T.R.P. Manual

DomDom

September 2014

Contents

Introduction

T.R.P. is an engine to make games for children so don't expect it to be the next UNITY killer ;-)

T.R.P. cannot compete with other engines like Cocos2dx or LibGDX but it should be enough to make small games in a relatively good environment.

T.R.P. makes you feel at home. You work on Windows or osx with your preferred text editor. You can then connect as many clients as you want (ios and/or android) to see the final result in real time.

Android and IOS versions are ready but still being tested and are not yet publicly available. But don't wait and start playing with the Win32 or OSX versions.

Rototo Revisions

Version : 0.2.0

```
- Radiobox
- Add GetRotation in all Widgets subclasses
- Watcher is not working in OSX
- Fix doc generation for global functions
- Better properties registration via scriptmanager
- Add property in generated doc
- ComboBox
- Localization system for Texts
- Add Set/Get in AngelScript to have nice scripts
- Add Vec2d versions of all AI functions
```

Version 0.1.1 :

```
- Add RadioBox
- Fix Shorcuts under OSX
```

Ratata Revisions

Version : 0.3.0

```
- Fix shorcuts on OSX (and in samples too)
- register file extension for ratata projects (OSX)
```

```
- See why errors are shown with a small font in ratata
- Icon is bad on osx for .rap projects
- Fix completion on Ratata
- Add the possibility to open script file by clicking on them in the finder
- Must better "find under cursor" in ratata
- Find in files in ratata
```

Documentation Revisions

Version 3:

```
- Remove Sublime Plugin
- Add Build Section
- The version is now on GitHub So a lot of small things changed.
```

Version 2:

```
- Rename all classes
- Add Android and IOS pages
```

Version 1:

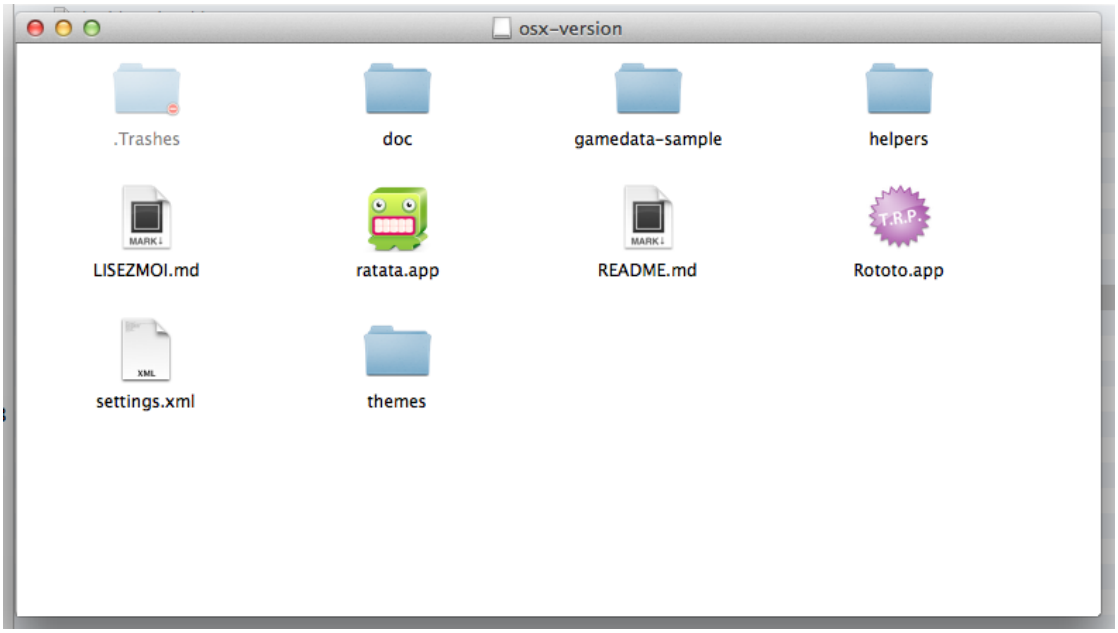
```
First Version
```

Installation

T.R.P. don't really need to be "installed".

