High Art
on top of
LOW-LEVEL APIS

Building Games with Ruby

Andrea O. K. Wright, Chariot Solutions
www.chariotsolutions.com

High Art

on top of

LOW-LEVEL APIS

Building Games with Ruby

Agenda

Text-based Games

2D Games

3D Games

4D Games

Game DSL

Created by Steven Hammond
http://www.northpub.com/pages/game_dsl

```
2.d6
```

Photo from http://upload.wikimedia.org/wikipedia/commons/6/6a/Dice.jpg

Game DSL

```
class Integer

  @@entropy_source = GameDSL::EntropySources::EntropySource.get_entropy_source

  # List of available dice types
  @@DICE = [2,4,6,8,10,12,20,30,100]

  # Generate dX methods for each of the allowable dice types.
  @@DICE.each do |sides|
    class_eval %{ def d#{sides}(opts = {})
                  value = 0
                  results = []
                  self.times do | i |
                    results << (@@entropy_source.get_random_integer % #{sides})+1
                    value += results[i]
                  end
                  if !opts[:keep_best].nil?
                    results.sort
                    value = 0
                    opts[:keep_best].times do | j |
                      value += results[j]
                    end
                  end
                  return value
                end }
   # Add dF method for rolling Fudge dice
    end
end
```

Source by Steven Hammond: http://www.northpub.com/pages/game_dsl

Game DSL

```
class Integer

  @@entropy_source = GameDSL::EntropySources::EntropySource.get_entropy_source

  # List of available dice types
  @@DICE = [2,4,6,8,10,12,20,30,100]

  # Generate dX methods for each of the allowable dice types.
  @@DICE.each do |sides|
    class_eval %{ def d#{sides}(opts = {})
                    value = 0
                    results = []
                    self.times do | i
                      results << (@@er                    #{sides})+1
                      value += results
                    end
                    if !opts[:keep_bes
                      results.sort
                      value = 0
                      opts[:keep_best]
                        value += resul
                      end
                    end
                    return value
                  end }
    # Add dF method for rolling Fudge dice
  end
end
```

Source by Steven Hammond: http://www.northpub.com/pages/game_dsl; Photo:http://www.colinfahey.com/sat/sat_fantasy_calc_dice.gif

Gambit

Created by James Edward Gray II and Gregory Brown

http://rubyforge.org/projects/gambit/

Board

[] []= cols each each_col each_location each_row each_with_location move neighboring_locations neighbors parse_location rows to_chess valid?

Cards

Card << [] count discard each new sort!

StandardCard <=> == each jokers_are_high jokers_are_high= name new suit_order suit_order= use_jokers use_jokers= value_order value_order=

Hand << [] count discard each new sort!

Pile << [] add each new remove

Deck << count cut! deal draw each new replace shuffle! sort!

Currency

+ - [] [] []= deposit each new relative_value withdraw

Dice

* + - / [] coerce count each fail matches? new reroll success sum

Move History

<< [] add_player add_players each new record

Score Card

add_player add_players loser new require_categories score set_total set_win start_at= total use_category winner

TeensyMUD

Created by Jon Lambert

http://teensymud.kicks-ass.org/wiki/show/HomePage

* Multiplexed single-threaded network driver using Acceptor/Reactor and Observer patterns
* Correct standard TELNET protocol implementation
* Supports TELNET NAWS, TTYPE, ECHO, SGA, and BINARY options
* VT-52/100/102/220 support
* Partial Xterm support
* ZMP protocol support
* Client detection (18+ clients tested so far)
* Fully persistent virtual world
* DBM, GDBM, SDBM, SQLite2, SQLite3 and YAML supported database back ends
* Includes database load and dump utilities
* Configurable object caching support for disk-based databases
* Object ownership
* Event-driven system with first class game objects communicating via events
* Supports room-based system currently with unlimited exits
* Offline creation in YAML
* Color and VT-100 highlighting support for builders using TML
* Currently supports chat, say and emote communication commands
* Currently supports player movement, inventory, get and drop commands
* Autolook and autoexits
* Runtime extensible commands. Add new commands without rebooting.
* Trigger script programming
* AOP-like cuts for PRE and POST event trigger processing
* Supports multiple persistent and variable timers

(partial list of features taken from the v.2.10.0 README)

TeensyMUD

```
require 'socket';require 'yaml';def q x;$m.find{|o|o.t==2&&x==o.n};end
def a r,t;$m.find_all{|o|t==o.t&&(!r||r==o.l)};end;def g z;a(nil,2).each{|p|
p.p z};end;class O;attr_accessor :i,:n,:l,:e,:s,:t;def initialize n,l=nil,t=0
@n,@l,@i,@t=n,l,$d+=1,t;@e={};end;def p s;@s.puts(s)if @s;end;def y m
v=$m.find{|o|@l==o.i};t=v.e.keys;case m;when/^q/;@s.close;@s=nil;
File.open('d','w'){|f|YAML::dump $m,f};when/^h/;p "i,l,d,g,c,h,q,<exit>,O,R"
when/^i/;a(@i,1).each{|o|p o.n};when/^c.* (.*)/;g "#{@n}:#{$1}"
when/^g/;a(@l,1).each{|q|q.l=@i};when/^d/;a(@i,1).each{|q|q.l=@l}
when/^O (.*)/;$m<<O.new($1,@l,1);when/^R (.*) (.*) (.*)/;$m<<d=O.new($1)
v.e[$2]=d.i;d.e[$3]=v.i;when/^l/;p v.n;(a(@l,2)+a(@l,1)).each{|x|
p x.n if x.s||x.t==1};p t.join '|';when/(^#{t.empty? ? "\1" : t.join('|^')})/
@l=v.e[$1];else;p "?";end;end;end;if !test ?e,'d';$d=0;$m=[O.new("Home")];else
$m=YAML::load_file 'd';end;$d=$m.size;z=TCPServer.new 0,4000;while k=z.accept
Thread.new(k){|s|s.puts "Name";s.gets;l=$_.chomp;d=q l;$m<<d=O.new(l,1,2)if !d
d.s=s;while s.gets;d.y $_.chomp;end;};end
```

```
Help (for the single character impaired)
i = displays player inventory
l = displays the contents of a room
d = drops all objects in your inventory into the room
g = gets all objects in the room into your inventory
c < message > = chat with other players h = displays help
q = quits the game (saves player)
O < object name > = creates a new object (ex. O rose)
R < room name > < exit name to > < exit name back > = creates a new room
and autolinks the exits using the exit names provided.
< exit name > = moves player through exit named (ex. south)
```

Source by Jon Lambert: http://teensymud.kicks-ass.org/wiki/show/TeensyMud1K

TeensyMUD

```
z=TCPServer.new 0,4000

while k=z.accept
  Thread.new(k){
        |s| s.puts "Name"
        s.gets
        l=$_.chomp
        d=q l
        $m<<d=O.new(l,1,2) if !d
        d.s=s
        while s.gets
              d.y $_.chomp
        end
        }
end
```

```
require 'socket';require 'yaml';def q x;$m
def a r,t;$m.find_all{|o|t==o.t&&(!r||r==o
p.p z};end;class O;attr_accessor :i,:n,:l,
@n,@l,@i,@t=n,l,$d+=1,t;@e={};end;def p s;
v=$m.find{|o|@l==o.i};t=v.e.keys;case m;wh
File.open('d','w'){|f|YAML::dump $m,f};whe
when/^i/;a(@i,1).each{|o|p o.n};when/^c.*
when/^g/;a(@l,1).each{|q|q.l=@i};when/^d/;
when/^O (.*)/;$m<<O.new($1,@l,1);when/^R (
v.e[$2]=d.i;d.e[$3]=v.i;when/^l/;p v.n;(a(@l,2)+a(@l,1)).each{|x|
p x.n if x.s||x.t==1};p t.join '|';when/(^#{t.empty? ? "\1" : t.join('|^')})/
@l=v.e[$1];else;p "?";end;end;end;if !test ?e,'d';$d=0;$m=[O.new("Home")];else
$m=YAML::load_file 'd';end;$d=$m.size;z=TCPServer.new 0,4000;while k=z.accept
Thread.new(k){|s|s.puts "Name";s.gets;l=$_.chomp;d=q l;$m<<d=O.new(l,1,2) if !d
d.s=s;while s.gets;d.y $_.chomp;end;};end
```

Source by Jon Lambert: http://teensymud.kicks-ass.org/wiki/show/TeensyMud1K

Ruby/SDL

Creator and Lead Developer: Ohai

http://www.kmc.gr.jp/~ohai/rubysdl.en.html

Ruby/SDL C Extension

```
static VALUE sdl_warpMouse(VALUE mod,VALUE x,VALUE y)
{
  SDL_WarpMouse( NUM2UINT(x),NUM2UINT(y) );
  return Qnil;
}
```

```
rb_define_module_function(mMouse,"warp",sdl_warpMouse,2);
```


R U
TIPE
CASTIN'
MEE???

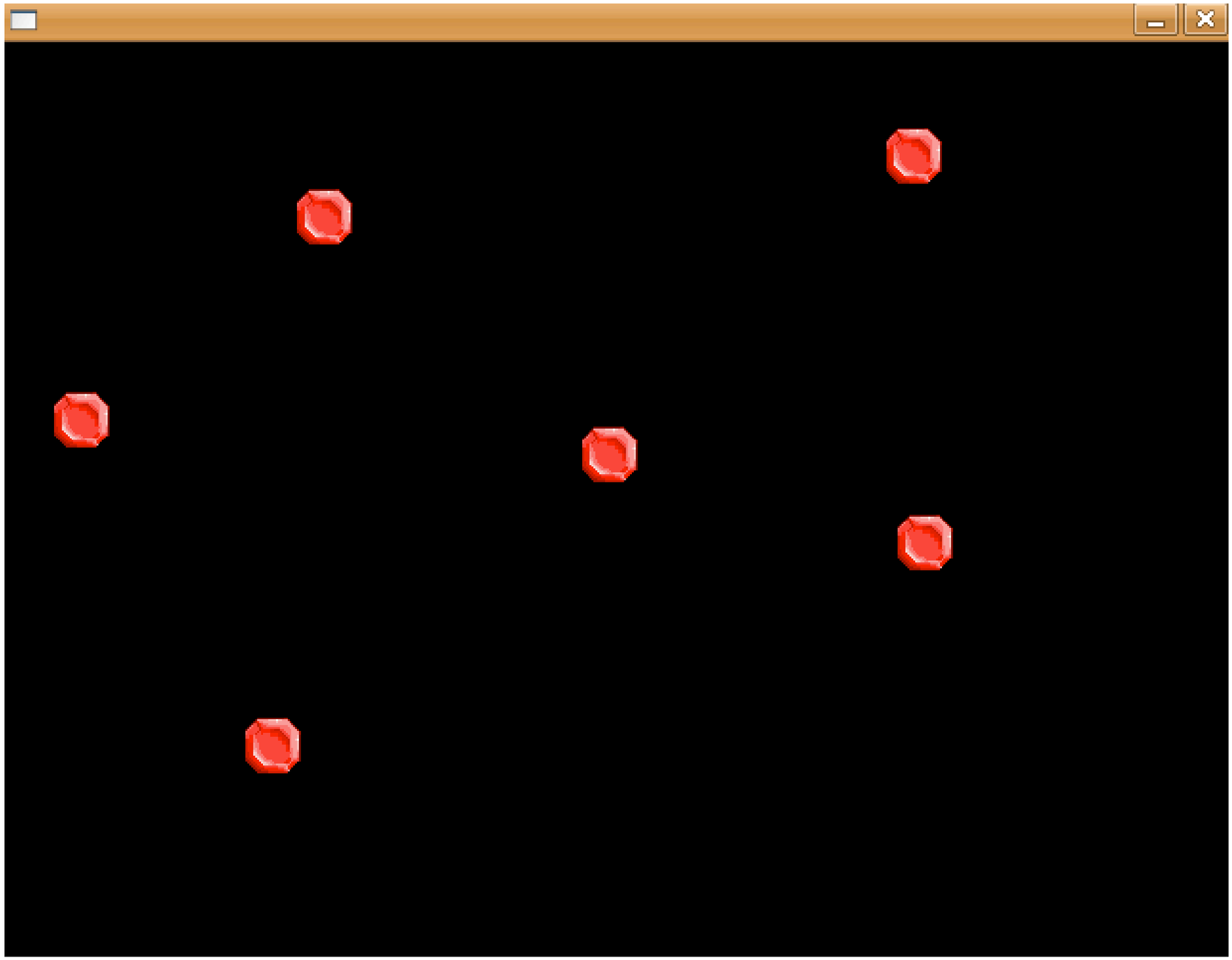
Ruby/SDL C Extension

```
static VALUE sdl_warpMouse(VALUE mod,VALUE x,VALUE y)
{
  SDL_WarpMouse( NUM2UINT(x),NUM2UINT(y) );
  return Qnil;
}
```

```
rb_define_module_function(mMouse,"warp",sdl_warpMouse,2);
```

