes')]),
    E([C('MouseButtonFromTextReleased', '', q('Left'))], [A('Scene', '', q('Fase1'), 'yes')]),
]
p.start_layout = 'Titulo'   # the game starts on the title screen

fim = p.layout('Fim', background=(40, 40, 70), layers=['Ceu', 'UI'])
fim.instance('Ceu', 0, 0, 'Ceu', width=960, height=540)
fim.instance('TextoTitulo', 0, 120, 'UI', width=960, height=80)
fim.instance('TextoInfo', 0, 230, 'UI', width=960, height=100)
fim.instance('TextoPiscando', 0, 380, 'UI', width=960, height=40)
fim.variables = [num('Recorde')]
fim.events = [
    COMMENT('Fim do jogo: mostra se venceu, as moedas e o recorde (salvo no navegador).'),
    E([at_start(), C('GroupExists', q('SapoSaltitante'), q('recorde'))],
      [A('LireFichierExp', q('SapoSaltitante'), q('recorde'), '', 'Recorde')]),
    E([at_start()], [set_text('TextoPiscando', '"Aperte ENTER ou clique para voltar ao início"')], [
        E([var_is('Venceu', '=', '1')], [set_text('TextoTitulo', '"VOCÊ VENCEU!"')]),
        ELSE([set_text('TextoTitulo', '"FIM DE JOGO"')]),
        E([compare('Moedas', '>', 'Recorde')], [set_var('Recorde', '=', 'Moedas'),
                                                A('EcrireFichierExp', q('SapoSaltitante'), q('recorde'), 'Recorde')]),
        E([], [set_text('TextoInfo', '"Moedas: " + ToString(Moedas) + NewLine() + "Recorde: " + ToString(Recorde)')]),
    ]),
    E([], [set_opacity('TextoPiscando', '=', '155 + 100 * sin(TimeFromStart() * 4)')]),
    E([C('KeyFromTextReleased', '', q('Return'))], [A('Scene', '', q('Titulo'), 'yes')]),
    E([C('MouseButtonFromTextReleased', '', q('Left'))], [A('Scene', '', q('Titulo'), 'yes')]),
]

out = os.path.join(HERE, 'sapo.json')
p.save_json(out)
print('events:', p.event_count(), '->', out)
```

samples/sapo/test.js

```
// Play-test scenario for Sapo Saltitante (guide section 15).
module.exports = async t => {
  const place = t.setPosition;
  const sapo = async () => (await t.objects('Sapo'))[0];
  const camX = t.cameraX;
  const globalVar = t.globalVariable;

  await t.boot(960, 540);
  await t.clearStorage();
  t.check('starts on the title screen', (await t.sceneName()) === 'Titulo', await t.sceneName());
  t.check('3 lives at start', (await globalVar('Vidas')) === 3);
  await t.shot('1-title');

  await t.key('Enter'); await t.sleep(900);
  t.check('ENTER starts Fase1', (await t.sceneName()) === 'Fase1', await t.sceneName());
  await t.shot('2-fase1-start');
  t.check('camera clamped at level start', Math.abs((await camX()) - 480) < 1, await camX());

  // walk right through the first coin row
  await t.key('ArrowRight', 1100); await t.sleep(200);
  const m = await globalVar('Moedas');
  t.check('walking collects coins (global Moedas)', m >= 3, m);
  t.check('running/idle animation names are used', ['Correndo', 'Parado', 'Pulando'].includes((await sapo()).animation), (await
» sapo()).animation);

  // jump
  const y0 = (await sapo()).y;
  await t.key('ArrowUp', 200); await t.sleep(100);
  const air = await sapo();
  t.check('up arrow jumps', air.y < y0 - 40, { y0, y: air.y });
  t.check('jump animation', air.animation === 'Pulando', air.animation);
  await t.sleep(900);

  // stomp the first beetle
  const b = (await t.objects('Besouro'))[0];
  await place('Sapo', 0, b.x, b.y - 140); await t.sleep(600);
  t.check('stomping kills the beetle', (await t.objectVariable('Besouro', 0, 'Morto')) === 1);
  t.check('stomping does not cost a life', (await globalVar('Vidas')) === 3, await globalVar('Vidas'));
  await t.shot('3-stomp');
  await t.sleep(1500);

  // beetles patrol
  const bs1 = await t.objects('Besouro'); await t.sleep(700); const bs2 = await t.objects('Besouro');
  t.check('live beetles move', bs1.some((o, i) => bs2[i] && Math.abs(o.x - bs2[i].x) > 10), bs1.map(o => Math.round(o.x)));

  // touching a live beetle from the side hurts
  const live = (await t.objects('Besouro'))[0];
  await place('Sapo', 0, live.x - 30, live.y - 6); await t.sleep(250);
  t.check('side contact costs a life', (await globalVar('Vidas')) === 2, await globalVar('Vidas'));
  t.check('hearts HUD shows 2 lives (tiled sprite width 42)', (await t.objects('Coracoes'))[0].w === 42, (await t.objects('Corac
» oes'))[0].w);
  t.check('invincibility after damage', (await t.variable('Invencivel')) > 0, await t.variable('Invencivel'));
  await t.shot('4-hurt-blink');
  await t.sleep(2300);

  // pit: fall between 900 and 1020
  await place('Sapo', 0, 960, 300); await t.sleep(1400);
  const s = await sapo();
  t.check('falling in a pit costs a life', (await globalVar('Vidas')) === 1, await globalVar('Vidas'));
  t.check('respawn at level start', s.x < 200, Math.round(s.x));
  await t.setVariable('Invencivel', 0);
  await t.setGlobalVariable('Vidas', 3);

  // jump-thru plank: land on top of the first plank (y=360)
  await place('Sapo', 0, 550, 250); await t.sleep(900);
  t.check('stands on the jump-thru plank', Math.abs((await sapo()).y - 342) < 4, (await sapo()).y);

  // moving platform carries the frog
  const mp0 = (await t.objects('PlataformaMovel'))[0];
  await place('Sapo', 0, mp0.cx, mp0.y - 40); await t.sleep(300);
```

```
  const ride0 = await sapo(); await t.sleep(1000); const ride1 = await sapo();
  const mp1 = (await t.objects('PlataformaMovel'))[0];
  t.check('moving platform moves', Math.abs(mp1.x - mp0.x) > 40, [mp0.x, mp1.x]);
  t.check('frog rides the platform (moves with it, does not fall)', Math.abs(ride1.x - ride0.x) > 40 && ride1.y < 440, { from: r
» ide0, to: [ride1.x, ride1.y] });
  t.check('camera follows the frog', Math.abs((await camX()) - Math.min(Math.max(ride1.x, 480), 2400)) < 30, await camX());
  t.check('parallax layer scrolls slower', Math.abs((await camX('Montanhas')) - (480 + ((await camX()) - 480) * 0.4)) < 2);
  await t.shot('5-moving-platform');

  // spikes hurt
  const vidas = await globalVar('Vidas');
  await place('Sapo', 0, 1466, 400); await t.sleep(600);
  t.check('spikes cost a life', (await globalVar('Vidas')) === vidas - 1, await globalVar('Vidas'));
  await t.setVariable('Invencivel', 0);

  // flag -> Fase2, coins kept
  const coinsBefore = await globalVar('Moedas');
  const flag = (await t.objects('Bandeira'))[0];
  await place('Sapo', 0, flag.x - 10, flag.y); await t.sleep(400);
  t.check('reaching the flag completes the level', (await t.variable('Completa')) === 1);
  await t.shot('6-level-complete');
  await t.sleep(2700);
  t.check('goes to Fase2', (await t.sceneName()) === 'Fase2', await t.sceneName());
  t.check('coins kept between levels', (await globalVar('Moedas')) === coinsBefore, await globalVar('Moedas'));
  t.check('HUD says Fase 2', (await t.objects('TextoFase'))[0].text === 'Fase 2');
  await t.sleep(400);
  await t.shot('7-fase2');

  // Fase2 flag -> Fim (victory), best score saved
  const flag2 = (await t.objects('Bandeira'))[0];
  await place('Sapo', 0, flag2.x - 10, flag2.y); await t.sleep(3300);
  t.check('Fase2 flag goes to Fim', (await t.sceneName()) === 'Fim', await t.sceneName());
  t.check('victory message', (await t.objects('TextoTitulo'))[0].text === 'VOCÊ VENCEU!', (await t.objects('TextoTitulo'))[0].te
» xt);
  t.check('best score saved', (await t.variable('Recorde')) === coinsBefore, await t.variable('Recorde'));
  await t.shot('8-victory');

  // back to title, then game over path
  await t.key('Enter', 300); await t.sleep(700);       // held for 300 ms: must not skip the title
  t.check('ENTER returns to the title (and does not skip it)', (await t.sceneName()) === 'Titulo', await t.sceneName());
  t.check('lives reset on the title', (await globalVar('Vidas')) === 3);
  await t.key('Enter'); await t.sleep(900);
  await t.setGlobalVariable('Vidas', 1);
  await place('Sapo', 0, 960, 300); await t.sleep(1500);
  t.check('losing the last life ends the game', (await t.sceneName()) === 'Fim', await t.sceneName());
  t.check('game over message', (await t.objects('TextoTitulo'))[0].text === 'FIM DE JOGO', (await t.objects('TextoTitulo'))[0].t
» ext);
  await t.shot('9-game-over');
};
```

Running the samples

```
cd work/samples
python3 sample_clicker.py                          # -> caca_estrelas.json (+ assets_clicker/)
node ../tools/validate.js caca_estrelas.json caca_estrelas.rt.json
node ../tools/export.js caca_estrelas.rt.json export_caca_estrelas
node ../tools/playtest.js export_caca_estrelas test_clicker.js
python3 ../tools/a3p.py encode caca_estrelas.rt.json CacaEstrelas.a3p
```

Appendix D - Keyboard key names

For `KeyPressed`/`KeyReleased` (unquoted) and `KeyFromTextPressed`/`KeyFromTextReleased` (quoted string):

