Retro Graphics Toolkit user's manual

Sega16

Contents

List of Figures

Introduction

Welcome to the Retro Graphics Toolkit offline manual. This manual contain information on using Retro Graphics Toolkit. It is recommended that you read these pages to get a better grasp of Retro Graphics Toolkit. Seeing the page count may cause a shock however easy page is short and easy to read. If anything needs clarified you can always contact me by opening up an issue, fixing the wiki yourself (which will in turn make it in this offline version) or reply to the Retro Graphics Toolkit forum topic.

There are a few programs that will import art for various Retro consoles but when it comes to converting art to currently supported systems no program does it better than Retro Graphics Toolkit.

Retro Graphics Toolkit was designed from the ground up to make managing and importing art to various gaming consoles and embedded devices a breeze. With its easy to use GUI you can jump right in and if you need help I can answer questions or you can read the tutorials that are here on the wiki

Features

Retro graphics toolkit has a rich feature set some of which are listed here and includes

- Importing common file formats such as png,jpg,bmp,tiff and more
- Several dithering choices and nearest color
- Multi-platform both in the sense that this runs on multiple operating systems and in the sense that you can create art for multiple systems.
- True color workflow allowing for easy changes in palette (to find out more read the article)
- Open source (GPLv3 licensed)
- Can easily make changes to tiles just by selecting a color and clicking on it.
- Two easy ways to place and modify tile placement. Either left click and the tile will placed on the current location using attributes that you selected or right click on the tile and then when you use the buttons it will affect tile with blue rectangle around it.
- Supports common compression featured in many sega genesis games.
- Good handling of alpha transparency. When you import an image with alpha transparency this is preserved and dithered.
- Tilemap blocks,chunks and advanced sprite editor

If that sounds like something you are interested in either compile the source code or if you are a windows user I supply up to date windows binaries here: https://github.com/ComputerNerd/Retro-Graphics-Toolkit/blob/master/RetroGraphicsToolkit.exe.7z

Chapter 1

Chunk editor

Chunks are what makes up levels. Chunks can be constructed with either blocks or tiles. The difference between blocks and chunks is that chunks contain level data and blocks are simply a plane map in the systems native format.

Like the plane mapping editor to place/edit tiles/blocks please read https://github.com/ComputerNerd/Retro-Graphics-Toolkit/wiki/Placing-tiles-and-blocks

Chapter 2

Development roadmap

2.1 Types of goals

Goals for Retro Graphics Toolkit are separated by short term goals and long term goals.

Short term goals relate to refining what already exist in Retro Graphics Toolkit.

Long term goals relate to new features.

2.2 Short term goals

- Finish TMS9918 support.
- Add more functions to the Lua bindings.
- Make sure everything can be undone.

2.3 Long term goals

- Animations.
- More dithering algorithms
- More color quantization algorithms.
- More chunk formats.
- More sprite formats.
- More game consoles supported.
- Adding a new game console via Lua scripting.
- More example code for all game consoles.

Chapter 3

Dithering algorithms

Retro Graphics Toolkit strives to offer flexibility and choices. Dithering is no exception.

You have some choices for dithering and I plan to add even more at some point.

Floyd Steinberg Very common error diffusion algorithm. This is the default dithering algorithm.

Riemersma Claims to combine the advantages of ordered dithering with the advantages of error diffusion.

Nearest color Does not do dithering of any sort, it just picks the nearest color.

Vertical dithering Kind of like ordered dithering but only with vertical lines.

Yliluoma 1 Part of the Yliluoma series of algorithms which are described here http://bisqwit.iki.fi/story/howto/dither/jy/. Old versions of Retro Graphics Toolkit had in implementation bug that had nothing to do with Yliluoma's code. If you are experiencing terrible quality upgrade to the latest version of Retro Graphics Toolkit.

Yliluoma 2 Another ordered dither algorithm. Note that which ordered dithering algorithm looks best will depend on the image and palette. Try them all and see what looks best, for best results.

Yliluoma 3 Good ordered dithering algorithm.

Thomas Knoll Also a work of Yliluoma. Another ordered dithering algorithm.

