Zappy

Generated by Doxygen 1.9.1

Chapter 1

raylib bindings and wrappers

Some people ported raylib to other languages in form of bindings or wrappers to the library. Here is a list with all the ports available. Feel free to send a PR if you know of any binding/wrapper not in this list.

1.0.1 Language Bindings

name	raylib version	language	license	repo
raylib	**4.2**	C/C++	Zlib	https://github.com/raysan5/raylib
raylib-boo	3.7	Boo	MIT	https://github.com/Rabios/raylib-boo
Raylib-cs	**4.2**	C#	Zlib	https://github.com/ChrisDill/Raylib-cs
Raylib-CsLo	**4.2**	C#	MPL-2.0	https://github.com/NotNotTech/Raylib-CsLo
cl-raylib	**4.0**	Common Lisp	MIT	https://github.com/longlene/cl-raylib
claylib/wrap	**4.2**	Common Lisp	Zlib	https://github.com/defun-games/claylib
chez-raylib	auto	Chez Scheme	GPLv3	https://github.com/Yunoinsky/chez-raylib
raylib-cr	**4.5-dev (7e7939e)**	Crystal	Apache-2.0	https://github.com/sol-vin/raylib-cr
ray-cyber	**4.2**	Cyber	MIT	https://github.com/fubark/ray-cyber

name	raylib version	language	license	repo
raylib-c3	**4.5-dev**	C3	MIT	https://github.com/Its-Kenta/Raylib-C3
dart-raylib	**4.0**	Dart	MIT	https://gitlab.com/wolfenrain/dart-raylib
bindbc-raylib3	**4.0**	D	BSL-1.0	https://github.com/o3o/bindbc-raylib3
dray	**4.2**	D	Apache-2.0	https://github.com/redthing1/dray
raylib-d	**4.2**	D	Zlib	https://github.com/schveiguy/raylib-d
dlang_raylib	**4.0**	D	MPL-2.0	https://github.com/rc-05/dlang_raylib
rayex	3.7	elixir	Apache-2.0	https://github.com/shiryel/rayex
raylib-factor	**4.0**	Factor	BSD	https://github.com/factor/factor/blob/maste factor
raylib-freebasic	**4.2**	FreeBASIC	MIT	https://github.com/WIITD/raylib-freebasic
raylib-go	**4.2**	Go	Zlib	https://github.com/gen2brain/raylib-go
raylib-guile	auto	Guile	Zlib	https://github.com/petelliott/raylib-guile
gforth-raylib	3.5	Gforth	MIT	https://github.com/Arnaut-Daniel/gforth-raylib
h-raylib	4.5-dev	Haskell	Apache-2.0	https://github.com/Anut-py/h-raylib
raylib-hx	**4.2**	Haxe	Zlib	https://github.com/foreignsasquatch/raylib-
hb-raylib	3.5	Harbour	MIT	https://github.com/Marcos-Leonardo-Mendez-Gerencir/hb-raylib

name	raylib version	language	license	repo
jaylib	**4.2**	Java	GPLv3+CE	https://github.com/electronstudio/jaylib/
raylib-j	**4.0**	Java	Zlib	https://github.com/CreedVI/Raylib-J
raylib.jl	**4.2**	Julia	Zlib	https://github.com/irishgreencitrus/raylib.jl
kaylib	3.7	Kotlin/native	?	https://github.com/electronstudio/kaylib
kaylib	**4.5-dev**	Kotlin/native	Zlib	https://codeberg.org/Kenta/Kaylib
raylib-lua	**4.2**	Lua	ISC	https://github.com/TSnake41/raylib-lua
raylua	**4.0**	Lua	MIT	https://github.com/Rabios/raylua
nelua-raylib	4.0	nelua	MIT	https://github.com/AKDev21/nelua-raylib
Raylib-Nelua	**4.5-dev**	nelua	MIT	https://github.com/Its-Kenta/Raylib-Nelua
NimraylibNow!	4.2	Nim	MIT	https://github.com/greenfork/nimraylib_now
raylib-Forever	auto	Nim	?	https://github.com/Guevara-chan/Raylib-Forever
naylib	auto	Nim	MIT	https://github.com/planetis-m/naylib
node-raylib	**4.0**	Node.js	Zlib	https://github.com/Rob-Loach/node-raylib