Ruby/SDL

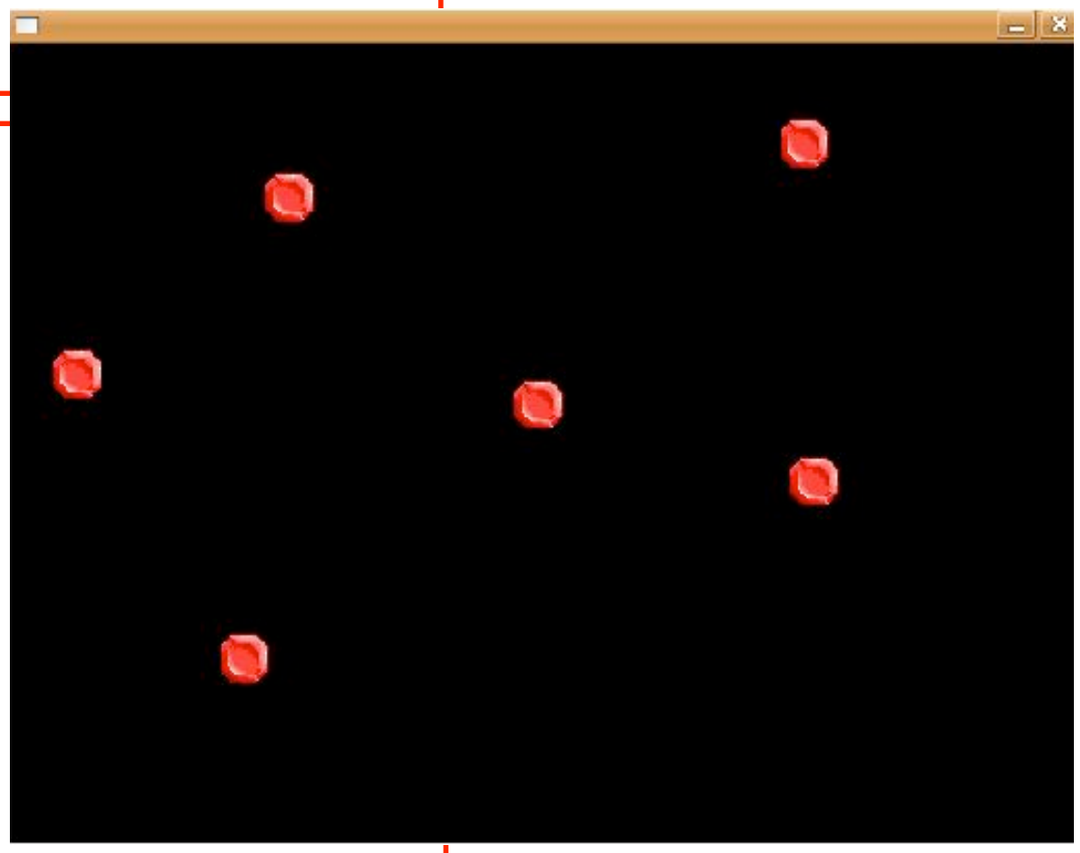
Sample



Ruby/SDL Sample

```
SDL.init( SDL::INIT_VIDEO )

screen = SDL::setVideoMode(640,480,16,SDL::SWSURFACE)

image = SDL::Surface.loadBMP("icon.bmp")
image.setColorKey( SDL::SRCCOLORKEY || SDL::RLEACCEL ,0)
$image = image.displayFormat
```

```
while true
  while event = SDL::Event2.poll
    case event
    when SDL::Event2::Quit
      exit
    when SDL::Event2::KeyDown
      exit if event.sym == SDL::Key::ESCAPE
    end
  end

  screen.fillRect(0,0,640,480,0)
  SDL::Key.scan

  sprites.each {|i|
    i.move
    i.draw(screen)
  }
  screen.updateRect(0,0,0,0)
end
```



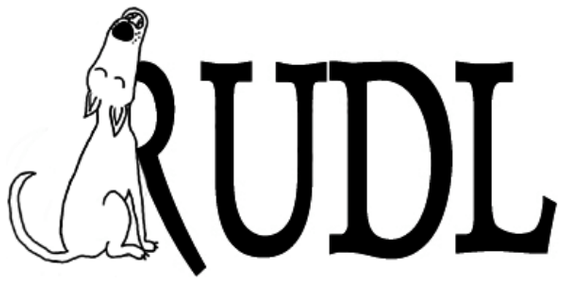
Ruby/SDL

Steven Davidovitz's Nebular Gauntlet

Project Home: http://www.nebulargauntlet.org/

Danny Van Bruggen, Creator

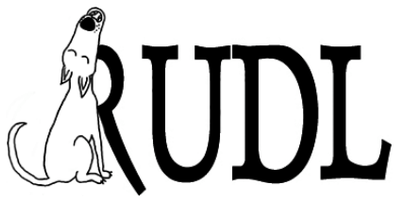
http://sourceforge.net/projects/rudl/



Principle of Least Surprise?

Ruby/SDL

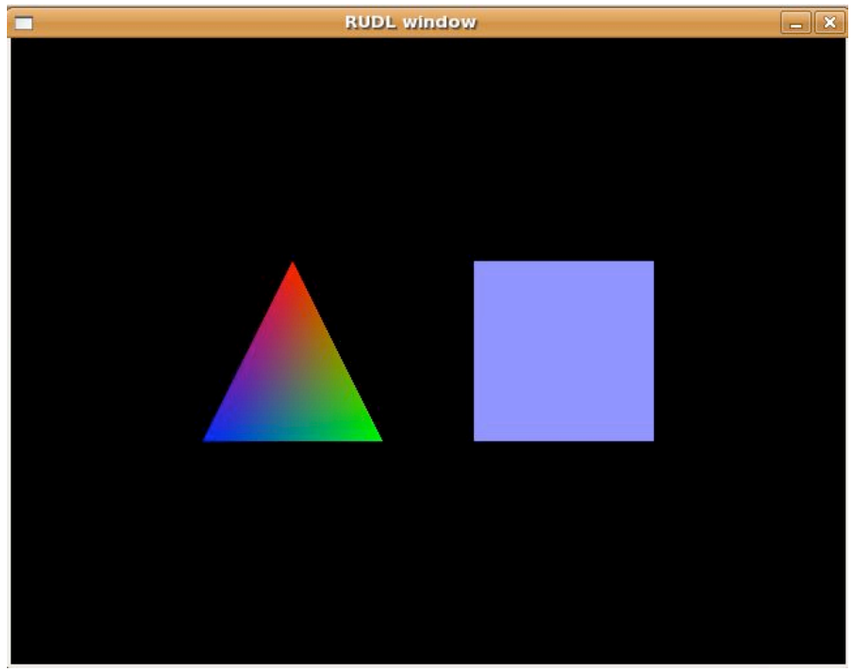
```
screen = SDL::setVideoMode(640,480,16,SDL::SWSURFACE)
```




```
display = DisplaySurface.new([640,480])
```

RUDL OpenGL

```
display = Proc.new {
GL.Clear(GL::COLOR_BUFFER_BIT | GL::DEPTH_BUFFER_BIT)
GL.Color(1.0, 1.0, 1.0)
GL.LoadIdentity()

GL.Translate(-1.5, 0.0, -6.0)

# draw a triangle
GL.Begin(GL::POLYGON)
GL.Vertex3f( 0.0, 1.0, 0.0)
GL.Vertex3f( 1.0,-1.0, 0.0)
GL.Vertex3f(-1.0,-1.0, 0.0)
GL.End()

# Move Right 3 Units
GL.Translate(3.0,0.0,0.0)

# draw a square (quadrilateral)
GL.Begin(GL::QUADS)                # start drawing a polygon (4 sided)
GL.Vertex3f(-1.0, 1.0, 0.0)          # Top Left
GL.Vertex3f( 1.0, 1.0, 0.0)          # Top Right
GL.Vertex3f( 1.0,-1.0, 0.0)          # Bottom Right
GL.Vertex3f(-1.0,-1.0, 0.0)          # Bottom Left
GL.End()                    # done with the quad
}
```

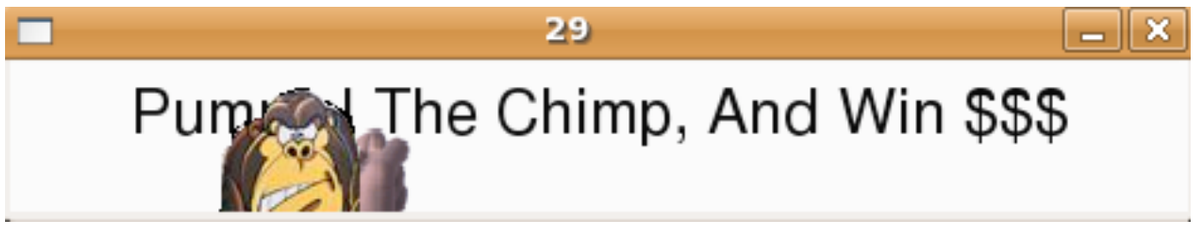


Source by Martin Stannard: from RUDL samples

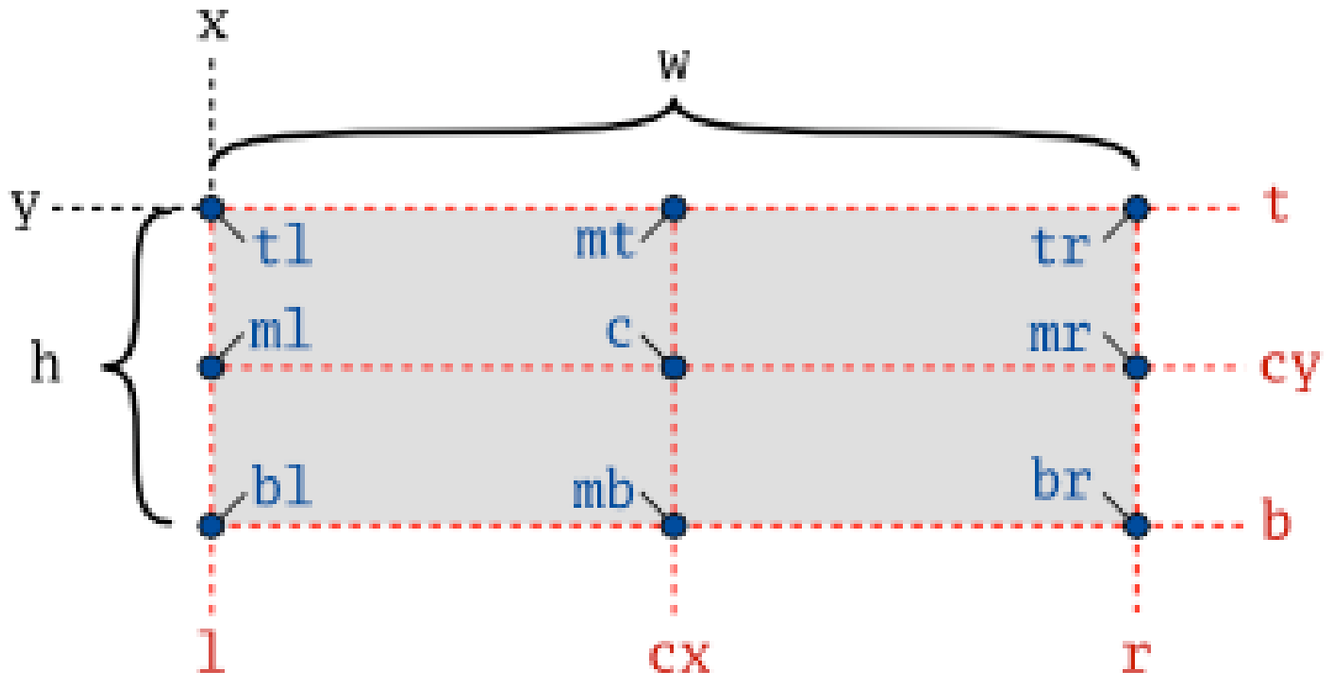
Lead Developer and Creator: John Croisant