Chapter 4

Download Retro Graphics Toolkit

For non-windows users you should compile Retro Graphics Toolkit yourself. It is very easy to do such just read the directions found in INSTALL. You don't even need to run as root and in fact there is only compiling to be done, Retro Graphics Toolkit does not depend on any other files besides the executable and it can be ran anywhere on your system.

For windows users due to the fact that many windows users are not prepared to compile c++ projects you can download a pre-compiled binary here https://github.com/ComputerNerd/Retro-Graphics-Toolkit/blob/master/RetroGraphicsToolkit.exe.7z just click "View Raw" to download.

Chapter 5

Example code

I now provide some example code showing how to display an image on supported platforms. You can get it here: https://github.com/ComputerNerd/Retro-Graphics-Toolkit/tree/master/examples

Chapter 6

File menu

This menu is dedicated to tasks that involve reading and/or writing to a file.

The menu items will use the following naming convention

Import Refers to an action in which a file loaded and the data is transformed in a manner of some sort
Export Refers to an action in which the data stored in ram is converted to a desired format.
Open Directly loading data and using it
Save Saving data in ram directly to a file

Due to the fact that there are many file options this menu is divided up into sub menus. Each submenu is given its own section

6.1 Tiles

Save tiles Loads tiles in the format of the selected system
Open truecolor tiles Loads tiles that contain rgba data.
Save tiles (append) Same as open tiles but instead of overwriting will append tiles.
Save tiles Like open tiles but saving instead of loading.
Save truecolor tiles Saves tiles are RGBA888 without any header
Import image to tiles. Utilizing an image file you can easily load tiles. This function is very similar to import image to tilemap except the tilemap is not affected.

6.2 Palettes

Open palette Opens palette saving in currently selected game system format.
Save palette Reverse of above.

6.3 Tilemaps

Import tile map or blocks and if NES attributes Loads tilemap or blocks and if system is set to Nintendo Entertainment System also load attributes.
Import image to tilemap Imports an image to tiles and changes tilemap so that the tiles are in the proper order on the tilemap.
Import image over current tilemap Replaces tiles on the tilemap with image data.
Export tilemap as image Saves the tilemap as an image. Good for ripping art or whatever you feel the need for this feature.
Export tilemap as with system color space More for testing purposes. Maybe someone will use this? What the function does is dithers the tilemap using all colors that the current system can display.

Export tile map and if NES attributes Reverse of opening tile map and if NES attributes

6.4 Projects

There is a dedicated article to this topic please read https://github.com/ComputerNerd/Retro-Graphics-Toolkit/wiki/Project-files-and-project-groups

6.5 Chunks

Import sonic 1 chunks Does what it says. If you have chunks that use the format of sonic 1 use the option.
Import sonic 1 chunks (append) Same as above but appends instead of overwriting.
Export chunks as sonic 1 format What comes up must come down. (Loading and saving).

6.6 Sprites

Import sprite from image Imports an image to a sprite group using one or multiple sprites depending on selected system and image size. Retro Graphics Toolkit will use as few sprites as possible.
Import sprite from image (append) Same as above but does not overwrite a sprite instead one is appended.
Import sprite sheet A sprite sheet is a large image that contains many sprites on this large image. Retro Graphics Toolkit can separate the image by each sprite.

Import mapping

Sonic 1 Imports sprite mapping data that conforms to the format used in Sonic 1 (16-bit)
Sonic 2 Imports sprite mapping data that conforms to the format used in Sonic 2 (16-bit)
Sonic 3 Imports sprite mapping data that conforms to the format used in Sonic 3 (16-bit) and also Sonic & Knuckles collection (16-bit)

Import DPLC

Sonic 1 Imports DPLC data that conforms to the format used in Sonic 1 (16-bit)
Sonic 2 Imports DPLC data that conforms to the format used in Sonic 2 (16-bit)
Sonic 3 Imports DPLC data that conforms to the format used in Sonic 3 (16-bit) and also Sonic & Knuckles collection (16-bit)

Export mapping

Sonic 1 Reverse of import mapping for Sonic 1
Sonic 2 Reverse of import mapping for Sonic 2
Sonic 3 Reverse of import mapping for Sonic 3

Export DPLC

Sonic 1 Reverse of import DPLC for Sonic 1
Sonic 2 Reverse of import DPLC for Sonic 2
Sonic 3 Reverse of import DPLC for Sonic 3

6.7 Scripts

Run Lua script Runs a Lua script from a file

Chapter 7

Importing an image

One of my motivations for creating Retro Graphics toolkit is to make importing images a simple process but at the same time yielding high quality results.

The first step of importing an image is getting the truecolor data into Retro Graphics Toolkit.

To do this you must first click on File->Tilemaps->Import image to tilemap.

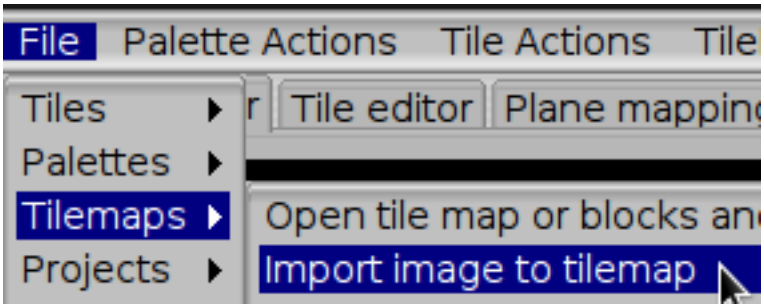

Figure 7.1: Image showing the menu item that you should click on

Just note that if you image is not a multiple of a tile (8x8 pixels) the image will be centered to make it aligned. The border will be assigned color 0 including alpha.

So you go to the plane mapping editor and are surprised to see a blank space where your imported image should go. Wondering why there is no image is a common mistake made by people new to Retro Graphics Toolkit. It is very likely that you did not make a mistake but instead you simply have not generated a palette.

To generate a palette you must first click on Palette Actions->Generate optimal palette with x amount of colors.

From there you will be prompted to pick from a choice of algorithms. However note the first menu. The default option "Don't change anything" is probably not what you want for a first import. Instead the Hue option is usually best. Which one works best depends on the image.

After that you are not done as you can till by the lack of an image. The image still needs to be dithered to fit the palette. You can do this by going to TileMap actions->Dither tilemap as image. Again pick the settings that result in the best looking image. After that you are done.

Here is an example image that I used it is based on https://raw.githubusercontent.com/ComputerNerd/Retro-Graphics-Toolkit/master/Docs/src/sega.png which is downscaled from https://raw.githubusercontent.com/ComputerNerd/Retro-Graphics-Toolkit/master/Docs/src/SEGA_logo.png I made the image except the sega text. Feel free to reuse the Sega logo.