OSX

Under OSX, T.R.P. comes as a disk image (dmg file). Just click on this dmg file to mount it. You will see something like this



Copy everything to a location of your choice Launch Rototo just to see the samples.

Windows

On Windows, T.R.P. comes as a zip file. Copy the content of the zip file to a location of your choice Launch Rototo just to see the samples.

However, it comes with a set of tools which, in turn, need to be. These tools are optional but T.R.P. was thought to use them. So your life will be much simpler if you do;-)

ShoeBox

Shoebox is a freeware tool(made with the Adobe Air SDK) for generating sprites sheets. T.R.P. use it a lot for sprites and animations.

To install it, you must first install the adobe air environment (if you don't already have it of course) Just go to http://get.adobe.com/fr/air/ and follow the instructions.

When the air environment is installed , take a version of Shoebox directly on the ShoeBox's website :

http://renderhjs.net/shoebox/

Tiled

Tiled is a well known tile editor. If you want to do a tile based game or just instantiate 2d objects with Tiled, take a version of Tiled directly on their website :

http://www.mapeditor.org

Configuration

You can edit the file settings.xml to configure T.R.P. For example, you can set the initial position of the window, it's scale, the default text editor, etc...

win32 sample :

```
<settings>
<position x = "50" y = "100"/>
<size w = "768" h="576"/>
<editor url="D:\tools\Sublime Text Build 3 3021 x64\sublime_text.exe" args="%s:%d:%d"/>
<datafolder directory="gamedata-shooter"/>
<autorestart value="0"/>
<allowdebug value="1"/>
<verbose value="0"/>
<logtofile value="1"/>
<server ip="192.168.0.1"/>
</settings>
```

OSX Sample :

```
<settings>
<position x = "50" y = "100"/>
<size w = "768" h="576"/>
<editor url="/usr/local/bin/subl" args="%s:%d:%d"/>
<datafolder directory="mygamedata"/>
<autorestart value="1"/>
<allowdebug value="0"/>
<verbose value="0"/>
<logtofile value="1"/>
<server ip="192.168.0.1"/>
</settings>
```

Here are the different options available :

- position : window's position in screen coordinates
- size : window's size in screen coordinates
- editor : url of your favorite text editor
- datafolder : folder name of your gamedata.
- autorestart : if value != 0 T.R.P. will restart automatically if the gamedata folder changed (add/modify/remove files)
- allowdebug : if value != 0 you can attach a debugger
- verbose : Set T.R.P. in verbose mode or not
- logtofile : Every T.R.P. log is written to a file called trp.log
- server : IP Address of the T.R.P. server (only if you want to connect to it as a client)

Keyboard Shortcuts

- Ctrl-R : restart
- Ctrl-F : Open the explorer (or Finder) into the script directory
- Ctrl-H : Open the HTML Help File

How to see Debug Logs

With Ratata

Simply open Ratata and connect it to Rototo. Then go to the console widget to see your logs.

Without Ratata - Pro way

If you want to see all logs (included TRP internal logs), the best option is to see OutputDebugString (and eventually turn on the verbose mode)

- Under Windows

Use the application Dbgview.exe located in the Tools directory

Download it Here

- Under OSX

Use the OSX Console Application located in Utility/Console.