name	raylib version	language	license	repo
raylib_odin_bindings	4.0-dev	Odin	MIT	https://github.com/Deathbat2190/raylib_odin_bindings
raylib-odin	**4.0**	Odin	BSD-3Clause	https://github.com/odin-lang/Odin/tree/master/vendor/ray
raylib-ocaml	**4.2**	OCaml	MIT	https://github.com/tjammer/raylib-ocaml
TurboRaylib	**4.2**	Object Pascal	MIT	https://github.com/turborium/TurboRaylib
Ray4Laz	**4.2**	Free Pascal	Zlib	https://github.com/Guva/Code/Ray4Laz
Raylib.4.0.Pascal	**4.0**	Free Pascal	Zlib	https://github.com/sysrpl/Raylib.4.0.Pascal
pyraylib	3.7	Python	Zlib	https://github.com/Ho011/pyraylib
raylib-python-cffi	**4.2**	Python	EPL-2.0	https://github.com/electronstudio/raylib-py
raylibpyctbg	**4.2**	Python	MIT	https://github.com/overdev/raylibpyctbg
raylib-py	**4.2**	Python	MIT	https://github.com/overdev/raylib-py
raylib-python-ctypes	**4.2**	Python	MIT	https://github.com/s/Dos280/raylib-python-ctypes
raylib-php	3.5	PHP	Zlib	https://github.com/joseph-montanez/raylib-p
raylib-phpcpp	3.5	PHP	Zlib	https://github.com/oraoto/raylib-phpcpp
raylibr	**4.0**	R	MIT	https://github.com/jeroenjanssens/raylibr
raylib-rs	3.5	Rust	Zlib	https://github.com/deltaphc/raylib-rs

name	raylib version	language	license	repo
Relib	3.5	ReCT	?	https://github.com/RedCube-Dev-Byte-Space/Relib
racket-raylib	**4.0**	Racket	MIT/Apache-2.0	https://github.com/eutro/racket-raylib
raylib-swift	**4.0**	Swift	MIT	https://github.com/STREGAs-Gate/Raylib
raylib-scopes	auto	Scopes	MIT	https://github.com/salotz/raylib-scopes
raylib-smallBasic	4.1-dev	SmallBASIC	GPLv3	https://github.com/smallbasic/smallbasic.plugins/tree/master/raylib
raylib-umka	**4.2**	Umka	Zlib	https://github.com/robloach/raylib-umka
raylib.v	**4.2**	V	Zlib	https://github.com/irishgreencitrus/raylib.v
raylib-vapi	**4.2**	Vala	Zlib	https://github.com/lxmcf/raylib-vapi
raylib-wren	**4.0**	Wren	ISC	https://github.com/TSnake41/raylib-wren
raylib-zig	**4.2**	Zig	MIT	https://github.com/Not-Nik/raylib-zig
raylib.zig	**4.2**	Zig	MIT	https://github.com/ryupold/raylib.zig
hare-raylib	auto	Hare	Zlib	https://git.sr.ht/~evantj/hare-raylib
raylib-sunder	auto	Sunder	0BSD	https://github.com/ashn-dot-dev/raylib-sunder
rayed-bqn	auto	BQN	MIT	https://github.com/Brian-ED/rayed-bqn

1.0.2 Utility Wrapers

These are utility wrappers for specific languages, they are not required to use raylib in the language but may adapt the raylib API to be more inline with the language's pardigm.

name	raylib version	language	license	repo
raylib-cpp	**4.2**	C++	Zlib	https://github.com/robloach/raylib-cpp
claylib	**4.2**	Common Lisp	Zlib	https://github.com/defun-games/claylib

1.0.3 Older or Unmaintained Language Bindings

These are older raylib bindings that are more than 2 versions old or have not been maintained.

name	raylib version	language	repo
raylib-cppsharp	2.5	C#	https://github.com/phxvyper/raylib-cppsharp
RaylibFS	2.5	F#	https://github.com/dallinbeutler/RaylibFS
raylib_d	2.5	D	https://github.com/Sepheus/raylib_d
bindbc-raylib	3.0	D	https://github.com/o3o/bindbc-raylib
go-raylib	3.5	Go	https://github.com/chunqian/go-raylib
raylib-goplus	2.6-dev	Go	https://github.com/Lachee/raylib-goplus
ray-go	2.6-dev	Go	https://github.com/hecate-tech/ray-go
raylib-luamore	3.0	Lua	https://github.com/HDPLocust/raylib-luamore
LuaJIT-Raylib	2.6	Lua	https://github.com/Bambofy/LuaJIT-Raylib
raylib-lua-sol	2.5	Lua	https://github.com/RobLoach/raylib-lua-sol
raylib-lua-ffi	2.0	Lua	https://github.com/raysan5/raylib/issues/693
raylib-lua	1.7	Lua	https://github.com/raysan5/raylib-lua
raylib-nelua	3.0	Nelua	https://github.com/Andre-LA/raylib-nelua
raylib-nim	2.0	Nim	https://github.com/Skrylar/raylib-nim
raylib-Nim	1.7	Nim	https://gitlab.com/define-private-public/raylib-Nim
nim-raylib	3.1-dev	Nim	https://github.com/tomc1998/nim-raylib
raylib-haskell	2.0	Haskell	https://github.com/DevJac/raylib-haskell

name	raylib version	language	repo
raylib-cr	2.5-dev	Crystal	https://github.com/AregevDev/raylib-cr
raylib.cr	2.0	Crystal	https://github.com/sam0x17/raylib.cr
cray	1.8	Crystal	https://gitlab.com/Zatherz/cray
raylib-pas	3.0	Pascal	https://github.com/tazdij/raylib-pas
raylib-pascal	2.0	Pascal	https://github.com/drezgames/raylib-pascal
Graphics-Raylib	1.4	Perl	https://github.com/athreef/Graphics-Raylib
raylib-ruby	2.6	Ruby	https://github.com/a0/raylib-ruby
raylib-ruby-ffi	2.0	Ruby	https://github.com/D3nX/raylib-ruby-ffi
raylib-mruby	2.5-dev	mruby	https://github.com/lihaochen910/raylib-mruby
raylib-java	2.0	Java	https://github.com/XoanaIO/raylib-java
clj-raylib	3.0	Clojure	https://github.com/lsevero/clj-raylib
QuickJS-raylib	3.0	QuickJS	https://github.com/sntg-p/QuickJS-raylib
raylib-duktape	2.6	JavaScript (Duktape)	https://github.com/RobLoach/raylib-duktape
raylib-v7	3.5	JavaScript (v7)	https://github.com/Rabios/raylib-v7
raylib-chaiscript	2.6	ChaiScript	https://github.com/RobLoach/raylib-chaiscript
raylib-squirrel	2.5	Squirrel	https://github.com/RobLoach/raylib-squirrel
racket-raylib-2d	2.5	Racket	https://github.com/arvyy/racket-raylib-2d
raylib-php-ffi	2.4-dev	PHP	https://github.com/oraoto/raylib-php-ffi
raylib-haxe	2.4	Haxe	https://github.com/ibilon/raylib-haxe
ringraylib	2.6	Ring	https://github.com/ringpackages/ringraylib
raylib-scm	2.5	Chicken Scheme	https://github.com/yashrk/raylib-scm
raylib-chibi	2.5	Chibi-Scheme	https://github.com/Vincent-Toups/raylib-chibi
raylib-gambit-scheme	3.1-dev	Gambit Scheme	https://github.com/georgjz/raylib-gambit-scheme
Euraylib	3.0	Euphoria	https://github.com/gAndy50/Euraylib

name	raylib version	language	repo
raylib-odin	3.0	Odin	https://github.com/kevinw/raylib-odin
vraylib	3.5	V	https://github.com/waotzi/vraylib
raylib-vala	3.0	Vala	https://code.guddler.uk/mart/raylibVapi
raylib-jai	3.1-dev	Jai	https://github.com/kujukuju/raylib-jai
ray.zig	2.5	Zig	https://github.com/Bit-Puffin/zig-raylib-experiments
raylib-Ada	3.0	Ada	https://github.com/mimo/raylib-Ada
jaylib	3.0	Janet	https://github.com/janet-lang/jaylib
raykit	?	Kit	https://github.com/Gamerfiend/raykit
ray.mod	3.0	BlitzMax	https://github.com/bmx-ng/ray.mod
raylib-mosaic	3.0	Mosaic	https://github.com/pluckyporcupine/raylib-mosaic
raylib-xdpw	2.6	XD Pascal	https://github.com/vtereshkov/raylib-xdpw
raylib-carp	3.0	Carp	https://github.com/pluckyporcupine/raylib-carp
raylib-fb	3.0	FreeBasic	https://github.com/IchMagBier/raylib-fb
raylib-purebasic	3.0	PureBasic	https://github.com/D-a-n-i-l-o/raylib-purebasic
raylib-ats2	3.0	ATS2	https://github.com/mephistopheles-8/raylib-ats2
raylib-beef	3.0	Beef	https://github.com/M0n7y5/raylib-beef
raylib-never	3.0	Never	https://github.com/never-lang/raylib-never
raylib.cbl	2.0	COBOL	*code examples*

Missing some language or wrapper? Feel free to create a new one! :)

Usually, raylib bindings follow the convention: `raylib-{language}`

Let me know if you're writing a new binding for raylib, I will list it here!

Chapter 2

Contributing to raylib

Hello contributors! Welcome to raylib!

Do you enjoy raylib and want to contribute? Nice! You can help with the following points:

- `C programming` - Can you write/review/test/improve the code?
- `Documentation/Tutorials/Example` - Can you write some tutorial/example?
- `Porting to other platforms` - Can you port/adapt/compile raylib on other systems?
- `Web Development` - Can you help `with the website`?
- `Testing` - Can you find some bugs in raylib?

This document contains a set of guidelines to contribute to the project. These are mostly guidelines, not rules. Use your best judgement, and feel free to propose changes to this document in a pull request.

2.0.1 raylib philosophy

- raylib is a tool to **ENJOY** videogames programming, every function in raylib is designed as a mini-tutorial on itself.
- raylib is **SIMPLE** and **EASY-TO-USE**, I tried to keep it compact with a small set of functions, if a function is too complex, better not including it.
- raylib is open source and free; educators and institutions can use this tool to **TEACH** videogames programming completely for free.
- raylib is collaborative; contribution of tutorials / code examples / bug fixes / code comments are highly appreciated.
- raylib's license (and its external libs respective licenses) allow using raylib on commercial projects.

2.0.2 Some interesting reads to start with

- raylib history
- `raylib architecture`
- [raylib license](LICENSE)
- raylib roadmap

`raylib Wiki` contains some information about the library and is open to anyone for edit. Feel free to review it if required, just take care not to break something.

2.0.3 raylib C coding conventions

Despite being written in C, raylib does not follow the standard Hungarian notation for C, it follows Pascal-case/camel-case notation, more common on C# language. All code formatting decisions have been carefully taken to make it easier for students/users to read, write and understand code.

Source code is extensively commented for that purpose, raylib primary learning method is:

```
Learn by reading code and examples
```

For detailed information on building raylib and examples, please check raylib Wiki.

2.0.4 Opening new Issues

To open new issue for raylib (bug, enhancement, discussion...), just try to follow these rules:

- Make sure the issue has not already been reported before by searching on GitHub under Issues.
- If you're unable to find an open issue addressing the problem, open a new one. Be sure to include a title and clear description, as much relevant information as possible, and a code sample demonstrating the unexpected behavior.
- If applies, attach some screenshot of the issue and a .zip file with the code sample and required resources.
- On issue description, add a brackets tag about the raylib module that relates to this issue. If don't know which module, just report the issue, I will review it.
- You can check other issues to see how is being done!