```
a-z (lowercase letters)  Num0-Num9  Numpad0-Numpad9  LShift RShift LControl RControl LAlt RAlt LSyst
» em RSystem SemiColon Comma Period Quote Slash BackSlash Equal Dash Menu LBracket RBracket Tilde Sp
» ace Back Tab Delete Insert Escape PageUp PageDown End Home Return NumpadPageUp NumpadPageDown Nump
» adEnd NumpadHome NumpadReturn Add Subtract Multiply Divide NumpadAdd NumpadSubtract NumpadMultiply
»  NumpadDivide Left Up Right Down NumpadLeft NumpadUp NumpadRight NumpadDown F1-F12 Pause
```

Mouse buttons: `Left`, `Right`, `Middle` (plus `Back`, `Forward`). In Playwright scenarios use DOM key names: `ArrowLeft`, `ArrowUp`, `Space`, `Enter`, `r`.

Appendix E - Instruction reference (generated from libGD)

Generated from libGD 5.6.282 metadata for the built-in (C++) extensions - the set the toolchain supports. For each instruction: the **type string** to put in `type.value`, its kind, its English name, and its parameters in order: `index:type[extra]? "description" default`. `?` = optional. Code-only parameters are shown as `type=""` - write `""` for them. **[legacy/hidden]** = still works but not offered by the editor picker; prefer a visible equivalent (Section 7.3). Object-scope instructions take the object name as parameter 0; behavior-scope ones take the object (0) and the behavior name (1). Expressions are called as `Name(args)` (free) or `Object.Name(args)` (object) or `Object.Behavior::Name(args)` (behavior), without the object/behavior parameters.

Events and control flow - BuiltinCommonInstructions

GDevelop comes with a set of events and conditions that allow to express the game logic and rules.

Free instructions

BuiltinCommonInstructions::And condition And

BuiltinCommonInstructions::CompareNumbers condition Compare two numbers
`0:expression "First expression" | 1:relationalOperator[number] "Sign of the test" | 2:expression "Second expression"`

BuiltinCommonInstructions::CompareStrings condition Compare two strings
`0:string "First string expression" | 1:relationalOperator[string] "Sign of the test" | 2:string "Second string expression"`

BuiltinCommonInstructions::Not condition Not

BuiltinCommonInstructions::Once condition Trigger once while true

BuiltinCommonInstructions::Or condition Or

BuiltinCommonInstructions::Always condition Always **[legacy/hidden]**
`0:conditionInverted=""`

Egal condition Compare two numbers **[legacy/hidden]**
`0:expression "First expression" | 1:relationalOperator[number] "Sign of the test" | 2:expression "Second expression"`

StrEqual condition Compare two strings **[legacy/hidden]**
`0:string "First string expression" | 1:relationalOperator[string] "Sign of the test" | 2:string "Second string expression"`

Toujours condition Always **[legacy/hidden]**
`0:conditionInverted=""`

Scene - BuiltinScene

Actions/conditions to change the current scene (or pause it and launch another one, or go back to the previous one), check if a scene or the game has just started/resumed, preload assets of a scene, get the current scene name or loading progress, quit the game, set background color, or disable input when focus is lost.

Free instructions

AreObjectAssetsLoaded condition Object preloaded
`0:currentScene="" | 1:string "Object name" | 2:sceneName? "Object scene"`

AreSceneAssetsLoaded condition Scene preloaded
`0:currentScene="" | 1:sceneName "Scene name"`

DoesSceneExist condition Does scene exist
`0:currentScene="" | 1:sceneName "Name of the scene to check"`

HasGameJustResumed condition Game has just resumed
`0:currentScene=""`

SceneJustBegins condition At the beginning of the scene
`0:currentScene=""`

SceneJustResumed condition Scene just resumed
`0:currentScene=""`

SceneLoadingProgress condition Scene loading progress
`0:currentScene="" | 1:sceneName "Scene name" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

DepartScene condition At the beginning of the scene **[legacy/hidden]**
`0:currentScene=""`

LoadObjectAssets action Preload object
`0:currentScene="" | 1:string "Object name" | 2:sceneName? "Object scene"`

PopScene action Stop and go back to previous scene
`0:currentScene=""`

PrioritizeLoadingOfScene action Preload scene
`0:currentScene="" | 1:sceneName "Name of the new scene"`

PushScene action Pause and start a new scene
`0:currentScene="" | 1:sceneName "Name of the new scene"`

Quit action Quit the game
`0:currentScene=""`

Scene action Change the scene
`0:currentScene="" | 1:sceneName "Name of the new scene" | 2:yesorno "Stop any other paused scenes?" default true`

SceneBackground action Background color
`0:currentScene="" | 1:color "Color"`

UnloadObjectAssets action Unload object
`0:currentScene="" | 1:string "Object name" | 2:sceneName? "Object scene"`

SceneLoadingProgress number expression Scene loading progress
`0:currentScene="" | 1:sceneName "Scene name"`

CurrentSceneName string expression Current scene name
`0:currentScene=""`

Objects - BuiltinObject

Common features that can be used for all objects in GDevelop.

Free instructions

CollisionNP condition Collision
`0:objectList "Object" | 1:objectList "Object" | 2:conditionInverted="" | 3:currentScene="" | 4:yesorno? "Ignore objects that are touching each other on their edges, but are not overlapping (default: no)" default no`

Distance condition Distance between two objects
`0:objectList "Object" | 1:objectList "Object 2" | 2:expression "Distance" | 3:conditionInverted=""`

IsCursorOnObject condition The cursor/touch is on an object
`0:objectList "Object" | 1:currentScene="" | 2:yesorno? "Accurate test (yes by default)" default yes | 3:conditionInverted=""`

IsTotalForceAngleTowardObject condition An object is moving toward another (using forces)
`0:objectList "Object" | 1:objectList "Object 2" | 2:expression "Tolerance, in degrees" | 3:conditionInverted=""`

IsTurnedTowardObject condition An object is turned toward another
`0:objectList "Name of the object" | 1:objectList "Name of the second object" | 2:expression "Angle of tolerance, in degrees (0: minimum tolerance)" | 3:conditionInverted=""`

PickAllInstances condition Pick all object instances
`0:objectsContext="" | 1:objectList "Object"`

PickNearest condition Pick nearest object
`0:objectList "Object" | 1:expression "X position" | 2:expression "Y position" | 3:conditionInverted=""`

PickRandomInstance condition Pick a random object
`0:objectsContext="" | 1:objectList "Object"`

PickedInstancesCount condition Number of object instances currently picked
`0:objectListOrEmptyWithoutPicking "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

Raycast condition Raycast
`0:objectList "Objects to test against the ray" | 1:expression "Ray source X position" | 2:expression "Ray source Y position" | 3:expression "Ray angle (in degrees)" | 4:expression "Ray maximum distance (in pixels)" | 5:scenevar "Result X position scene variable" | 6:scenevar "Result Y position scene variable" | 7:conditionInverted=""`

RaycastToPosition condition Raycast to position
`0:objectList "Objects to test against the ray" | 1:expression "Ray source X position" | 2:expression "Ray source Y position" | 3:expression "Ray target X position" | 4:expression "Ray target Y position" | 5:scenevar "Result X position scene variable" | 6:scenevar "Result Y position scene variable" | 7:conditionInverted=""`

SceneInstancesCount condition Number of object instances on the scene
`0:objectsContext="" | 1:objectListOrEmptyWithoutPicking "Object" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

AjoutHasard condition Pick a random object **[legacy/hidden]**
`0:objectsContext="" | 1:objectList "Object"`

AjoutObjConcern condition Pick all object instances **[legacy/hidden]**
`0:objectsContext="" | 1:objectList "Object"`

EstTourne condition An object is turned toward another **[legacy/hidden]**
`0:objectList "Name of the object" | 1:objectList "Name of the second object" | 2:expression "Angle of tolerance, in degrees (0: minimum tolerance)" | 3:conditionInverted=""`

NbObjet condition Number of objects **[legacy/hidden]**
`0:objectList "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

SeDirige condition An object is moving toward another (using forces) **[legacy/hidden]**
`0:objectList "Object" | 1:objectList "Object 2" | 2:expression "Tolerance, in degrees" | 3:conditionInverted=""`

SourisSurObjet condition The cursor/touch is on an object **[legacy/hidden]**
`0:objectList "Object" | 1:currentScene="" | 2:yesorno? "Accurate test (yes by default)" default yes | 3:conditionInverted=""`

Create action Create an object
`0:objectsContext="" | 1:objectListOrEmptyIfJustDeclared "Object to create" | 2:expression "X position" | 3:expression "Y position" | 4:layer? "Layer" default ""`

CreateByName action Create an object from its name
`0:objectsContext="" | 1:objectListOrEmptyIfJustDeclared "Group of potential objects" | 2:string "Name of the object to create" | 3:expression "X position" | 4:expression "Y position" | 5:layer? "Layer" default ""`

MoveObjects action Apply movement to all objects
`0:currentScene=""`

PickAllInstances action Pick all object instances
`0:objectsContext="" | 1:objectList "Object"`

PickNearest action Pick nearest object
`0:objectList "Object" | 1:expression "X position" | 2:expression "Y position"`

PickRandomInstance action Pick a random object
`0:objectsContext="" | 1:objectList "Object"`

AjoutHasard action Pick a random object **[legacy/hidden]**
`0:objectsContext="" | 1:objectList "Object"`

AjoutObjConcern action Pick all object instances **[legacy/hidden]**
`0:objectsContext="" | 1:objectList "Object"`

PickedInstancesCount number expression Number of object instances currently picked
`0:objectListOrEmptyWithoutPicking "Object"`

SceneInstancesCount number expression Number of object instances on the scene
`0:objectsContext="" | 1:objectListOrEmptyWithoutPicking "Object"`

Count number expression Number of objects **[legacy/hidden]**
`0:objectList "Object"`

Object: (any object)

Angle condition Angle
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle to compare to (in degrees)"`

BehaviorActivated condition Behavior activated
`0:object "Object" | 1:behavior "Behavior"`

BooleanObjectVariable condition Object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:trueorfalse "Check if the value is" default true | 3:yesorno=""`

BoundingBoxBottom condition Bounding box bottom position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxCenterX condition Bounding box center X position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxCenterY condition Bounding box center Y position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxLeft condition Bounding box left position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxRight condition Bounding box right position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxTop condition Bounding box top position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

CenterX condition Center X position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

CenterY condition Center Y position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

CollisionPoint condition Point inside object
`0:object "Object" | 1:expression "X position of the point" | 2:expression "Y position of the point"`

CompareObjectTimer condition Value of an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name" | 2:relationalOperator[time] "Sign of the test" | 3:expression "Time in seconds"`

HasNoForces condition Object is stopped (no forces applied on it)
`0:object "Object"`

IsTotalForceAngleAround condition Angle of movement (using forces)
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Tolerance, in degrees"`

Layer condition Current layer
`0:object "Object" | 1:layer "Layer"`

NumberObjectVariable condition Object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

ObjectTimerPaused condition Object timer paused
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

ObjectVariableAsBoolean condition Boolean variable
`0:object "Object" | 1:objectvar "Variable" | 2:trueorfalse "Check if the value is" default true`

ObjectVariableChildCount condition Number of children
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

ObjectVariableChildExists condition Child existence
`0:object "Object" | 1:objectvar "Structure variable" | 2:string "Name of the child"`

PosX condition X position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

PosY condition Y position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

Speed condition Speed (from forces)
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

StringObjectVariable condition Object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[string] "Sign of the test" | 3:string "Value to compare"`

VarObjet condition Number variable
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

VarObjetTxt condition Text variable
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[string] "Sign of the test" | 3:string "Value to compare"`

Visible condition Visibility
`0:object "Object"`

ZOrder condition Z-order
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

AngleOfDisplacement condition Angle of movement (using forces) **[legacy/hidden]**
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Tolerance, in degrees"`

Arret condition Object is stopped (no forces applied on it) **[legacy/hidden]**
`0:object "Object"`

Invisible condition Invisibility of an object **[legacy/hidden]**
`0:object "Object"`

IsEffectEnabled condition Effect is enabled **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name"`

ObjectTimer condition Value of an object timer **[legacy/hidden]**
`0:object "Object" | 1:identifier[objectTimer] "Timer's name" | 2:expression "Time in seconds"`

Plan condition Z-order **[legacy/hidden]**
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

VarObjetDef condition Variable defined **[legacy/hidden]**
`0:object "Object" | 1:string "Variable"`

Vitesse condition Speed (from forces) **[legacy/hidden]**
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ActivateBehavior action De/activate a behavior
`0:object "Object" | 1:behavior "Behavior" | 2:yesorno "Activate?"`

AddForceAL action Add a force (angle)
`0:object "Object" | 1:expression "Angle" | 2:expression "Speed (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

AddForceTowardObject action Add a force to move toward an object
`0:object "Object" | 1:objectPtr "Target Object" | 2:expression "Speed (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

AddForceTowardPosition action Add a force to move toward a position
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Speed (in pixels per second)" | 4:forceMultiplier? "Force multiplier" default 0`

AddForceXY action Add a force
`0:object "Object" | 1:expression "Speed on X axis (in pixels per second)" | 2:expression "Speed on Y axis (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

ChangeLayer action Layer
`0:object "Object" | 1:layer "Move it to this layer" default ""`

ClearForces action Stop the object
`0:object "Object"`

Delete action Delete the object
`0:object "Object" | 1:currentScene=""`

Hide action Hide
`0:object "Object"`

ModVarObjet action Change number variable
`0:object "Object" | 1:objectvar "Variable" | 2:operator[number] "Modification's sign" | 3:number "Value"`

ModVarObjetTxt action Change text variable
`0:object "Object" | 1:objectvar "Variable" | 2:operator[string] "Modification's sign" | 3:string "Value"`

ObjectVariableClearChildren action Clear children
`0:object "Object" | 1:objectvar "Array or structure variable"`

ObjectVariablePush2 action Add existing variable
`0:object "Object" | 1:objectvar "Array variable" | 2:variable "Scene variable with the content to add"`

ObjectVariablePushBool action Add boolean variable
`0:object "Object" | 1:objectvar "Array variable" | 2:trueorfalse "Boolean to add"`

ObjectVariablePushNumber action Add number variable
`0:object "Object" | 1:objectvar "Array variable" | 2:expression "Number to add"`

ObjectVariablePushString action Add text variable
`0:object "Object" | 1:objectvar "Array variable" | 2:string "Text to add"`

ObjectVariableRemoveAt action Remove variable by index
`0:object "Object" | 1:objectvar "Array variable" | 2:expression "Index to remove"`

ObjectVariableRemoveChild action Remove a child
`0:object "Object" | 1:objectvar "Structure variable" | 2:string "Child's name"`

PauseObjectTimer action Pause an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

PushBooleanToObjectVariable action Add value to object array variable
`0:object "Object" | 1:objectvar "Array variable" | 2:trueorfalse "Boolean to add"`

PushNumberToObjectVariable action Add value to object array variable
`0:object "Object" | 1:objectvar "Array variable" | 2:expression "Number to add"`

PushStringToObjectVariable action Add value to object array variable
`0:object "Object" | 1:objectvar "Array variable" | 2:string "Text to add"`

PutAroundObject action Put the object around another
`0:object "Object" | 1:objectPtr ""Center" Object" | 2:expression "Distance" | 3:expression "Angle, in degrees"`

PutAroundPosition action Put around a position
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Distance" | 4:expression "Angle, in degrees"`

RemoveObjectTimer action Delete an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

ResetObjectTimer action Start (or reset) an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

Rotate action Rotate
`0:object "Object" | 1:expression "Angular speed (in degrees per second)" | 2:currentScene=""`

RotateTowardAngle action Rotate toward angle
`0:object "Object" | 1:expression "Angle to rotate towards (in degrees)" | 2:expression "Angular speed (in degrees per second)" | 3:currentScene=""`

RotateTowardObject action Rotate toward another object
`0:object "Object" | 1:objectPtr "Target object" | 2:expression "Angular speed (in degrees per second)" | 3:currentScene=""`

RotateTowardPosition action Rotate toward position
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Angular speed (in degrees per second)" | 4:currentScene=""`

SeparateFromObjects action Separate objects
`0:object "Object" | 1:objectList "Objects (won't move)" | 2:yesorno? "Ignore objects that are touching each other on their edges, but are not overlapping (default: no)" default no`

SetAngle action Angle
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

SetBooleanObjectVariable action Change object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:operator[boolean] "Value" | 3:yesorno=""`

SetCenter action Center position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:expression "X position" | 3:operator[number] "Modification's sign" | 4:expression "Y position"`

SetCenterX action Center X position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetCenterY action Center Y position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetIncludedInParentCollisionMask action Include in parent collision mask
`0:object "Object" | 1:yesorno "Include in parent collision mask"`

SetNumberObjectVariable action Change object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:operator[number] "Modification's sign" | 3:number "Value"`

SetObjectVariableAsBoolean action Change boolean variable
`0:object "Object" | 1:objectvar "Variable" | 2:trueorfalse "New Value:"`

SetStringObjectVariable action Change object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:operator[string] "Modification's sign" | 3:string "Value"`

SetX action X position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetXY action Position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:expression "X position" | 3:operator[number] "Modification's sign" | 4:expression "Y position"`

SetY action Y position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetZOrder action Z order
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

Show action Show
`0:object "Object" | 1:inlineCode=""`

ToggleObjectVariableAsBoolean action Toggle boolean variable
`0:object "Object" | 1:objectvar "Variable"`

UnPauseObjectTimer action Unpause an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

AddForceVers action Add a force to move toward an object **[legacy/hidden]**
`0:object "Object" | 1:objectPtr "Target Object" | 2:expression "Speed (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

AddForceVersPos action Add a force to move toward a position **[legacy/hidden]**
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Speed (in pixels per second)" | 4:forceMultiplier? "Force multiplier" default 0`

Arreter action Stop the object **[legacy/hidden]**
`0:object "Object"`

Cache action Hide **[legacy/hidden]**
`0:object "Object"`

ChangePlan action Z order **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

Ecarter action Move an object away from another **[legacy/hidden]**
`0:object "Object" | 1:objectList "Object 2 (won't move)"`

EnableEffect action Enable an object effect **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:yesorno "Enable?"`

MettreAutour action Put the object around another **[legacy/hidden]**
`0:object "Object" | 1:objectPtr ""Center" Object" | 2:expression "Distance" | 3:expression "Angle, in degrees"`

MettreAutourPos action Put around a position **[legacy/hidden]**
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Distance" | 4:expression "Angle, in degrees"`

MettreX action X position **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

MettreXY action Position **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:expression "X position" | 3:operator[number] "Modification's sign" | 4:expression "Y position"`

MettreY action Y position **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

Montre action Show **[legacy/hidden]**
`0:object "Object" | 1:inlineCode=""`

ObjectVariablePush action Add existing variable **[legacy/hidden]**
`0:object "Object" | 1:objectvar "Array variable" | 2:scenevar "Scene variable with the content to add"`

Rebondir action Move an object away from another **[legacy/hidden]**
`0:object "Object" | 1:objectList "Object 2 (won't move)"`

SetEffectBooleanParameter action Effect property (enable or disable) **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:objectEffectParameterName "Property name" | 3:yesorno "Enable this property"`

SetEffectDoubleParameter action Effect property (number) **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:objectEffectParameterName "Property name" | 3:expression "New value"`

SetEffectStringParameter action Effect property (string) **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:objectEffectParameterName "Property name" | 3:string "New value"`

Angle number expression Angle
`0:object "Object"`

AngleToObject number expression Angle between two objects
`0:object "Object" | 1:objectPtr "Object"`

AngleToPosition number expression Angle between an object and a position
`0:object "Object" | 1:expression "Target X position" | 2:expression "Target Y position"`

ArrayVariableFirstNumber number expression First number child
`0:object "Object" | 1:objectvar "Array variable"`

ArrayVariableLastNumber number expression Last number child
`0:object "Object" | 1:objectvar "Array variable"`

BoundingBoxBottom number expression Bounding box bottom position
`0:object "Object"`

BoundingBoxCenterX number expression Bounding box center X position
`0:object "Object"`

BoundingBoxCenterY number expression Bounding box center Y position
`0:object "Object"`

BoundingBoxLeft number expression Bounding box left position
`0:object "Object"`

BoundingBoxRight number expression Bounding box right position
`0:object "Object"`

BoundingBoxTop number expression Bounding box top position
`0:object "Object"`

CenterX number expression Center X position
`0:object "Object"`

CenterY number expression Center Y position
`0:object "Object"`

Distance number expression Distance between two objects
`0:object "Object" | 1:objectPtr "Object"`

DistanceToPosition number expression Distance between an object and a position
`0:object "Object" | 1:expression "Target X position" | 2:expression "Target Y position"`

ForceAngle number expression Angle of the sum of forces
`0:object "Object"`

ForceLength number expression Length of the sum of forces
`0:object "Object"`

ForceX number expression X coordinate of the sum of forces
`0:object "Object"`

ForceY number expression Y coordinate of the sum of forces
`0:object "Object"`

Height number expression Height
`0:object "Object"`

ObjectTimerElapsedTime number expression Object timer value
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

SqDistance number expression Square distance between two objects
`0:object "Object" | 1:objectPtr "Object"`

SqDistanceToPosition number expression Square distance between an object and a position
`0:object "Object" | 1:expression "Target X position" | 2:expression "Target Y position"`

Variable number expression Number variable
`0:object "Object" | 1:objectvar "Variable"`