http://sourceforge.net/projects/rubygame/

Rubygame Samples



Rubygame and pygame




```
@rect.topleft = 10,10
```

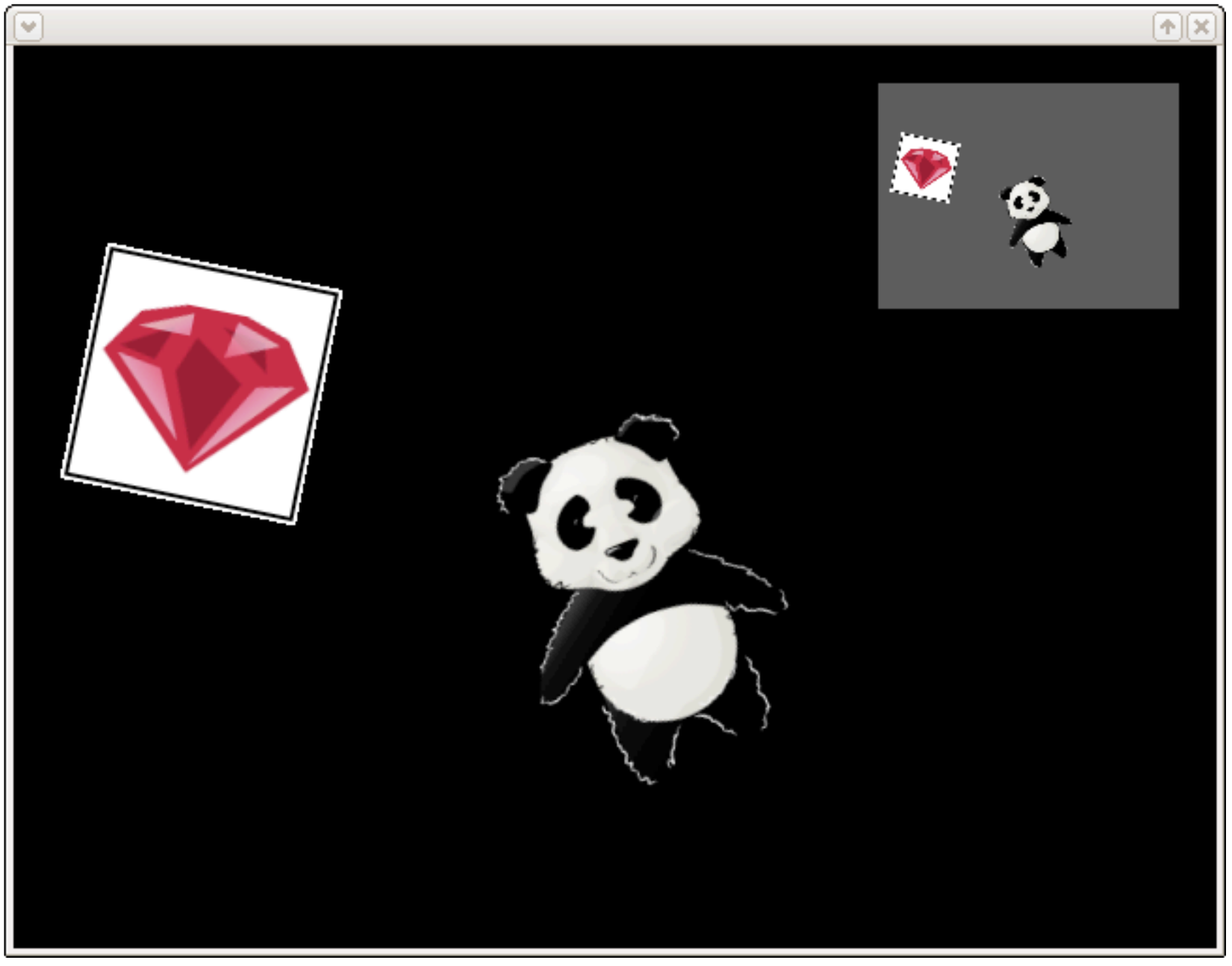
```
# Attempt to punch a target. Returns true if it hit or false if not.
def punch(target)
    @punching = true
    # use a smaller rect to check if we collided with the target
    return @rect.inflate(-5, -5).collide_rect?(target.rect)
end
```

Rubygame and pygame

```
def load_image(name, colorkey=nil)
    image = Rubygame::Surface.load_image(name)
    if colorkey != nil
        if colorkey == -1
            colorkey = image.get_at([0,0])
        end
        image.set_colorkey(colorkey)
    end
    return image, Rubygame::Rect.new(0,0,*image.size)
end
```

Edge Rubygame

Edge Rubygame OpenGL Support



Edge Rubygame Event Management

Register an EventHook with the EventHandler to Link an Action and a Trigger

```
scene = self
@event_handler.append_hook do
  @owner = scene.objects
  @trigger = TickTrigger.new
  @action = MethodAction.new(:update, true)
end
```

```
handler.append_hook do
  @owner = scene
  @trigger = AnyTrigger.new(KeyPressTrigger.new( :q ),
                            KeyPressTrigger.new( :escape ),
                            InstanceTrigger.new( QuitEvent ))
  @action = BlockAction.new {|owner, event| throw :quit }
end
```

If the Event Meets the Hook's Trigger's Criteria, Invoke the Hook's Action...

```
def handle(event)
  matching_hooks = @hooks.select {|hook| hook.match?( event )}
  matching_hooks.each {|hook| hook.perform( event )}
  return nil
end
```

Edge Rubygame Event Management

```
scene = self
@event_handler.append_hook do
  @owner = scene.objects
  @trigger = TickTrigger.new
  @action = MethodAction.new(:up
end
```

```
class TickTrigger
    def match?( event )
        event.kind_of?( Rubygame::TickEvent )
    end
end
```

```
handler.append_hook do
  @owner = scene
  @trigger = AnyTrigger.new(KeyPressTrigger.new( :q ),
                            KeyPressTrigger.new( :escape ),
                            InstanceTrigger.new( QuitEvent ))
  @action = Block
end
```

```
class AnyTrigger
    def initialize( *triggers )
        @triggers = triggers
    end

    def match?( event )
        @triggers.any? { |trigger| trigger.match? event }
    end
end
```

Lead Developer and Creator: John Croisant

http://sourceforge.net/projects/rubygame/

gosu

coolest gamedev library around.

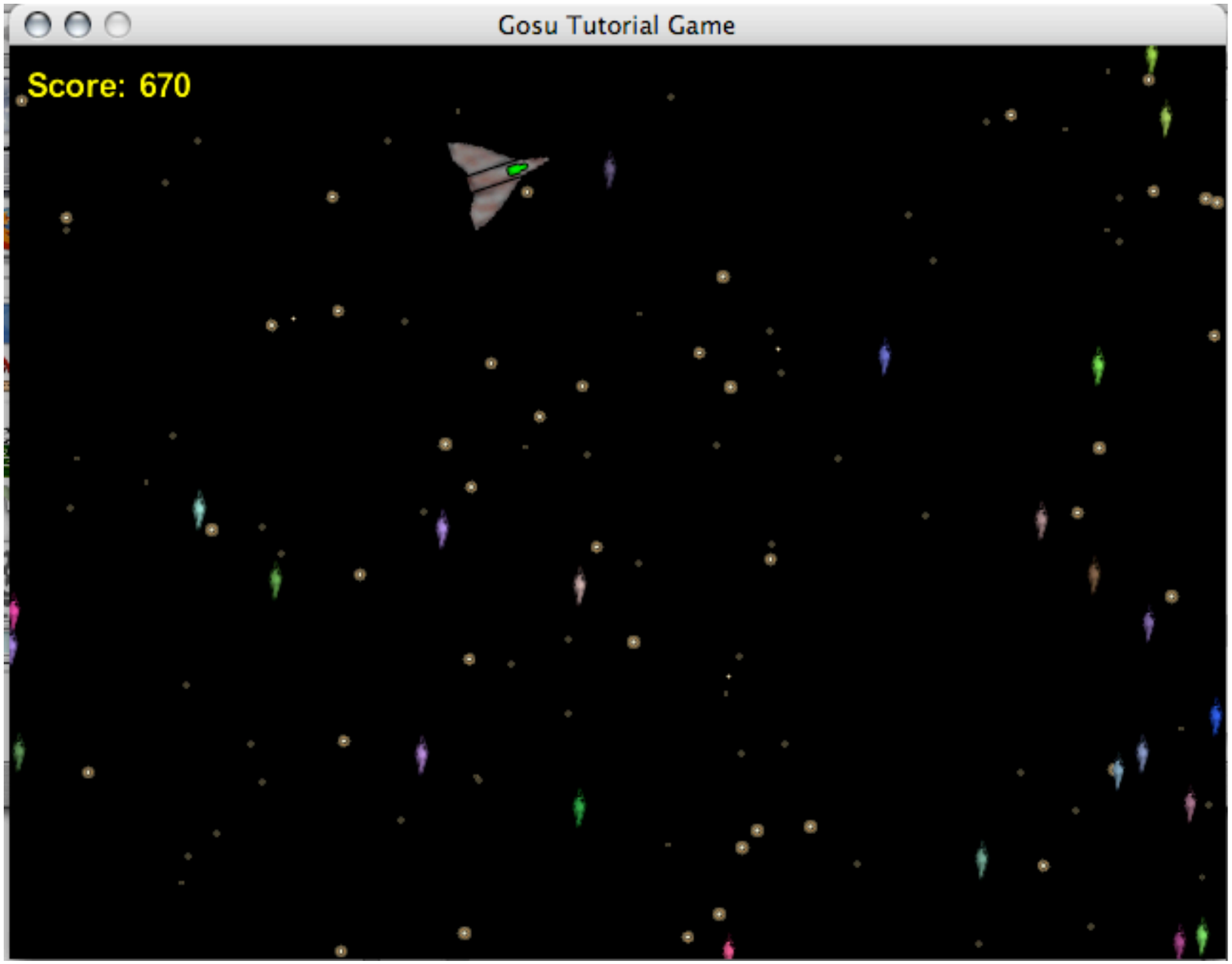
Co- Creators: Julian Raschke & Jan Lücker

Lead Developer: Julian Raschke

http://code.google.com/p/gosu/


gosu
coolest gamedev
library around.