The Easy way

If you only want to see your logs, it's easier to turn on the logtofile option and use a file logger to see trp.log modifications in real time. For example you can use the free version of BareTail

Download it Here

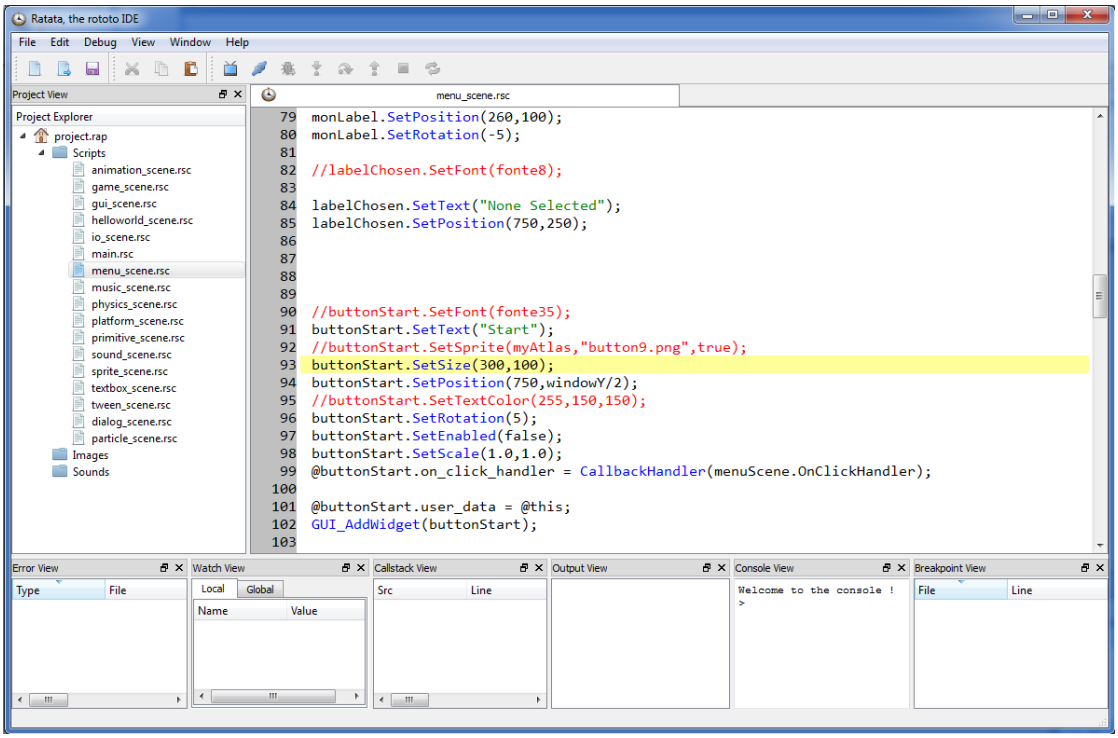
or Glogg

Download it Here

Build

Building T.R.P. is quite complex beacause ituses quite a lot dependencies. If you don't succeed, please don't hesitate to contact me on (www.veed.fr)[www.veed.fr]

Ratata : the I.D.E.



Tutorial

Basic principle

T.R.P. use a script langage(AngelScript) and search for its data in the gamedata directory (at the same level as the executable directory). If you want to use another folder (or if you work on multiple projects), you can indicate your gamedata folder in the file settings.xml

```
<settings>
...
<datafolder directory="mygamedata"/>
...
</settings>
```

Learn the AngelScript syntax

AngelScript is very close to the c++ langage but there are still some differences (such as references) Basically, no new or delete. Pointers are replaced by @ You can simulate new and delete by usgin a scope trick . Let's see this example :

```
class A;

A @refA; // This is a reference (= a pointer)

// This is a new

{
A a;    // The object a is instantiated, it should be deleted at the end of the scope

@refA = @a; // But in fact, it will stay allocated because there is still a reference on it
}

// This is a delete

@refA = NULL; // No more reference on object 'a' so we can finally call delete
```

For any additionnal informations , please go directly to the AngelScript website

Here

Try the samples

T.R.P. comes by default with a lot of samples.

Simply launch the application, you'll fall over them.

It is strongly advised to open these script files (extension .rsc = Rototo Script File) and study them.

Your first T.R.P. project (aka HelloWorld version 1)

T.R.P. needs at least one file called main.rsc in the gamedata directory

Take your favorite text editor and create a file called main.rsc. Paste the code below. This is the minimal TRP program.