VariableChildCount number expression Number of children
`0:object "Object" | 1:objectvar "Array or structure variable"`

Width number expression Width
`0:object "Object"`

X number expression X position
`0:object "Object"`

XFromAngleAndDistance number expression X position from angle and distance
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Distance"`

Y number expression Y position
`0:object "Object"`

YFromAngleAndDistance number expression Y position from angle and distance
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Distance"`

ZOrder number expression Z-order
`0:object "Object"`

Hauteur number expression Height **[legacy/hidden]**
`0:object "Object"`

Largeur number expression Width **[legacy/hidden]**
`0:object "Object"`

Longueur number expression Length of the sum of forces **[legacy/hidden]**
`0:object "Object"`

Plan number expression Z-order **[legacy/hidden]**
`0:object "Object"`

ArrayVariableFirstString string expression First text child
`0:object "Object" | 1:objectvar "Array variable"`

ArrayVariableLastString string expression Last text child
`0:object "Object" | 1:objectvar "Array variable"`

Layer string expression Object layer
`0:object "Object"`

ObjectName string expression Object name
`0:object "Object"`

VariableString string expression Text variable
`0:object "Object" | 1:objectvar "Variable"`

Object: (any object)

Angle condition Angle
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle to compare to (in degrees)"`

BehaviorActivated condition Behavior activated
`0:object "Object" | 1:behavior "Behavior"`

BooleanObjectVariable condition Object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:trueorfalse "Check if the value is" default true | 3:yesorno=""`

BoundingBoxBottom condition Bounding box bottom position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxCenterX condition Bounding box center X position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxCenterY condition Bounding box center Y position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxLeft condition Bounding box left position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxRight condition Bounding box right position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

BoundingBoxTop condition Bounding box top position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

CenterX condition Center X position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

CenterY condition Center Y position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

CollisionPoint condition Point inside object
`0:object "Object" | 1:expression "X position of the point" | 2:expression "Y position of the point"`

CompareObjectTimer condition Value of an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name" | 2:relationalOperator[time] "Sign of the test" | 3:expression "Time in seconds"`

HasNoForces condition Object is stopped (no forces applied on it)
`0:object "Object"`

IsTotalForceAngleAround condition Angle of movement (using forces)
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Tolerance, in degrees"`

Layer condition Current layer
`0:object "Object" | 1:layer "Layer"`

NumberObjectVariable condition Object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

ObjectTimerPaused condition Object timer paused
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

ObjectVariableAsBoolean condition Boolean variable
`0:object "Object" | 1:objectvar "Variable" | 2:trueorfalse "Check if the value is" default true`

ObjectVariableChildCount condition Number of children
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

ObjectVariableChildExists condition Child existence
`0:object "Object" | 1:objectvar "Structure variable" | 2:string "Name of the child"`

PosX condition X position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

PosY condition Y position
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

Speed condition Speed (from forces)
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

StringObjectVariable condition Object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[string] "Sign of the test" | 3:string "Value to compare"`

VarObjet condition Number variable
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

VarObjetTxt condition Text variable
`0:object "Object" | 1:objectvar "Variable" | 2:relationalOperator[string] "Sign of the test" | 3:string "Value to compare"`

Visible condition Visibility
`0:object "Object"`

ZOrder condition Z-order
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

AngleOfDisplacement condition Angle of movement (using forces) **[legacy/hidden]**
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Tolerance, in degrees"`

Arret condition Object is stopped (no forces applied on it) **[legacy/hidden]**
`0:object "Object"`

Invisible condition Invisibility of an object **[legacy/hidden]**
`0:object "Object"`

IsEffectEnabled condition Effect is enabled **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name"`

ObjectTimer condition Value of an object timer **[legacy/hidden]**
`0:object "Object" | 1:identifier[objectTimer] "Timer's name" | 2:expression "Time in seconds"`

Plan condition Z-order **[legacy/hidden]**
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

VarObjetDef condition Variable defined **[legacy/hidden]**
`0:object "Object" | 1:string "Variable"`

Vitesse condition Speed (from forces) **[legacy/hidden]**
`0:object "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ActivateBehavior action De/activate a behavior
`0:object "Object" | 1:behavior "Behavior" | 2:yesorno "Activate?"`

AddForceAL action Add a force (angle)
`0:object "Object" | 1:expression "Angle" | 2:expression "Speed (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

AddForceTowardObject action Add a force to move toward an object
`0:object "Object" | 1:objectPtr "Target Object" | 2:expression "Speed (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

AddForceTowardPosition action Add a force to move toward a position
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Speed (in pixels per second)" | 4:forceMultiplier? "Force multiplier" default 0`

AddForceXY action Add a force
`0:object "Object" | 1:expression "Speed on X axis (in pixels per second)" | 2:expression "Speed on Y axis (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

ChangeLayer action Layer
`0:object "Object" | 1:layer "Move it to this layer" default ""`

ClearForces action Stop the object
`0:object "Object"`

Delete action Delete the object
`0:object "Object" | 1:currentScene=""`

Hide action Hide
`0:object "Object"`

ModVarObjet action Change number variable
`0:object "Object" | 1:objectvar "Variable" | 2:operator[number] "Modification's sign" | 3:number "Value"`

ModVarObjetTxt action Change text variable
`0:object "Object" | 1:objectvar "Variable" | 2:operator[string] "Modification's sign" | 3:string "Value"`

ObjectVariableClearChildren action Clear children
`0:object "Object" | 1:objectvar "Array or structure variable"`

ObjectVariablePush2 action Add existing variable
`0:object "Object" | 1:objectvar "Array variable" | 2:variable "Scene variable with the content to add"`

ObjectVariablePushBool action Add boolean variable
`0:object "Object" | 1:objectvar "Array variable" | 2:trueorfalse "Boolean to add"`

ObjectVariablePushNumber action Add number variable
`0:object "Object" | 1:objectvar "Array variable" | 2:expression "Number to add"`

ObjectVariablePushString action Add text variable
`0:object "Object" | 1:objectvar "Array variable" | 2:string "Text to add"`

ObjectVariableRemoveAt action Remove variable by index
`0:object "Object" | 1:objectvar "Array variable" | 2:expression "Index to remove"`

ObjectVariableRemoveChild action Remove a child
`0:object "Object" | 1:objectvar "Structure variable" | 2:string "Child's name"`

PauseObjectTimer action Pause an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

PushBooleanToObjectVariable action Add value to object array variable
`0:object "Object" | 1:objectvar "Array variable" | 2:trueorfalse "Boolean to add"`

PushNumberToObjectVariable action Add value to object array variable
`0:object "Object" | 1:objectvar "Array variable" | 2:expression "Number to add"`

PushStringToObjectVariable action Add value to object array variable
`0:object "Object" | 1:objectvar "Array variable" | 2:string "Text to add"`

PutAroundObject action Put the object around another
`0:object "Object" | 1:objectPtr ""Center" Object" | 2:expression "Distance" | 3:expression "Angle, in degrees"`

PutAroundPosition action Put around a position
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Distance" | 4:expression "Angle, in degrees"`

RemoveObjectTimer action Delete an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

ResetObjectTimer action Start (or reset) an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

Rotate action Rotate
`0:object "Object" | 1:expression "Angular speed (in degrees per second)" | 2:currentScene=""`

RotateTowardAngle action Rotate toward angle
`0:object "Object" | 1:expression "Angle to rotate towards (in degrees)" | 2:expression "Angular speed (in degrees per second)" | 3:currentScene=""`

RotateTowardObject action Rotate toward another object
`0:object "Object" | 1:objectPtr "Target object" | 2:expression "Angular speed (in degrees per second)" | 3:currentScene=""`

RotateTowardPosition action Rotate toward position
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Angular speed (in degrees per second)" | 4:currentScene=""`

SeparateFromObjects action Separate objects
`0:object "Object" | 1:objectList "Objects (won't move)" | 2:yesorno? "Ignore objects that are touching each other on their edges, but are not overlapping (default: no)" default no`

SetAngle action Angle
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

SetBooleanObjectVariable action Change object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:operator[boolean] "Value" | 3:yesorno=""`

SetCenter action Center position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:expression "X position" | 3:operator[number] "Modification's sign" | 4:expression "Y position"`

SetCenterX action Center X position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetCenterY action Center Y position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetIncludedInParentCollisionMask action Include in parent collision mask
`0:object "Object" | 1:yesorno "Include in parent collision mask"`

SetNumberObjectVariable action Change object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:operator[number] "Modification's sign" | 3:number "Value"`

SetObjectVariableAsBoolean action Change boolean variable
`0:object "Object" | 1:objectvar "Variable" | 2:trueorfalse "New Value:"`

SetStringObjectVariable action Change object variable value
`0:object "Object" | 1:objectvar "Variable" | 2:operator[string] "Modification's sign" | 3:string "Value"`

SetX action X position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetXY action Position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:expression "X position" | 3:operator[number] "Modification's sign" | 4:expression "Y position"`

SetY action Y position
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetZOrder action Z order
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

Show action Show
`0:object "Object" | 1:inlineCode=""`

ToggleObjectVariableAsBoolean action Toggle boolean variable
`0:object "Object" | 1:objectvar "Variable"`

UnPauseObjectTimer action Unpause an object timer
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

AddForceVers action Add a force to move toward an object **[legacy/hidden]**
`0:object "Object" | 1:objectPtr "Target Object" | 2:expression "Speed (in pixels per second)" | 3:forceMultiplier? "Force multiplier" default 0`

AddForceVersPos action Add a force to move toward a position **[legacy/hidden]**
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Speed (in pixels per second)" | 4:forceMultiplier? "Force multiplier" default 0`

Arreter action Stop the object **[legacy/hidden]**
`0:object "Object"`

Cache action Hide **[legacy/hidden]**
`0:object "Object"`

ChangePlan action Z order **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

Ecarter action Move an object away from another **[legacy/hidden]**
`0:object "Object" | 1:objectList "Object 2 (won't move)"`

EnableEffect action Enable an object effect **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:yesorno "Enable?"`

MettreAutour action Put the object around another **[legacy/hidden]**
`0:object "Object" | 1:objectPtr ""Center" Object" | 2:expression "Distance" | 3:expression "Angle, in degrees"`

MettreAutourPos action Put around a position **[legacy/hidden]**
`0:object "Object" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Distance" | 4:expression "Angle, in degrees"`

MettreX action X position **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

MettreXY action Position **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:expression "X position" | 3:operator[number] "Modification's sign" | 4:expression "Y position"`

MettreY action Y position **[legacy/hidden]**
`0:object "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

Montre action Show **[legacy/hidden]**
`0:object "Object" | 1:inlineCode=""`

ObjectVariablePush action Add existing variable **[legacy/hidden]**
`0:object "Object" | 1:objectvar "Array variable" | 2:scenevar "Scene variable with the content to add"`

Rebondir action Move an object away from another **[legacy/hidden]**
`0:object "Object" | 1:objectList "Object 2 (won't move)"`

SetEffectBooleanParameter action Effect property (enable or disable) **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:objectEffectParameterName "Property name" | 3:yesorno "Enable this property"`

SetEffectDoubleParameter action Effect property (number) **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:objectEffectParameterName "Property name" | 3:expression "New value"`

SetEffectStringParameter action Effect property (string) **[legacy/hidden]**
`0:object "Object" | 1:objectEffectName "Effect name" | 2:objectEffectParameterName "Property name" | 3:string "New value"`

Angle number expression Angle
`0:object "Object"`

AngleToObject number expression Angle between two objects
`0:object "Object" | 1:objectPtr "Object"`

AngleToPosition number expression Angle between an object and a position
`0:object "Object" | 1:expression "Target X position" | 2:expression "Target Y position"`

ArrayVariableFirstNumber number expression First number child
`0:object "Object" | 1:objectvar "Array variable"`

ArrayVariableLastNumber number expression Last number child
`0:object "Object" | 1:objectvar "Array variable"`

BoundingBoxBottom number expression Bounding box bottom position
`0:object "Object"`

BoundingBoxCenterX number expression Bounding box center X position
`0:object "Object"`

BoundingBoxCenterY number expression Bounding box center Y position
`0:object "Object"`

BoundingBoxLeft number expression Bounding box left position
`0:object "Object"`

BoundingBoxRight number expression Bounding box right position
`0:object "Object"`

BoundingBoxTop number expression Bounding box top position
`0:object "Object"`

CenterX number expression Center X position
`0:object "Object"`

CenterY number expression Center Y position
`0:object "Object"`

Distance number expression Distance between two objects
`0:object "Object" | 1:objectPtr "Object"`

DistanceToPosition number expression Distance between an object and a position
`0:object "Object" | 1:expression "Target X position" | 2:expression "Target Y position"`

ForceAngle number expression Angle of the sum of forces
`0:object "Object"`

ForceLength number expression Length of the sum of forces
`0:object "Object"`

ForceX number expression X coordinate of the sum of forces
`0:object "Object"`

ForceY number expression Y coordinate of the sum of forces
`0:object "Object"`

Height number expression Height
`0:object "Object"`

ObjectTimerElapsedTime number expression Object timer value
`0:object "Object" | 1:identifier[objectTimer] "Timer's name"`

SqDistance number expression Square distance between two objects
`0:object "Object" | 1:objectPtr "Object"`

SqDistanceToPosition number expression Square distance between an object and a position
`0:object "Object" | 1:expression "Target X position" | 2:expression "Target Y position"`

Variable number expression Number variable
`0:object "Object" | 1:objectvar "Variable"`

VariableChildCount number expression Number of children
`0:object "Object" | 1:objectvar "Array or structure variable"`

Width number expression Width
`0:object "Object"`

X number expression X position
`0:object "Object"`

XFromAngleAndDistance number expression X position from angle and distance
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Distance"`

Y number expression Y position
`0:object "Object"`

YFromAngleAndDistance number expression Y position from angle and distance
`0:object "Object" | 1:expression "Angle, in degrees" | 2:expression "Distance"`

ZOrder number expression Z-order
`0:object "Object"`

Hauteur number expression Height **[legacy/hidden]**
`0:object "Object"`

Largeur number expression Width **[legacy/hidden]**
`0:object "Object"`

Longueur number expression Length of the sum of forces **[legacy/hidden]**
`0:object "Object"`

Plan number expression Z-order **[legacy/hidden]**
`0:object "Object"`

ArrayVariableFirstString string expression First text child
`0:object "Object" | 1:objectvar "Array variable"`

ArrayVariableLastString string expression Last text child
`0:object "Object" | 1:objectvar "Array variable"`

Layer string expression Object layer
`0:object "Object"`

ObjectName string expression Object name
`0:object "Object"`

VariableString string expression Text variable
`0:object "Object" | 1:objectvar "Variable"`

Sprite - Sprite

Sprite are animated objects which can be used for most elements of a 2D game.

Free instructions

Collision condition Collision (Pixel perfect) **[legacy/hidden]**
`0:objectList[Sprite] "Object 1" | 1:objectList[Sprite] "Object 2" | 2:conditionInverted="" | 3:currentScene="" | 4:yesorno "Consider objects touching each other, but not overlapping, as in collision (default: no)"`

Object: Sprite

BlendMode condition Blend mode
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

Sprite condition Current frame
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

AnimStopped condition Animation paused **[legacy/hidden]**
`0:object[Sprite] "Object"`

Animation condition Current animation **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

AnimationEnded condition Animation finished **[legacy/hidden]**
`0:object[Sprite] "Object"`

AnimationEnded2 condition Animation finished **[legacy/hidden]**
`0:object[Sprite] "Object"`

AnimationName condition Current animation name **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:objectAnimationName "Animation name"`

Direction condition Current direction **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

FlippedX condition Horizontally flipped **[legacy/hidden]**
`0:object[Sprite] "Object"`

FlippedY condition Vertically flipped **[legacy/hidden]**
`0:object[Sprite] "Object"`

Height condition Height **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

Opacity condition Opacity **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Opacity to compare to (0-255)"`

ScaleHeight condition Scale on Y axis **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Scale (1 by default)"`

ScaleWidth condition Scale on X axis **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Scale (1 by default)"`

Width condition Width **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ChangeBlendMode action Blend mode
`0:object[Sprite] "Object" | 1:expression "Mode (0: Normal, 1: Add, 2: Multiply, 3: Screen)"`

ChangeColor action Tint color
`0:object[Sprite] "Object" | 1:color "Tint"`

ChangeSprite action Current frame
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ChangeAnimation action Change the animation **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ChangeAnimationSpeedScale action Animation speed scale **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Speed scale"`

ChangeDirection action Change the direction **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ChangeHeight action Height **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ChangeScale action Scale **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

ChangeScaleHeight action Scale on Y axis **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

ChangeScaleWidth action Scale on X axis **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

ChangeWidth action Width **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

FlipX action Flip the object horizontally **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:yesorno "Activate flipping"`

FlipY action Flip the object vertically **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:yesorno "Activate flipping"`

Opacity action Sprite opacity **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Opacity (0-255)"`

PauseAnimation action Pause the animation **[legacy/hidden]**
`0:object[Sprite] "Object"`

PlayAnimation action Play the animation **[legacy/hidden]**
`0:object[Sprite] "Object"`

SetAnimationName action Change the animation (by name) **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:objectAnimationName "Animation name"`

SetSize action Size **[legacy/hidden]**
`0:object "Object" | 1:expression "Width" | 2:expression "Height"`

TourneVers action Rotate an object toward another **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:objectPtr "Rotate toward this object" | 2:currentScene=""`

TourneVersPos action Rotate an object toward a position **[legacy/hidden]**
`0:object[Sprite] "Object to be rotated" | 1:expression "X position" | 2:expression "Y position" | 3:expression "Angular speed (degrees per second)" default 0 | 4:currentScene=""`

AnimationFrameCount number expression Number of frames
`0:object[Sprite] "Object"`

PointX number expression X position of a point
`0:object[Sprite] "Object" | 1:objectPointName "Name of the point"`

PointY number expression Y position of a point
`0:object[Sprite] "Object" | 1:objectPointName "Name of the point"`

Sprite number expression Image
`0:object[Sprite] "Object"`

Anim number expression Animation **[legacy/hidden]**
`0:object[Sprite] "Object"`

Animation number expression Animation **[legacy/hidden]**
`0:object[Sprite] "Object"`

AnimationSpeedScale number expression Animation speed scale **[legacy/hidden]**
`0:object[Sprite] "Object"`

Direc number expression Direction **[legacy/hidden]**
`0:object[Sprite] "Object"`

Direction number expression Direction **[legacy/hidden]**
`0:object[Sprite] "Object"`

Opacity number expression Opacity **[legacy/hidden]**
`0:object[Sprite] "Object"`

ScaleX number expression Scale of the width of an object **[legacy/hidden]**
`0:object[Sprite] "Object"`

ScaleY number expression Scale of the height of an object **[legacy/hidden]**
`0:object[Sprite] "Object"`

X number expression X position of a point **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:objectPointName? "Name of the point"`

Y number expression Y position of a point **[legacy/hidden]**
`0:object[Sprite] "Object" | 1:objectPointName? "Name of the point"`

AnimationName string expression Animation name **[legacy/hidden]**
`0:object[Sprite] "Object"`

Text object - TextObject

An object that can be used to display any text on the screen: remaining life counter, some indicators, menu buttons, dialogues...

Object: TextObject::Text

TextObject::IsBold condition Bold
`0:object[TextObject::Text] "Object"`

TextObject::IsItalic condition Italic
`0:object[TextObject::Text] "Object"`

TextObject::IsWrapping condition Wrapping
`0:object[TextObject::Text] "Object"`

TextObject::Padding condition Padding
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TextObject::Text::FontSize condition Font size
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TextObject::Text::IsOutlineEnabled condition Outline enabled
`0:object[TextObject::Text] "Object"`

TextObject::Text::IsShadowEnabled condition Shadow enabled
`0:object[TextObject::Text] "Object"`

TextObject::Text::LineHeight condition Line height
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TextObject::Text::OutlineThickness condition Outline thickness
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Thickness"`

TextObject::Text::ShadowAngle condition Shadow angle
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle (in degrees)"`

TextObject::Text::ShadowBlurRadius condition Shadow blur radius
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Blur radius"`

TextObject::Text::ShadowDistance condition Shadow distance