Tutorial


Gosu Tutorial Game
Score: 670

Main Window

```
require 'gosu'

class MainWindow < Gosu::Window
  def initialize(width, height, fullscreen, tick_len)
    super
  end

  def update
    # Does anything need to be repositioned, resized
    # or tinted differently? If so how?
  end

  def draw
    # Display calls go here.
  end

  def button_down(id)
    # Event-handling code goes here.
    # Buttons in Gosu include keys, mouse buttons
    # and game pad controls.
  end
end

w = MyWindow.new(640, 480, false, 20)
w.show
```

Main Window

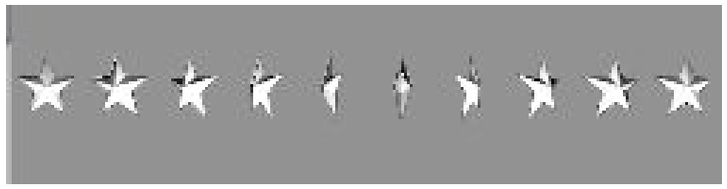
```
def draw
  @font.draw("Score: #{@score}", 10, 10, ZOrder::UI, 1.0, 1.0, 0xffffff00)
  @background_image.draw(0, 0, ZOrder::Background)
  @player.draw
  @stars.each { |star| star.draw }
end
```

Main Window

```
def draw
  @font.draw("Score: #{@score}", 10, 10, ZOrder::UI, 1.0, 1.0, 0xffffff00)
  @background_image.draw(0, 0, ZOrder::Background)
  @player.draw
  @stars.each { |star| star.draw }
end
```

```
module ZOrder
  Background, Stars, Player, UI = *0..3
end
```

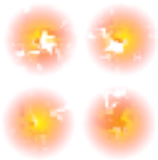
Techniques



```
class Star
  attr_reader :x, :y

  def initialize(animation)
    @animation = animation
    @color = Gosu::Color.new(0xff000000)
    @color.red = rand(255 - 40) + 40
    @color.green = rand(255 - 40) + 40
    @color.blue = rand(255 - 40) + 40
    @x = rand * 640
    @y = rand * 480
  end

  def draw
    img = @animation[Gosu::milliseconds / 100 % @animation.size];
    img.draw(@x - img.width / 2.0, @y - img.height / 2.0,
        ZOrder::Stars, 1, 1, @color, :additive)
  end
end
```

Janis Born's Escave

Techniques



Nightly Travels of a Witch

Team Tortilla: Florian Gross, Julian Raschke, Alexander Post

Techniques

```
####################
#..................#
#..................#
#..................#
#..................#
#..................#
#..................#
#..................#
#.........C.C......#
#..P.....T..T......#
#######.........*...#
####################
```


```
table = { #abbreviated list of mappings; just showing those needed for level 1
  ...
  '*' => lambda { |x, y| Bed.new(game, x * 50 + 25, y * 50) }, # Bed
  'C' => lambda { |x, y| set(x, y, 5) }, # Chair
  'T' => lambda { |x, y| set(x, y, 1); set(x + 1, y, 2); set(x, y + 1, 3);
      set(x + 1, y + 1, 4) }, # Table
  '#' => lambda { |x, y| set(x, y, 0) }, # Wall grey
  'P' => lambda { |x, y| Player.new(game, x * 50 + 25, y * 50 - 51) }, # witch
  ...
}
```

gosu

coolest gamedev library around.

Co- Creators: Julian Raschke & Jan Lücker
Lead Developer: Julian Raschke

http://code.google.com/p/gosu/

gosu

coolest gamedev
library around.

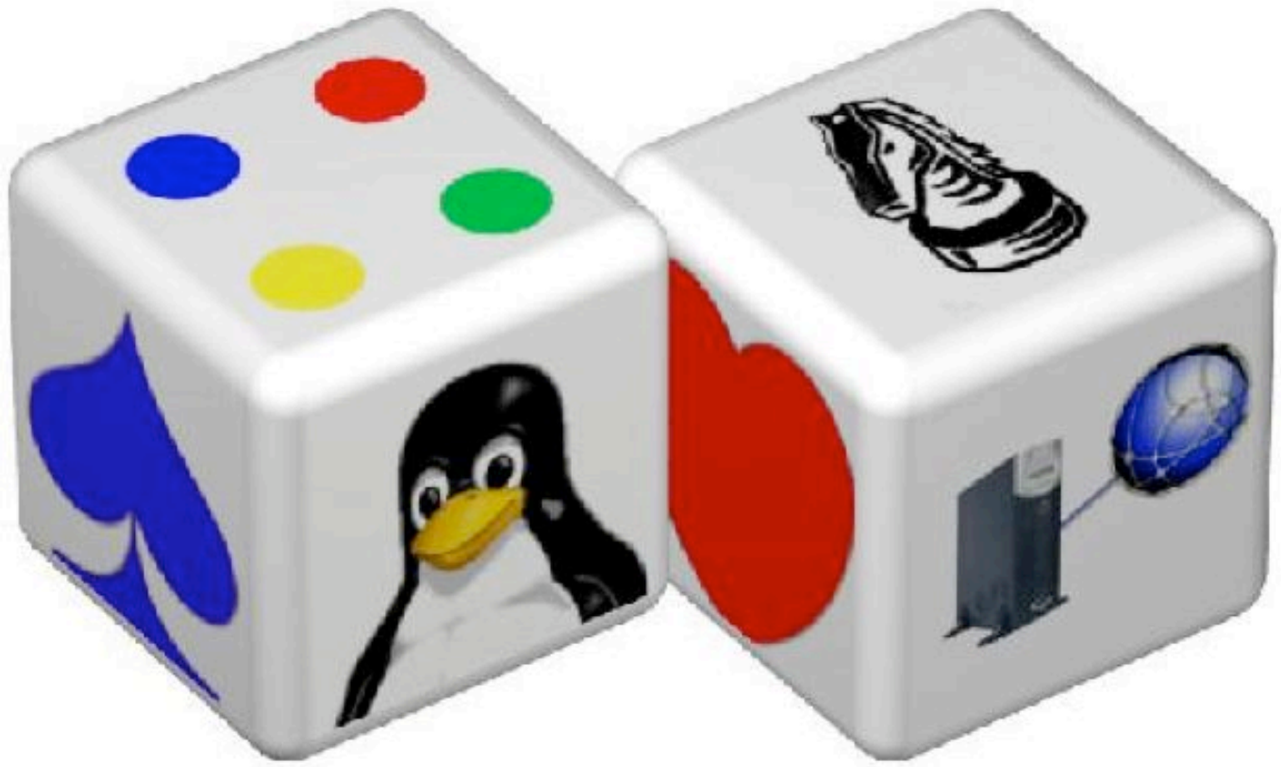
Ruby/SDL

Rubygame

?


llor.nu
Simple huge.

