

Animation and Animated Games

By Mike Hahn

Copyright © 2009 JZoneGamer.com

Date Last Edited: 23-Feb-2009

Table of Contents

Overview	1
<u>Introduction</u>	<u>1</u>
<u>Animation Overview.....</u>	<u>1</u>
Animation.....	2
<u>Animation Low-Level</u>	<u>2</u>
<u>Animation High-Level.....</u>	<u>4</u>
<u>Event Handling</u>	<u>5</u>
<u>Level Editor.....</u>	<u>6</u>

Overview

Introduction

[[Home](#)]

JZoneGamer is a software tool that lets you create multiplayer board games, as well as games with 2D, and ultimately 3D animation. You can log on to JZoneGamer.com and play these games with other JZoneGamer users. These games are coded in a built-in scripting language called Treescrypt, and then uploaded to the JZoneGamer web site. Non-programmers can use the Level Editor and the Vector Editor to create simple animated games, and programmers can add functionality to these games.

Animation Overview

Low-Level Components

These low-level components are the basic building blocks of animated games, otherwise known as "atoms."

Coord: Point on screen

Rect: Rectangle

Quad: Rectangle with fill color

Ellipse: Circle or oval

Disc: Ellipse with fill color

Polyline: Set of connected line segments

Shape: Set of connected line segments (polygon with fill)

Bitmap: Rectangular array of pixels (colored dots)

Text: Label containing text

High-Level Components

Vector: collection of atoms and/or other vectors

Window: Rectangular area on screen, static or animated.

User: Game player

Level: Arena containing game action

Game: Collection of levels making up a complete game

In-Progress: Game in progress – web site includes list of these

Event Handling

Keyboard: These events are handled at the user level.

Mouse: These events are handled at the vector level.

Collision Detection: These events are triggered when 2 atoms collide.

Timer: These events are triggered after every N frame changes.

Animation

Animation Low-Level

JZoneGamer can be used to make multiplayer animated games. Non-programmers can use the Level Editor and the Vector Editor to create simple animated games, and programmers can add functionality to these games, using a built-in scripting language called Treescript.

The following low-level components are the basic building blocks of animated games, otherwise known as “atoms.”

Coord:

Point on screen

- **x** – floating-point number (float)
- **y** – float

Rect:

Rectangle

- **x** – float
- **y** – float
- **width** – float
- **height** – float
- **color**
- **mouse** – yes/no
- **collision** – yes/no
- **border** – yes/no
- **visible** – yes/no

Quad:

Rectangle with fill color

- **x** – float
- **y** – float
- **width** – float
- **height** – float
- **pen-color**
- **fill-color**
- **mouse** – yes/no
- **collision** – yes/no
- **border** – yes/no
- **visible** – yes/no

Ellipse:

Circle or oval

- **x** – float
- **y** – float
- **width** – float
- **height** – float

- **color**
- **mouse** – yes/no
- **collision** – yes/no
- **border** – yes/no
- **visible** – yes/no

Disc:

Ellipse with fill color

- **x** – float
- **y** – float
- **width** – float
- **height** – float
- **pen-color**
- **fill-color**
- **mouse** – yes/no
- **collision** – yes/no
- **border** – yes/no
- **visible** – yes/no

Polyline:

Set of connected line segments

- **x** – float
- **y** – float
- **coord-list** – list of coordinates
- **color**
- **trajectory** – yes/no (parent vector)
- **mouse** – yes/no
- **collision** – yes/no
- **border** – yes/no
- **visible** – yes/no

Shape:

Set of connected line segments (polygon with fill)

- **x** – float
- **y** – float
- **coord-list** – list of coordinates
- **pen-color**
- **fill-color**
- **mouse** – yes/no
- **collision** – yes/no
- **border** – yes/no
- **visible** – yes/no

Bitmap:

Rectangular array of pixels (colored dots)

- **x** – float
- **y** – float
- **width** – number
- **height** – number
- **plane-count** – no. of bit planes

- **array of pixels**
- **transparent** – yes/no
- **mouse** – yes/no
- **collision** – yes/no
- **visible** – yes/no

Text:

Label containing text

- **x** – float
- **y** – float
- **width** – number
- **height** – number
- **pen-color**
- **fill-color**
- **caption** – string
- **font**
- **transparent** – yes/no
- **mouse** – yes/no
- **collision** – yes/no
- **visible** – yes/no

Animation High-Level

Vector:

This is a collection of atoms and/or other vectors

- **x** – float
- **y** – float
- **dx** – float: x-velocity
- **dy** – float: y-velocity
- **parent** – parent vector
- **vector-list** – list of components
- **trans-mat** – 3 x 3 linear transformation matrix
- **object** – pointer to associated Treescrpt object, if any
- **class-name** – name of class to which object belongs
- **name** – string: name of vector, if any
- **stationary** – yes/no
- **mouse** – yes/no
- **collision** – yes/no
- **visible** – yes/no

Window:

Rectangular area on screen, static or animated. Board games always take place in static windows.