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Distance"`

TextObject::Text::ShadowOpacity condition Shadow opacity
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Opacity (0 - 255)"`

TextObject::TextAlignment condition Alignment
`0:object[TextObject::Text] "Object" | 1:relationalOperator[string] "Sign of the test" | 2:stringWithSelector[["left", "center", "right"]] "Alignment"`

TextObject::WrappingWidth condition Wrapping width
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TextObject::Angle condition Angle **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle to compare to (in degrees)"`

TextObject::Opacity condition Opacity **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Opacity to compare to (0-255)"`

TextObject::ScaleX condition Scale on X axis **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Scale to compare to (1 by default)"`

TextObject::ScaleY condition Scale on Y axis **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Scale to compare to (1 by default)"`

TextObject::Size condition Font size **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TextObject::String condition Compare the text **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:relationalOperator[string] "Sign of the test" | 2:string "Text to compare to"`

TextObject::ChangeColor action Color
`0:object[TextObject::Text] "Object" | 1:color "Color"`

TextObject::Font action Font
`0:object[TextObject::Text] "Object" | 1:fontResource "Font resource name"`

TextObject::SetBold action Bold
`0:object[TextObject::Text] "Object" | 1:yesorno "Set bold style"`

TextObject::SetGradient action Gradient
`0:object[TextObject::Text] "Object" | 1:stringWithSelector[["LINEAR_VERTICAL", "LINEAR_HORIZONTAL"]] "Gradient type" | 2:color "First Color" | 3:color "Second Color" | 4:color "Third Color" | 5:color "Fourth Color"`

TextObject::SetItalic action Italic
`0:object[TextObject::Text] "Object" | 1:yesorno "Set italic"`

TextObject::SetPadding action Padding
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TextObject::SetTextAlignment action Alignment
`0:object[TextObject::Text] "Object" | 1:stringWithSelector[["left", "center", "right"]] "Alignment"`

TextObject::SetWrapping action Word wrapping
`0:object[TextObject::Text] "Object" | 1:yesorno "Wrapping"`

TextObject::ShowShadow action Enable shadow
`0:object[TextObject::Text] "Object" | 1:yesorno? "Show the shadow" default yes`

TextObject::Text::SetFontSize action Font size
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TextObject::Text::SetLineHeight action Line height
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TextObject::Text::SetOutlineColor action Outline color
`0:object[TextObject::Text] "Object" | 1:color "Color"`

TextObject::Text::SetOutlineEnabled action Enable outline
`0:object[TextObject::Text] "Object" | 1:yesorno? "Enable outline" default yes`

TextObject::Text::SetOutlineThickness action Outline thickness
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Thickness"`

TextObject::Text::SetShadowAngle action Shadow angle
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

TextObject::Text::SetShadowBlurRadius action Shadow blur radius
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Blur radius"`

TextObject::Text::SetShadowColor action Shadow color
`0:object[TextObject::Text] "Object" | 1:color "Color"`

TextObject::Text::SetShadowDistance action Shadow distance
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Distance"`

TextObject::Text::SetShadowOpacity action Shadow opacity
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Opacity (0 - 255)"`

TextObject::WrappingWidth action Wrapping width
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TextObject::Angle action Angle **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

TextObject::Opacity action Text opacity **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Opacity (0-255)"`

TextObject::Scale action Scale **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

TextObject::ScaleX action Scale on X axis **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

TextObject::ScaleY action Scale on Y axis **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

TextObject::SetOutline action Outline **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:color "Color" | 2:expression "Thickness"`

TextObject::SetShadow action Text shadow **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:color "Color" | 2:expression "Distance" | 3:expression "Blur" | 4:expression "Angle"`

TextObject::Size action Font size **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TextObject::String action Modify the text **[legacy/hidden]**
`0:object[TextObject::Text] "Object" | 1:operator[string] "Modification's sign" | 2:string "Text"`

FontSize number expression Font size
`0:object[TextObject::Text] "Object"`

LineHeight number expression Line height
`0:object[TextObject::Text] "Object"`

OutlineThickness number expression Outline thickness
`0:object[TextObject::Text] "Object"`

Padding number expression Padding
`0:object[TextObject::Text] "Object"`

ShadowAngle number expression Shadow angle
`0:object[TextObject::Text] "Object"`

ShadowBlurRadius number expression Shadow blur radius
`0:object[TextObject::Text] "Object"`

ShadowDistance number expression Shadow distance
`0:object[TextObject::Text] "Object"`

ShadowOpacity number expression Shadow opacity
`0:object[TextObject::Text] "Object"`

Angle number expression Angle **[legacy/hidden]**
`0:object[TextObject::Text] "Object"`

Opacity number expression Opacity of a Text object **[legacy/hidden]**
`0:object[TextObject::Text] "Object"`

ScaleX number expression X Scale of a Text object **[legacy/hidden]**
`0:object[TextObject::Text] "Object"`

ScaleY number expression Y Scale of a Text object **[legacy/hidden]**
`0:object[TextObject::Text] "Object"`

String string expression Text **[legacy/hidden]**
`0:object[TextObject::Text] "Object"`

Variables - BuiltinVariables

Actions, conditions and expressions to handle variables, from simple variables like the player score, the number of remaining lives to complex variables containing arbitrary data like an inventory or the result of a web request.

Free instructions

BooleanVariable condition Variable value
`0:variableOrPropertyOrParameter "Variable" | 1:trueorfalse "Check if the value is" default true | 2:trueorfalse=""`

NumberVariable condition Variable value
`0:variableOrPropertyOrParameter "Variable" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

StringVariable condition Variable value
`0:variableOrPropertyOrParameter "Variable" | 1:relationalOperator[string] "Sign of the test" | 2:string "Value to compare"`

VariableChildCount condition Number of children
`0:variable "Array variable" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

VariableChildExists2 condition Child existence
`0:variable "Variable" | 1:string "Name of the child"`

GlobalVariableAsBoolean condition Boolean variable **[legacy/hidden]**
`0:globalvar "Variable" | 1:trueorfalse "Check if the value is" default true`

GlobalVariableChildCount condition Number of children **[legacy/hidden]**
`0:globalvar "Array variable" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

GlobalVariableChildExists condition Child existence **[legacy/hidden]**
`0:globalvar "Variable" | 1:string "Name of the child"`

SceneVariableAsBoolean condition Boolean variable **[legacy/hidden]**
`0:scenevar "Variable" | 1:trueorfalse "Check if the value is" default true`

SceneVariableChildCount condition Number of children **[legacy/hidden]**
`0:scenevar "Array variable" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

VarGlobal condition Number variable **[legacy/hidden]**
`0:globalvar "Variable" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

VarGlobalDef condition Variable defined **[legacy/hidden]**
`0:currentScene="" | 1:string "Variable"`

VarGlobalTxt condition Text variable **[legacy/hidden]**
`0:globalvar "Variable" | 1:relationalOperator[string] "Sign of the test" | 2:string "Value to compare"`

VarScene condition Number variable **[legacy/hidden]**
`0:scenevar "Variable" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

VarSceneDef condition Variable defined **[legacy/hidden]**
`0:currentScene="" | 1:string "Variable"`

VarSceneTxt condition Text variable **[legacy/hidden]**
`0:scenevar "Variable" | 1:relationalOperator[string] "Sign of the test" | 2:string "Value to compare"`

VariableChildExists condition Child existence **[legacy/hidden]**
`0:scenevar "Variable" | 1:string "Name of the child"`

ClearVariableChildren action Clear children
`0:variable "Structure or array variable"`

PushBoolean action Add value to array variable
`0:variable "Array variable" | 1:trueorfalse "Boolean to add"`

PushNumber action Add value to array variable
`0:variable "Array variable" | 1:expression "Number to add"`

PushString action Add value to array variable
`0:variable "Array variable" | 1:string "Text to add"`

PushVariable action Add existing variable
`0:variable "Array variable" | 1:variable "Variable with the content to add"`

RemoveVariableAt action Remove variable by index
`0:variable "Array variable" | 1:expression "Index to remove"`

RemoveVariableChild action Remove a child
`0:variable "Structure variable" | 1:string "Child's name"`

SetBooleanVariable action Change variable value
`0:variableOrProperty "Variable" | 1:operator[boolean] "Value" | 2:trueorfalse=""`

SetNumberVariable action Change variable value
`0:variableOrProperty "Variable" | 1:operator[number] "Modification's sign" | 2:number "Value"`

SetStringVariable action Change variable value
`0:variableOrProperty "Variable" | 1:operator[string] "Modification's sign" | 2:string "Value"`

GlobalVariableClearChildren action Clear children **[legacy/hidden]**
`0:globalvar "Structure or array variable"`

GlobalVariablePush action Add existing variable **[legacy/hidden]**
`0:globalvar "Array variable" | 1:scenevar "Scene variable with the content to add"`

GlobalVariablePushBool action Add boolean variable **[legacy/hidden]**
`0:globalvar "Array variable" | 1:trueorfalse "Boolean to add"`

GlobalVariablePushNumber action Add number variable **[legacy/hidden]**
`0:globalvar "Array variable" | 1:expression "Number to add"`

GlobalVariablePushString action Add text variable **[legacy/hidden]**
`0:globalvar "Array variable" | 1:string "Text to add"`

GlobalVariableRemoveAt action Remove variable by index **[legacy/hidden]**
`0:globalvar "Array variable" | 1:expression "Index to remove"`

GlobalVariableRemoveChild action Remove a child **[legacy/hidden]**
`0:globalvar "Structure variable" | 1:string "Child's name"`

ModVarGlobal action Change number variable **[legacy/hidden]**
`0:globalvar "Variable" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ModVarGlobalTxt action Change text variable **[legacy/hidden]**
`0:globalvar "Variable" | 1:operator[string] "Modification's sign" | 2:string "Value"`

ModVarScene action Change number variable **[legacy/hidden]**
`0:scenevar "Variable" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ModVarSceneTxt action Change text variable **[legacy/hidden]**
`0:scenevar "Variable" | 1:operator[string] "Modification's sign" | 2:string "Value"`

SceneVariablePush action Add existing variable **[legacy/hidden]**
`0:scenevar "Array variable" | 1:scenevar "Scene variable with the content to add"`

SceneVariablePushBool action Add boolean variable **[legacy/hidden]**
`0:scenevar "Array variable" | 1:trueorfalse "Boolean to add"`

SceneVariablePushNumber action Add number variable **[legacy/hidden]**
`0:scenevar "Array variable" | 1:expression "Number to add"`

SceneVariablePushString action Add text variable **[legacy/hidden]**
`0:scenevar "Array variable" | 1:string "Text to add"`

SceneVariableRemoveAt action Remove variable by index **[legacy/hidden]**
`0:scenevar "Array variable" | 1:expression "Index to remove"`

SetGlobalVariableAsBoolean action Change boolean variable **[legacy/hidden]**
`0:globalvar "Variable" | 1:trueorfalse "New Value:"`

SetSceneVariableAsBoolean action Change boolean variable **[legacy/hidden]**
`0:scenevar "Variable" | 1:trueorfalse "New Value:"`

ToggleGlobalVariableAsBoolean action Toggle boolean variable **[legacy/hidden]**
`0:globalvar "Variable"`

ToggleSceneVariableAsBoolean action Toggle boolean variable **[legacy/hidden]**
`0:scenevar "Variable"`

VariableClearChildren action Clear children **[legacy/hidden]**
`0:scenevar "Structure or array variable"`

VariableRemoveChild action Remove a child **[legacy/hidden]**
`0:scenevar "Structure variable" | 1:string "Child's name"`

VariableChildCount number expression Number of children
`0:variable[AllowUndeclaredVariable] "Array or structure variable"`

VariableFirstNumber number expression First number child
`0:variable "Array variable"`

VariableLastNumber number expression Last number child
`0:variable "Array variable"`

GlobalVariable number expression Number variable **[legacy/hidden]**
`0:globalvar "Name of the global variable"`

GlobalVariableChildCount number expression Number of children **[legacy/hidden]**
`0:globalvar "Array or structure variable"`

GlobalVariableFirstNumber number expression First number child **[legacy/hidden]**
`0:globalvar "Array variable"`

GlobalVariableLastNumber number expression Last number child **[legacy/hidden]**
`0:globalvar "Array variable"`

SceneVariableFirstNumber number expression First number child **[legacy/hidden]**
`0:scenevar "Array variable"`

SceneVariableLastNumber number expression Last number child **[legacy/hidden]**
`0:scenevar "Array variable"`

Variable number expression Number variable **[legacy/hidden]**
`0:scenevar "Variable"`

VariableFirstString string expression First text child
`0:variable "Array variable"`

VariableLastString string expression Last text child
`0:variable "Array variable"`

GlobalVariableFirstString string expression First text child **[legacy/hidden]**
`0:globalvar "Array variable"`

GlobalVariableLastString string expression Last text child **[legacy/hidden]**
`0:globalvar "Array variable"`

GlobalVariableString string expression Text variable **[legacy/hidden]**
`0:globalvar "Variable"`

SceneVariableFirstString string expression First text child **[legacy/hidden]**
`0:scenevar "Array variable"`

SceneVariableLastString string expression Last text child **[legacy/hidden]**
`0:scenevar "Array variable"`

VariableString string expression Text variable **[legacy/hidden]**
`0:scenevar "Variable"`

Mouse and touch - BuiltinMouse

Conditions, actions and expressions to handle either the mouse or touches on a touchscreen. Notably: cursor position, mouse wheel, mouse buttons, touch positions, started/end touches, etc... By default, conditions related to the mouse will also handle the touches - so that it's easier to handle both in your game. You can disable this behavior if you want to handle them separately in different events.

Free instructions

CursorX condition Cursor X position
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CursorY condition Cursor Y position
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

HasAnyTouchOrMouseStarted condition A new touch has started
`0:currentScene=""`

HasTouchEnded condition A touch has ended
0:currentScene="" | 1:expression "Touch identifier"

IsMouseInsideCanvas condition Mouse cursor is inside the window
0:currentScene=""

IsMouseWheelScrollingDown condition The mouse wheel is scrolling down
0:currentScene=""

IsMouseWheelScrollingUp condition The mouse wheel is scrolling up
0:currentScene=""

MouseButtonFromTextPressed condition Mouse button pressed or touch held
0:currentScene="" | 1:mouseButton "Button to check"

MouseButtonFromTextReleased condition Mouse button released
0:currentScene="" | 1:mouseButton "Button to check"

TouchX condition Touch X position
0:currentScene="" | 1:expression "Touch identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare" | 4:layer? "Layer" default "" | 5:expression? "Camera number (default : 0)" default 0

TouchY condition Touch Y position
0:currentScene="" | 1:expression "Touch identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare" | 4:layer? "Layer" default "" | 5:expression? "Camera number (default : 0)" default 0

HasAnyTouchStarted condition A new touch has started **[legacy/hidden]**
0:currentScene=""

MouseButtonPressed condition Mouse button pressed or touch held **[legacy/hidden]**
0:currentScene="" | 1:mouse "Button to check"

MouseButtonReleased condition Mouse button released **[legacy/hidden]**
0:currentScene="" | 1:mouse "Button to check"

MouseOnlyCursorY condition Mouse cursor Y position **[legacy/hidden]**
0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0

MouseOnlyOnlyCursorX condition **[legacy/hidden]**

MouseX condition Cursor X position **[legacy/hidden]**
0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0

MouseY condition Cursor Y position **[legacy/hidden]**
0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0

PopEndedTouch condition A touch has ended **[legacy/hidden]**
0:currentScene=""

PopStartedTouch condition A new touch has started **[legacy/hidden]**
0:currentScene=""

SourisBouton condition Mouse button pressed or touch held **[legacy/hidden]**
0:currentScene="" | 1:mouse "Button to check"

SourisX condition Cursor X position **[legacy/hidden]**
0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0

SourisY condition Cursor Y position **[legacy/hidden]**
0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0

HideCursor action Hide the cursor
0:currentScene=""

ShowCursor action Show the cursor
0:currentScene=""

TouchSimulateMouse action De/activate moving the mouse cursor with touches
0:currentScene="" | 1:yesorno "Activate (yes by default when game is launched)" default yes

CacheSouris action Hide the cursor **[legacy/hidden]**
0:currentScene=""

MontreSouris action Show the cursor **[legacy/hidden]**
0:currentScene=""

CursorX number expression Cursor X position
0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0

CursorY number expression Cursor Y position
0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0

MouseWheelDelta number expression Mouse wheel: Displacement
0:currentScene=""

StartedTouchOrMouseCount number expression Started touch count
0:currentScene=""

StartedTouchOrMouseId number expression Started touch identifier
0:currentScene="" | 1:expression "Touch index"

TouchX number expression Touch X position
0:currentScene="" | 1:expression "Touch identifier" | 2:layer? "Layer" default "" | 3:expression? "Camera number (default : 0)" default 0

TouchY number expression Touch Y position
0:currentScene="" | 1:expression "Touch identifier" | 2:layer? "Layer" default "" | 3:expression? "Camera number (default : 0)" default 0

LastEndedTouchId number expression Identifier of the last ended touch **[legacy/hidden]**
`0:currentScene=""`

LastTouchId number expression Identifier of the last touch **[legacy/hidden]**
`0:currentScene=""`

MouseOnlyCursorX number expression Mouse cursor X position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

MouseOnlyCursorY number expression Mouse cursor Y position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

MouseX number expression Cursor X position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

MouseY number expression Cursor Y position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

SourisX number expression Cursor X position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

SourisY number expression Cursor Y position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

StartedTouchCount number expression Started touch count **[legacy/hidden]**
`0:currentScene=""`

StartedTouchId number expression Started touch identifier **[legacy/hidden]**
`0:currentScene="" | 1:expression "Touch index"`

Keyboard - BuiltinKeyboard

Conditions to check keys pressed on a keyboard. Note that this does not work with on-screen keyboard on touch devices: use instead mouse/touch conditions when making a game for mobile/touchscreen devices or when making a new game from scratch.

Free instructions

AnyKeyPressed condition Any key pressed
`0:currentScene=""`

AnyKeyReleased condition Any key released
`0:currentScene=""`

KeyFromTextJustPressed condition Key just pressed
`0:currentScene="" | 1:keyboardKey "Key to check"`

KeyFromTextPressed condition Key pressed
`0:currentScene="" | 1:keyboardKey "Key to check"`

KeyFromTextReleased condition Key released
`0:currentScene="" | 1:keyboardKey "Key to check"`

KeyPressed condition Key pressed **[legacy/hidden]**
`0:currentScene="" | 1:key "Key to check"`

KeyReleased condition Key released **[legacy/hidden]**
`0:currentScene="" | 1:key "Key to check"`

LastPressedKey string expression Last pressed key
`0:currentScene=""`

Timers and time - BuiltinTime

Actions and conditions to start, pause or reset scene timers, modify the time scale (speed at which the game is running - useful for slow motion effects). Also contains an action that wait for a delay before running the next actions and sub-events and expressions to read the time scale, time delta of the last frame or timer elapsed time.

Free instructions

CompareTimer condition Value of a scene timer
`0:currentScene="" | 1:identifier[sceneTimer] "Timer's name" | 2:relationalOperator[time] "Sign of the test" | 3:expression "Time in seconds"`

TimeScale condition Time scale
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Time scale (1 by default)"`

TimerPaused condition Scene timer paused
`0:currentScene="" | 1:identifier[sceneTimer] "Timer's name"`

Timer condition Value of a scene timer **[legacy/hidden]**
`0:currentScene="" | 1:expression "Time in seconds" | 2:identifier[sceneTimer] "Timer's name"`

ChangeTimeScale action Time scale
`0:currentScene="" | 1:expression "Scale (1: Default, 2: 2x faster, 0.5: 2x slower...)"`

PauseTimer action Pause a scene timer
`0:currentScene="" | 1:identifier[sceneTimer] "Timer's name"`

RemoveTimer action Delete a scene timer
`0:currentScene="" | 1:identifier[sceneTimer] "Timer's name"`

ResetTimer action Start (or reset) a scene timer
`0:currentScene="" | 1:identifier[sceneTimer] "Timer's name"`

UnPauseTimer action Unpause a scene timer
`0:currentScene="" | 1:identifier[sceneTimer] "Timer's name"`

Wait action Wait X seconds
`0:expression "Time to wait in seconds"`

Time number expression Current time
`0:currentScene="" | 1:stringWithSelector[["hour", "min", "sec", "mon", "year", "wday", "mday", "yday", "timestamp"]] "- Hour of the day: "hour" - Minutes: "min" - Seconds: "sec" - Day of month: "mday" - Months since January: "mon" - Year since 1900: "year" - Days since Sunday: "wday" - Days since Jan 1st: "yday" - Timestamp (ms): "timestamp""`

TimeDelta number expression Time elapsed since the last frame
`0:currentScene=""`

TimeFromStart number expression Time elapsed since the beginning of the scene (in seconds).
`0:currentScene=""`

TimeScale number expression Time scale
`0:currentScene=""`

TimerElapsedTime number expression Scene timer value
`0:currentScene="" | 1:identifier[sceneTimer] "Timer's name"`

ElapsedTime number expression Time elapsed since the last frame **[legacy/hidden]**
`0:currentScene=""`

TempsDebut number expression Time elapsed since the beginning of the scene (in seconds). **[legacy/hidden]**
`0:currentScene=""`

TempsFrame number expression Time elapsed since the last frame **[legacy/hidden]**
`0:currentScene=""`

Mathematical tools - BuiltinMathematicalTools

A set of mathematical functions that can be used in expressions.

Free instructions

AngleBetweenPositions number expression Angle between two positions
`0:expression "First point X position" | 1:expression "First point Y position" | 2:expression "Second point X position" | 3:expression "Second point Y position"`

AngleDifference number expression Difference between two angles
`0:expression "First angle, in degrees" | 1:expression "Second angle, in degrees"`

DistanceBetweenPositions number expression Distance between two positions
`0:expression "First point X position" | 1:expression "First point Y position" | 2:expression "Second point X position" | 3:expression "Second point Y position"`

Pi number expression Number Pi (3.1415...)

Random number expression Random integer
`0:expression "Maximum value"`

RandomFloat number expression Random float
`0:expression "Maximum value"`

RandomFloatInRange number expression Random float in range
`0:expression "Minimum value" | 1:expression "Maximum value"`

RandomInRange number expression Random integer in range
`0:expression "Minimum value" | 1:expression "Maximum value"`

RandomWithStep number expression Random value in steps
`0:expression "Minimum value" | 1:expression "Maximum value" | 2:expression "Step"`

XFromAngleAndDistance number expression X position from angle and distance
`0:expression "Angle, in degrees" | 1:expression "Distance"`

YFromAngleAndDistance number expression Y position from angle and distance
`0:expression "Angle, in degrees" | 1:expression "Distance"`

abs number expression Absolute value
`0:expression "Expression"`

acos number expression Arccosine
`0:expression "Expression"`

acosh number expression Hyperbolic arccosine
`0:expression "Expression"`

asin number expression Arcsine
`0:expression "Expression"`

asinh number expression Arcsine
`0:expression "Expression"`

atan number expression Arctangent
`0:expression "Expression"`

atan2 number expression 2 argument arctangent
`0:expression "Y" | 1:expression "X"`

atanh number expression Hyperbolic arctangent
`0:expression "Expression"`

cbrt number expression Cube root
`0:expression "Expression"`

ceil number expression Ceil (round up)
`0:expression "Expression"`

ceilTo number expression Ceil (round up) to a decimal point
`0:expression "Expression" | 1:expression? "Expression"`

clamp number expression Clamp (restrict a value to a given range)
`0:expression "Value" | 1:expression "Min" | 2:expression "Max"`

cos number expression Cosine
`0:expression "Expression"`

cosh number expression Hyperbolic cosine
`0:expression "Expression"`

cot number expression Cotangent
`0:expression "Expression"`

csc number expression Cosecant
`0:expression "Expression"`

exp number expression Exponential
`0:expression "Expression"`

floor number expression Floor (round down)
`0:expression "Expression"`

floorTo number expression Floor (round down) to a decimal point
`0:expression "Expression" | 1:expression? "Expression"`

lerp number expression Lerp (Linear interpolation)
`0:expression "a (in a+(b-a)*x)" | 1:expression "b (in a+(b-a)*x)" | 2:expression "x (in a+(b-a)*x)"`

lerpAngle number expression Lerp (Linear interpolation) between two angles
`0:expression "Starting angle, in degrees" | 1:expression "Destination angle, in degrees" | 2:expression "Interpolation value between 0 and 1."`

log number expression Logarithm
`0:expression "Expression"`

log10 number expression Base-10 logarithm
`0:expression "Expression"`

log2 number expression Base-2 logarithm
`0:expression "Expression"`

max number expression Maximum of two numbers
`0:expression "First expression" | 1:expression "Second expression"`

min number expression Minimum of two numbers
`0:expression "First expression" | 1:expression "Second expression"`

mod number expression Modulo
`0:expression "x (as in x mod y)" | 1:expression "y (as in x mod y)"`

normalize number expression Normalize a value between `min` and `max` to a value between 0 and 1.
`0:expression "Value" | 1:expression "Min" | 2:expression "Max"`

nthroot number expression Nth root
`0:expression "Number" | 1:expression "N"`

pow number expression Power
`0:expression "Number" | 1:expression "The exponent (n in x^n)"`

round number expression Round
`0:expression "Expression"`

roundTo number expression Round to a decimal point
`0:expression "Number to Round" | 1:expression? "Decimal Places"`

sec number expression Secant
`0:expression "Expression"`

sign number expression Sign of a number
`0:expression "Expression"`

sin number expression Sine
`0:expression "Expression"`

sinh number expression Hyperbolic sine
`0:expression "Expression"`

sqrt number expression Square root
`0:expression "Expression"`

tan number expression Tangent
`0:expression "Expression"`

tanh number expression Hyperbolic tangent
`0:expression "Expression"`

trunc number expression Truncation
`0:expression "Expression"`

int number expression Round **[legacy/hidden]**
`0:expression "Expression"`

ln number expression Logarithm **[legacy/hidden]**
`0:expression "Expression"`

rint number expression Round **[legacy/hidden]**
`0:expression "Expression"`

Conversion - BuiltinCommonConversions

Expressions to convert numbers to string, strings to numbers, angles (degrees from/to radians) and a GDevelop variable to/from a JSON string.

Free instructions

JSONToObjectVariableStructure action Convert JSON to object variable
`0:string "JSON string" | 1:objectPtr "Object" | 2:objectvar "Object variable where store the JSON object"`

JSONToVariableStructure2 action Convert JSON to a variable
`0:string "JSON string" | 1:variable "Variable where to store the JSON object"`

JSONToGlobalVariableStructure action Convert JSON to global variable **[legacy/hidden]**
`0:string "JSON string" | 1:globalvar "Global variable where store the JSON object"`

JSONToVariableStructure action Convert JSON to a scene variable **[legacy/hidden]**
`0:string "JSON string" | 1:scenevar "Variable where store the JSON object"`

ToDeg number expression Radians > Degrees
`0:expression "Angle, in radians"`

ToNumber number expression Text > Number
`0:string "Text to convert to a number"`

ToRad number expression Degrees > Radians
`0:expression "Angle, in degrees"`

LargeNumberToString string expression Number > Text (without scientific notation)
`0:expression "Expression to be converted to text"`

ObjectVarToJSON string expression Convert object variable to JSON
`0:objectPtr "The object with the variable" | 1:objectvar "The object variable to be stringified"`

ToJSON string expression Convert variable to JSON
`0:variable[AllowUndeclaredVariable] "The variable to be stringified"`

ToString string expression Number > Text
`0:expression "Expression to be converted to text"`

GlobalVarToJSON string expression Convert global variable to JSON **[legacy/hidden]**
`0:globalvar "The global variable to be stringified"`

Text manipulation - BuiltinStringInstructions

Provides expressions to manipulate strings (also called texts): new line, upper/lowercase, substring, find, replace, etc...

Free instructions

StrFind number expression Search in a text
`0:string "Text" | 1:string "Text to search for"`

StrFindFrom number expression Search in a text, starting from a position
`0:string "Text" | 1:string "Text to search for" | 2:expression "Position of the first character in the string to be considered in the search"`

StrFindLast number expression Search the last occurrence in a text
`0:string "Text" | 1:string "Text to search for"`

StrFindLastFrom number expression Search the last occurrence in a text, starting from a position
`0:string "Text" | 1:string "Text to search for" | 2:expression "Position of the last character in the string to be considered in the search"`

StrLength number expression Length of a text
`0:string "Text"`

StrRFind number expression Search in a text from the end **[legacy/hidden]**
`0:string "Text" | 1:string "Text to search for"`

StrRFindFrom number expression Search in a text from the end, starting from a position **[legacy/hidden]**
`0:string "Text" | 1:string "Text to search for" | 2:expression "Position of the last character in the string to be considered in the search"`

FromCodePoint string expression Get character from code point
`0:expression "Code point"`

NewLine string expression Insert a new line

StrAt string expression Get a character from a text
`0:string "Text" | 1:expression "Position of the character (the first letter is at position 0)"`

StrRepeat string expression Repeat a text
`0:string "Text to repeat" | 1:expression "Repetition count"`

StrReplaceAll string expression Replace all occurrences of a text by another.
`0:string "Text in which the replacement(s) must be done" | 1:string "Text to find inside the first text" | 2:string "Replacement to put instead of the text to find"`

StrReplaceOne string expression Replace the first occurrence of a text by another.
`0:string "Text in which the replacement must be done" | 1:string "Text to find inside the first text" | 2:string "Replacement to put instead of the text to find"`

SubStr string expression Get a portion of a text
`0:string "Text" | 1:expression "Start position of the portion (the first letter is at position 0)" | 2:expression "Length of the portion"`

ToLowerCase string expression Lowercase a text
`0:string "Text"`

ToUpperCase string expression Uppercase a text
`0:string "Text"`

Sounds and music - BuiltinAudio

GDevelop provides several conditions and actions to play audio files. They can be either long music or short sound effects.

Free instructions

GlobalVolume condition Global volume
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Volume to compare to (0-100)"`

MusicChannelPitch condition Pitch of the music on a channel
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Pitch to compare to (1 by default)"`

MusicChannelPlayingOffset condition Playing offset of the music on a channel
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Position to compare to (in seconds)"`

MusicChannelVolume condition Volume of the music on a channel
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Volume to compare to (0-100)"`

MusicPaused condition A music file is paused
`0:currentScene="" | 1:expression "Channel"`

MusicPlaying condition A music file is being played
`0:currentScene="" | 1:expression "Channel"`

MusicStopped condition A music file is stopped
`0:currentScene="" | 1:expression "Channel"`

SoundChannelPitch condition Pitch of the sound of a channel
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Pitch to compare to (1 by default)"`

SoundChannelPlayingOffset condition Playing offset of the sound on a channel
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Position to compare to (in seconds)"`

SoundChannelVolume condition Volume of the sound on a channel
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Volume to compare to (0-100)"`

SoundPaused condition A sound is paused
`0:currentScene="" | 1:expression "Channel"`

SoundPlaying condition A sound is being played
`0:currentScene="" | 1:expression "Channel"`

SoundStopped condition A sound is stopped
`0:currentScene="" | 1:expression "Channel"`

MusicCanalVolume condition Volume of the music on a channel **[legacy/hidden]**
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Volume to compare to (0-100)"`

SoundCanalVolume condition Volume of the sound on a channel **[legacy/hidden]**
`0:currentScene="" | 1:expression "Channel identifier" | 2:relationalOperator[number] "Sign of the test" | 3:number "Volume to compare to (0-100)"`

FadeMusicVolume action Fade the volume of a music played on a channel.
`0:currentScene="" | 1:expression "Channel identifier" | 2:expression "Final volume (0-100)" | 3:expression "Fading time in seconds"`

FadeSoundVolume action Fade the volume of a sound played on a channel.
`0:currentScene="" | 1:expression "Channel identifier" | 2:expression "Final volume (0-100)" | 3:expression "Fading time in seconds"`

PauseMusicChannel action Pause the music of a channel
`0:currentScene="" | 1:expression "Channel identifier"`

PauseSoundChannel action Pause the sound of a channel
`0:currentScene="" | 1:expression "Channel identifier"`

PlayMusic action Play a music file
`0:currentScene="" | 1:musicfile "Audio file (or audio resource name)" | 2:yesorno? "Repeat the sound" default no | 3:expression? "Volume" default 100 | 4:expression? "Pitch (speed)" default 1`

PlayMusicOnChannel action Play a music file on a channel
`0:currentScene="" | 1:musicfile "Audio file (or audio resource name)" | 2:expression "Channel identifier" | 3:yesorno? "Repeat the sound" default no | 4:expression? "Volume" default 100 | 5:expression? "Pitch (speed)" default 1`

PlaySound action Play a sound
`0:currentScene="" | 1:soundfile "Audio file (or audio resource name)" | 2:yesorno? "Repeat the sound" default no | 3:expression? "Volume" default 100 | 4:expression? "Pitch (speed)" default 1`

PlaySoundOnChannel action Play a sound on a channel
`0:currentScene="" | 1:soundfile "Audio file (or audio resource name)" | 2:expression "Channel identifier" | 3:yesorno? "Repeat the sound" default no | 4:expression? "Volume" default 100 | 5:expression? "Pitch (speed)" default 1`

PreloadMusic action Preload a music file
`0:currentScene="" | 1:musicfile "Audio file (or audio resource name)"`

PreloadSound action Preload a sound file
`0:currentScene="" | 1:soundfile "Sound file (or sound resource name)"`

ResumeMusicChannel action Resume playing a music on a channel
0:currentScene="" | 1:expression "Channel identifier"

ResumeSoundChannel action Resume playing a sound on a channel
0:currentScene="" | 1:expression "Channel identifier"

SetGlobalVolume action Game global volume
0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Volume (0-100)"

SetMusicChannelPitch action Pitch of the music on a channel
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Pitch (1 by default)"

SetMusicChannelPlayingOffset action Playing offset of the music on a channel
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Position (in seconds)"

SetMusicChannelVolume action Volume of the music on a channel
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Volume (0-100)"

SetSoundChannelPitch action Pitch of the sound of a channel
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Pitch (1 by default)"

SetSoundChannelPlayingOffset action Playing offset of the sound on a channel
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Position (in seconds)"

SetSoundChannelVolume action Volume of the sound on a channel
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Volume (0-100)"

StopMusicChannel action Stop the music on a channel
0:currentScene="" | 1:expression "Channel identifier"

StopSoundChannel action Stop the sound of a channel
0:currentScene="" | 1:expression "Channel identifier"

UnloadAllAudio action Unload all audio
0:currentScene=""

UnloadMusic action Unload a music file
0:currentScene="" | 1:musicfile "Audio file (or audio resource name)"

UnloadSound action Unload a sound file
0:currentScene="" | 1:soundfile "Sound file (or sound resource name)"

ModGlobalVolume action Game global volume **[legacy/hidden]**
0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Volume (0-100)"

ModPitchMusicChannel action Pitch of the music on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Pitch (1 by default)"

ModPitchSoundChannel action Pitch of the sound of a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Pitch (1 by default)"

ModPlayingOffsetMusicChannel action Playing offset of the music on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Position (in seconds)"

ModPlayingOffsetSoundChannel action Playing offset of the sound on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Position (in seconds)"

ModVolumeMusicCanal action Volume of the music on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Volume (0-100)"

ModVolumeSoundCanal action Volume of the sound on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier" | 2:operator[number] "Modification's sign" | 3:number "Volume (0-100)"

PauseMusicCanal action Pause the music of a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier"

PauseSoundCanal action Pause the sound of a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier"

PlayMusicCanal action Play a music file on a channel **[legacy/hidden]**
0:currentScene="" | 1:musicfile "Audio file (or audio resource name)" | 2:expression "Channel identifier" | 3:yesorno? "Repeat the sound" default no | 4:expression? "Volume" default 100 | 5:expression? "Pitch (speed)" default 1

PlaySoundCanal action Play a sound on a channel **[legacy/hidden]**
0:currentScene="" | 1:soundfile "Audio file (or audio resource name)" | 2:expression "Channel identifier" | 3:yesorno? "Repeat the sound" default no | 4:expression? "Volume" default 100 | 5:expression? "Pitch (speed)" default 1

RePlayMusicCanal action Resume playing a music on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier"

RePlaySoundCanal action Resume playing a sound on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier"

StopMusicCanal action Stop the music on a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier"

StopSoundCanal action Stop the sound of a channel **[legacy/hidden]**
0:currentScene="" | 1:expression "Channel identifier"

GlobalVolume number expression Global volume
`0:currentScene=""`

MusicChannelPitch number expression Music's pitch
`0:currentScene="" | 1:expression "Channel"`

MusicChannelPlayingOffset number expression Music playing offset
`0:currentScene="" | 1:expression "Channel"`

MusicChannelVolume number expression Music volume
`0:currentScene="" | 1:expression "Channel"`

SoundChannelPitch number expression Sound's pitch
`0:currentScene="" | 1:expression "Channel"`

SoundChannelPlayingOffset number expression Sound playing offset
`0:currentScene="" | 1:expression "Channel"`

SoundChannelVolume number expression Sound volume
`0:currentScene="" | 1:expression "Channel"`

Storage - BuiltinFile

Actions and conditions to store data (like the player progress or anything else to be persisted across game sessions). Data are stored on the device and erased when the game is uninstalled.

Free instructions

GroupExists condition Existence of a group
`0:string "Storage name" | 1:string "Group"`

DeleteFichier action Clear a storage
`0:string "Storage name"`

DeleteGroupFichier action Delete an element
`0:string "Storage name" | 1:string "Group"`

EcrireFichierExp action Save a value
`0:string "Storage name" | 1:string "Group" | 2:expression "Expression"`

EcrireFichierTxt action Save a text
`0:string "Storage name" | 1:string "Group" | 2:string "Text"`

LoadFile action Manually preload a storage in memory
`0:string "Storage name"`

ReadNumberFromStorage action Load a value
`0:string "Storage name" | 1:string "Group" | 2:currentScene="" | 3:variable "Variable"`

ReadStringFromStorage action Load a text
`0:string "Storage name" | 1:string "Group" | 2:currentScene="" | 3:variable "Variable"`

UnloadFile action Manually unload and persist a storage
`0:string "Storage name"`

LireFichierExp action Load a value **[legacy/hidden]**
`0:string "Storage name" | 1:string "Group" | 2:currentScene="" | 3:scenevar "Scene variable"`

LireFichierTxt action Load a text **[legacy/hidden]**
`0:string "Storage name" | 1:string "Group" | 2:currentScene="" | 3:scenevar "Scene variable"`

Layers and cameras - BuiltinCamera

Each scene can be composed of multiple layers. These conditions and actions allow to manipulate them during the game. In particular, you can move the camera of a layer to center it on an object or a position. The zoom of a layer camera (in 2D) or its position in the 3D space (for 3D games) can be changed. Each layer can be hidden or shown (but layers do not have opacity). Actions/conditions are available to enable/disable 2D/3D effects on layers. Parameters of each effect can be modified with actions.

Free instructions

CameraAngle condition Angle of a camera of a layer
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CameraBorderBottom condition Camera bottom border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number" | 3:relationalOperator[number] "Sign of the test" | 4:number "Value to compare"`

CameraBorderLeft condition Camera left border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number" | 3:relationalOperator[number] "Sign of the test" | 4:number "Value to compare"`

CameraBorderRight condition Camera right border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number" | 3:relationalOperator[number] "Sign of the test" | 4:number "Value to compare"`

CameraBorderTop condition Camera top border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number" | 3:relationalOperator[number] "Sign of the test" | 4:number "Value to compare"`

CameraCenterX condition Camera center X position
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CameraCenterY condition Camera center Y position
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CameraHeight condition Height of a camera
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number" | 3:relationalOperator[number] "Sign of the test" | 4:number "Value to compare"`

CameraWidth condition Width of a camera
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number" | 3:relationalOperator[number] "Sign of the test" | 4:number "Value to compare"`

CameraZoom condition Camera zoom
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0 | 3:relationalOperator[number] "Sign of the test" | 4:number "Zoom"`

LayerDefaultZOrder condition Layer default Z order
`0:currentScene="" | 1:layer? "Layer" default "" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

LayerEffectEnabled condition Layer effect is enabled
`0:currentScene="" | 1:layer? "Layer" default "" | 2:layerEffectName "Effect name"`

LayerTimeScale condition Layer time scale
`0:currentScene="" | 1:layer? "Layer" default "" | 2:relationalOperator[number] "Sign of the test" | 3:number "Time scale (1 by default)"`

LayerVisible condition Visibility of a layer
`0:currentScene="" | 1:layer "Layer" default ""`

CameraX condition Camera center X position **[legacy/hidden]**
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CameraY condition Camera center Y position **[legacy/hidden]**
`0:currentScene="" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CenterCameraOnObject action Center the camera on an object
`0:currentScene="" | 1:objectPtr "Object" | 2:yesorno? "Anticipate the movement of the object (yes by default)" default yes | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

ChangeLayerTimeScale action Layer time scale
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression "Scale (1: Default, 2: 2x faster, 0.5: 2x slower...)"`

ClampCamera action Enforce camera boundaries
`0:currentScene="" | 1:expression "Left bound X Position" | 2:expression "Top bound Y Position" | 3:expression "Right bound X Position" | 4:expression "Bottom bound Y Position" | 5:layer? "Layer" default "" | 6:expression? "Camera number (default : 0)" default 0`

EnableLayerEffect action Enable layer effect
`0:currentScene="" | 1:layer? "Layer" default "" | 2:layerEffectName "Effect name" | 3:yesorno? "Enable"`

HideLayer action Hide a layer
`0:currentScene="" | 1:layer "Layer" default ""`

SetCameraAngle action Angle of a camera of a layer
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

SetCameraCenterX action Camera center X position
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

SetCameraCenterY action Camera center Y position
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

SetLayerAmbientLightColor action Ambient light color
`0:currentScene="" | 1:layer? "Layer" default "Lighting" | 2:color "Color"`

SetLayerDefaultZOrder action Layer default Z order
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression "New default Z order"`

SetLayerEffectBooleanParameter action Effect property (enable or disable)
`0:currentScene="" | 1:layer? "Layer" default "" | 2:layerEffectName "Effect name" | 3:layerEffectParameterName "Property name" | 4:yesorno "Enable this property"`

SetLayerEffectParameter action Effect property (number)
`0:currentScene="" | 1:layer? "Layer" default "" | 2:layerEffectName "Effect name" | 3:layerEffectParameterName "Property name" | 4:expression "New value"`

SetLayerEffectStringParameter action Effect property (string)
`0:currentScene="" | 1:layer? "Layer" default "" | 2:layerEffectName "Effect name" | 3:layerEffectParameterName "Property name" | 4:string "New value"`

ShowLayer action Show a layer
`0:currentScene="" | 1:layer "Layer" default ""`

ZoomCamera action Camera zoom
`0:currentScene="" | 1:expression "Value (1:Initial zoom, 2:Zoom x2, 0.5:Unzoom x2...)" | 2:layer? "Layer" default "" | 3:expression? "Camera number (default : 0)" default 0`

CameraX action Camera center X position **[legacy/hidden]**
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CameraY action Camera center Y position **[legacy/hidden]**
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CentreCamera action Center the camera on an object **[legacy/hidden]**
`0:currentScene="" | 1:objectPtr "Object" | 2:yesorno? "Anticipate the movement of the object (yes by default)" default yes | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

FixCamera action Center the camera on an object within limits **[legacy/hidden]**
`0:currentScene="" | 1:objectPtr "Object" | 2:expression "Top left side of the boundary: X Position" | 3:expression "Top left side of the boundary: Y Position" | 4:expression "Bottom right side of the boundary: X Position" | 5:expression "Bottom right side of the boundary: Y Position" | 6:yesorno? "Anticipate the movement of the object (yes by default)" default yes | 7:layer? "Layer" default "" | 8:expression? "Camera number (default : 0)" default 0`

RotateCamera action Angle of a camera of a layer **[legacy/hidden]**
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

SetCameraX action Camera center X position **[legacy/hidden]**
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

SetCameraY action Camera center Y position **[legacy/hidden]**
`0:currentScene="" | 1:operator[number] "Modification's sign" | 2:number "Value" | 3:layer? "Layer" default "" | 4:expression? "Camera number (default : 0)" default 0`

CameraAngle number expression Angle of a camera of a layer
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

CameraBorderBottom number expression Camera bottom border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number"`

CameraBorderLeft number expression Camera left border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number"`

CameraBorderRight number expression Camera right border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number"`

CameraBorderTop number expression Camera top border position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number"`

CameraCenterX number expression Camera center X position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

CameraCenterY number expression Camera center Y position
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

CameraHeight number expression Height of a camera
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number"`

CameraWidth number expression Width of a camera
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number"`

CameraZoom number expression Zoom of a camera of a layer
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

LayerDefaultZOrder number expression Default Z Order for a layer
`0:currentScene="" | 1:layer "Layer"`

LayerTimeScale number expression Layer time scale
`0:currentScene="" | 1:layer "Layer"`

CameraRotation number expression Angle of a camera of a layer **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

CameraX number expression Camera center X position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

CameraY number expression Camera center Y position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

VueRotation number expression Angle of a camera of a layer **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

VueX number expression Camera center X position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

VueY number expression Camera center Y position **[legacy/hidden]**
`0:currentScene="" | 1:layer? "Layer" default "" | 2:expression? "Camera number (default : 0)" default 0`

Game window and resolution - BuiltinWindow

Actions and conditions to manipulate the game window or change how the game is resized according to the screen size. Depending on the platform on which the game is running, not all of these features can be applied. Also contains expressions to read the screen size.

Free instructions

IsFullScreen condition Fullscreen activated?
`0:currentScene=""`

CenterWindow action Center the game window on the screen
`0:currentScene=""`

SetAdaptGameResolutionAtRuntime action Automatically adapt the game resolution
`0:currentScene="" | 1:yesorno "Update resolution during the game to fit the screen or window size?"`

SetFullScreen action De/activate fullscreen
`0:currentScene="" | 1:yesorno "Activate fullscreen" | 2:yesorno? "Keep aspect ratio (HTML5 games only, yes by default)" default yes`

SetGameResolutionResizeMode action Game resolution resize mode
`0:currentScene="" | 1:stringWithSelector[["adaptWidth", "adaptHeight", ""]] "Resize mode"`

SetGameResolutionSize action Game resolution
0:currentScene="" | 1:expression "Width" | 2:expression "Height"

SetWindowMargins action Window's margins
0:currentScene="" | 1:expression "Top" | 2:expression "Right" | 3:expression "Bottom" | 4:expression "Left"

SetWindowSize action Game window size
0:currentScene="" | 1:expression "Width" | 2:expression "Height" | 3:yesorno "Also update the game resolution? If not, the game will be stretched or reduced to fit in the window."

SetWindowTitle action Window's title
0:currentScene="" | 1:string "New title"

SceneWindowHeight number expression Height of the scene window
0:currentScene=""

SceneWindowWidth number expression Width of the scene window
0:currentScene=""

ScreenHeight number expression Height of the screen/page

ScreenWidth number expression Width of the screen/page

WindowTitle string expression Window's title
0:currentScene=""

Objects with animations - AnimatableCapability

Actions and conditions for objects having animations (sprite, 3D models...).

Behavior: AnimatableCapability::AnimatableBehavior

AnimatableCapability::AnimatableBehavior::ElapsedTime condition Animation elapsed time
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Elapsed time (in seconds)"

AnimatableCapability::AnimatableBehavior::HasAnimationEnded condition Animation finished
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

AnimatableCapability::AnimatableBehavior::Index condition Animation (by number)
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Animation index"

AnimatableCapability::AnimatableBehavior::IsAnimationPaused condition Animation paused
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

AnimatableCapability::AnimatableBehavior::Name condition Animation (by name)
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:relationalOperator[string] "Sign of the test" | 3:objectAnimationName "Animation name"

AnimatableCapability::AnimatableBehavior::SpeedScale condition Animation speed scale
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed scale"

AnimatableCapability::AnimatableBehavior::PauseAnimation action Pause the animation
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

AnimatableCapability::AnimatableBehavior::PlayAnimation action Resume the animation
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

AnimatableCapability::AnimatableBehavior::SetElapsedTime action Animation elapsed time
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Elapsed time (in seconds)"

AnimatableCapability::AnimatableBehavior::SetIndex action Animation (by number)
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Animation index"

AnimatableCapability::AnimatableBehavior::SetName action Animation (by name)
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:operator[string] "Modification's sign" | 3:objectAnimationName "Animation name"

AnimatableCapability::AnimatableBehavior::SetSpeedScale action Animation speed scale
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed scale"

Duration number expression Animation duration
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

ElapsedTime number expression Animation elapsed time
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

Index number expression Animation (by number)
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

SpeedScale number expression Animation speed scale
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

Name string expression Animation (by name)
0:object "Object" | 1:behavior[AnimatableCapability::AnimatableBehavior] "Behavior"

Scalable objects - ScalableCapability

Actions/conditions/expression to change or check the scale of an object (default: 1).

Behavior: ScalableCapability::ScalableBehavior

ScalableCapability::ScalableBehavior::Value condition Scale
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Scale (1 by default)"`

ScalableCapability::ScalableBehavior::X condition Scale on X axis
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Scale (1 by default)"`

ScalableCapability::ScalableBehavior::Y condition Scale on Y axis
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Scale (1 by default)"`

ScalableCapability::ScalableBehavior::SetValue action Scale
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Scale (1 by default)"`

ScalableCapability::ScalableBehavior::SetX action Scale on X axis
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Scale (1 by default)"`

ScalableCapability::ScalableBehavior::SetY action Scale on Y axis
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Scale (1 by default)"`

Value number expression Scale
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior"`

X number expression Scale on X axis
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior"`

Y number expression Scale on Y axis
`0:object "Object" | 1:behavior[ScalableCapability::ScalableBehavior] "Behavior"`

Objects with opacity - OpacityCapability

Action/condition/expression to change or check the opacity of an object (0-255).

Behavior: OpacityCapability::OpacityBehavior

OpacityCapability::OpacityBehavior::Value condition Opacity
`0:object "Object" | 1:behavior[OpacityCapability::OpacityBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Opacity (0-255)"`

OpacityCapability::OpacityBehavior::SetValue action Opacity
`0:object "Object" | 1:behavior[OpacityCapability::OpacityBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Opacity (0-255)"`

Value number expression Opacity
`0:object "Object" | 1:behavior[OpacityCapability::OpacityBehavior] "Behavior"`

Resizable objects - ResizableCapability

Change or compare the size (width/height) of an object which can be resized (i.e: most objects).

Behavior: ResizableCapability::ResizableBehavior

ResizableCapability::ResizableBehavior::Height condition Height
`0:object "Object" | 1:behavior[ResizableCapability::ResizableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Height"`

ResizableCapability::ResizableBehavior::Width condition Width
`0:object "Object" | 1:behavior[ResizableCapability::ResizableBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Width"`

ResizableCapability::ResizableBehavior::SetHeight action Height
`0:object "Object" | 1:behavior[ResizableCapability::ResizableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Height"`

ResizableCapability::ResizableBehavior::SetSize action Size
`0:object "Object" | 1:behavior[ResizableCapability::ResizableBehavior] "Behavior" | 2:expression "Width" | 3:expression "Height"`

ResizableCapability::ResizableBehavior::SetWidth action Width
`0:object "Object" | 1:behavior[ResizableCapability::ResizableBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Width"`

Flippable objects - FlippableCapability

Actions/conditions for objects which can be flipped horizontally or vertically.

Behavior: FlippableCapability::FlippableBehavior

FlippableCapability::FlippableBehavior::FlippedX condition Horizontally flipped
`0:object "Object" | 1:behavior[FlippableCapability::FlippableBehavior] "Behavior"`

FlippableCapability::FlippableBehavior::FlippedY condition Vertically flipped
`0:object "Object" | 1:behavior[FlippableCapability::FlippableBehavior] "Behavior"`

FlippableCapability::FlippableBehavior::FlipX action Flip the object horizontally
`0:object "Object" | 1:behavior[FlippableCapability::FlippableBehavior] "Behavior" | 2:yesorno "Activate flipping"`

FlippableCapability::FlippableBehavior::FlipY action Flip the object vertically
`0:object "Object" | 1:behavior[FlippableCapability::FlippableBehavior] "Behavior" | 2:yesorno "Activate flipping"`

Objects containing a text - TextContainerCapability

Allows an object to contain a text, usually shown on screen, that can be modified.

Behavior: TextContainerCapability::TextContainerBehavior

TextContainerCapability::TextContainerBehavior::Value condition Text
`0:object "Object" | 1:behavior[TextContainerCapability::TextContainerBehavior] "Behavior" | 2:relationalOperator[string] "Sign of the test" | 3:string "Text"`

TextContainerCapability::TextContainerBehavior::SetValue action Text
`0:object "Object" | 1:behavior[TextContainerCapability::TextContainerBehavior] "Behavior" | 2:operator[string] "Modification's sign" | 3:string "Text"`

Value string expression Text
`0:object "Object" | 1:behavior[TextContainerCapability::TextContainerBehavior] "Behavior"`

Objects with effects - EffectCapability

Actions/conditions to enable/disable and change parameters of visual effects applied on objects.

Behavior: EffectCapability::EffectBehavior

EffectCapability::EffectBehavior::IsEffectEnabled condition Effect is enabled
`0:object "Object" | 1:behavior[EffectCapability::EffectBehavior] "Behavior" | 2:objectEffectName "Effect name"`

EffectCapability::EffectBehavior::EnableEffect action Enable an object effect
`0:object "Object" | 1:behavior[EffectCapability::EffectBehavior] "Behavior" | 2:objectEffectName "Effect name" | 3:yesorno "Enable?"`

EffectCapability::EffectBehavior::SetEffectBooleanParameter action Effect property (enable or disable)
`0:object "Object" | 1:behavior[EffectCapability::EffectBehavior] "Behavior" | 2:objectEffectName "Effect name" | 3:objectEffectParameterName "Property name" | 4:yesorno "Enable this property"`

EffectCapability::EffectBehavior::SetEffectDoubleParameter action Effect property (number)
`0:object "Object" | 1:behavior[EffectCapability::EffectBehavior] "Behavior" | 2:objectEffectName "Effect name" | 3:objectEffectParameterName "Property name" | 4:expression "New value"`

EffectCapability::EffectBehavior::SetEffectStringParameter action Effect property (string)
`0:object "Object" | 1:behavior[EffectCapability::EffectBehavior] "Behavior" | 2:objectEffectName "Effect name" | 3:objectEffectParameterName "Property name" | 4:string "New value"`

Platform behavior - PlatformBehavior

The platformer engine allows to create controllable objects that can run and jump on other objects that are marked as platforms. It supports various features commonly found in platformers: grabbing the edge of a platform, sustaining the jump while a key is held, customizable gravity... It can be used for the player, but also for other objects moving on platforms. In this case though, it's recommended to first check if there is a simpler behavior that could be used. Default controls for keyboards are included. For touch or gamepads, use the "Multitouch Joystick" objects and the associated "mapper" behaviors.

Free instructions

PlatformBehavior::IsObjectOnGivenFloor condition Character is on given platform
`0:objectList "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:objectList "Platforms" | 3:conditionInverted=""`

Behavior: PlatformBehavior::PlatformBehavior

PlatformBehavior::ChangePlatformType action Platform type
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformBehavior] "Behavior" | 2:stringWithSelector[["Platform","Jumpthru","Ladder"]] "Platform type"`

Behavior: PlatformBehavior::PlatformerObjectBehavior

PlatformBehavior::Acceleration condition Acceleration
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Acceleration to compare to (in pixels per second per second)"`

PlatformBehavior::CanGrabPlatforms condition Can grab platforms
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::CanJump condition Can jump
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::CurrentFallSpeed condition Current falling speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"

PlatformBehavior::CurrentJumpSpeed condition Current jump speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"

PlatformBehavior::CurrentSpeed condition Current horizontal speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"

PlatformBehavior::Deceleration condition Deceleration
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Deceleration to compare to (in pixels per second per second)"

PlatformBehavior::Gravity condition Gravity
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Gravity to compare to (in pixels per second per second)"

PlatformBehavior::IsFalling condition Is falling
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"

PlatformBehavior::IsGrabbingPlatform condition Is grabbing platform ledge
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"

PlatformBehavior::IsJumping condition Is jumping
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"

PlatformBehavior::IsOnFloor condition Is on floor
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"

PlatformBehavior::IsOnLadder condition Is on ladder
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"

PlatformBehavior::JumpSpeed condition Jump speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"

PlatformBehavior::JumpSustainTime condition Jump sustain time
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Duration to compare to (in seconds)"

PlatformBehavior::LadderClimbingSpeed condition Ladder climbing speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"

PlatformBehavior::MaxFallingSpeed condition Maximum falling speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Max speed to compare to (in pixels per second)"

PlatformBehavior::MaxSpeed condition Maximum horizontal speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Max speed to compare to (in pixels per second)"

PlatformBehavior::PlatformerObjectBehavior::IsMovingEvenALittle condition Is moving
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"

PlatformBehavior::PlatformerObjectBehavior::IsUsingControl condition Control pressed or simulated
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:stringWithSelector[["Left", "Right", "Jump", "Ladder", "Release Ladder", "Up", "Down"]] "Key"

PlatformBehavior::IsMoving condition Is moving **[legacy/hidden]**
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"

PlatformBehavior::Acceleration action Acceleration
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Acceleration (in pixels per second per second)"

PlatformBehavior::CanGrabPlatforms action Platform grabbing
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:yesorno "Can grab platforms"

PlatformBehavior::Deceleration action Deceleration
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Deceleration (in pixels per second per second)"

PlatformBehavior::Gravity action Gravity
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Gravity (in pixels per second per second)"

PlatformBehavior::IgnoreDefaultControls action Ignore default controls
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:yesorno "Ignore controls"

PlatformBehavior::JumpSpeed action Jump speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed (in pixels per second)"

PlatformBehavior::JumpSustainTime action Jump sustain time
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Duration (in seconds)"

PlatformBehavior::LadderClimbingSpeed action Ladder climbing speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed (in pixels per second)"

PlatformBehavior::MaxFallingSpeed action Maximum falling speed
0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Max speed (in pixels per second)" | 4:yesorno "If jumping, try to preserve the current speed in the air"

PlatformBehavior::MaxSpeed action Maximum horizontal speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Max speed (in pixels per second)"`

PlatformBehavior::PlatformerObjectBehavior::AbortJump action Abort jump
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::PlatformerObjectBehavior::SetCanNotAirJump action Forbid jumping again in the air
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::PlatformerObjectBehavior::SetCurrentFallSpeed action Current falling speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed (in pixels per second)"`

PlatformBehavior::PlatformerObjectBehavior::SetCurrentSpeed action Current horizontal speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed (in pixels per second)"`

PlatformBehavior::SetCanJump action Allow jumping again
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateControl action Simulate control
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior" | 2:stringWithSelector[["Left", "Right", "Jump", "Ladder", "Release Ladder", "Up", "Down"]] "Key"`

PlatformBehavior::SimulateDownKey action Simulate down key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateJumpKey action Simulate jump key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateLadderKey action Simulate ladder key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateLeftKey action Simulate left key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateReleaseLadderKey action Simulate release ladder key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateReleasePlatformKey action Simulate release platform key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateRightKey action Simulate right key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateUpKey action Simulate up key press
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

PlatformBehavior::SimulateReleaseKey action Simulate release platform key press **[legacy/hidden]**
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

Acceleration number expression Acceleration
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

CurrentFallSpeed number expression Current fall speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

CurrentJumpSpeed number expression Current jump speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

CurrentSpeed number expression Current horizontal speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

Deceleration number expression Deceleration
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

Gravity number expression Gravity
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

JumpSpeed number expression Jump speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

JumpSustainTime number expression Jump sustain time
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

LadderClimbingSpeed number expression Ladder climbing speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

MaxFallingSpeed number expression Maximum falling speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

MaxSpeed number expression Maximum horizontal speed
`0:object "Object" | 1:behavior[PlatformBehavior::PlatformerObjectBehavior] "Behavior"`

Top-down movement - TopDownMovementBehavior

Allows to move an object in either 4 or 8 directions, with the keyboard (default), a virtual stick (for this, also add the "Top-down multitouch controller mapper" behavior and a"Multitouch Joystick" object), gamepad or manually using events.

Behavior: TopDownMovementBehavior::TopDownMovementBehavior

TopDownMovementBehavior::Acceleration condition Acceleration
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Acceleration to compare to (in pixels per second per second)"`

TopDownMovementBehavior::AngleOffset condition Rotation offset
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Angle to compare to (in degrees)"`

TopDownMovementBehavior::AngularMaxSpeed condition Angular maximum speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Max angular speed to compare to (in degrees per second)"`

TopDownMovementBehavior::Deceleration condition Deceleration
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Deceleration to compare to (in pixels per second per second)"`

TopDownMovementBehavior::DiagonalsAllowed condition Diagonal movement
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

TopDownMovementBehavior::IsMoving condition Is moving
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

TopDownMovementBehavior::MaxSpeed condition Maximum speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Max speed to compare to (in pixels per second)"`

TopDownMovementBehavior::ObjectRotated condition Object rotated
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

TopDownMovementBehavior::Speed condition Speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"`

TopDownMovementBehavior::TopDownMovementBehavior::IsMovementAngleAround condition Angle of movement
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:expression "Angle (in degrees)" | 3:expression "Tolerance (in degrees)"`

TopDownMovementBehavior::TopDownMovementBehavior::IsUsingControl condition Control pressed or simulated
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:stringWithSelector[["Left", "Right", "Up", "Down", "Stick"]] "Key"`

TopDownMovementBehavior::TopDownMovementBehavior::MovementAngleOffset condition Movement angle offset
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Angle (in degrees)"`

TopDownMovementBehavior::XVelocity condition Speed on X axis
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"`

TopDownMovementBehavior::YVelocity condition Speed on Y axis
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"`

TopDownMovementBehavior::Angle condition Angle of movement **[legacy/hidden]**
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Angle to compare to (in degrees)"`

TopDownMovementBehavior::Acceleration action Acceleration
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Acceleration (in pixels per second per second)"`

TopDownMovementBehavior::AllowDiagonals action Diagonal movement
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:yesorno "Allow?"`

TopDownMovementBehavior::AngleOffset action Rotation offset
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Angle (in degrees)"`

TopDownMovementBehavior::AngularMaxSpeed action Angular maximum speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Max angular speed (in degrees per second)"`

TopDownMovementBehavior::Deceleration action Deceleration
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Deceleration (in pixels per second per second)"`

TopDownMovementBehavior::IgnoreDefaultControls action Ignore default controls