The GGZ Gaming Zone Project

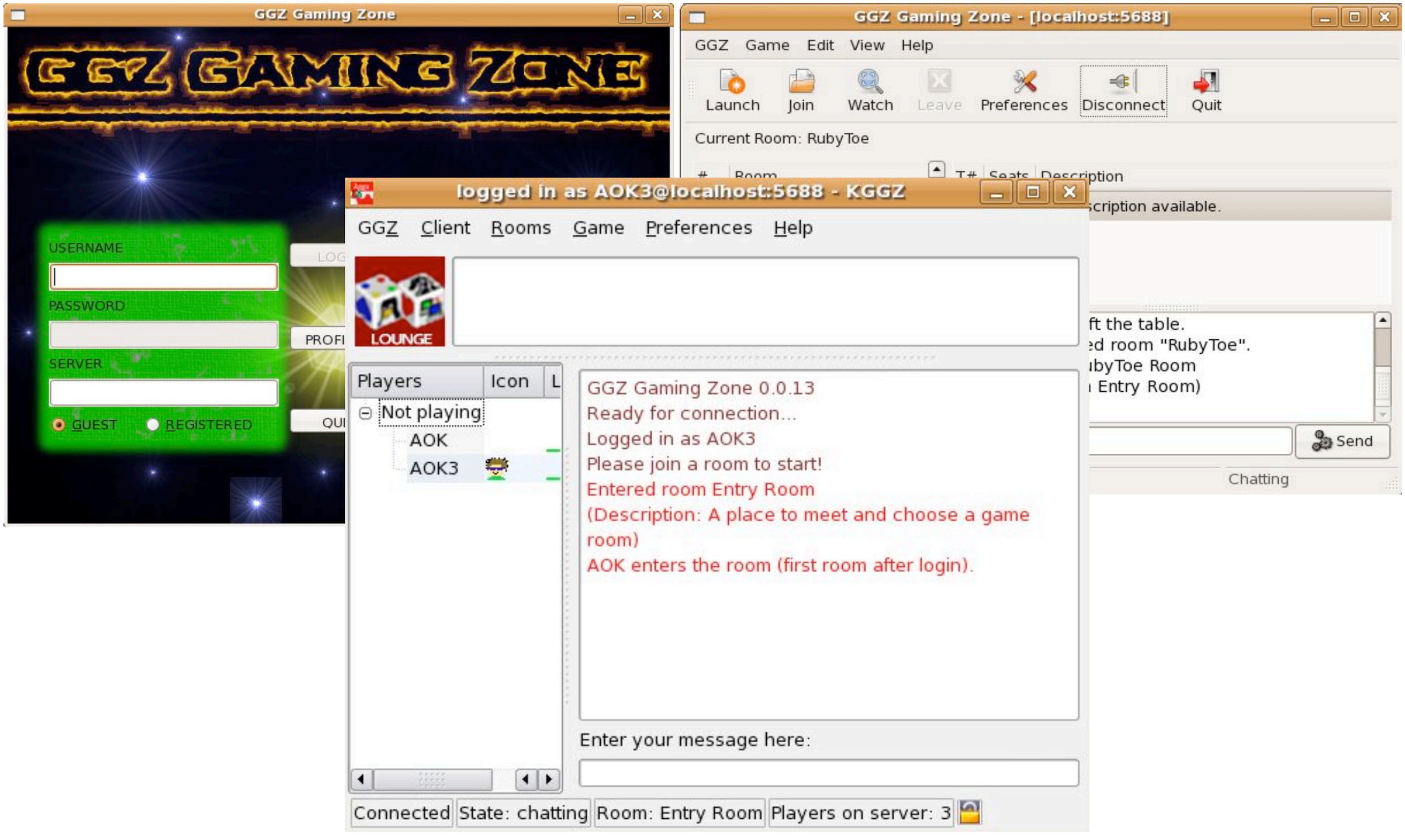


Founders: Brent M. Hendricks
Rich Gade

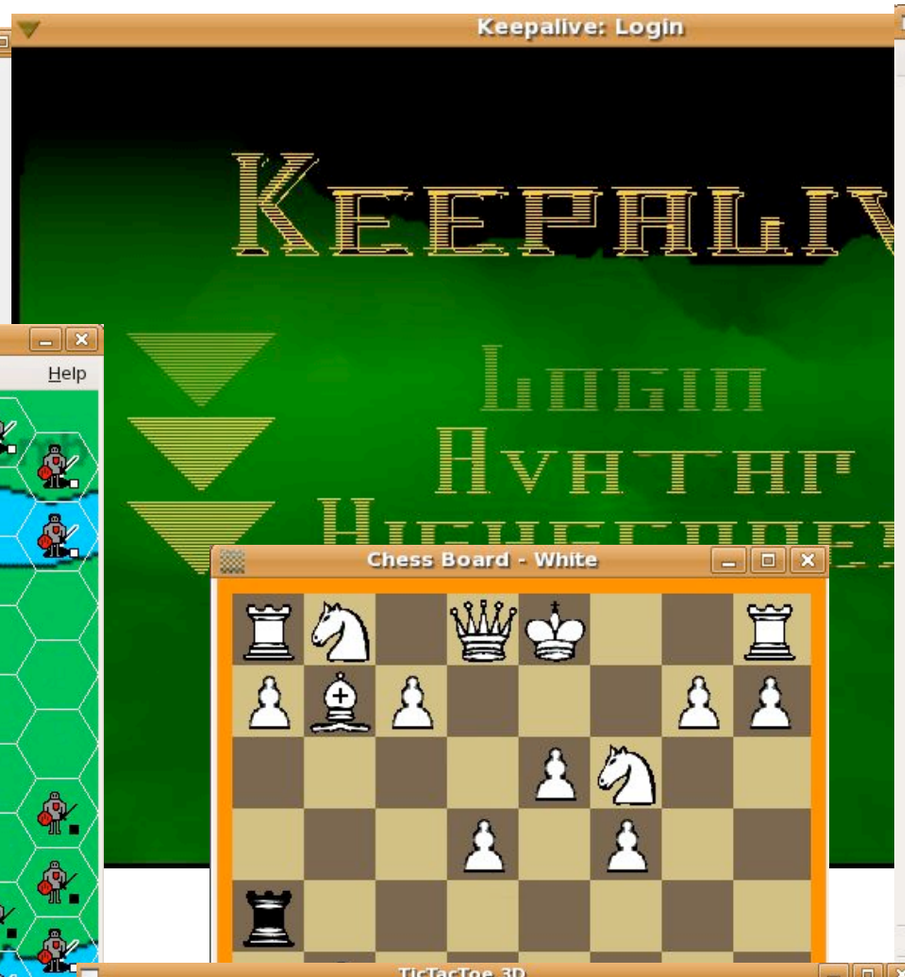
Developed and Maintained by an International Team

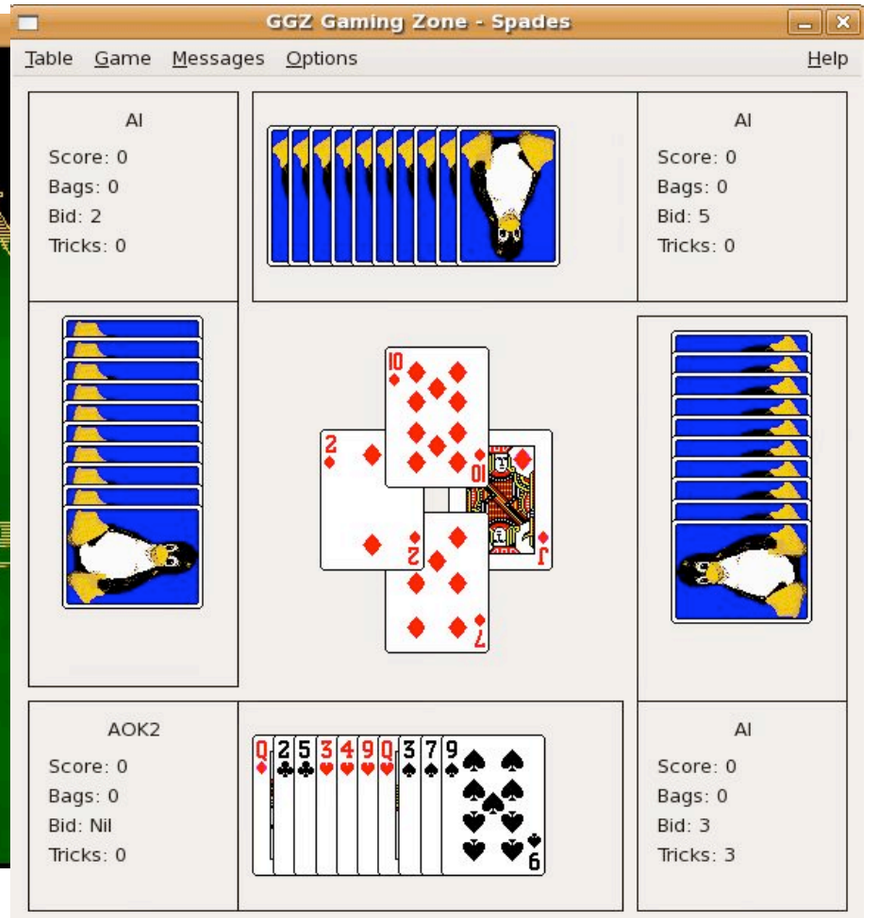
http://www.ggzgamingzone.org/

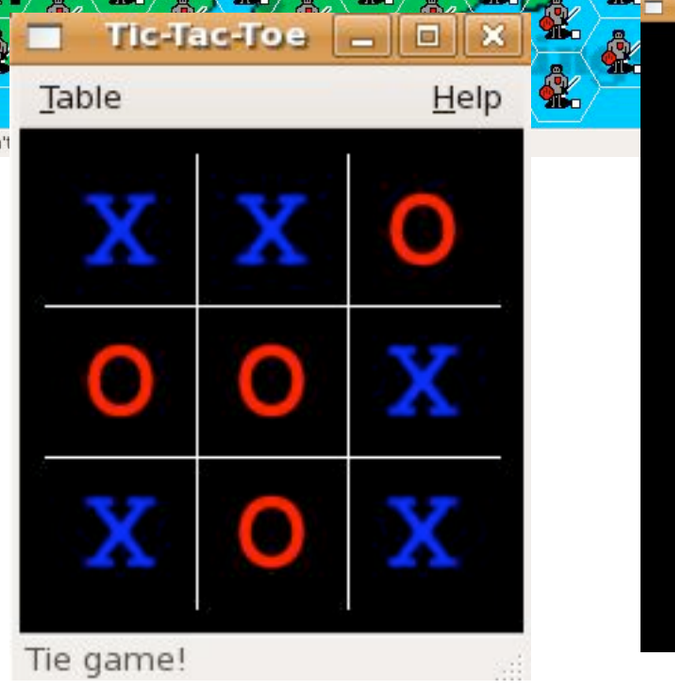
GGZ Gaming Zone

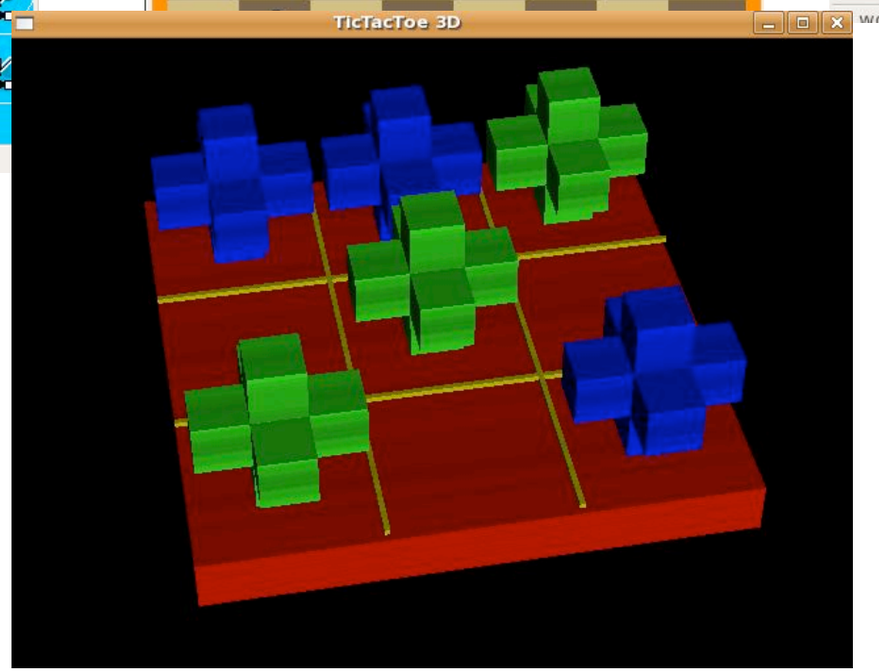


GGZ Gaming Zone



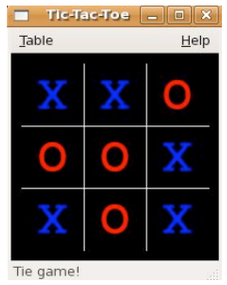






GGZ Gaming Zone

game client



ggzmod/game

net.c **ggzcommgen** **net.c**

Game Client/Game Server
Individual Protocol
TCP/IP or UDP/IP

game server (tictactoe)

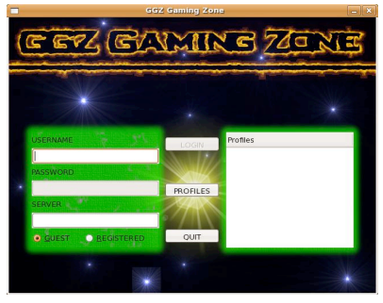
ggzdmod/game

Core Client/Game Client
Binary Protocol
Local Connection

GGZ Server/Game Server
Binary Protocol
Local Connection

ggzmod/core

core client



Core Client/GGZ Server
XML Protocol
TCP/IP

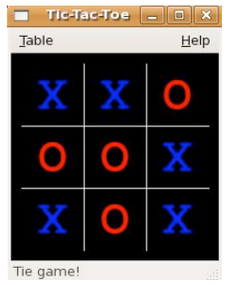
ggzcore

ggzdmod/ggz

ggz server

GGZ Gaming Zone

game client



ggzmod/game

ggzcommgen

net.c ⟷ **net.c**

Game Client/Game Server

Individual Protocol

TCP/IP or UDP/IP

rggzdmod

game server (tictactoe)

ggzdmod/game

Core Client/Game Client

Binary Protocol

Local Connection

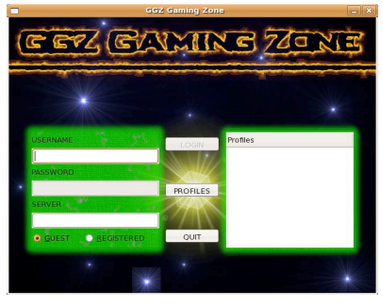
GGZ Server/Game Server

Binary Protocol

Local Connection

ggzmod/core

core client



Core Client/GGZ Server

XML Protocol

TCP/IP

ggzcore

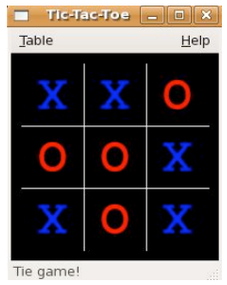
ggzdmod/ggz

ggz server

GGZ Gaming Zone

ggzcommgen

game client



ggzmod/game

net.c ↔ **rubytoenet.rb**

Game Client/Game Server

Individual Protocol

TCP/IP or UDP/IP

game server (RubyToe)

rggzdmod

Core Client/Game Client

Binary Protocol

Local Connection

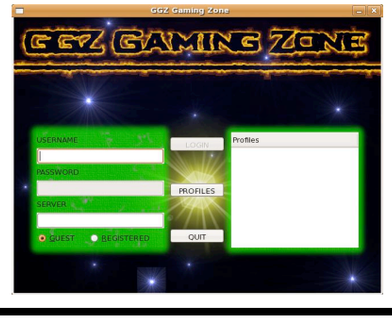
GGZ Server/Game Server

Binary Protocol

Local Connection

ggzmod/core

core client



Core Client/GGZ Server

XML Protocol

TCP/IP

ggzcore ↔

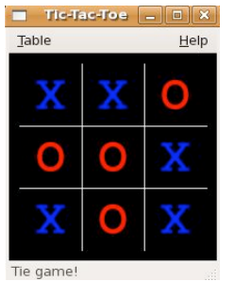
ggzdmod/ggz

ggz server

GGZ Gaming Zone

ggzcommgen

game client



net.c ⟷ **rubytoenet.rb**

Game Client/Game Server

Individual Protocol

TCP/IP or UDP/IP

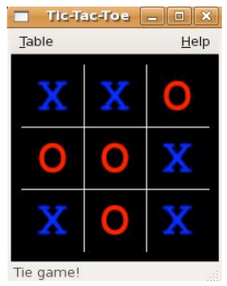
game server (RubyToe)

```
def send_gameover(p, winner)
  ...
  s = GGZRawSocket.for_fd($server.get_seat_fd(p))
  s.puti(::MSG_GAMEOVER)
  s.putb(winner)
end
```

GGZ Gaming Zone

ggzcommgen

game client



net.c ⟷ **net.c**

Game Client/Game Server

Individual Protocol

TCP/IP or UDP/IP

game server (tictactoe)

```
<ggzcomm engine="tictactoe" version="4">
    <definitions>
        <def name="msggameover" value="3"/>
    </definitions>
    <server>
    ... <event name="msggameover">
            <data name="winner" type="byte"/>
        </event>
    </server>
</ggzcomm>
```

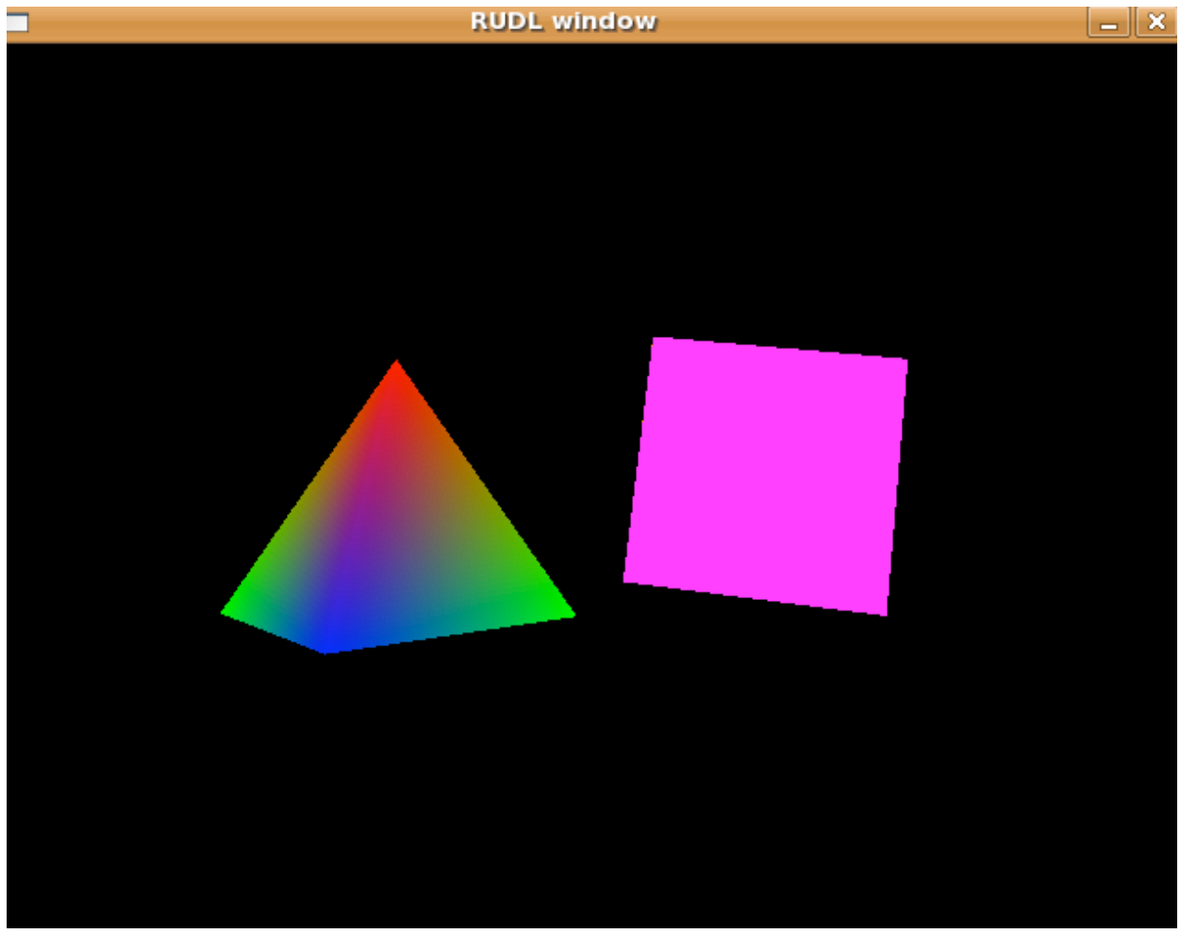
```
void ggzcomm_msggameover(GGZCommIO * io) {
    ret = ggz_write_int(io->fd, msggameover);
    if(ret < 0)
        ggzcomm_error();
    ret = ggz_write_char(io->fd,variables.winner);
    if(ret < 0)
        ggzcomm_error();
}
```

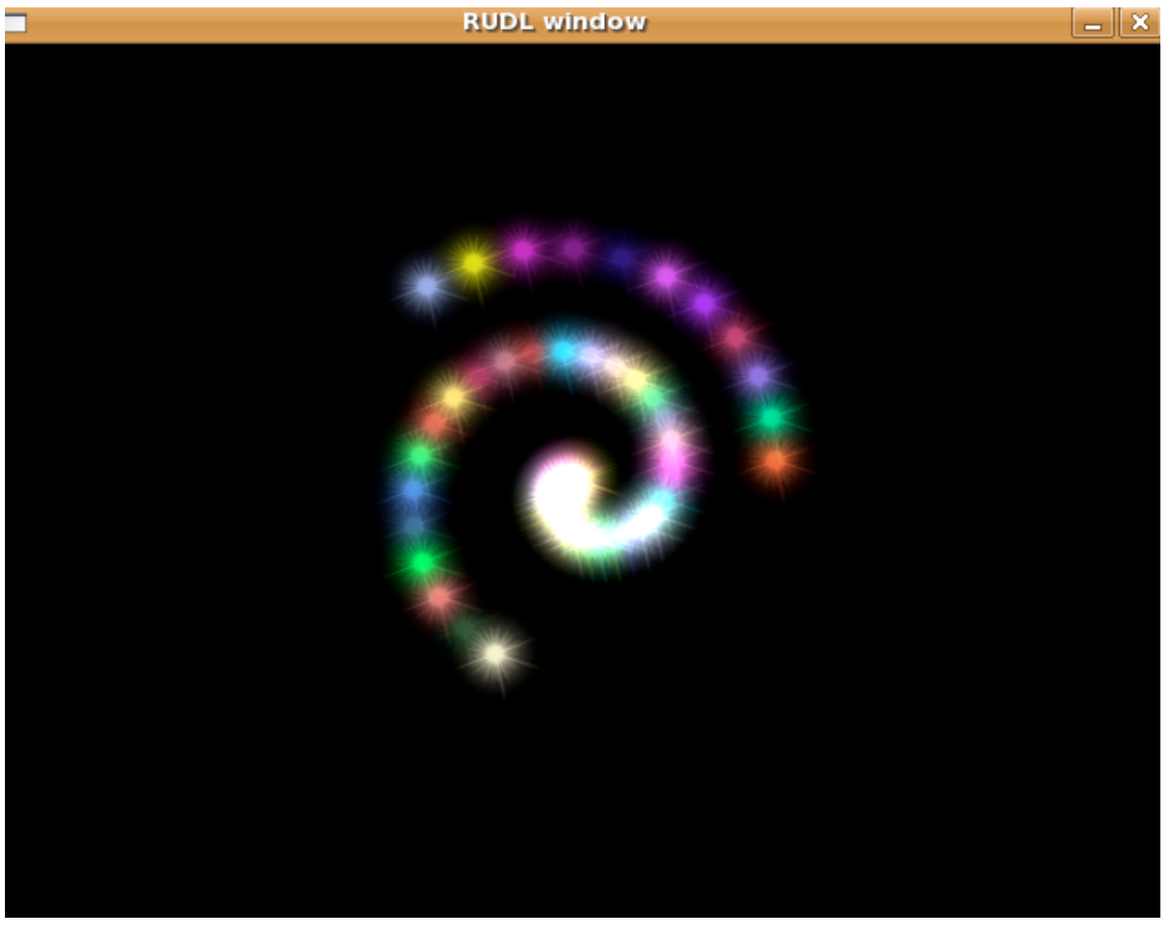
GGZ Gaming Zone

```
begin
  require "TTTAI"
  $ai = TTTAI.new
  puts "## Info: Using TTT-AI plugin for higher intelligence ##"
rescue LoadError
  $ai = nil
  puts "## Warning: TTT-AI not found, using internal random AI ##"
end
```

```
def find_move(p)
  if $ai then
    return $ai.ai_findmove(p, 1, @board)
  end

  for i in 0..100
    move = rand(9)
    if evaluate_move(p, move) == ::MOVE_OK then
      return move
    end
   end
   return -1
  end
end
```

3D with OpenGL





RUDL Samples

Drawing a Cube with OpenGL

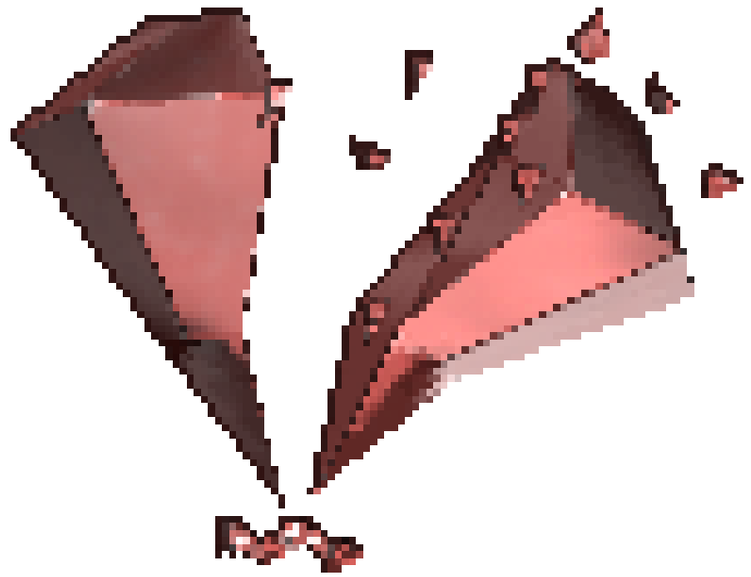
```
# draw a cube (6 quadrilaterals)
GL.Begin(GL::QUADS);                    # start drawing the cube.

# top of cube
GL.Color3f(0.0, 1.0, 0.0)               # Set The Color To Blue
GL.Vertex3f( 1.0,  1.0, -1.0)           # Top Right Of The Quad (Top)
GL.Vertex3f(-1.0,  1.0, -1.0)           # Top Left Of The Quad (Top)
GL.Vertex3f(-1.0,  1.0,  1.0)           # Bottom Left Of The Quad (Top)
GL.Vertex3f( 1.0,  1.0,  1.0)           # Bottom Right Of The Quad (Top)

# bottom of cube
GL.Color3f(1.0, 0.5, 0.0)               # Set The Color To Orange
GL.Vertex3f( 1.0, -1.0,  1.0)           # Top Right Of The Quad (Bottom)
GL.Vertex3f(-1.0, -1.0,  1.0)           # Top Left Of The Quad (Bottom)
GL.Vertex3f(-1.0, -1.0, -1.0)           # Bottom Left Of The Quad (Bottom)
GL.Vertex3f( 1.0, -1.0, -1.0)           # Bottom Right Of The Quad (Bottom)

# front of cube
# back of cube
# left
#right

GL.End();
```