2.0.5 Sending a Pull-Request

- Make sure the PR description clearly describes the problem and solution. Include the relevant issue number if applicable.
- Don't send big pull requests (lots of changelists), they are difficult to review. It's better to send small pull requests, one at a time.
- Verify that changes don't break the build (at least on Windows platform). As many platforms where you can test it, the better, but don't worry if you cannot test all the platforms.

2.0.6 Contact information

If you have any doubt, don't hesitate to contact me!. You can write me a direct mail but you can also contact me on the following networks:

- raylib Discord - A direct communication channel for project discussions.
- raylib twitter - My personal twitter account, I usually post about raylib, you can send me PMs.
- raylib reddit - A good place for discussions or to ask for help.
- raylib web - On top-right corner there is a bunch of networks where you can find me.

Thank you very much for your time! :)

Chapter 3

C Coding Style Conventions

Here it is a list with some of the code conventions used by raylib:

Code element	Convention	Example
Defines	ALL_CAPS	`#define PLATFORM_DESKTOP`
Macros	ALL_CAPS	`#define MIN(a,b) (((a)<(b))?(a):(b))`
Variables	lowerCase	`int screenWidth = 0;, float targetFrameTime = 0.016f;`
Local variables	lowerCase	`Vector2 playerPosition = { 0 };`
Global variables	lowerCase	`bool windowReady = false;`
Constants	lowerCase	`const int maxValue = 8;`
Pointers	MyType *pointer	`Texture2D *array = NULL;`
float values	always x.xf	`float gravity = 10.0f`
Operators	value1*value2	`int product = value*6;`
Operators	value1/value2	`int division = value/4;`
Operators	value1 + value2	`int sum = value + 10;`
Operators	value1 - value2	`int res = value - 5;`
Enum	TitleCase	`enum TextureFormat`
Enum members	ALL_CAPS	`PIXELFORMAT_UNCOMPRESSED_R8G8B8`
Struct	TitleCase	`struct Texture2D`, `struct Material`
Struct members	lowerCase	`texture.width`, `color.r`
Functions	TitleCase	`InitWindow()`, `LoadImageFromMemory()`
Functions params	lowerCase	`width`, `height`
Ternary Operator	(condition)? result1 : result2	`printf("Value is 0: %s", (value == 0)? "yes" : "no");`

Some other conventions to follow:

- **ALWAYS** initialize all defined variables.
- **Do not use TABS**, use 4 spaces instead.
- Avoid trailing spaces, please, avoid them
- Control flow statements always are followed **by a space**:

```
if (condition) value = 0;
while (!WindowShouldClose())
{
}
```

```
for (int i = 0; i < NUM_VALUES; i++) printf("%i", i);
// Be careful with the switch formatting!
switch (value)
{
    case 0:
    {
    } break;
    case 2: break;
    default: break;
}
```

- All conditions checks are **always between parenthesis** but not boolean values:
```
if ((value > 1) && (value < 50) && valueActive)
{
}
```

- When dealing with braces or curly brackets, open-close them in aligned mode:
```
void SomeFunction()
{
   // TODO: Do something here!
}
```

If proposing new functions, please try to use a clear naming for function-name and functions-parameters, in case of doubt, open an issue for discussion.

3.1 Files and Directories Naming Conventions

- Directories will be named using `snake_case`: `resources/models`, `resources/fonts`
- Files will be named using `snake_case`: `main_title.png`, `cubicmap.png`, `sound.wav`

NOTE: Avoid any space or special character in the files/dir naming!

3.2 Games/Examples Directories Organization Conventions

- Data files should be organized by context and usage in the game, think about the loading requirements for data and put all the resources that need to be loaded at the same time together.
- Use descriptive names for the files, it would be perfect if just reading the name of the file, it was possible to know what is that file and where fits in the game.
- Here it is an example, note that some resources require to be loaded all at once while other require to be loaded only at initialization (gui, font).

```
resources/audio/fx/long_jump.wav
resources/audio/music/main_theme.ogg
resources/screens/logo/logo.png
resources/screens/title/title.png
resources/screens/gameplay/background.png
resources/characters/player.png
resources/characters/enemy_slime.png
resources/common/font_arial.ttf
resources/common/gui.png
```