- **x** – number
- **y** – number
- **width** – number
- **height** – number
- **static** – yes/no
- **scrolling** – yes/no
- **vector-list** – list of vectors
- **component-list** – list of static components

User:

Game player

- **user-no** – number
- **user-id** – string
- **password**
- **first-name**
- **last-name**
- **email**
- **start-date** – date of registration
- **active** – yes/no
- **main-vec** – vector
- **vector-list** – list of vectors in user's possession

Level:

Arena containing game action

- **level-no** – number
- **level-name** – string
- **width** – float
- **height** – float
- **win-list** – list of windows
- **user-list** – list of users
- **vector-list** – list of vectors

Game:

Collection of levels making up a complete game

- **game-no** - number
- **name** – name of the game
- **descr** – description of game
- **level-list** – list of levels
- **start-date** – date game went live
- **active** – yes/no

In-Progress:

Game in progress – web site includes list of these

- **game-no** – number
- **level-list** – list of levels
- **winner-list** – list of users who are winners

Event Handling

Keyboard:

Keyboard events are handled at the user level. The default action of the arrow keys (and joystick) is to move the user in the desired direction. The default action of the Enter key is to perform the current default command. If chat is enabled, users may type in messages to each other.

Mouse:

Mouse events are handled at the vector level. Those child objects of the vector with (mouse = yes) transmit mouse messages to their parent vector. Mouse events include Click, Dbl-click, Mouse-down, Mouse-up, and Mouse-move. Each mouse event is an interior event, or optionally a border event, of the low-level component.

Collision Detection:

A Collision Detection event is triggered when 2 low-level components collide, and both components are (collision = yes) and (visible = yes). Each collision event is an interior event, or optionally a border event, of each of the 2 low-level components.

Timer:

Timer events are triggered after every N frame changes, where N is a number greater than or equal to one. Treescrypt attempts to display 30 frames per second. If the user's computer is too slow, then it skips frames.

Vector Events:

- **OnCreate**
- **OnDestroy**
- **OnShow** – visible changes to yes
- **OnHide** – visible changes to no
- **OnEnter** – enters user's field of view
- **OnExit** – exits user's field of view
- **OnStart** – begins new trajectory, or velocity becomes non-zero
- **OnStop** – ends old trajectory, or velocity becomes zero
- **OnReparent** – new parent vector
- **OnAcquire** – acquires new child vector/atom
- **OnLose** – loses old child vector/atom
- **OnCollide** – collision detected
- **OnClick**
- **OnDbIcIck**
- **OnMouseUp**
- **OnMouseDown**
- **OnMouseMove**
- **OnKeyUp** – user event
- **OnKeyDown** – user event
- **OnKeyPress** – user event

Level Editor

Design-Time Commands:

- **Up Arrow** – Select parent vector
- **Down Arrow** – Select first child vector/atom
- **Left Arrow** – Select previous sibling vector/atom
- **Right Arrow** – Select next sibling vector/atom
- **Shift+Left Arrow** – Move up in display order

- **Shift+Right Arrow** – Move down in display order
- **Shift+Ctrl+Arrow Key** – Nudge by one grid unit
- **Left Click** – make selection
- **Right Click** – popup menu of additional commands
- **Shift+Click** – toggle selection on/off
- **Drag** interior of vector/atom – move it
- **Drag** one of 8 black/grey squares surrounding selection – resize it
Black squares: single vector/atom selected
Grey squares: multiple vectors/atoms selected
- **Drag** one of 4 double arrows surrounding selection – rotate it

Right-Click Atom:

- Cut
- Copy
- Paste
- Delete
- Align – multiple selection
- Size – multiple selection
- Convert to different type of atom
- Edit parameters
- Pen Color
- Fill Color
- Toggle Resize/Rotate

Right-Click Vector:

- Cut
- Copy
- Paste
- Delete
- Align – multiple selection
- Size – multiple selection
- Edit parameters
- Cycle: Resize/Rotate/Multi-Select
- Velocity