Source by Martin Stannard: from RUDL samples

Creator and Lead Developer:
Jason Roelofs

http://ogrerb.rubyforge.org/index.html

OGRE.RB Samples

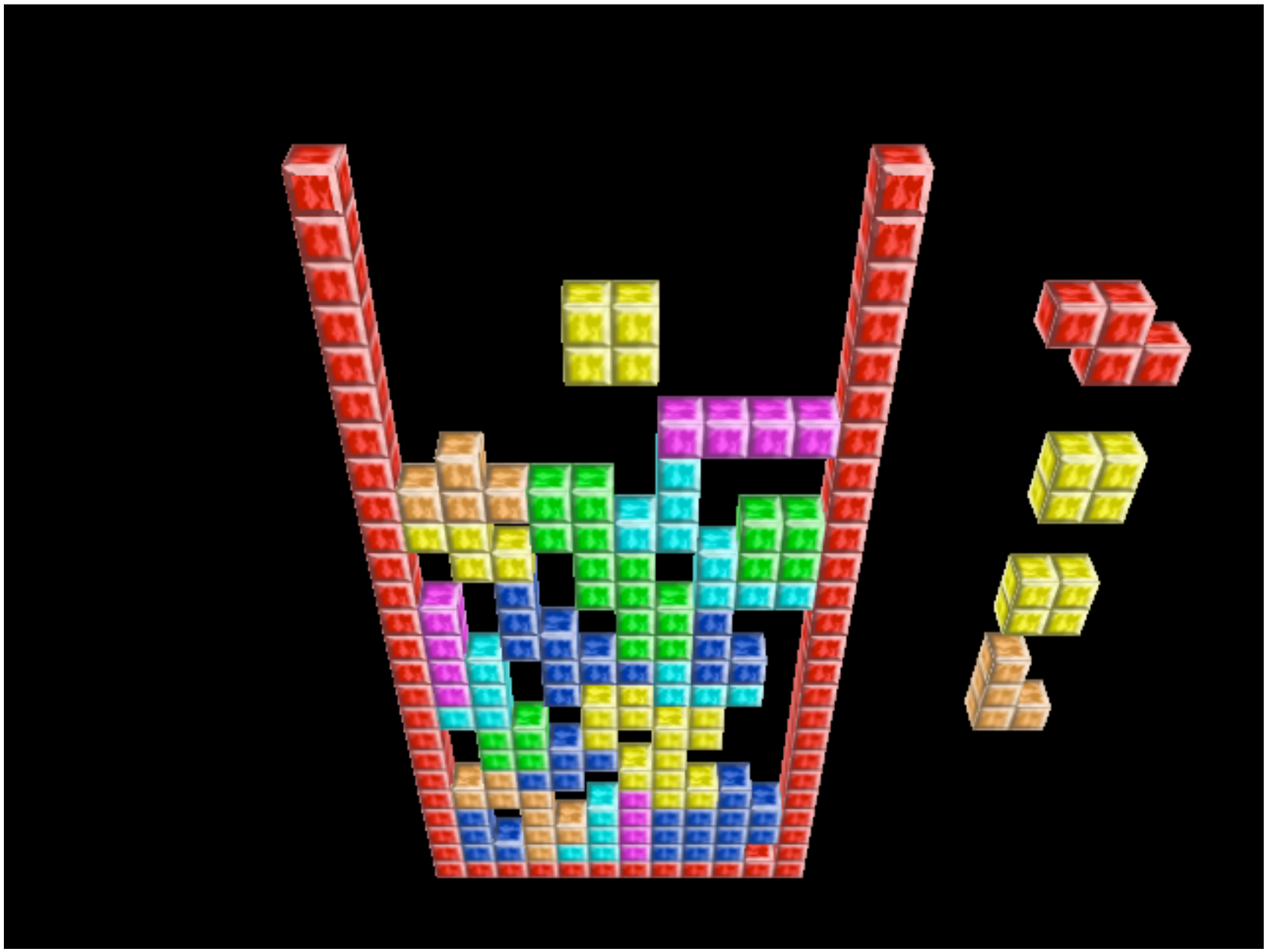


ShatteredRuby

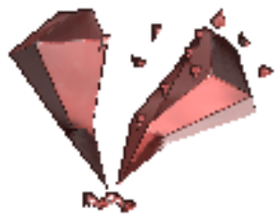
Co-Creators and Co-Lead Developers:
Martyn Garcia and Mikkel Garcia

http://groups.google.com/group/shatteredruby

ShatteredRuby



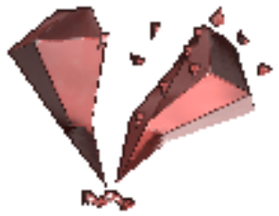
Shattered Tetris Tutorial



ShatteredRuby

Martyn Garcia on Rails vs. Shattered

Rails:
Databases and Tables (Model)
HTML/Javascript (View)

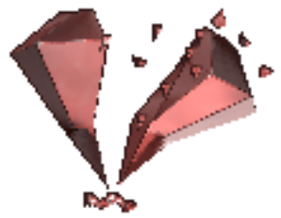


Shattered:
Game Logic (Model)
3D Rendering (View)

Looking at these two lists, it's easy to see that, while Rails has a simple text output view, Shattered has a very complex 3d rendering. And while Shattered has simple game logic, Rails has a very complex database backend.

Our focus is swapped**.**

The way Rails gains a ton of power is by making assumptions on the model. ..[o]ur view is the largest part..



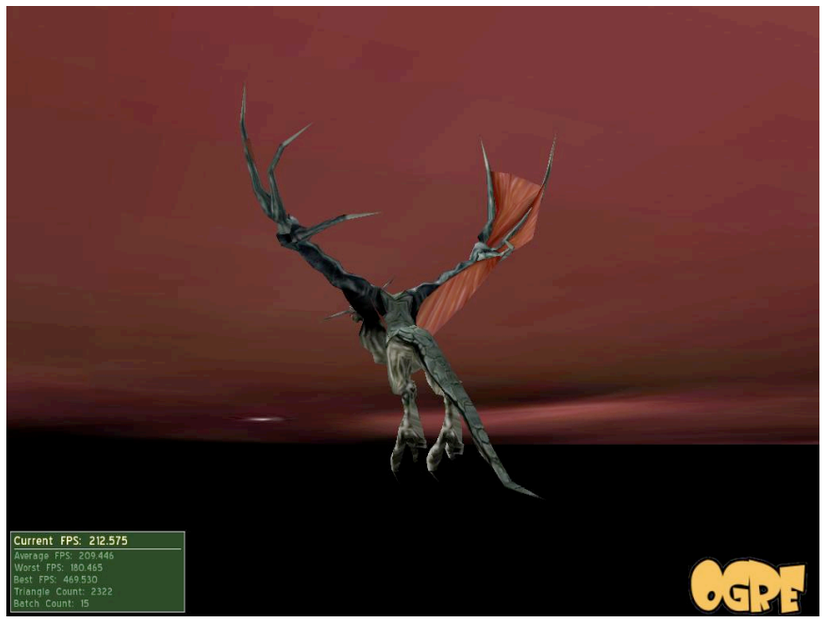
ShatteredRuby

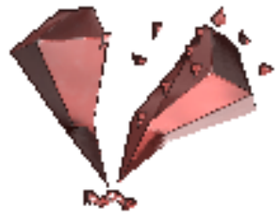
From the Ogre.rb Samples

```
class SkyPlaneApplication < Application
  def create_scene
    scene_manager.set_ambient_light ColourValue.new(0.5, 0.5, 0.5)
    plane = Plane.new
    plane.d = 5000
    plane.normal = -Vector3.UNIT_Y
    scene_manager.set_sky_plane(true, plane, "SpaceSkyPlane", 10000, 3)
    light = scene_manager.create_light("MainLight")
    light.set_position(20, 80, 50)
    dragon = scene_manager.create_entity("dragon", "dragon.mesh")
    scene_manager.root_scene_node.attach_object(dragon)
  end
end
```

Shattering the Code....

```
class SkyPlane
  light “light”, :position => (20,80,50)
end

class Dragon
end
```





ShatteredRuby

Symbols to Vector

```
:x.to_v          #v(1,0,0)
:y.to_v          #v(0,1,0)
:z.to_v          #v(0,0,1)
:up.to_v         #v(0,1,0)
:down.to_v       #v(0,-1,0)
:left.to_v       #v(-1,0,0)
:right.to_v      #v(1,0,0)
:forward.to_v    #v(0,0,1)
:backward.to_v   #v(0,0,-1)
:zero.to_v       #v(0,0,0)
```

ShatteredRuby

Timers

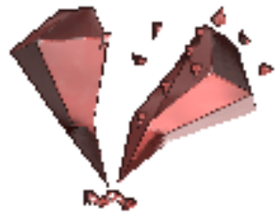
```
class Bomb < ...
  timer :in =>  30.seconds, :action => :blow_up
  timer :every => 1.second, :action => :shorten_fuse
  #
  def blow_up
  ...
  end
  #
  def shorten_fuse
  ...
  end
end
```

```
class Hero < ...
   key :pressed => :tab, :action => :power_up
   #
   def power_up
     ... # make the character stronger
     timer.in(30.seconds) do
       ... #make the character power down
     end
   end
   #
   def poison
     timer.every(1.second) do
       ... #make the character's life go down
     end
   end
 end
```

Examples from the Shattered Wiki: http://wiki.shatteredruby.com/index.php?title=Timers

ShatteredRuby

```
> shatter space_race
```



ShatteredRuby

```
> shatter space_race

create
create  app/models
create  app/views
create  app/media/common/programs
create  app/media/common/templates
create  doc
create  config
create  log
create  script
create  test/unit
create  vendor
create  vendor/plugins
create  Rakefile
create  README
create  config/ogre_plugins.windows.cfg
create  config/ogre.cfg
create  config/boot.rb
create  config/environment.rb
create  test/test_helper.rb
create  script/generate
create  script/runner
create  script/console
create  script/destroy
create  script/runner.rb
create  doc/README_FOR_APP
create  log/ogre.log
create  log/shattered.log
create  app/media/common/templates/basic.rmaterial
create  app/media/common/templates/offset_map.rmaterial
```

ShatteredRuby

Model\View\Actor Architecture

Actor

Model

View

```
space_race>script/generate actor flying_carpet
      exists  app/models/
      exists  test/unit/
      create  app/models/flying_carpet.rb
      create  test/unit/flying_carpet_test.rb
      exists  app/views/
      create  app/media/flying_carpet
      create  app/views/flying_carpet_view.rb
      create  app/media/flying_carpet/flying_carpet.mesh
      create  app/media/flying_carpet/flying_carpet.png
```

Model

Contains game logic.
Handles user input events.
When a method is called on the Model, and a method with the same name exists on the View -- the View's method is automatically invoked.

View

When a method is called on the Model, and a method with the same name exists on the View -- the View's method is automatically invoked.

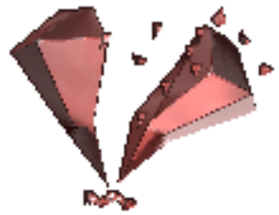
ShatteredRuby

State Management

```
space_race>script/generate state menu
      create  app/states/
      create  app/states/menu_state.rb

space_race>script/generate state game
      exists  app/states/
      create  app/states/game_state.rb
```

State

Represents routes through the application
Provides a top level where actors can interact

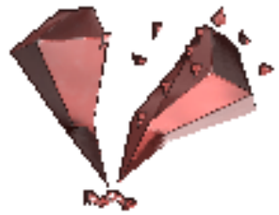


ShatteredRuby

OGRE Materials Format

```
// This is a comment
material walls/funkywall1
{
    // first, preferred technique
    technique
    {
        // first pass
        pass
        {
            ambient 0.5 0.5 0.5
            diffuse 1.0 1.0 1.0

            // Texture unit 0
            texture_unit
            {
                texture wibbly.jpg
                scroll_anim 0.1 0.0
                wave_xform scale sine 0.0 0.7 0.0 1.0
            }
            // Texture unit 1 (this is a multitexture pass)
            texture_unit
            {
                texture wobbly.png
                rotate_anim 0.25
                colour_op add
            }
        }
    }

    // Second technique, can be used as a fallback or LOD level
    technique
    {
        // .. and so on
    }

}
```

(from the OGRE manual:
http://www.ogre3d.org/docs/manual/manual_14.html)



ShatteredRuby

OGRE Materials Format

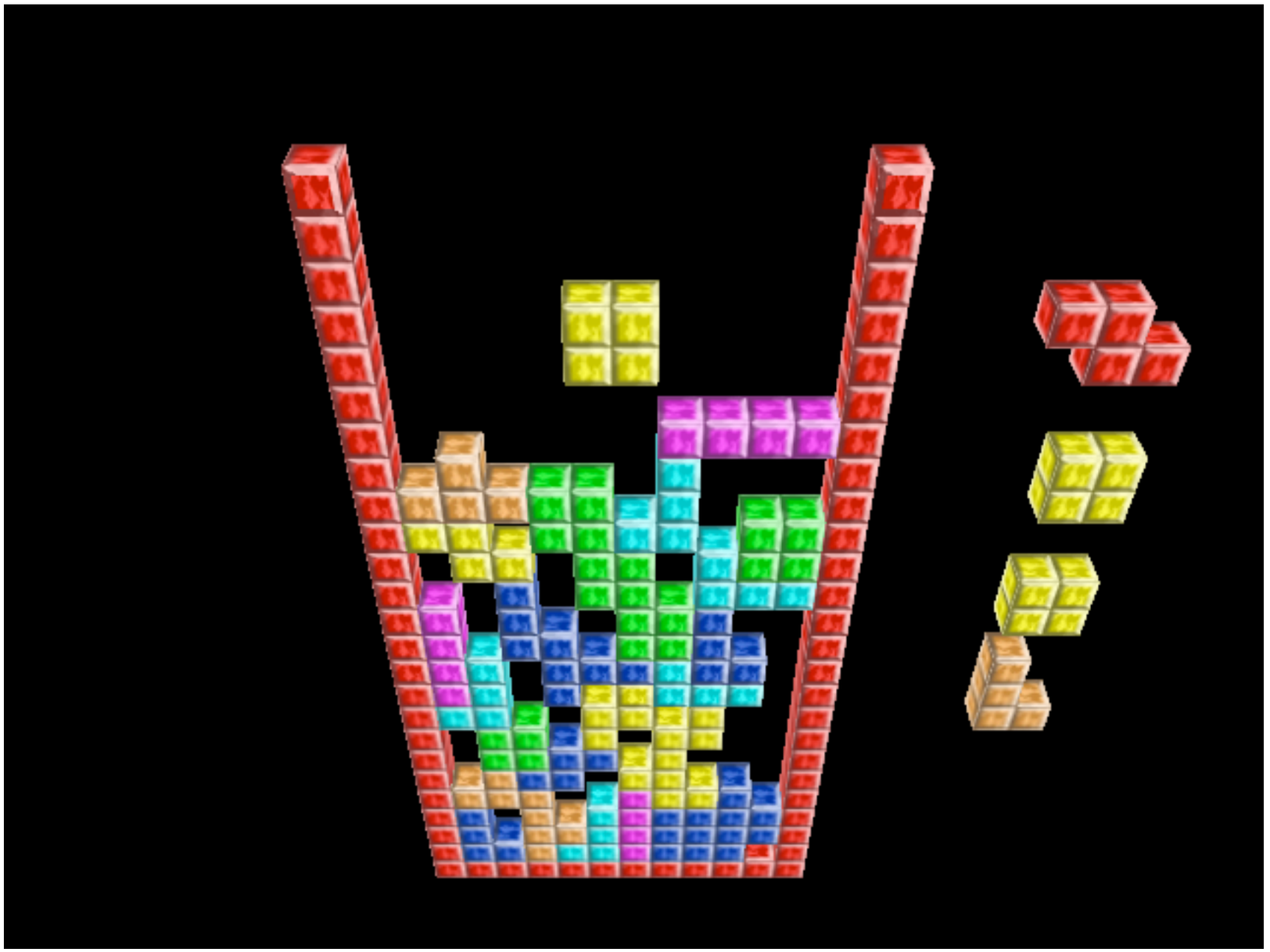
```
// This is a comment
material walls/funkywall1
{
    // first, preferred technique
    technique
    {
        // first pass
        pass
        {
            ambient 0.5 0.5 0.5
            diffuse 1.0 1.0 1.0

            // Texture unit 0
            texture_unit
            {
                texture wibbly.jpg
                scroll_anim 0.1 0.0
                wave_xform scale sine 0.0 0.7 0.0 1.0
            }
            // Texture unit 1 (this is a multitexture pass)
            texture_unit
            {
                texture wobbly.png
                rotate_anim 0.25
                colour_op add
            }
        }
    }

    // Second technique, can be used as a fallback or LOD level
    technique
    {
        // .. and so on
    }
}
```

```
// This is an example rmaterial, a basic single textured, uniformly lit mesh.
// required fields: texture

material <%= name %>
{
    technique
    {
        pass
        {
            lighting off
            texture_unit
            {
                texture <%= texture %>
            }
        }
    }
}
```

```
material :flying_carpet, :template => 'basic', :texture => 'tapestry.png'
```

(from the OGRE manual:
http://www.ogre3d.org/docs/manual/manual_14.html)

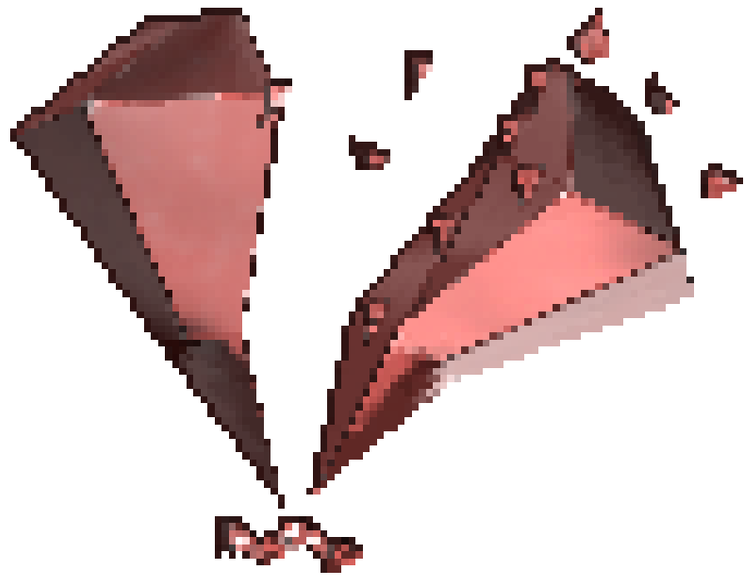
ShatteredRuby



Shattered Tetris Tutorial

That's right, there are no more "What, Shattered doesn't support per vertex bi-quadruple render transmutation?! - that's outragous!". Anything supported in Ogre will be supported in Shattered!

-- Mikkel Garcia

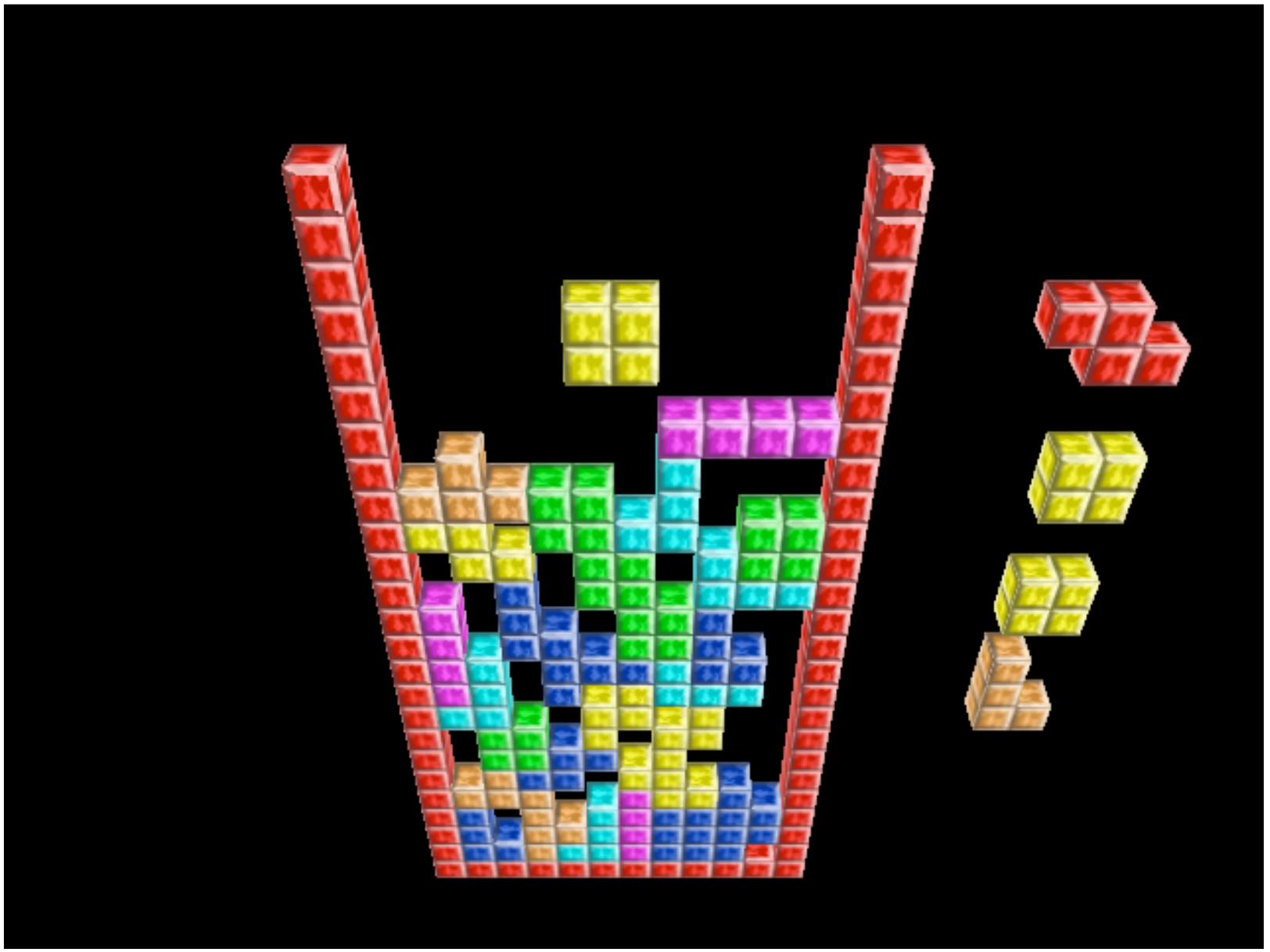


ShatteredRuby

Co-Creators and Co-Lead Developers:
Martyn Garcia and Mikkel Garcia

http://groups.google.com/group/shatteredruby

ShatteredRuby



Shattered Tetris Tutorial

High Art

on top of

LOW-LEVEL APIS

Building Games with Ruby

Agenda

Text-based Games

2D Games

3D Games

4D Games