- OnInit
- OnUpdate
- OnRender
- OnShutdown

```
void OnInit()
{
UTI_Log("Hello World");
}

void OnUpdate(uint64 _delta)
```

```
{
UTI_Exit();
}

void OnRender(uint64 _delta)
{

}

void OnShutdown()
{
}
```

You must provide at least those 4 functions , they will be automatically called by T.R.P. during the execution of your game.

- OnInit is called only once at the beginning of the game (or each time you restart the game)
- OnUpdate is called every engine frame before OnRender
- OnRender is called every engine frame
- OnShutdown is called only once at the end of the game (when the window close or if you call UTI_Exit)

Enhanced HelloWorld (Version 2)

Let's do the same HelloWorld program but this time by using the scene manager provided in the samples and with a Label (to have something more graphical than the system logs)

In the gamedata directory you will find a minimal scene manager. So let's open the file scene_manager.rsc to see what's inside ;-)

We have 2 classes :

- a SceneManager class
- a Scene class

The SceneManager class includes a "ChangeScene" method void ChangeScene(Scene @ _newScene) wich is more interesting than the others.

The global idea is to have multiple scenes, 1 for the game, 1 for menus, ... and to navigate from one to another by calling the ChangeScene method.

Let's do this 2 steps process :

1. Write a game scene wich inherits from Scene
2. Write a new "main.rsc" file to launch directly our new scene (e.g. by calling a changeScene with it)

Create our game scene

Create a text file called “game_scene.rsc” paste the following code into it:

```
class GameScene:Scene
    {


    void Init()
        {
        maFonte.Load("casual.ttf",50);

        monLabel.SetFont(maFonte);
        monLabel.SetText("Hello World");

        }

    void OnUpdate(uint64 _delta)
        {

        }

    void OnRender(uint64 _delta)
        {
        monLabel.Render();
        }

    CFont   maFonte;
    CLabel  monLabel;
    }
```

Font Label

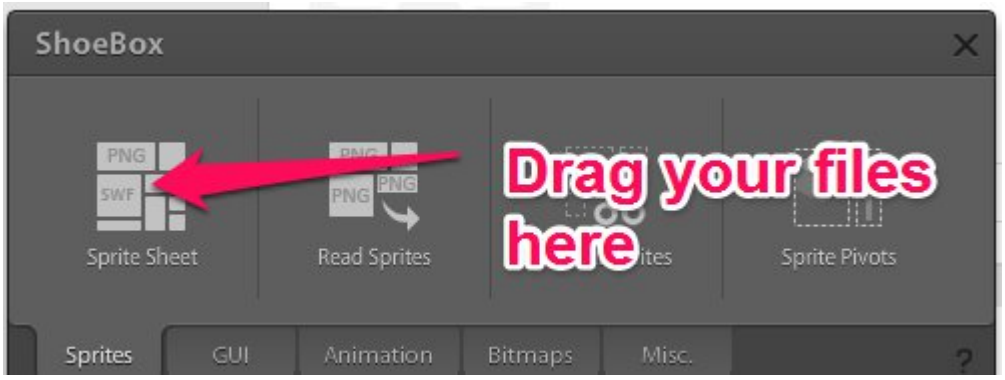
Of course, you noticed that the scene manager provides is minimalist, nothing prevents you to create your own with more features (of pushScenes / Popscene example)

Create the new main.rsc file

create a text file called main.rsc. Paste the code below.

Your first Sprite

T.R.P. does not use images directly. It uses atlas (= spriteSheets) to optimize drawings. The recommended tool to create spritesheets is Shoebox. Drag your images into shoebox



Shoebox will create 2 files for you.

- sheet.png : contains all your images into one big spritesheet
- sheet.xml : an xml file containing the location of each of your files.