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:yesorno "Ignore controls"`

TopDownMovementBehavior::MaxSpeed action Maximum speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Max speed (in pixels per second)"`

TopDownMovementBehavior::RotateObject action Rotate the object
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:yesorno "Rotate object?"`

TopDownMovementBehavior::SimulateControl action Simulate control
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:stringWithSelector[["Left", "Right", "Up", "Down"]] "Key"`

TopDownMovementBehavior::SimulateDownKey action Simulate down key press
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

TopDownMovementBehavior::SimulateLeftKey action Simulate left key press
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

TopDownMovementBehavior::SimulateRightKey action Simulate right key press
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

TopDownMovementBehavior::SimulateStick action Simulate stick control
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:expression "Stick angle (in degrees)" | 3:expression "Stick force (between 0 and 1)"`

TopDownMovementBehavior::SimulateUpKey action Simulate up key press
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

TopDownMovementBehavior::TopDownMovementBehavior::SetMovementAngleOffset action Movement angle offset
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Angle (in degrees)"`

TopDownMovementBehavior::TopDownMovementBehavior::SetVelocityX action Speed on the X axis
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed (in pixels per second)"`

TopDownMovementBehavior::TopDownMovementBehavior::SetVelocityY action Speed on the Y axis
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed (in pixels per second)"`

Acceleration number expression Acceleration
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

Angle number expression Angle of the movement
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

AngleOffset number expression Rotation offset
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

AngularMaxSpeed number expression Angular maximum speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

Deceleration number expression Deceleration
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

MaxSpeed number expression Maximum speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

MovementAngleOffset number expression Movement angle offset
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

Speed number expression Speed
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

StickAngle number expression Stick angle
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

XVelocity number expression Speed on the X axis
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

YVelocity number expression Speed on the Y axis
`0:object "Object" | 1:behavior[TopDownMovementBehavior::TopDownMovementBehavior] "Behavior"`

Draggable Behavior - DraggableBehavior

Allows objects to be moved using the mouse (or touch). Add the behavior to an object to make it draggable. Use events to enable or disable the behavior when needed.

Behavior: DraggableBehavior::Draggable

DraggableBehavior::Dragged condition Being dragged
`0:object "Object" | 1:behavior[DraggableBehavior::Draggable] "Behavior"`

DraggableBehavior::Dropped condition Was just dropped
`0:object "Object" | 1:behavior[DraggableBehavior::Draggable] "Behavior"`

Destroy Outside Screen Behavior - DestroyOutsideBehavior

This behavior can be used to destroy objects when they go outside of the bounds of the 2D camera. Useful for 2D bullets or other short-lived objects. Don't use it for 3D objects in a FPS/TPS game or any game with a camera not being a top view (for 3D objects, prefer comparing the position, for example Z position to see if an object goes outside of the bound of the map). If the object appears outside of the screen, it's not removed unless it goes beyond the unseen object grace distance.

Behavior: DestroyOutsideBehavior::DestroyOutside

DestroyOutsideBehavior::DestroyOutside::ExtraBorder condition Additional border (extra distance before deletion)
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

DestroyOutsideBehavior::DestroyOutside::UnseenGraceDistance condition Unseen object grace distance
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

DestroyOutsideBehavior::ExtraBorder condition Additional border (extra distance before deletion) **[legacy/hidden]**
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

DestroyOutsideBehavior::DestroyOutside::SetExtraBorder action Additional border (extra distance before deletion)
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"`

DestroyOutsideBehavior::DestroyOutside::SetUnseenGraceDistance action Unseen object grace distance
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"`

DestroyOutsideBehavior::ExtraBorder action Additional border (extra distance before deletion) **[legacy/hidden]**
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"`

ExtraBorder number expression Additional border (extra distance before deletion)
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior"`

UnseenGraceDistance number expression Unseen object grace distance
`0:object "Object" | 1:behavior[DestroyOutsideBehavior::DestroyOutside] "Behavior"`

Anchor - AnchorBehavior

Anchor objects to the window's bounds.

Tiled Sprite Object - TiledSpriteObject

Displays a 2D image in a repeating pattern over an area. Useful for making backgrounds, including background that are scrolling when the camera moves. This is more performant than using multiple Sprite objects.

Object: TiledSpriteObject::TiledSprite

TiledSpriteObject::XOffset condition Image X Offset
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TiledSpriteObject::YOffset condition Image Y Offset
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TiledSpriteObject::Angle condition Angle **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TiledSpriteObject::Height condition Height **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TiledSpriteObject::Opacity condition Opacity **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Opacity to compare to (0-255)"`

TiledSpriteObject::Width condition Width **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

TiledSpriteObject::SetColor action Tint color
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:color "Tint"`

TiledSpriteObject::SetImageFromResource action Image
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:imageResource "Image resource" | 2:currentScene=""`

TiledSpriteObject::XOffset action Image X Offset
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TiledSpriteObject::YOffset action Image Y Offset
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TiledSpriteObject::Angle action Angle **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TiledSpriteObject::Height action Height **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

TiledSpriteObject::SetOpacity action Change Tiled Sprite opacity **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Opacity (0-255)"`

TiledSpriteObject::SetSize action Size **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:expression "Width" | 2:expression "Height"`

TiledSpriteObject::Width action Width **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

XOffset number expression Image X Offset
`0:object[TiledSpriteObject::TiledSprite] "Object"`

YOffset number expression Image Y Offset
`0:object[TiledSpriteObject::TiledSprite] "Object"`

Opacity number expression Opacity **[legacy/hidden]**
`0:object[TiledSpriteObject::TiledSprite] "Object"`

Panel Sprite (9-patch) Object - PanelSpriteObject

Panel Sprite, also called 9-patch, is an object showing an image that can be resized by stretching or repeating the edges and corners as well as the filling.

Object: PanelSpriteObject::PanelSprite

PanelSpriteObject::Angle condition Angle **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle to compare to (in degrees)"`

PanelSpriteObject::Height condition Height **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

PanelSpriteObject::Opacity condition Opacity **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Opacity to compare to (0-255)"`

PanelSpriteObject::Width condition Width **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

PanelSpriteObject::SetColor action Tint color
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:color "Tint"`

PanelSpriteObject::SetImageFromResource action Image name
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:imageResource "Image file (or image resource name)" | 2:currentScene=""`

PanelSpriteObject::Angle action Angle **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

PanelSpriteObject::Height action Height **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

PanelSpriteObject::Image action Image name (deprecated) **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:string "Image name" | 2:currentScene=""`

PanelSpriteObject::SetOpacity action Change Panel Sprite opacity **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Opacity (0-255)"`

PanelSpriteObject::Width action Width **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

Opacity number expression Opacity **[legacy/hidden]**
`0:object[PanelSpriteObject::PanelSprite] "Panel Sprite"`

Shape painter - PrimitiveDrawing

An object that can be used to draw arbitrary 2D shapes on the screen using events.

Object: PrimitiveDrawing::Drawer

PrimitiveDrawing::AreCoordinatesRelative condition Relative coordinates
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

PrimitiveDrawing::CheckAntialiasing condition Anti-aliasing type
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:stringWithSelector[["none","low","medium","high"]] "Type of anti-aliasing to check the object against"`

PrimitiveDrawing::ClearBetweenFrames condition Clear between frames
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

PrimitiveDrawing::FillOpacity condition Fill opacity
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Opacity to compare to (0-255)"`

PrimitiveDrawing::OutlineOpacity condition Outline opacity
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Opacity to compare to (0-255)"`

PrimitiveDrawing::OutlineSize condition Outline size
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

PrimitiveDrawing::Drawer::ScaleX condition Scale on X axis **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Scale (1 by default)"`

PrimitiveDrawing::Drawer::ScaleY condition Scale on Y axis **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Scale (1 by default)"`

PrimitiveDrawing::FlippedX condition Horizontally flipped **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object"`

PrimitiveDrawing::FlippedY condition Vertically flipped **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object"`

PrimitiveDrawing::ScaleX condition **[legacy/hidden]**

PrimitiveDrawing::ScaleY condition **[legacy/hidden]**

PrimitiveDrawing::Arc action Arc
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of center" | 2:expression "Y position of center" | 3:expression "Radius (in pixels)" | 4:expression "Start angle of the arc (in degrees)" | 5:expression "End angle of the arc (in degrees)" | 6:yesorno "Anticlockwise" | 7:yesorno "Close path"`

PrimitiveDrawing::BeginFillPath action Begin fill path
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "Start drawing x" | 2:expression "Start drawing y"`

PrimitiveDrawing::BezierCurve action Bezier curve
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of start point" | 2:expression "Y position of start point" | 3:expression "First control point x" | 4:expression "First control point y" | 5:expression "Second Control point x" | 6:expression "Second Control point y" | 7:expression "Destination point x" | 8:expression "Destination point y"`

PrimitiveDrawing::ChamferRectangle action Chamfer Rectangle
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "Left X position" | 2:expression "Top Y position" | 3:expression "Right X position" | 4:expression "Bottom Y position" | 5:expression "Chamfer (in pixels)"`

PrimitiveDrawing::Circle action Circle
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of center" | 2:expression "Y position of center" | 3:expression "Radius (in pixels)"`

PrimitiveDrawing::ClearBetweenFrames action Clear between frames
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:yesorno? "Clear between each frame" default yes`

PrimitiveDrawing::Drawer::ClearShapes action Clear shapes
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

PrimitiveDrawing::Ellipse action Ellipse
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of center" | 2:expression "Y position of center" | 3:expression "The width of the ellipse" | 4:expression "The height of the ellipse"`

PrimitiveDrawing::EndFillPath action End fill path
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

PrimitiveDrawing::FillColor action Fill color
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:color "Fill color"`

PrimitiveDrawing::FillOpacity action Fill opacity
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:operator[number] "Modification's sign" | 2:number "Opacity (0-255)"`

PrimitiveDrawing::FilletRectangle action Fillet Rectangle
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "Left X position" | 2:expression "Top Y position" | 3:expression "Right X position" | 4:expression "Bottom Y position" | 5:expression "Fillet (in pixels)"`

PrimitiveDrawing::LineV2 action Line
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of start point" | 2:expression "Y position of start point" | 3:expression "X position of end point" | 4:expression "Y position of end point" | 5:expression "Thickness (in pixels)"`

PrimitiveDrawing::MovePathTo action Move path drawing position
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of start point" | 2:expression "Y position of start point"`

PrimitiveDrawing::OutlineColor action Outline color
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:color "Color"`

PrimitiveDrawing::OutlineOpacity action Outline opacity
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:operator[number] "Modification's sign" | 2:number "Opacity (0-255)"`

PrimitiveDrawing::OutlineSize action Outline size
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

PrimitiveDrawing::PathArc action Path arc
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "Center x of circle" | 2:expression "Center y of circle" | 3:expression "Radius (in pixels)" | 4:expression "Start angle" | 5:expression "End angle" | 6:yesorno "Anticlockwise"`

PrimitiveDrawing::PathBezierCurveTo action Path bezier curve
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "First control point x" | 2:expression "First control point y" | 3:expression "Second Control point x" | 4:expression "Second Control point y" | 5:expression "Destination point x" | 6:expression "Destination point y"`

PrimitiveDrawing::PathLineTo action Path line
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of start point" | 2:expression "Y position of start point"`

PrimitiveDrawing::PathQuadraticCurveTo action Path quadratic curve
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "Control point x" | 2:expression "Control point y" | 3:expression "Destination point x" | 4:expression "Destination point y"`

PrimitiveDrawing::QuadraticCurve action Quadratic curve
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of start point" | 2:expression "Y position of start point" | 3:expression "Control point x" | 4:expression "Control point y" | 5:expression "Destination point x" | 6:expression "Destination point y"`

PrimitiveDrawing::Rectangle action Rectangle
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "Left X position" | 2:expression "Top Y position" | 3:expression "Right X position" | 4:expression "Bottom Y position"`

PrimitiveDrawing::RegularPolygon action Regular Polygon
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of center" | 2:expression "Y position of center" | 3:expression "Number of sides of the polygon (minimum: 3)" | 4:expression "Radius (in pixels)" | 5:expression "Rotation (in degrees)"`

PrimitiveDrawing::RoundedRectangle action Rounded rectangle
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "Left X position" | 2:expression "Top Y position" | 3:expression "Right X position" | 4:expression "Bottom Y position" | 5:expression "Radius (in pixels)"`

PrimitiveDrawing::SetAntialiasing action Anti-aliasing
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:stringWithSelector[["none","low","medium","high"]] "Anti-aliasing quality level"`

PrimitiveDrawing::SetRectangularCollisionMask action Collision Mask
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:expression "Left X position" | 2:expression "Top Y position" | 3:expression "Right X position" | 4:expression "Bottom Y position"`

PrimitiveDrawing::SetRotationCenter action Center of rotation
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:expression "X position" | 2:expression "Y position"`

PrimitiveDrawing::Star action Star
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of center" | 2:expression "Y position of center" | 3:expression "Number of points of the star (minimum: 2)" | 4:expression "Radius (in pixels)" | 5:expression "Inner radius (in pixels, half radius by default)" | 6:expression "Rotation (in degrees)"`

PrimitiveDrawing::Torus action Torus
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of center" | 2:expression "Y position of center" | 3:expression "Inner Radius (in pixels)" | 4:expression "Outer Radius (in pixels)" | 5:expression "Start Arc (in degrees)" | 6:expression "End Arc (in degrees)"`

PrimitiveDrawing::UseRelativeCoordinates action Use relative coordinates
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:yesorno "Use relative coordinates?" default true`

PrimitiveDrawing::closePath action Close Path
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

PrimitiveDrawing::ClosePath action **[legacy/hidden]**

PrimitiveDrawing::Drawer::SetScaleX action Scale on X axis **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

PrimitiveDrawing::Drawer::SetScaleY action Scale on Y axis **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

PrimitiveDrawing::FlipX action Flip the object horizontally **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:yesorno "Activate flipping"`

PrimitiveDrawing::FlipY action Flip the object vertically **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:yesorno "Activate flipping"`

PrimitiveDrawing::Height action Height **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

PrimitiveDrawing::Line action Line **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object" | 1:expression "X position of start point" | 2:expression "Y position of start point" | 3:expression "X position of end point" | 4:expression "Y position of end point" | 5:expression "Thickness (in pixels)"`

PrimitiveDrawing::PathMoveTo action **[legacy/hidden]**

PrimitiveDrawing::Scale action Scale **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:operator[number] "Modification's sign" | 2:number "Scale (1 by default)"`

PrimitiveDrawing::Width action Width **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

FillColorBlue number expression Filing color blue component
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

FillColorGreen number expression Filing color green component
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

FillColorRed number expression Filing color red component
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

FillOpacity number expression Filling opacity
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

OutlineColorBlue number expression Outline color blue component
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

OutlineColorGreen number expression Outline color green component
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

OutlineColorRed number expression Outline color red component
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

OutlineOpacity number expression Outline opacity
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

OutlineSize number expression Outline size
`0:object[PrimitiveDrawing::Drawer] "Shape Painter object"`

ToDrawingX number expression X drawing coordinate of a point from the scene
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:expression "X scene position" | 2:expression "Y scene position"`

ToDrawingY number expression Y drawing coordinate of a point from the scene
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:expression "X scene position" | 2:expression "Y scene position"`

ToSceneX number expression X scene coordinate of a point from the drawing
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:expression "X drawing position" | 2:expression "Y drawing position"`

ToSceneY number expression Y scene coordinate of a point from the drawing
`0:object[PrimitiveDrawing::Drawer] "Object" | 1:expression "X drawing position" | 2:expression "Y drawing position"`

PrimitiveDrawing::ScaleX number expression **[legacy/hidden]**

PrimitiveDrawing::ScaleY number expression **[legacy/hidden]**

ScaleX number expression Scale on X axis **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object"`

ScaleY number expression Scale on Y axis **[legacy/hidden]**
`0:object[PrimitiveDrawing::Drawer] "Object"`

Antialiasing string expression Type of anti-aliasing used by a shape painter
`0:object[PrimitiveDrawing::Drawer] "Object"`

Particle system - ParticleSystem

A 2D particle emitter allows to create various effects by showing a lot of tiny images called particles. It's ideal for fires, smoke, explosions, magical effects, etc... in 2D games. For 3D games, use the 3D particle emitter instead.

Object: ParticleSystem::ParticleEmitter

ParticleSystem::ConeSprayAngle condition Angle of the spray cone
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle to compare to (in degrees)"`

ParticleSystem::EmitterAngle condition Emission angle
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle to compare to (in degrees)"`

ParticleSystem::Flow condition Flow
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Flow to compare to (in particles per second)"`

ParticleSystem::NoMoreParticles condition No more particles
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleSystem::ParticleAlpha1 condition Start opacity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleAlpha2 condition End opacity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleBlue1 condition Start color blue component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleBlue2 condition End color blue component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleEmitter::AdditiveRendering condition Activate particles additive rendering
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleSystem::ParticleEmitter::MaxParticlesCount condition Number of displayed particles
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::ParticleEmitter::ParticleRotationMaxSpeed condition Particle rotation max speed
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angular speed (in degrees per second)"`

ParticleSystem::ParticleEmitter::ParticleRotationMinSpeed condition Particle rotation min speed
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angular speed (in degrees per second)"`

ParticleSystem::ParticleGravityAngle condition Gravity angle
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Angle to compare to (in degrees)"`

ParticleSystem::ParticleGravityLength condition Gravity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::ParticleGravityX condition Gravity value on X axis
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::ParticleGravityY condition Gravity value on Y axis
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::ParticleGreen1 condition Start color green component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleGreen2 condition End color green component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleLifeTimeMax condition Maximum lifetime
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Duration to compare to (in seconds)"`

ParticleSystem::ParticleLifeTimeMin condition Minimum lifetime
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Duration to compare to (in seconds)"`

ParticleSystem::ParticleRed1 condition Start color red component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleRed2 condition End color red component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare to (0-255)"`

ParticleSystem::ParticleSize1 condition Start size
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::ParticleSize2 condition End size
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::Tank condition Capacity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Capacity to compare to"`

ParticleSystem::Texture condition Image