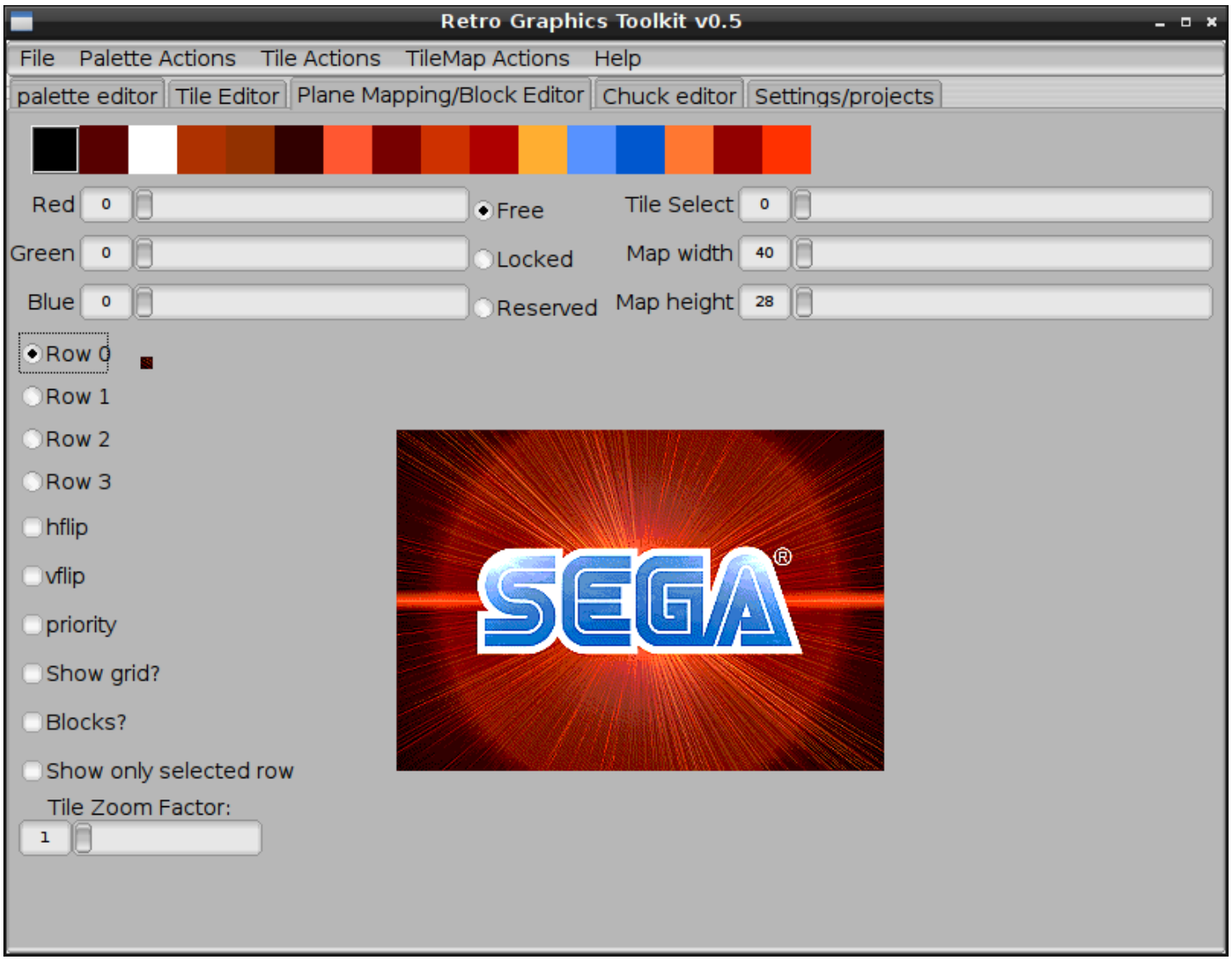

Figure 7.2: An example showing off the high quality output.

Chapter 8

Internal representation

Note this article does **not** apply to palettes and regular tiles.

One important concept to grasp is the fact that in Retro Graphics Toolkit when you load file it is actually converted to it's own internal representation which may support more information than the target console. For example you can load far more tiles than the target system can store in ram and then assign each tile on the map using a tile that exceeds the maximum index for a tile [1].

So why does Retro Graphics Toolkit let you do this? The reason is so that you have the flexibility to get the art the way you want it to be and then let Retro Graphics Toolkit automatically optimize you are for the system by removing duplicate tiles and maybe in future versions very similar tiles if removing duplicates is not enough. By letting Retro Graphics Toolkit automatically deal with system limitations you preserve the intent of the artists to the fullest extent possible. For example you create a tilemap that uses many tiles then you save your project then use the remove duplicate tiles function and now your art will fit in VRAM and you just saved yourself lots of time and got the same end result and from there all you need to do is export your data and you are done.

As an extreme example I imported a tileset ripped from a game which was stored as a PNG file. Upon importing the tileset it took 57600 tiles but after generating a palette for the image and dithering it and removing duplicate tiles it now takes only 657 tiles. You would have to be very crazy to want to do that by hand. Don't waste hours of your time doing something by hand that Retro Graphics Toolkit can do very quickly.

[1] On the Sega genesis the tile's value for each tilemap entry shall not exceed 2047 and on the NES the limit is 255.

Chapter 9

Lua API

This page is about how to use Lua API functions if you want general information on Lua such as how to run one and use them please read [[Lua scripts|Lua-scripts]]

The goal of the Lua API is to provide extensive support and to provide both low level and high level access to all internal data stored in Retro Graphics Toolkit, and to allow for both platform (the game console you have currently selected) depended and independent functions.

Another note about the API these functions are designed to not cause a crash even when called in an invalid/incorrect parameters, should this happen it is a bug and should be reported.

Currently the best documentation on these functions is the source code runlua.cpp. I have also wrote many examples of calling these API functions here luaExamples however these do not necessarily cover all functions.

Chapter 10

Lua scripts

Previously Retro Graphics Toolkit has been static in nature this meaning that you were limited to what I have programmed in Retro Graphics Toolkit in regards to automated functions. For example I added an option to sort each palette row by either hue, lightness or saturation. Before I added this trivial feature despite it's simplicity you could not that in an automated way in Retro Graphics Toolkit you would have to do it manually.

As of now Retro Graphics Toolkit uses Lua 5.2.3 (the latest stable version) so keep that in mind while writing scripts.

Also an API of functions is provided and will be defined soon.

Chapter 11

Metrics based sprite optimizations

This feature is not yet implemented.

There are metrics that can be used to optimize sprites, this determines how the sprites will be optimized. The options are as follows and can be combined

Minimum tiles

When specified alone does whatever it takes to reduce tiles. This includes using more sprites.

Minimum sprites

Uses less sprites even if it means more tiles. When both minimum tiles and minimum sprites are combined a balance will be found

Minimum DPLC entries

Adds some optimizations in hopes to minimize DPLC entries even if it means modifying mapping that was a duplicate but is not anymore by using this option. You will not notice all effects until you export the mapping as a flag is also set with the option. However this option can affect tile arrangement and possibly other stuff.

Maximum tiles (global)

Sets the maximum amount of tiles that can be used for all sprites, has highest priority.

Maximum tiles (local)

Sets the maximum amount of tiles that can be used for a sprite group, has the second highest priority.

Chapter 12

Palette actions menu

The palette actions menu contains stuff that pertain to palette operations.

Generate optimal palette with x amount of colors Allows you to generate a palette using what is on the tilemap (works with blocks just as well).

Clear entire Palette Will set all palette entries to black. If you change your mind depending on when you change your mind you can either press No or press CTRL+Z

Pick nearest color algorithm Retro Graphics Toolkit supports a verity different nearest color algorithms. Try them all and see which looks best.

RGB color to entry When in the tile editor you can use the RGB sliders to select a color and using this option will convert that color to the closest color that can be displayed on the current system and put that in the currently selected palette entry.

Entry to RGB color Opposite of above. Will get the RGB values for the current palette entry and set that value to the RGB sliders.

Chapter 13

Palette editor

13.1 Introduction to the palette tab

This tab is what you will see at first when launching Retro Graphics Toolkit

The main purpose of this tab is to edit the currently loaded palette. Also in this tab you can change some settings.

Although you can edit the loaded palette in a similar manner in other tabs, this tab allows you to see the entire palette.

13.2 How to edit a palette.

First of all click on the color you want to change. Now drag the sliders to change said value. This will update automatically

As with all sliders in Retro Graphics Toolkit you can also use the keyboard to change the values, just make sure the sliders are in focus and press the left and right arrow keys.

You can undo and redo palette changes just note that for example lets say you draw red from 0 to 7 although as you dragged your mouse you saw the color change while dragging only the old value of zero will be stored in the undo buffer, the immediate values are assumed to not be intended by the user.

Chapter 14

Placing tiles and blocks

Regardless of weather you want to edit the tilemap/blocks or chunks there are two ways to do such

The first way is to use radio boxes and tile/block select slider. Once you have what you need left click on where you want your specified tile to go

The second way is to right click on a tile this will draw a blue box around the tile now when you change the radio boxes or tile select slider it will directly affect that tile. To exit this mode left/right click on the same tile.

Chapter 15

Plane mapping block editor

This editor serves two purposes

- Editing a plane data
- Editing block data

To place place/modify tiles it is best to read this https://github.com/ComputerNerd/Retro-Graphics-Toolkit/wiki/Placing-tiles-and-blocks

Chapter 16

Project files and project groups

Project files

Retro graphics toolkit operates in projects and project groups. Project files are very simple conceptually. They store all data including tiles (both converted and truecolor) palettes plane mapping and so on. I will use the file extension .rgp for project files but you can use whatever you want. Retro Graphics toolkit does not check file extension for project files. Project files do not support sharing. The data that was not shared will be saved in the file and when loaded appear as one project. If you have multiple projects that share data it would be better to use a project group.

Project groups

Another cool feature is project groups. A project group is multiple projects in one file. When saving project groups I will use the file extension .rgg for project groups but you can use whatever you want. Retro Graphics toolkit does not check file extension for project groups. Project groups do support sharing data.

Settings/project

The settings/project tab contains all you need to control you project and to switch what project you are on. To add more projects you simply press Append blank project and it will do just as it said.

You can move the "Current project" slider and press "Delete selected project" to delete the selected project.

Also per project you can add text to the project describing it.

The Have (item) check-boxes do exactly what they say. If you check them that means data for that is stored. If unchecked it is ignored.

The share with (item) check-boxes allow part of one project's data to be in another project. When editing the shared data it affects all projects that share it as the data is stored in ram only once. A copy is not made. However if you unchecked share with and it is previously checked the (formally) shared data will be copied into the currently selected project.

Applications

You are designing a game. The first project (project 0) contains the title screen the next project in the group contains the first level and so on. The second level takes place at night it uses the same art as the first level but a different palette so you press Share tiles and select project 1.

Backwards compatibility

Retro Graphics Toolkit is designed to accept older versions of projects. If I change the format that will be okay as the projects store which version was used to save it. This means that if I add a new feature that adds more data that could be stored when you load the project the version will be read and attempting to read the new data will

be skipped. If this involves a have option this will be deselected. For example older versions of Retro Graphics Toolkit did not support chunks so when I saved a project before adding support and I want to add chunks to an old project I will need to go to the Settings/projects tab and click on Have chunks. When you save a project the latest version will always be used. There is no point in supporting saving old project files. If there was a change I made that made Retro Graphics Toolkit worse (which I doubt will happen) instead of just using an older inferior version open an issue on github or post a message in a Retro Graphics Toolkit thread. Even better submit a pull request. Remember I am always open to pull requests.

Chapter 17

Redoing and undoing

To undo an action press and hold the control key (CTRL) and press Z from now on this will be represented as CTRL+Z

After undoing an action if you change your mind you can press CTRL+Y

Just a word of caution although most actions can be undone I may have missed some stuff. It is recommended to save a project file regularly to avoid any accidents. Also Retro Graphics Toolkit supports “unlimited” history if this uses too much ram you can clear it by using the undo/redo menu clicking on “Clear undo buffer” without the quotes.

Chapter 18

Settings projects tab

This section covers the tab called “Settings/projects” if you want information on projects in general please read

https://github.com/ComputerNerd/Retro-Graphics-Toolkit/wiki/Project-files-and-project-groups

for a better understanding of projects and project groups.

This tab contains stuff for projects letting you share or disable having data also you can add a description to the project and also since I ran out of room for the sprite editor you can also set the global sprite name for the project here.

Chapter 19

Sprite actions menu

Generate optimal palette for selected sprite Creates a palette for the selected sprite.
Dither sprite as image Dithers the sprite to the current palette using the dither algorithm that you selected.

Chapter 20

Sprite editor

Starting with v0.7 of Retro Graphics Toolkit the sprite editor has been improved to make managing large objects that are made of smaller sprites much easier. To do this Retro Graphics Toolkit divides sprites up into two categories:

- Sprites
- Sprite groups

A sprite is simply what the hardware in question can display. This is the smallest unit if you will.

A sprite group contain can contain one sprite or many sprites. How many sprites are in the group is a choice made by the user as there is no set limit. Generally it is best to have each frame in a different group. To get a good idea of what a group entails lets consider a 2D Sonic game and more specifiably lets consider Sonic's sprite. Sonic is not just one hardware sprite. Instead multiple sprites comprise a group. In the case of Sonic the reason for him being divided up into multiple sprites was to reuse more tiles. Yes that is one reason why you may have multiple sprites in a group however lets consider a fighting game. Generally speaking fighting games have large sprites. To make managing this easier you can have the entire character in one group.

Chapter 21

Text boxes for resizing

Starting in version 0.7 Retro Graphics Toolkit now uses text boxes to resize the tilemap and chunks. If you are here right now it may be because you are surprised that you typed in a number and nothing happened that is because you did not press the enter key after you typed in the number that you want. If you are wondering why I changed this it is because I found it was a faster way to get precise values. With sliders you would have to use the mouse at a rough position and then maybe adjust with arrow keys or move the mouse carefully. In all other cases I have found sliders to be more efficient.

Chapter 22

Tile actions menu

This menu contains actions that pertain to tiles.

Append blank tile to end of buffer This will create a new blank tile placing it after all tiles that exist in ram.

Fill tile with selected color Will make the tile have the selected color

Fill tile with color 0 Similar to above but makes the tile black

Remove duplicate truecolor tiles Removes truecolor tiles that are duplicate also checks for flipped tiles. For some reason I forgot to add a progress bar will add that next release. I don't want to do a release with such a minor change.

Remove duplicate tiles Like above but has a progress bar and checks regular tiles instead of truecolor tiles.

Update dither all tiles This option is largely obsolete but what it does is for each tile it dithers it using the first palette row. Use the dither tilemap/sprite as image for better quality

Delete currently selected tile Should be self explanatory. If you did not mean to press this or you changed your mind undo the action by pressing holding control and pressing the letter 'Z' without the quotes.

Create new tiles for flipped tiles This is for NES users. For systems that support tile flipping this action will still run but it is useless doing so. The NES appears to not support flipped tiles so this will create new variants of existing tiles that are flipped by software instead of would be hardware.

Chapter 23

Tile editor

The first thing that you will notice about the tile editor is the two tiles that are next to each other. The tile on the left is the truecolor tile and the tile on the right is the tile that uses the current palette. When you click on the tile on the left the pixel you clicked on will be set using the RGB sliders and the tile will be dithered using the currently selected dithering algorithm. When you click on the tile on the right both tiles will be set using the currently selected palette entry. The truecolor tile's alpha value for that pixel will be set to 255.

Chapter 24

Tilemap actions menu

This menu contains actions that pertain to the plane map/blocks.

Remove tile from tilemap This is usually paired with a tile delete. What this does if for example you enter 1 all tiles greater than will be subtracted by one.

Toggle TrueColor Viewing (defaults to off) Shows truecolor tiles instead of regular tiles

Pick Tile row based on delta Tries to pick the best row for all tile on the tilemap by trying to dither to all rows and seeing which one results in less error.

Auto determine if use shadow highlight Specific to the sega genesis allows Retro Graphics Toolkit to decided if the tile should be normal or shadowed useful for the sega genesis's shadow highlight mode.

Dither tilemap as image Converts the tilemap to an image, dithers it and converts it back to tiles. Much better quality than just doing each tile one by one.

File tile map with selection including attributes Will erase everything on the tilemap and replace it with what you have selected as in radio boxes and check boxes in the plane editor.

Fix out of range tiles (replace with current attributes in plane editor) If something on the tilemap refers to a non existent tile you can use this to replace it.

Chapter 25

Transparency notes

On the Sega Genesis and NES the first color for each row is transparent.

When you import a png file it may contain transparent data. It is better to use a png file with transparency than a mask color however a mask color will also work if that color is the first color for affected palette rows.

When the same color occurs multiple times in the same palette row the last color will be used. You can take advantage of this if you want the transparent color to be the same as a solid color. In sonic one the sprite of sonic uses black for transparency and solids. This also applies to all sprites that share the same palette with sonic. The reason this works is due to the fact that transparent pixels are *always* assigned to color 0. Therefore any non transparent pixels will not use said transparent color.

Chapter 26

Understanding true color data

Retro Graphics toolkit stores true color data in addition to the converted tiles. This makes it easier to change the palette. Simply re-dither. Also note that alpha data is also stored. Transparent data will be dithered and assigned color 0.

The advantage of preserving this data is traditionally when you wanted to make a change to the palette you would have to redraw all graphics or go through a (less but still very laborious) process of manually loading the image in an image editor and applying the new palette for all images. Retro Graphics Toolkit automates this process.