Let's see an example of xml file produced by ShoeBox

```
<TextureAtlas imagePath="sheet.png">
    <SubTexture name="Bullet.png" x="0" y="116" width="28" height="9"/>
    <SubTexture name="Player.png" x="0" y="74" width="39" height="40"/>
    <SubTexture name="Pointer.png" x="41" y="74" width="22" height="31"/>
    <SubTexture name="Seeker.png" x="0" y="0" width="40" height="30"/>
    <SubTexture name="Wanderer.png" x="0" y="32" width="40" height="40"/>
</TextureAtlas>
```

You can see that the name of your images stay the same, but now each image has coordinates into the big spritesheet.

Thus to load your image into T.R.P. this is a 2 steps process

- Load the Atlas (= the spritesheet)
- Load the sprite

1. Load the Atlas

```
Atlas atlas;
atlas.Load("sheet"); //without extension because trp will read both the xml and image file
```

2. Load the sprite

```
Sprite mySprite;
mySprite.Load(atlas,"Player.png"); // atlas is the preloaded atlas, the 2nd param is the orig
```

Here's the minimal sample for sprites

```
class SpriteScene:Scene
    {
    Sprite mySprite;
    Atlas  myAtlas;

    void Init()
        {
        myAtlas.Load("graphics/sheet");
        mySprite.Load(myAtlas,"sheep.png");
        mySprite.SetPosition(windowX/2,windowY/2);
        }

    void OnUpdate(uint64 _delta)
        {

        }

    void OnRender(uint64 _delta)
        {
        WND_ClearWithColor(80,80,80,255);
        mySprite.Render();
        }

    void OnShutdown()
        {
        myAtlas.UnLoad();
        }

    }
```

Of course, this is a minimal sample, you can do a lot more with sprites, Please take a look at the reference documentation for sprites

Sprite

Your First Animation

Concerning animations, T.R.P. uses sequences of images. Again, T.R.P. gets all images from a spritesheet.

In order to play an animation you must

1. Export your animation as a sequence of images
2. Build a spritesheet with all your images (again use shoebox for this)

You can use any nomenclature you want, the only constraint is to start at frame 0 and end at nbframe-1; The magic then append in the Load function where you can explain T.R.P. the naming you want to use. Let me try to explain this better throught an example :

Let's say you have a ten frames animation called horse_run.

you can use any naming and any image format you want for your frames for example horse-001.png Horse_00001.jpg but you must tell T.R.P. how your namaing works.

For example, if you want to call each frame like this : horse-000.png ... horse-009.png

then the "FORMAT" will be "horse","%s-%03d.png"

some other samples :

for Horse_00000.jpg to Horse_00009.jpg the format will be "horse","%s_%05d.jpg"

Load(myAtlas,"cocci","%s_%05d.png",10);

The minimal sample you can have to play an animation is this one

```
class AnimationScene:Scene
    {
    Animation myAnim;
    Atlas  myAtlas;

    void Init()
        {
        myAtlas.Load("graphics/broute");
        myAnim.Load(myAtlas,"broute","%s_%05d.tga",30);
        myAnim.SetFPS(12);
        myAnim.Play(E_MODE_NORMAL,C_INFINITE_LOOP);
        }

    void OnUpdate(uint64 _delta)
        {
        myAnim.Update(_delta);
        }

    void OnRender(uint64 _delta)
        {
        WND_ClearWithColor(80,80,80,255);
        myAnim.Render();
        }

    void OnShutdown()
        {
        myAtlas.UnLoad();
        myAnim.UnLoad();
        }

    }
```

Of course, this is a minimal sample, you can do a lot more with animations, Please take a look at the reference documentation for animations

Animation

Your First Sound

```
class SoundScene:Scene
    {


    void Init()
        {
        mySFX.Load("sounds/sheep.ogg");
        mySFX.Play();
        }

    void OnUpdate(uint64 _delta)
        {

        }

    void OnRender(uint64 _delta)
        {
        WND_ClearWithColor(80,80,80,255);
        }

    void OnShutdown()
        {
        }

    Sound  mySFX;
    }
```

Of course, this is a minimal sample, you can do a lot more with sounds, Please take a look at the reference documentation for sound