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[string] "Sign of the test" | 2:string "Value to compare"`

ParticleSystem::ZoneRadius condition Creation radius
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::EmitterAngleA condition Emission angle 1 **[legacy/hidden]**
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::EmitterAngleB condition Emission angle 2 **[legacy/hidden]**
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

ParticleSystem::ConeSprayAngle action Angle of the spray cone
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

ParticleSystem::EmitterAngle action Emission angle
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

ParticleSystem::EmitterForceMax action Emission maximal force
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::EmitterForceMin action Emission minimal force
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::JumpEmitterForwardInTime action Jump emitter forward in time
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:number "Seconds of time"`

ParticleSystem::ParticleAlpha1 action Start opacity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleAlpha2 action End opacity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleBlue1 action Start color blue component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleBlue2 action End color blue component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleColor1 action Start color
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:color "Start color"`

ParticleSystem::ParticleColor2 action End color
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:color "End color"`

ParticleSystem::ParticleEmitter::SetAdditiveRendering action Activate particles additive rendering
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:yesorno "New value"`

ParticleSystem::ParticleEmitter::SetMaxParticlesCount action Number of displayed particles
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::ParticleEmitter::SetParticleRotationMaxSpeed action Particle rotation max speed
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angular speed (in degrees per second)"`

ParticleSystem::ParticleEmitter::SetParticleRotationMinSpeed action Particle rotation min speed
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angular speed (in degrees per second)"`

ParticleSystem::ParticleGravityAngle action Gravity angle
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Angle (in degrees)"`

ParticleSystem::ParticleGravityLength action Gravity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::ParticleGravityX action Gravity value on X axis
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::ParticleGravityY action Gravity value on Y axis
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::ParticleGreen1 action Start color green component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleGreen2 action End color green component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleLifeTimeMax action Maximum lifetime
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Duration (in seconds)"`

ParticleSystem::ParticleLifeTimeMin action Minimum lifetime
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Duration (in seconds)"`

ParticleSystem::ParticleRed1 action Start color red component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleRed2 action End color red component
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value (0-255)"`

ParticleSystem::ParticleSize1 action Start size
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::ParticleSize2 action End size
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::RecreateParticleSystem action Recreate particles
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleSystem::SetFlow action Flow
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Flow (in particles per second)"`

ParticleSystem::SetTank action Capacity
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Capacity"`

ParticleSystem::SetTextureFromResource action Particle image
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:imageResource "Image file (or image resource name)" | 2:currentScene=""`

ParticleSystem::StartEmission action Start emission
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleSystem::StopEmission action Stop emission
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleSystem::ZoneRadius action Creation radius
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::EmitterAngleA action Emission angle 1 **[legacy/hidden]**
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::EmitterAngleB action Emission angle 2 **[legacy/hidden]**
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::Flow action Flow **[legacy/hidden]**
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::Tank action Capacity **[legacy/hidden]**
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:operator[number] "Modification's sign" | 2:number "Value"`

ParticleSystem::Texture action Particle image (deprecated) **[legacy/hidden]**
`0:object[ParticleSystem::ParticleEmitter] "Object" | 1:string "Image to use" | 2:currentScene=""`

ConeSprayAngle number expression Angle of the spray cone
`0:object[ParticleSystem::ParticleEmitter] "Object"`

CurrentParticleCount number expression Particles count
`0:object[ParticleSystem::ParticleEmitter] "Object"`

EmitterAngle number expression Emission angle
`0:object[ParticleSystem::ParticleEmitter] "Object"`

EmitterForceMax number expression Emission maximal force
`0:object[ParticleSystem::ParticleEmitter] "Object"`

EmitterForceMin number expression Emission minimal force
`0:object[ParticleSystem::ParticleEmitter] "Object"`

Flow number expression Flow
`0:object[ParticleSystem::ParticleEmitter] "Object"`

MaxParticlesCount number expression Number of displayed particles
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleAlpha1 number expression Start opacity
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleAlpha2 number expression End opacity
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleBlue1 number expression Start color blue component
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleBlue2 number expression End color blue component
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleGravityAngle number expression Gravity angle
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleGravityLength number expression Gravity
`0:object[ParticleSystem::ParticleEmitter] "Object"`

ParticleGravityX number expression X gravity
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleGravityY number expression Y gravity
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleGreen1 number expression Start color green component
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleGreen2 number expression End color green component
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleLifeTimeMax number expression Maximum lifetime of particles
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleLifeTimeMin number expression Minimum lifetime of particles
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleRed1 number expression Start color red component
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleRed2 number expression End color red component
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleRotationMaxSpeed number expression Particle rotation max speed
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleRotationMinSpeed number expression Particle rotation min speed
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleSize1 number expression Start size
0:object[ParticleSystem::ParticleEmitter] "Object"

ParticleSize2 number expression End size
0:object[ParticleSystem::ParticleEmitter] "Object"

RendererParam1 number expression Rendering first parameter
0:object[ParticleSystem::ParticleEmitter] "Object"

RendererParam2 number expression Rendering second parameter
0:object[ParticleSystem::ParticleEmitter] "Object"

Tank number expression Capacity
0:object[ParticleSystem::ParticleEmitter] "Object"

ZoneRadius number expression Radius of emission zone
0:object[ParticleSystem::ParticleEmitter] "Object"

EmitterAngleA number expression Emission angle A **[legacy/hidden]**
0:object[ParticleSystem::ParticleEmitter] "Object"

EmitterAngleB number expression Emission angle B **[legacy/hidden]**
0:object[ParticleSystem::ParticleEmitter] "Object"

NbParticles number expression Particles number **[legacy/hidden]**
0:object[ParticleSystem::ParticleEmitter] "Object"

Texture string expression Particles image
0:object[ParticleSystem::ParticleEmitter] "Object"

Text entry object - TextEntryObject

Deprecated object that can be used to capture the text entered with a keyboard by a player.

Object: TextEntryObject::TextEntry

TextEntryObject::Activated condition Text input
0:object[TextEntryObject::TextEntry] "Object"

TextEntryObject::String condition Text in memory
0:object[TextEntryObject::TextEntry] "Object" | 1:relationalOperator[string] "Sign of the test" | 2:string "Text to compare to"

TextEntryObject::Activate action De/activate capturing text input
0:object[TextEntryObject::TextEntry] "Object" | 1:yesorno "Activate"

TextEntryObject::String action Text in memory
0:object[TextEntryObject::TextEntry] "Object" | 1:operator[string] "Modification's sign" | 2:string "Text"

String string expression Text entered with keyboard
0:object[TextEntryObject::TextEntry] "Object"

Pathfinding behavior - PathfindingBehavior

Pathfinding allows to compute an efficient path for objects, avoiding obstacles on the way.

Behavior: PathfindingBehavior::PathfindingBehavior

PathfindingBehavior::Acceleration condition Acceleration
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"

PathfindingBehavior::AngleOffset condition Rotation offset
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Angle to compare to (in degrees)"

PathfindingBehavior::AngularMaxSpeed condition Angular maximum speed
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Max angular speed to compare to (in degrees per second)"

PathfindingBehavior::CellHeight condition Height of the virtual grid
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"

PathfindingBehavior::CellWidth condition Width of the virtual grid
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"

PathfindingBehavior::DestinationReached condition Destination reached
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"

PathfindingBehavior::DiagonalsAllowed condition Diagonal movement
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"

PathfindingBehavior::ExtraBorder condition Extra border
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"

PathfindingBehavior::MaxSpeed condition Maximum speed
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Max speed to compare to (in pixels per second)"

PathfindingBehavior::ObjectRotated condition Object rotated
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"

PathfindingBehavior::PathFound condition Path found
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"

PathfindingBehavior::PathfindingBehavior::GridOffsetX condition Grid X offset
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"

PathfindingBehavior::PathfindingBehavior::GridOffsetY condition Grid Y offset
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"

PathfindingBehavior::PathfindingBehavior::MovementAngleIsAround condition Angle of movement on its path
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:expression "Angle, in degrees" | 3:expression "Tolerance, in degrees"

PathfindingBehavior::Speed condition Speed on its path
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Speed to compare to (in pixels per second)"

PathfindingBehavior::Acceleration action Acceleration
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"

PathfindingBehavior::AllowDiagonals action Diagonal movement
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:yesorno "Allow?"

PathfindingBehavior::AngleOffset action Rotation offset
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Angle (in degrees)"

PathfindingBehavior::AngularMaxSpeed action Angular maximum speed
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Max angular speed (in degrees per second)"

PathfindingBehavior::CellHeight action Height of the cells
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"

PathfindingBehavior::CellWidth action Width of the cells
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"

PathfindingBehavior::ExtraBorder action Extra border
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"

PathfindingBehavior::MaxSpeed action Maximum speed
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Max speed (in pixels per second)"

PathfindingBehavior::PathfindingBehavior::SetGridOffsetX action Grid X offset
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"

PathfindingBehavior::PathfindingBehavior::SetGridOffsetY action Grid Y offset
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"

PathfindingBehavior::RotateObject action Rotate the object
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:yesorno "Rotate object?"

PathfindingBehavior::SetDestination action Move to a position
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:currentScene="" | 3:expression "Destination X position" | 4:expression "Destination Y position"

PathfindingBehavior::Speed action Speed
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Speed (in pixels per second)"

Acceleration number expression Acceleration
0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"

AngleOffset number expression Rotation offset
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

AngularMaxSpeed number expression Angular maximum speed
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

CellHeight number expression Height of a cell
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

CellWidth number expression Width of a cell
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

DestinationX number expression Destination X position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

DestinationY number expression Destination Y position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

ExtraBorder number expression Extra border size
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

GetNodeX number expression Get a waypoint X position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:expression "Node index (start at 0!)"`

GetNodeY number expression Get a waypoint Y position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior" | 2:expression "Node index (start at 0!)"`

GridOffsetX number expression Grid X offset
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

GridOffsetY number expression Grid Y offset
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

LastNodeX number expression Last waypoint X position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

LastNodeY number expression Last waypoint Y position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

MaxSpeed number expression Maximum speed
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

MovementAngle number expression Angle of movement on its path
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

NextNodeIndex number expression Index of the next waypoint
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

NextNodeX number expression Get next waypoint X position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

NextNodeY number expression Get next waypoint Y position
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

NodeCount number expression Waypoint count
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

Speed number expression Speed
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingBehavior] "Behavior"`

Behavior: PathfindingBehavior::PathfindingObstacleBehavior

PathfindingBehavior::Cost condition Cost
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingObstacleBehavior] "Behavior" | 2:relationalOperator[number] "Sign of the test" | 3:number "Value to compare"`

PathfindingBehavior::IsImpassable condition Impassable obstacle
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingObstacleBehavior] "Behavior"`

PathfindingBehavior::Cost action Cost
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingObstacleBehavior] "Behavior" | 2:operator[number] "Modification's sign" | 3:number "Value"`

PathfindingBehavior::SetImpassable action Should object be impassable
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingObstacleBehavior] "Behavior" | 2:yesorno "Impassable"`

Cost number expression Cost
`0:object "Object" | 1:behavior[PathfindingBehavior::PathfindingObstacleBehavior] "Behavior"`

Linked objects - LinkedObjects

These actions and conditions allow to virtually link two objects. It's then useful in the events to quickly retrieve one or more objects attached to another. For example, this can be used to link some equipment objects with the character holding them. More generally, this can be used to store and retrieve objects in a way that is more efficient than using variables.

Free instructions

LinkedObjects::PickObjectsLinkedTo condition Take into account linked objects
`0:currentScene="" | 1:objectList "Pick these objects..." | 2:objectPtr "...if they are linked to this object" | 3:eventsFunctionContext=""`

LinkedObjects::LinkObjects action Link two objects
`0:currentScene="" | 1:objectPtr "Object 1" | 2:objectPtr "Object 2"`

LinkedObjects::PickObjectsLinkedTo action Take into account linked objects
`0:currentScene="" | 1:objectList "Pick these objects..." | 2:objectPtr "...if they are linked to this object" | 3:eventsFunctionContext=""`

LinkedObjects::RemoveAllLinksOf action Unlink all objects from an object
`0:currentScene="" | 1:objectPtr "Object"`

LinkedObjects::RemoveLinkBetween action Unlink two objects
`0:currentScene="" | 1:objectPtr "Object 1" | 2:objectPtr "Object 2"`

Inventories - Inventory

Actions and conditions to store named inventories in memory, with items (indexed by their name), a count for each of them, a maximum count and an equipped state. Can be loaded/saved from/to a GDevelop variable.

Free instructions

Inventory::Count condition Item count
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name" | 3:relationalOperator[number] "Sign of the test" | 4:number "Value to compare"`

Inventory::Has condition Has an item
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name"`

Inventory::IsEquipped condition Item equipped
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name"`

Inventory::IsFull condition Item full
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name"`

Inventory::Add action Add an item
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name"`

Inventory::Equip action Equip an item
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name" | 3:yesorno "Equip?"`

Inventory::Remove action Remove an item
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name"`

Inventory::SerializeToVariable action Save an inventory in a scene variable
`0:currentScene="" | 1:string "Inventory name" | 2:scenevar "Scene variable"`

Inventory::SetMaximum action Set a maximum count for an item
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name" | 3:expression "Maximum count"`

Inventory::SetUnlimited action Set unlimited count for an item
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name" | 3:yesorno "Allow an unlimited amount?"`

Inventory::UnserializeFromVariable action Load an inventory from a scene variable
`0:currentScene="" | 1:string "Inventory name" | 2:scenevar "Scene variable"`

Inventory::Count number expression Item count
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name"`

Inventory::Maximum number expression Item maximum
`0:currentScene="" | 1:string "Inventory name" | 2:string "Item name"`

System information - SystemInfo

Conditions to check if the device has a touchscreen, is a mobile, or if the game runs as a preview.

Free instructions

SystemInfo::HasTouchScreen condition Device has a touchscreen
`0:currentScene=""`

SystemInfo::IsMobile condition Is a mobile device

SystemInfo::IsNativeDesktopApp condition Is a native desktop app
`0:currentScene=""`

SystemInfo::IsNativeMobileApp condition Is a native mobile app

SystemInfo::IsPreview condition Is the game running as a preview
`0:currentScene=""`

SystemInfo::IsWebGLSupported condition Is WebGL supported
`0:currentScene=""`

Network - BuiltinNetwork

Actions to send web requests, communicate with external "APIs" and other network related tasks. Also contains an action to open a URL on the device browser.

Free instructions

EnableMetrics action Enable (or disable) metrics collection
`0:currentScene="" | 1:yesorno "Enable the metrics?"`

LaunchFile action Open a URL (web page) or a file
`0:string "URL (or filename)" | 1:currentScene=""`

SendAsyncRequest action Send a request to a web page
`0:string "URL (API or web-page address)" | 1:string "Request body content" | 2:stringWithSelector[["GET", "POST", "PUT", "HEAD", "DELETE", "PATCH", "OPTIONS"]] "Request method" default "GET" | 3:string? "Content type" | 4:scenevar? "Variable where to store the response" | 5:scenevar? "Variable where to store the error message"`

SendRequest action Send a request to a web page **[legacy/hidden]**
`0:string "Host, with protocol" | 1:string "Path" | 2:string "Request body content" | 3:string? "Method: "POST" or "GET"" default "GET" | 4:string? "Content type" | 5:scenevar? "Response scene variable"`

Event functions - BuiltinAdvanced

Advanced control features for functions made with events.

Free instructions

CompareArgumentAsNumber condition Compare function parameter value **[legacy/hidden]**
`0:functionParameterName[number,string,boolean] "Parameter name" | 1:relationalOperator[number] "Sign of the test" | 2:number "Value to compare"`

CompareArgumentAsString condition Compare function parameter text **[legacy/hidden]**
`0:functionParameterName[number,string,boolean] "Parameter name" | 1:relationalOperator[string] "Sign of the test" | 2:string "Value to compare"`

GetArgumentAsBoolean condition Check if a function parameter is set to true (or yes) **[legacy/hidden]**
`0:functionParameterName[number,string,boolean] "Parameter name"`

CopyArgumentToVariable2 action Copy function parameter to variable
`0:functionParameterName[variable] "Parameter name" | 1:variable "Variable"`

CopyVariableToArgument2 action Copy variable to function parameter
`0:functionParameterName[variable] "Parameter name" | 1:variable "Variable"`

SetReturnBoolean action Set condition return value
`0:trueorfalse "Should the condition be true or false?"`

SetReturnNumber action Set number return value
`0:expression "The number to be returned"`

SetReturnString action Set text return value
`0:string "The text to be returned"`

CopyArgumentToVariable action Copy function parameter to variable **[legacy/hidden]**
`0:functionParameterName[variable] "Parameter name" | 1:scenevar "Scene variable"`

CopyVariableToArgument action Copy variable to function parameter **[legacy/hidden]**
`0:functionParameterName[variable] "Parameter name" | 1:scenevar "Scene variable"`

GetArgumentAsNumber number expression Get function parameter value **[legacy/hidden]**
`0:functionParameterName[number,string,boolean] "Parameter name"`

GetArgumentAsString string expression Get function parameter text **[legacy/hidden]**
`0:functionParameterName[number,string,boolean] "Parameter name"`

External layouts - BuiltinExternalLayouts

External layouts hold instances placed apart from a scene, using the objects and layers of the scene they are associated with. Their instances are not created when the scene starts: the scene creates them when needed, with the action to create objects from an external layout, at their editor positions (offset by an origin), on their layer, with their size, angle, Z order, visibility and instance variables. The objects and layers used must exist in the scene creating them (any external layout of the project can be loaded, not only the ones associated with the scene). Nothing is removed first: creating the same external layout again duplicates its objects, so delete the previous instances (for example the previous level) before loading new ones. Typical uses: the levels of a game built in a single scene (one external layout per level, loaded at the beginning of the scene according to a variable), a room or an enemy wave spawned on demand. For a UI panel, a custom object is usually a better fit (it is reusable, resizable and can hold its own logic); use an external layout for a simple, one-off UI screen.

Free instructions

BuiltinExternalLayouts::CreateObjectsFromExternalLayout action Create objects from an external layout
`0:currentScene="" | 1:externalLayoutName "Name of the external layout" | 2:expression? "X position of the origin" default 0 | 3:expression? "Y position of the origin" default 0 | 4:expression? "Z position of the origin" default 0`