Sound

Your First Music

```
class MusicScene:Scene
    {


    void Init()
```

```
    {
    myMusic.Load("sounds/invitation.mod");
    myMusic.Play();
    }

void OnUpdate(uint64 _delta)
    {

    }

void OnRender(uint64 _delta)
    {
    WND_ClearWithColor(80,80,80,255);
    }

void OnShutdown()
    {
    }

Music  myMusic;
}
```

Of course, this is a minimal sample, you can do a lot more with music, Please take a look at the reference documentation for music

Music

Your First Button

Of course, this is a minimal sample, you can do a lot more with buttons, Please take a look at the reference documentation for buttons

Button

Your First ListBox

Of course, this is a minimal sample, you can do a lot more with listboxes, Please take a look at the reference documentation for listboxes

ListBox

Input/Output

```
int handle;
handle  = IO_Open("slot0.txt","w");
string ts = "Handle =  "+formatInt(handle,"");
UTI_Log(ts);
IO_WriteString(handle,"coucou");
IO_WriteString(handle,"coucou2");
IO_WriteInt(handle,24);
IO_Close(handle);
```

```
int handle;
handle  = IO_Open("slot0.txt","r");
ts = "Handle =  "+formatInt(handle,"");
UTI_Log(ts);
string test;
int testInt;
IO_ReadString(handle,test);
UTI_Log(test);
IO_ReadString(handle,test);
UTI_Log(test);
IO_ReadInt(handle,testInt);
ts = "testInt =  "+formatInt(testInt,"");
UTI_Log(ts);

IO_Close(handle);
```

Android

IOS

Script Reference

Class Animation

Members:

void Load(Atlas @ ***atlas, string &in*** **name,string &in** ***format,int*** **nbFrames)**

Role : Load an animation sheet from an atlas given a specific format and a frame number

- _atlas : Reference to the atlas containing the animation
- _name : name of the animation

Sample :

```
int toto
toto = Load(titi);
tata
```

void SetFPS(int fps)

Role : Set the speed of the animation in frames per second

- _fps : frames per second

void Update(uint64 elapsed)

void Render()

Class Atlas

Members:

Atlas()

void Load(string &in *file,int* flags=13)

void LoadFromImage(string &in *file,int* flags=13)

void UnLoad()

Class Button

Members:

void SetText(string &in newText)

void Render()

void SetSize(int w,int h)

void SetPosition(int x,int y,int from=0)

void SetFont(Font @ font)

void SetSprite(int index,Atlas @ atlas, string &in name, bool ninePatch = false)

void SetTextColor(uint8 r=255,uint8 g=255,uint8 b=255,uint8 a=255)

void SetRotation(float angle)

bool Touched(int x,int y)

void SetScale(double xFactor,double yFactor)

void SetEnabled(bool value)

void SetType(int type)

void SetState(int state)

Class Font

Members:

Class Label

Members:

void SetText(string &in *newText,bool* justified = true)

void Render()

void SetFont(Font @ font)

void SetColor(uint8 *r=255,uint8* g=255,uint8 *b=255,uint8* a=255)

void SetPosition(int x,int y,int from = 0)

bool Touched(int *x,int* y)

void SetRotation(float angle)

void SetShaded(bool value)

Class ListBox

Members:

Class Emitter

Members:

Emitter()

void Load(Atlas @ atlas,string &in file,int flags=13)

void SetPosition(int x,int y,int from=0)

void Update(uint64 elapsed)

void Render()

Class Body

Members:

Class Primitive

Members:

void Render()

void SetColor(uint8 r=255,uint8 g=255,uint8 b=255,uint8 a=255)

void SetPosition(int x,int y,int from = 0)

Class Music

Members:

Class Sound

Members:

Class Sprite

Members:

void Load(Atlas @ atlas, string &in name)

void SetPosition(int x,int y,int from = 0)

bool Touched(int x,int y)

void SetScale(double xFactor,double yFactor)

void SetRotation(float angle)

Class TextBox

Members: