# Shield Cat Instruction Manual

**Or “How to make the otter spin”**

**Created by CyanSorcery**

# Table of Contents



# Special Note

As Shield Cat is currently in development, some parts of this manual may be out of date, or refer to features which no longer exist. Similarly, this manual may also refer to upcoming features which are not actually implemented in the game yet.

# Contact

If you wish to contact the developer about this game, or wish to connect with other people who like the game, the easiest and fastest way is through the official Discord chatroom:

https://discord.gg/shieldcat

You may also use the following to get more information or support with the game:

**Website**
https://shieldcat.com

**Twitter**
https://twitter.com/ShieldCatGame

**Support Email**
support@shieldcat.com

**Bug Report Email**
bugreport@shieldcat.com

**Give Feedback**
feedback@shieldcat.com

**Developer Contact**
roxy@shieldcat.com

# About

Shield Cat is a game starring Lance the Otter. It takes place in the Cat Kingdom, a magical place in the far off World of the Metamagicia. The Cat Kingdom is protected by the Knights of the Shield Cats, who in addition to protecting the kingdom, are also responsible for railroad, mail, and air services. There are many members of the Shield Cats, from knights to scientists.

However, lately strange things are happening. Creatures made of dark magic are appearing everywhere. Parker, who is in charge of the railroad, has disappeared. Nobody can get into the Cat Castle, and the ruler, King King Cat the Cat, has gone missing.

One day, Lance has a mysterious dream where Parker calls out to him, warning that something terrible was happening, and asking for Lance's help. After waking up with a start, Lance gets out of bed and heads into the village where Parker lives.

This is where our story begins...

# Installation

Thank you for downloading and installing Shield Cat! Here are some things to know:

- You will find the main configuration file in the save data directory. It will be found in `/config/main.ini`. See Editing the Configuration for more details. Most of these options can be configured from within the game, either through the pause menu or using the Debug Console.
- This game is **NOT** freeware! If you downloaded this game from a website besides itch or Steam, **you may have received a pirated copy!** The developer is **NOT** responsible if a pirated version of the game hacks your computer.

# Platform Specific Notes

Below are notes specific to each platform that Shield Cat is available on.

## Windows

- Shield Cat is supported on Windows 10. It should run on older or newer versions of Windows, but this isn't guaranteed. Your version of Windows MUST be 64bit. Shield Cat will **not** run on 32bit Windows.
- All versions of Shield Cat use the same save data folder, regardless of if it is a demo version or the full version. **The save data folder is located in** `%localappdata%\Shield_Cat`. Configuration files are also stored there.
- When running the game, you may get a notice about running an app from an unknown publisher. Click "Run Anyways" or "Learn More" to reveal the option to run the application. **This is not an issue when running through Steam.**
- Your version of Windows must support DirectX 11.

# Controls

Below you will find the default controls for the game. You are able to customize these controls from within the game as well. It should be noted that additional control options (such as double-tapping to sprint) are available in the Options menu of the Pause menu.

# Computers

These are the controls for the game when played on the PC.

## XInput

XInput refers to controllers such as XBOX 360, One, and Series gamepads. They can also be compatible 3rd party gamepads, such as 8bitdo. Some gamepads support XInput and DirectInput modes. Please refer to the instructions for your gamepad for configuration.

| Button | Button Usage |
|---|---|
| **Directional Pad or Analog Sticks** | **(During gameplay)** Move your character around.<br>**(in menus or questions)** Change your current selection. |
| **A (Green) or Left Trigger** | **(Pressed)** Use MoveTech abilities, talk to people, investigate objects, confirm menu selections.<br><br>**(Hold, in dialogue)** Cause Dialogue to speed up. |
| **B (Red) or Right Trigger** | Use Thrown Abilities, go back in menus, cancel questions, skip through dialogue boxes. |
| **X (Blue)** | **(Held)** Sprint, make dialogue and menus go faster. |
| **Left Bumper or Right Bumper** | Swap between different Thrown Abilities. |
| **Y (Yellow) or Options** | Open the pause menu. |
| **Back** | Open the full map screen. |
| **Analog Stick Press** | Preview the collectibles you have in the current stage. |

*XBOX, XBOX 360, XBOX One, and XBOX Series and related trademarks are copyright Microsoft.

## DualShock 4

| Button | Button Usage |
|---|---|
| **Directional Pad or Analog Sticks** | **(During gameplay)** Move your character around. **(in menus or questions)** Change your current selection. |
| **Cross or L2** | **(Pressed)** Use MoveTech abilities, talk to people, investigate objects, confirm menu selections.<br><br>**(Hold, in dialogue)** Cause Dialogue to speed up. |
| **Circle or R2** | Use Thrown Abilities, go back in menus, cancel questions, skip through dialogue boxes. |
| **Square** | **(Held)** Sprint, make dialogue and menus go faster. |
| **L1 or R1** | Swap between different Thrown Abilities. |
| **Triangle or Options** | Open the pause menu. |
| **Touchpad** | Open the full map screen. |
| **Analog Stick Press** | Preview the collectibles you have in the current stage. |

## Switch Pro, JoyCon L/R

| Button | Button Usage |
|---|---|
| **Directional Pad or Analog Sticks** | **(During gameplay)** Move your character around. **(in menus or questions)** Change your current selection. |
| **A, ZL** | **(Pressed)** Use MoveTech abilities, talk to people, investigate objects, confirm menu selections.<br><br>**(Hold, in dialogue)** Cause Dialogue to speed up. |
| **B, ZR** | Use Thrown Abilities, go back in menus, cancel questions, skip through dialogue boxes. |
| **Y** | **(Held)** Sprint, make dialogue and menus go faster. |
| **L, R** | Swap between different Thrown Abilities. |
| **X, Plus** | Open the pause menu. |
| **Minus** | Open the full map screen. |
| **Analog Stick Press** | Preview the collectibles you have in the current stage. |

**NOTE:** Shield Cat is able to recognize the Joycon L and R as a single gamepad without the need for additional software. If you wish to use a single Joycon half, you will have to configure it yourself.

# Unrecognized Controllers

The game should be able to automatically recognize most controllers and assign a default configuration to them. However, sometimes the controller may simply not be recognized. If this happens, the game will tell you so when you plug in the gamepad, or at the start of the game if it was plugged in before you started playing. For these gamepads, you'll have to configure them manually.

**Please note,** the game will **not** work with Playstation 3 gamepads. **Do not** attempt to use Playstation 3 gamepads with the game.

If the game does not have a preset configuration for your gamepad, and you wish to contribute a configuration to the game, please do so! You can export the configuration ini from the Input Remapping menu from within the game, or additionally you may simply copy it from the game's save folder. You may send configuration files to `support@shieldcat.com`.

## DirectInput

When you plug in an unrecognized DirectInput gamepad, the following controls will be assigned to it.

| Button | Button Usage |
|---|---|
| **POV Hat 1 or Analog Stick 1** | **(During gameplay)** Move your character around.<br>**(in menus or questions)** Change your current selection. |
| **Button 1** | **(Pressed)** Use MoveTech abilities, talk to people, investigate objects, confirm menu selections.<br><br>**(Hold, in dialogue)** Cause Dialogue to speed up. |
| **Button 2** | Use Thrown Abilities, go back in menus, cancel questions, skip through dialogue boxes. |
| **Button 3** | **(Held)** Sprint, make dialogue and menus go faster. |
| **Button 4** | Open the pause menu. |
| **Button 5** | Open the full map screen. |
| **Button 6 and Button 7** | Swap between different Thrown Abilities. |

# Mouse

There are two different mouse modes that you can use which define how the mouse will interact with the game. They are as follows:

**Action Mode**

Left click will use MoveTech abilities, talk to NPCs and investigate objects. Right click uses Throwables. Middle click opens the menu. Scroll wheel swaps between thrown abilities.

**Click'n'Drag**

Left click and dragging around will move the player character. Middle click opens the menu. Scroll wheel swaps thrown abilities.

In either mode, you can navigate the menus or dialogue boxes with the mouse. The setting you choose does not affect your interaction with menus or dialogue sequences.

# Characters

There are many different people you'll meet throughout your journey! Here are some of them, but there are many more people for you to get to know!

## Lance Otterton

The protagonist of this adventure, Lance, is a mail carrier who lives in the Lignum Village, located in the Deep Woods. One day, he has a mysterious dream where his partner Parker tells him that something bad is happening in the Cat Kingdom. Startled awake by the dream, Lance goes to see Parker and to find out what's going on. He has no idea what events will be set in motion on that day

| Character Bio | | | | | |
|---|---|---|---|---|---|
| **Age** | 22 | **Species** | Otter | **Birthday** | November 9th |
| **Gender** | Male | **Pronouns** | He/Him | **Height** | 5'4" (152cm) |
| **Orientation** | Pan | **Job** | Mail Carrier | **Favorite Music** | Rock |
| **Friends** | Parker (Partner), Trent, Olivia | | | | |
| **Enemies** | None | | | | |
| **Bio** | Loves to travel, being in the water, and adventuring. Caused mischief in his younger days. Has grown up, but still keeps his hair cool and stylish. | | | | |

# Parker Railway

Parker works on the Roo Express, a family-owned railroad company specializing in magical travel across the Cat Kingdom. After their father got into an accident, Parker, alongside their sister Julie, has been running the railroad. Recently, however, the siblings have gone missing. Now all of the trains have stopped, and it's up to Lance to figure out what's going on.

| Character Bio | | | | | |
|---|---|---|---|---|---|
| **Age** | 21 | **Species** | Kangaroo | **Birthday** | June 7th |
| **Gender** | Nonbinary | **Pronouns** | They/Them | **Height** | 6'0" (183cm) |
| **Orientation** | Pan | **Job** | Operator of the Roo Express | **Favorite Music** | Classical |
| **Friends** | Lance (Boyfriend), Suzette | | | | |
| **Enemies** | None | | | | |
| **Bio** | Loves trains, seeing new sights, and quiet evenings. Can create magic portals that they can jump through to appear somewhere else. | | | | |

# Suzette

Suzette is a geologist for the Shield Cats. She studies rocks, stones, and the earth to find out the effects of hard Metamagic and how it differs from other types. She lives in the Underground Forest, a place surrounded by lots of ground for her to study. She has the ability to summon and throw large boulders, which her girlfriend thinks is really cool.

| Character Bio | | | | | |
|---|---|---|---|---|---|
| **Age** | 25 | **Species** | Leopard Gecko | **Birthday** | January 29th |
| **Gender** | Female | **Pronouns** | She/Her | **Height** | 5'3" (160cm) |
| **Orientation** | Pan (Female leaning) | **Job** | Keeper of the Underground Forest | **Favorite Music** | Jazz, Piano |
| **Friends** | Olivia (Girlfriend) Parker | | | | |
| **Enemies** | None | | | | |
| **Bio** | Enjoys quiet and solitude, but sometimes enjoys a good party. Can manipulate the earth, walk on walls, and swim through the ground. | | | | |

# Ruby

Ruby works for the Shield Cats as the keeper of the dam at Lake Fromonda. There is also an aquatic exhibit there where people can come see the local wildlife.

| Character Bio | | | | | |
|---|---|---|---|---|---|
| **Age** | 22 | **Species** | Beaver | **Birthday** | July 27th |
| **Gender** | Female | **Pronouns** | She/Her | **Height** | 4'6" (137cm) |
| **Orientation** | Pan/Ace/Aro | **Job** | Protector of the Lake Fromonda dam | **Favorite Music** | Zydeco |
| **Friends** | Trent, Olivia | | | | |
| **Enemies** | None | | | | |
| **Bio** | Very energetic and rowdy. Carries a lasso at all times. Has great mobility in the water. | | | | |

# Trent

Trent is a member of the Knights of the Shield Cats, who uses his command over fire and ice to study the effects of different types of Metamagic in extreme environments. He takes up his studies in Mt. Bophades, but is not very serious about it. Recently, a huge blizzard has taken over Mt. Bophades and has been making it very cold. Trent, who normally would have such situations under control, has not been seen.

| Character Bio | | | | | |
|---|---|---|---|---|---|
| **Age** | 23 | **Species** | Fox | **Birthday** | June 28th |
| **Gender** | Male | **Pronouns** | He/Him | **Height** | 5'9" (175cm) |
| **Orientation** | Pan (Male leaning) | **Job** | Scientist, Elemental Effects | **Favorite Music** | EDM, Dubstep |
| **Friends** | Olivia, Lance, Ruby | | | | |
| **Enemies** | None | | | | |
| **Bio** | Loves showing off, makes huge displays of his powers, overly confident. | | | | |

# Olivia

Olivia is the captain of the Sky Cat Airship Fleet, the air forces of the Knights of the Shield Cats. Although young, her natural leadership and skill in the air accelerated her rise to the top. Because the kingdom is at peace, the Airship Fleet is mainly responsible for providing air travel and leisure cruises through the sky. However, they are always ready to go in case they are needed.

| Character Bio | | | | | |
|---|---|---|---|---|---|
| **Age** | 24 | **Species** | Cockatiel | **Birthday** | March 24th |
| **Gender** | Female (Trans) | **Pronouns** | She/Her | **Height** | 6'0" (183cm) (excludes tail and spikes) |
| **Orientation** | Pan (Female leaning) | **Job** | Airship Captain, Knights of the Shield Cats | **Favorite Music** | Heavy Metal |
| **Friends** | Trent, Ruby, Suzette (Girlfriend) | | | | |
| **Enemies** | None | | | | |
| **Bio** | Loves to fight, acts first and asks questions later. Extremely skilled in the air. Her nickname is Ollie. | | | | |

# Other Characters

Aside from the characters mentioned above, you'll also meet many other characters in your quest.

**Willow**

Willow is a magical white fox who appears only to certain people. She has the ability to help you record your progress, and will also help you out when you're in a bind - as long as you have a cookie for repayment.

**Claire**

Claire is the leader of the Knights of the Shield Cats. As the head of the protectors of the Throne, she is responsible for all of the operations of the Shield Cats, keeping the peace in the kingdom, and making sure everything is running smoothly. Lately, however, some have reported her causing mischief. Rumors have spread that she may have staged a coup to take over the Cat Kingdom.

**King King Cat the Cat**

King Cat is the elected ruler of the Cat Kingdom. After his friends joked around in school that he should try to become king because of his name, he decided to learn about law and rulership and eventually became the king. King is generally loved around the kingdom, and despite being a bit of an airhead, is surprisingly capable of his job. Lately, however, he hasn't been seen around at all.

# Locations

There are many locations to visit in the Cat Kingdom! Here are just some of the places you can find.

**Lignum Woods**

A forest full of mystery and enchantment, Lignum Woods is home to the Lignum Village, where Lance and Parker live.

**Mt. Bophades**

A dormant volcano surrounded by snow and ice, Mt. Bophades is the home of Trent as well as where the docks for the Sky Cat Airship Fleet are.

**Lake Fromonda**

It is a beautiful lake and is where Lance was born.

**Cat Castle**

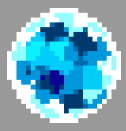
The center of the Cat Kingdom.

**Underground Forest**

A beautiful underground area full of tunnels and skylights.

# Collectibles

There are a large amount of collectibles for you to find, hidden throughout the Cat Kingdom! Here are just some of the collectibles you can find.

| | |
|---|---|
|  | **Pretty Petals**<br><br>The currency of the Cat Kingdom. Use these to buy abilities and upgrades at shops. |
|  | **Fish Scales**<br><br>Each stage has a fixed number of Fish Scales. Use these to unlock routes for the train to travel. Can you find all of them in each area? |
|  | **Big Jewels**<br><br>Beautiful large jewels are hidden throughout each area. You are awarded a Cat Coin if you can find all 5 of them in an area. |
|  | **Cat Coin**<br><br>Special currency of the Cat Kingdom. Use these to open certain doors to progress throughout the game. |
|  | **Keys**<br><br>There are many different types of keys hidden throughout the Cat Kingdom that unlock various types of keys. Remember, **keys found in one area will not work in another area!** |
|  | **Treasure Chests and Treasure Orbs**<br><br>You never know what these will contain, so keep an eye out for them! |
|  | **Fish Statue**<br><br>This large fish statue will permanently increase your health! |

# Abilities

During your quest, you will find many different abilities that you can make use of! On top of finding these abilities, you can also find upgrades to these abilities. Here are just some of the abilities you can find. Can you find them all?

| | |
|---|---|
|  | **Tubular Twister**<br>Lance's spin attack! You can mash the button to go faster and increase in speed, but you will stop whenever you come into contact with an enemy or obstacle. |
|  | **Shield**<br>Lance can throw his shield to hit enemies from far away. He can also use it to solve puzzles, and upgrade it to give it more distance and range when he throws it. |
|  | **Magic Cards**<br>Lance throws a continual stream of cards from a seemingly endless deck. He can use this ability to take down multiple enemies at once. |
|  | **Bomb**<br>Lance can use these bombs to open up the path to new areas, or to damage enemies from a distance. It's explosive fun! |
| | **Boomerang**<br>Lance throws a boomerang that surrounds him in a circle. This is good for attacking enemies around corners. |

# Passives

These are abilities that give you a passive bonus when you get them! There are several types, and some can be upgraded to become stronger.

| | |
|---|---|
| **Lantern** | See more around you in dark places. |
| **Healer** | Gradually recover health over time. |
| **Distance Traveler** | Increases your top speed when spinning. |
| **Magnet** | Draws in nearby collectibles. Equip more to enhance the effect. |
| **Lucky** | Increase the chances that you'll find Pretty Petals, Health Fish, and other nice stuff when cutting grass, defeating enemies, etc. Equip more to increase the effect. |
| **Magic Master** | Recover magic at a faster rate. |
| **Infinite Abilities** | Use any Ability as much as you want without consuming magic. |

# Technical Stuff

The following pages contain, well, really technical stuff about how the game works. Most likely you won't need to read any of this unless you're editing the game's configuration files directly. Please take care when editing these files to not make a mistake!

## Main Configuration

The configuration file can be found in the save folder at `/config/main.ini` (Please refer to the Platform Specific Notes for details.) This file is the main configuration file for the game, and contains many settings which are detailed below. You can erase a value and it will be reset next time you run the game. It is also important to note that **the game will not automatically write these values to the configuration file unless they're changed**, so please keep this in mind!

Please note, most of these values can be changed from within the game, so it's not necessary to modify the configuration file directly.

## Video

These settings control how the game is drawn to the screen.

| **Preset** | **Default: high** |
|---|---|
| Specifies the built-in preset to use for video settings. When set to `custom` the game will use the settings you specify in the configuration file. Otherwise, it will use the settings included within the game's installation directory. | |

| **RetroPreset** | **Default: off** |
|---|---|
| Specifies the built in preset for giving the game a retro look and feel. When the `Preset` option is set to `lowest` or `custom` this setting is ignored. | |

| **AllowAutoAdjust** | **Default: true** |
|---|---|
| Allow the game to automatically change the graphics preset depending on the performance. When "Preset" is set to `custom` this setting is ignored. | |

**`VSync`** **`Default: true`**

Enables VSync (Vertical Sync,) which can help prevent screen tearing between frames. Can slow down the game when enabled, though this is very unlikely. When `AllowAutoAdjust` is enabled, VSync is automatically configured as necessary.

**`TexFilter`** **`Default: true`**

Enables Bilinear Interpolation texture filtering in some parts of the game as needed. Performance impact is extremely minor.

**`UpscaleMethod`** **`Default: 2`**

Decides how the game fills the screen on monitors which are not an exact multiple of the base resolution of 360p (Otherwise stated, monitors which are not quite 720p, 1080p, 1440p, or 4k.) The values are as follows:

`0` **- Integer Upscale**
The game will always upscale to be an exact multiplier of the base resolution. Black borders will be inserted around the game view. This is the fastest rendering method.

`1` **- Fractional (Fast)**
The game will upscale to the integer multiple lower than the target resolution, and then upscale from there to fill the screen. There is a small performance impact, and the result can be slightly blurry.

`2` **- Fractional (High Quality)**
The game will upscale to the integer multiple higher than the target resolution, and then downscale from there to fill the screen. The quality is much better, but there can also be a huge performance impact on older computers or less powerful hardware.

It should be noted that, when the game is played on a screen that is an exact multiple of the base resolution (For example playing on a 1080p display,) this setting is ignored and Integer upscaling is applied.

**`InternalUpscale`** **`Default: true`**

When enabled, the game renders and then upscales as normal. If disabled, the game is only rendered at 360p and then upscaled using Integer Upscaling. It should be noted that, though this will enable the fastest drawing possible, several other settings will be ignored, and **all** Post Processing effects will be ignored.

| `HiResRender` | `Default: true` |
|---|---|
| When enabled, the game renders at the target screen resolution, increasing the quality of 3D models drawn, as well as allowing much smoother movement of sprites and the camera. In addition, high-resolution fonts are used. However, this can cause a performance dip on less powerful hardware.<br><br>When disabled, the game renders at 360p and then upscales. This allows for a more "authentic" experience where 3D objects are rendered at low res, and sprites and the camera move with pixel perfect precision. This mode is also much faster to render. However, some fonts may be rendered unreadable at such a low resolution. | |

| `Reflections` | `Default: true` |
|---|---|
| Enables reflections of objects in the water and on other reflective surfaces. Can have a noticeable performance impact when enabled. | |

# Window

`Width` `Height` and `Fullscreen` store the state of the game window on the previous run of the game. You can set them to any value, but they can't be smaller than 640x360, and can't be bigger than your monitor's resolution. Additionally, the values will always be adjusted to be an even number. Otherwise, there can be weird pixel interpolation. It is best to **not modify these values** unless you really need to.

# Audio

The following control the gain (volume level) of different types of audio. They are all values between `0` and `1`, where `1` is full volume and `0` disables playback. For example, if you wanted to set one of them to half volume, you would set it to `0.5`.

| **MSTGain** | **Default: 1** |
|---|---|
| Controls the volume for all audio within the game. All other volume levels below depend on the level of this setting. | |

| **OSTGain** | **Default: 0.8** |
|---|---|
| Controls the volume for all music played back in the game. | |

| **SFXGain** | **Default: 1** |
|---|---|
| Controls the volume for all sound effects in the game. | |

| **TXTGain** | **Default: 1** |
|---|---|
| Controls the volume for the "voice blips" played while reading messages. | |

| **ENVGain** | **Default: 1** |
|---|---|
| Controls the volume for environmental sound effects, like birds chirping in the trees, waterfalls, riverbanks etc. | |

In addition, the following commands are available.

| **SFXStereo** | **Default: true** |
|---|---|
| Whether to play back sound effects in stereo or not. Music and environmental sounds are not affected. | |

| **Echoes** | **Default: true** |
|---|---|
| Whether or not to use echoes in certain areas, such as caverns. Can sound nice, but may sound garbled on some devices. If you have trouble with audio, try disabling this first. | |

# Game

These are various options which affect general gameplay.

| `CamShakeStrength` | `Default: 1` |
|---|---|
| Allows the camera to shake as a reaction to certain events or situations. A value of `0` means that it's disabled, `0.5` is half strength, and `1` means full strength. You can also double the strength by setting it to `2`. | |

| `InputTimeout` | `Default: 18000` |
|---|---|
| When absolutely no input is being received by the game, this sets the amount of time (in frames) before the screen dims. As the game runs at 60 frames per second, the default time of `18000` is 5 minutes. You can disable this function by setting it to `0`. | |

| `Seed` | `Default: 0` |
|---|---|
| Sets the seed used for randomization throughout the game. Mostly an unnecessary setting as the game does not use procedural generation in any way. It should be noted that, unlike some engines that take a value of `0` to mean "make up a value every time," Gamemaker uses `0` as just another seed for randomization. | |

| `TextAA` | `Default: true` |
|---|---|
| Allows AntiAliasing on fonts that support it. When disabled, fonts may render poorly. There is no performance impact from turning this on or off. | |

| `EnterRoomMarker` | `Default: true` |
|---|---|
| Shows an indicator to let you know where you are on the screen when stepping into a new stage or going indoors or outdoors. | |

| `ConfigVer` | `Default: varies` |
|---|---|
| This is the config/save version of the version of Shield Cat you last played. This is set so that the game can know if changes/updates need to be made when updating the game to a newer version. **UNDER NO CIRCUMSTANCES SHOULD YOU CHANGE THIS VALUE.** | |

| `MaxSlots` | `Default: 20` |
|---|---|
| How many save slots show in the title screen menu. Save slots that are outside of this number are not deleted - they are simply ignored by the game. | |

| `SaveSlot` | `Default: varies` |
|---|---|
| This is the save slot you played on last. | |

| `SkipIntros` | `Default: false` |
|---|---|
| Makes the game skip all introduction sequences and load into a predefined location using the last save file opened. **This is mainly used for debugging, and can cause the game to become unplayable and/or loss of save data.** Enable it at your own risk! | |

| `ReduceParticles` | `Default: false` |
|---|---|
| When enabled, the game does not spawn as many particles as it normally would. This can help boost performance or make the visuals less distracting. | |

| `ShowDevMsg` | `Default: true` |
|---|---|
| Shows a message on the screen that this is a development build of Shield Cat. If you plan on streaming or sharing screenshots/video of the game, you are asked to keep this enabled. If you do disable it, you **must** make it clear that the build you are playing is a development build if you intend to stream or share screenshots or pictures of the game. | |

| `Language` | `Default: en_us` |
|---|---|
| Specify the language folder to read language data from. | |

# Accessibility

These are various accessibility options that you can enable.

| `UseAltColors` | `Default: false` |
|---|---|
| Uses alternate color schemes on various elements throughout the game to help with color blindness and to make game elements easier to see.<br>**Please Note:** This was added before there was a dedicated Accessibility section in the configuration file. Therefore, you'll find this in the `[Game]` section. | |

| `ReducedAnim` | `Default: false` |
|---|---|
| When enabled, it reduces animations used in the game to be less distracting or flashy. | |

# Input

These options define how the game handles gamepads, the keyboard, mice, and touchscreens. Please be aware that disabling input types **will** cause the game to become **unplayable** by those devices, so they should only be disabled for debugging!

| `AllowKeyboard` | `Default: true` |
|---|---|
| Allows input from the keyboard. Please note that keyboard input is disabled on non-desktop based platforms. Additionally, note that keyboard input is always allowed for debug functions, regardless of this setting. | |

| `AllowGamepad` | `Default: true` |
|---|---|
| Allows input from connected gamepads. As Shield Cat will read configured input from gamepads each frame, performance will be impacted the more gamepads you have. You can either remove gamepads from your PC you're not using, or disable this option entirely.<br><br>It should be noted that certain non-gamepad devices may be detected, but Shield Cat will ignore input from these devices. To qualify as an input device for Shield Cat, the gamepad must have at least one analog stick or POV hat, as well as 5 buttons minimum. | |

| `AllowTouch` | `Default: true` |
|---|---|
| Allows input from the mouse, or touch screen on some devices. Please note that Shield Cat can only recognize a single touch on desktop devices. | |

| `AllowVibration` | `Default: true` |
|---|---|
| Although individual gamepads can be set with their vibration strengths, this can override all gamepad settings to disable vibration on all gamepads. | |

| `AnalogDeadzone` | `Default: 0.2` |
|---|---|
| The deadzone for analog sticks on gamepads. Can be a number from `0` to `1`, where `0` allows any input to be read from the stick in any position, and `1` makes it pretty much impossible to read input from the stick. | |

**MobileMode** **Default: false**

Splits the screen into different regions which do different things when interacted with. **This option is mostly depreciated and may be removed in the future.**

**AnalogRunEnabled** **Default: false**

When enabled, the player character will run when tilting the analog stick far enough in the direction you want to go.

**AnalogRunThreshold** **Default: 0.7**

Sets how much you have to tilt the analog stick to make the player character run. Larger values mean you have to tilt the stick further.

**DoubleTapTime** **Default: off**

This sets the frame window in which you can double tap in the same direction to cause the player character to start running. Possible values are as follows:

`off, tight, normal, loose, very_loose`

**EnhancedRumble** **Default: true**

Uses subtle rumble when grabbing collectibles, walking on surfaces etc. Only works on relatively newer gamepads, so should be disabled for older ones.

**XDisplayPref** **Default: 0**

Sets the in-game button prompts used for XInput devices (only in Windows.) These are the possible values:

`0`: XBawks
`1`: Braystation
`2`: Swatch

**RunToggle** **Default: false**

When enabled, you can simply press the run button to start running, and press it again to stop running. Certain actions will cancel out this function.

| `MouseMode` | `Default: 0` |
|---|---|

There are two different mouse modes that you can use which define how the mouse will interact with the game. They are as follows:

**Mode `0` - Action Mode**
Left click will use MoveTech abilities, talk to NPCs and investigate objects. Right click uses Throwables. Middle click opens the menu. Scroll wheel swaps between thrown abilities.

**Mode `1` - Click'n'Drag**
Left click and dragging around will move the player character. Middle click opens the menu. Scroll wheel swaps thrown abilities.

In either mode, you can navigate the menus or dialogue boxes with the mouse. The setting you choose here does not affect your interaction with menus or dialogue sequences.

| `HoldMoveTech` | `Default: false` |
|---|---|

When enabled, you can simply hold the MoveTech button to continually use the ability you have equipped until you run out of magic.

# PostProc

These settings are applied after the game is drawn. They can look nice, but **can have a big performance impact** (detailed in each section.) Please note that if `InternalUpscale` is disabled, you can't use these settings. Also, these settings will only be used if `Preset` is set to `custom`.

| **TotBlurQuality** | **Default: 4** |
|---|---|

Sets the quality for the blur used on the edge of the screen in certain areas, as well as when the game is paused. The following is a list of values:

`0` - Blur Disabled
`1` - Low Quality
`2` - Medium Quality
`3` - High Quality
`4` - Higher Quality
`5` - Extreme Quality

Please note that **the higher the quality, the bigger the performance impact.** Also, if `HiResRender` and `TexFilter` are both enabled, it will further improve the quality of the blur (additional performance cost is extremely minor.)

| **EdgeBlur** | **Default: true** |
|---|---|

Enables a decorative blur around the edges of the screen in certain areas. Please note that this will not work if `TotBlurQuality` is set to `0`.

| **ScanlinesEnabled** | **Default: false** |
|---|---|

Adds scanlines to the display output. It should be noted that this function will not work if the game view is too small.

| **ScanlinesStrength** | **Default: 0.5** |
|---|---|

The strength of the scanlines being applied to the screen. Please note that this will not work if `ScanlinesEnabled` is `false`.

| **CRTBend** | **Default: false** |
|---|---|

Bends the edges of the screen to simulate an old style CRT display.

| **CRTBlur** | **Default: false** |
|---|---|
| Adds a glow to the screen to simulate the display of an old CRT. However, this setting can have a big performance impact. Also, please note that this will not work if `TotBlurQuality` is set to `0`. | |

| **BitCrush** | **Default: 256** |
|---|---|
| Reduces the amount of colors used in the game. Valid range is any number between `1` and `256`, with `256` disabling the effect and `1` making the game completely unplayable. Good values for this are `2`, `4`, and `8`. | |

| **InvertColors** | **Default: false** |
|---|---|
| Inverts all colors on the screen. | |

| **Grayscale** | **Default: false** |
|---|---|
| Converts the game image to grayscale. | |

| **Brightness** | **Default: 1** |
|---|---|
| This sets the brightness for the game. Possible values are between `0.1` and `2`. | |

# RetroPal

These options control how the Retro Palette mode operates.

| **Enable** | **Default: false** |
|---|---|
| Enables the Retro Palette filter. As a Post Processing filter, this filter can only be used if `Preset` is set to `custom` and `InternalUpscale` is enabled. | |

| **PalInd** | **Default: 1** |
|---|---|
| The palette that the game uses when this filter is enabled. By default, the game will automatically choose a palette based on predefined colors for each area. Setting a value above `1` will allow you to use one of the built in palettes, while setting it to `0` will allow you to use your own. There are over 69 different palettes to choose from, so they are not all listed in this manual. Feel free to experiment! | |

| **ColStr** | **Default: 9BBC0F7DA4003062300F380F** |
|---|---|
| When `PalInd` is set to `0`, it will use the color string here. The format is 4 sets of HTML Hex colors, ordered from lightest to darkest. You can read this value like so:<br><br>`9BBC0F 7DA400 306230 0F380F` | |

# MenuText

These options specify how text is rendered in the menu.

| `Font` | `Default: default.ini` |
|---|---|
| This is the font pack to be used in the menu. Please see the section on Font Packs for more information. | |

| `Size` | `Default: regular` |
|---|---|
| The size of the font to use. The following values are valid:<br><br>`smaller small regular large larger` | |

# Dialogue

These options specify how text is rendered during dialogue sequences.

| `TextSpeed` | `Default: 2` |
|---|---|
| This sets how slow or fast dialogue is rendered. Positive values represent how many frames to wait between writing a letter to the screen, while negative values represent how many letters to process in a single frame. Built-in values are as follows (Please note you can **only** use the number, **not** text:)<br>- `14`: Slowest<br>- `7`: Slower<br>- `4`: Slow<br>- `2`: Regular<br>- `-3`: Fast<br>- `-9`: Faster<br>- `-200`: Instant | |

| `Font` | `Default: default.ini` |
|---|---|
| This is the font pack to be used in the menu. Please see the section on Font Packs for more information. | |

| `Size` | `Default: regular` |
|---|---|
| The size of the font to use. The following values are valid:<br><br>`smaller small regular large larger` | |

# Debug

The following are a list of debug options that are available. These options shouldn't be enabled in the game unless you know what you're doing, as **they can make the game unplayable or corrupt your save file!**

| `EnableDebug` | `Default: false` |
|---|---|

When enabled, the following options are available when you press these keys:

**F9 - Debug Console**

Opens the game's Debug Console, a text-based prompt with many commands available.. Type `help` for a list of commands available, and `help command` to find out about the specific command. Due to the fact that the options available in this debug console change frequently, documentation for all commands available are not included in this guide.

**F10 - Debug Overlay**

Shows detailed information about the game. Due to the changing nature of the debug overlay, you'll need to refer to the debug overlay itself to see what it displays and what features are available.

Please note, the Debug Console is **only available in the full version of the game!** It will not work in the Demo or Playtest version of the game. Also, using many of the commands available will mark your save file as using cheats, so **don't use these functions on the save file you intend to play the game with, as leaderboards and achievements will become unavailable!**

| `DisplayFPS` | `Default: false` |
|---|---|

Shows the following values on screen:

**FPS**

The current speed at which the game is running (should be 60)

**TFPS**

This number represents how much overhead is available after rendering the game. The larger the percentage, the better.

**FT**

The frame time it takes to render each frame. Ideally should be around 16.6666667ms. Higher values mean that the game is not reaching its 60fps target.

| `DirectRender` | `Default: false` |
|---|---|
| Renders the game directly to the back buffer. Can allow for a significant speed increase, but it is also rather buggy. Certain menus will not render, all Video and Post Process options are ignored, and the game is rendered at the screen's native resolution. It is recommended to keep this disabled except as a last resort. | |

| `Page` | `Default: 0` |
|---|---|
| When using the Debug Overlay, this is the "Page" you're on. Controls what the Debug Overlay shows you. | |

| `HalfFPSMode` | `Default: true` |
|---|---|
| When enabled, allows the game to drop into 30fps rendering when the performance target isn't being met. **This does not force the game to run at 30fps!** It only locks the framerate to 30fps if 60fps cannot be achieved. | |

| `ToolPath` | `Default: datafiles/` |
|---|---|
| Shield Cat has the capability to self-rewrite its own code to keep track of the location of collectibles, update cached files, etc.. By setting this path and then creating a symlink to where the game's project or installation folder is located, you can enable this functionality as well as additional debug tools. **However, messing with this function can cause your game to become completely unplayable and corrupted as well as destroy your save files and possibly irreversibly damage your install!** **Do not use this unless you understand the consequences!** | |

In addition to these, there are the following values for the Debug Console:

`ConsoleBGC`
: Controls the background color of the Debug Console. Accepts an HTML Hex value.

`ConsoleBGA`
: Controls the transparency of the Debug Console's background. Accepts a value between `0` and `1`.

`ConsoleText`
: Controls the color of the text in the Debug Console. Accepts an HTML Hex value.

Please note that you can also change these from within the Debug Console itself.

# Input Configuration

In Shield Cat, it is possible to set whatever kind of actions you like to whatever kind of buttons/keyboard controls you like. To facilitate this, there is the `keyboard.ini` file, as well as individual ini files for each gamepad you connect to the game.

It is important to note that, when changing these files, the entries can be in any order, and the game may rearrange the order randomly. Do not worry about making the ini files neat!

When assigning keys or gamepad values, the format goes as follows:

For keys: `T="pause"`
For Gamepads: `gp_plus="pause"`

The key or gamepad button will be on the left, while the action will be on the right in quotation marks. Below I have detailed all the game's actions, as well as all the possible keys and gamepad values you can use.

**Please note, you can easily remap input devices from within the game,** so it is usually not necessary to edit the files directly. However, this reference is provided in the unlikely event that you do need to do so.

When using the in-game remapper, it will attempt to make sure that you cannot make the game unplayable, but this security is not guaranteed when editing the files directly. If you accidentally make the game unplayable, simply delete the configuration files and the game will regenerate them when you next run the game. (Do not delete any configuration files while the game is open or it may crash!)

# Game Actions

Everything you can do in the game is considered an "action." Such actions can be "attack" "move left" etc. They are detailed below.

| Action String | Description |
|---|---|
| `action_1` | Cause Lance to use his currently equipped MoveTech. |
| `action_2` | Cause Lance to use his current Throwable Ability. |
| `action_3` | Causes Lance to run when held. |
| `pause` | Pauses the game. |
| `talk` | Allows Lance to talk to people. |
| `ok` | Accept button for menus, dialog boxes, etc. |
| `cancel` | Cancel button for menus, dialog boxes, etc. |
| `select` | Opens up the full map screen. |
| `show_ui` | Opens a small preview to show stats for the current stage without pausing the game. |
| `shoulder_l` | Cycles to Lance's previous ability in the list. |
| `shoulder_r` | Cycles to Lance's next ability in the list. |
| `dir_up`<br>`dir_left`<br>`dir_down`<br>`dir_right`<br>`dir_up_left`<br>`dir_up_right`<br>`dir_down_left`<br>`dir_down_right` | Make Lance go different directions, select items on menus. Please note that it is not required to map the diagonal directions, as holding dir_up and dir_right will automatically activate dir_up_right. |

When assigning abilities to a key or gamepad button, you can combine them. For instance, if you want the "Q" key to make Lance attack, talk, and press ok, then you can do the following:

```
Q="attack,talk,ok"
```

**IMPORTANT: You MUST put all action keywords in lower case, or the game will not read them!**

# Gamepad Strings

The following details various strings and what they represent on a gamepad. Please note that there are different values for XInput and DirectInput devices, and using one set of values on the other may result in unpredictable results.

The game will attempt to automatically map actions to the controller, but some controllers will have to be mapped manually. The game will let you know if this is the case.

In addition to these, you can set the vibration like this:

```
gp_vibration_allowed=100
```

It is a value between `0` and `100`, where `0` is no vibration, and `100` is full.

Please note that **ANY** connected controller set up to receive vibration will vibrate. If you find this undesirable, you can either disable vibration on that controller, or simply disconnect it from your computer. You may also edit the main.ini file and disable controller vibration there for all gamepads.

Each controller configuration has its own configuration file. You can find them in the game's Save Directory, in the subdirectory `/config/gamepads`. Each configuration file will have a name that contains its unique identifier. **Do not change the name that the game gives the configuration files, or it will no longer read them!** Also, please note that **all XInput devices will use** `gp_none_xinput_standard_gamepad.ini`**.**

Note that the game will attempt to load SDL 2 Gamepad mappings and, if successful, will populate the INI file with default XInput mappings for that device. This will allow some non-XInput devices to use XInput strings, but it is preferred that these are changed to DirectInput for future compatibility.

It should be noted that you can **NOT** use Gamemaker built-in constants in the configuration file! These will be ignored.

To help with configuration, the game features a debug overlay that you may activate. On one of the overlay's subscreens, the game will display any input that is currently being pressed, regardless of if it's actually mapped or not.

# XInput Configuration Table

| XInput String | Description |
|---|---|
| `gp_face_bottom` | The bottom button on the face of the controller. On XBOX Controllers, it is the Green A button. On PS, it is the cross. On Switch, it is the B button. |
| `gp_face_right` | The right button on the face of the controller. On XBOX Controllers, it is the Red B button. On PS, it is the circle. On Switch, it is the A button. |
| `gp_face_left` | The left button on the face of the controller. On XBOX Controllers, it is the Blue X button. On PS, it is the square. On Switch, it is the Y button. |
| `gp_face_top` | The top button on the face of the controller. On XBOX Controllers, it is the Yellow Y button. On PS, it is the triangle. On Switch, it is the X button. |
| `gp_shoulder_l1` | The front left bumper button on the controller. On XBOX Controllers, it is the LB button. On PS, it is L1. On Switch, it is L. |
| `gp_shoulder_l2` | The rear left bumper trigger on the controller. On XBOX Controllers, it is the LT button. On PS, it is L2. On Switch, it is ZL. |
| `gp_shoulder_r1` | The front right bumper button on the controller. On XBOX Controllers, it is the RB button. On PS, it is R1. On Switch, it is R. |
| `gp_shoulder_r2` | The rear right bumper trigger on the controller. On XBOX Controllers, it is the RT button. On PS, it is R2. On Switch, it is ZR. |
| `gp_plus` | The start button on an XBOX controller, Options on the PS4 controller, and Plus on the Switch controller. |
| `gp_minus` | The back button on an XBOX controller, Touchpad button on the PS4 controller, and Minus on Switch. |
| `gp_l3` | The button when you press the left analog stick. |
| `gp_r3` | The button when you press the right analog stick. |
| `gp_dpad_up`<br>`gp_dpad_right`<br>`gp_dpad_bottom`<br>`gp_dpad_left` | The directional pad on the controller. |
| `gp_lstick_up`<br>`gp_lstick_down`<br>`gp_lstick_left`<br>`gp_lstick_right` | The left analog stick. Please note that the game only supports 8 movement directions, so precise analog movement is not possible. |

| | |
|---|---|
| `gp_rstick_up`<br>`gp_rstick_down`<br>`gp_rstick_left`<br>`gp_rstick_right` | The right analog stick. |

## DirectInput Configuration Table

| DirectInput String | Description |
|---|---|
| `gp_b0 thru gp_b29` | The number of the button on the DirectInput device. If you wanted to check button 5 for example, you would use `gp_b5`. Please note that button assignment starts at `0`, and may not always match what the gamepad may physically say, nor what configuration tools such as Windows' USB Gamepad Devices screen may say. |
| `gp_h[1-4]_up`<br>`gp_h[1-4]_right`<br>`gp_h[1-4]_down`<br>`gp_h[1-4]_left` | The index of the Point of View hat (POV) on the controller. For example, if you wanted to configure "up" on the secondary hat, you would use `gp_h2_up`. |
| `gp_a[1-4]_up`<br>`gp_a[1-4]_right`<br>`gp_a[1-4]_down`<br>`gp_a[1-4]_left` | The index of the Analog stick/trigger on the controller. For example, if you wanted to configure "down" on the third one, you'd use `gp_a3_down`. |

## Keyboard Strings

The following is a list of strings that you can use to configure the keyboard.

```
left right up down enter space shift control alt backspace tab
home end delete insert pageup pagedown pause printscreen f1 f2
f3 f4 f5 f6 f7 f8 f9 f10 f11 f12 numpad0 numpad1 numpad2
numpad3 numpad4 numpad5 numpad6 numpad7 numpad8 numpad9
multiply divide add subtract decimal
```

In addition to these, you can use any letter between `A` and `Z`, as well as any number between `0` and `9`.

Please note that keyboard strings must ALWAYS be in lowercase, while the individual letters must ALWAYS be in upper case. Otherwise, the game may not read them.

# SDL Game Controller Database

Shield Cat has the ability to attempt to load a predefined input configuration from the SDL Game Controller Database file included in the game. This is a community sourced file which has presets for many different types of gamepads. This allows the gamepad database to be updated easily after the game has been released with new gamepads that didn't exist at the time of release.

## Updating the Database

You may find the database file within the installation directory of the game, located at `/gamepad_db/gamecontrollerdb.txt`. However, it is not recommended to replace this file or add custom mappings to it, as they can be overwritten when the game updates! Instead, you may download the file to the game's Save Directory, which is located in `[SaveDir]/gamepad_db/custom_gp.txt`. Please see the section about Platform Specific notes to find out where the save location is on your computer.

You can download the latest version of the SDL Game Controller DB here:
https://github.com/gabomdq/SDL_GameControllerDB

## Built In Presets

In addition to the SDL database, there are a number of additional mappings included in the game. You can locate these in the installation folder, under `/gamepad_db/ini`. These are INI files which were contributed by the community, and contain mappings specific for Shield Cat. If you wish to contribute your own mapping, please see the segment about Unrecognized Controllers.

As data here can be overwritten, it is recommended that you do not store your own gamepad configurations here! Instead, these can be found in `[SaveDir]/config/gamepads/`.

# Save Files

The game uses JSON files to save the player's progress. **This documentation assumes you are already familiar with JSON files.** If you are not familiar with JSON files, it is recommended that you become familiar with them before reading this part of the manual.

The save file location depends on the platform. Please see the Platform Specific notes to find out where they are saved. All Save Data files will be located in `[SaveDir]/savedata/`, and are saved in the format `ottsav01.json`, `ottsave02.json`, etc. It is important to note that these are Hexadecimal numbers. For instance, the 12th save file will be saved as `ottsav0c.json`, and the 50th save file will be `ottsav32.json`. **All save files must be lowercase or the game will not read them!**

When saving the game, backup files are created. This allows the game to revert to how it was before the save in case the save was not successful, to prevent loss of data.

**Important**: In the Steam version of the game, you may see a file called `steam_autocloud.vdf`. **Do not modify or remove this file.** It is used to allow save files to sync to the Steam Cloud, and is required for the Steam Cloud function to work.

# Loading Save Files

When loading save files into the game, they are passed through a strict validation check. This validation makes sure that all the values are correct, and initializes them to their defaults if not. It also removes any part of the save file that isn't used in the current version of the game. **This is why the game will refuse to load newer save files from an older version, as the validation routine will remove otherwise valid data from the future.**

# Save File Format

There are two main JSON objects in the save file. They are:

- `save_data`: Contains general information about the game state.
- `lance_data`: Contains data specific to Lance.

## Save Data

The save_data object contains general information about the game state. This is a table of the variables you'll find in there, as well as what they're used for. **Due to the complexity and differences of data types, specific values or types cannot be given.** Ideally, you should never have to edit the save file directly and it is also a bad idea to do so anyways.

| | |
|---|---|
| **version** | The save file version. **UNDER NO CIRCUMSTANCES SHOULD YOU CHANGE THIS.** This is used to upgrade the save file to newer versions as needed, and setting it to an older version **WILL** cause loss of data. |
| **game_play_time** | How long the player has been playing (in seconds.) |
| **game_microtime** | The milliseconds portion of how long the player has been playing. |
| **real_game_play_time** | The actual play time, including during menus and dialogue sequences. The microtime is included in this number as well. |
| **game_save_year**<br>**game_save_month**<br>**game_save_day**<br>**game_save_hour**<br>**game_save_minute** | The time at which the player saved the game. |
| **game_state_flags** | Keeps track of various information about the game, such as the difficulty mode. |
| **save_location** | Where the player last saved their game. |
| **show_speedrun_clock** | Whether or not to show game play time during gameplay. |
| **room_vars** | Contains a list of all variables for every stage in the game, where the key is the internal name for the stage, and the value is another object in which the data for that stage is stored. Please see Room Vars for more details. |

| | |
|---|---|
| **minigame_data** | This contains the player's times for all of the minigames. |
| **global_flags** | The Global Flag table. See Flag Tables for more information. |
| **map_marker_data** | This contains information about the Map Markers that the player can place on the map screen. If the position is set to `undefined` then the player isn't using this map marker. All Map Markers are relative to wherever Lance's house is on the map screen. This allows the map to be changed and expanded without invalidating the existing map markers. |

## Room Vars

This object contains information about every stage in the game and what the player has done in that stage. The key is the internal name of the stage. For instance, the key for the Lignum Woods - Deep Woods area is `room_lignum_stage_1`. Each one of these will have the following values:

| | |
|---|---|
| **flag_buffer** | Buffer containing all the room-specific flags for this stage. Please see Flag Tables for more information. |
| **scale_buffer** | Buffer containing all the Fish Scale information for this stage. Please see Flag Tables for more information. |
| **key_count** | How many keys the player has for this stage. |
| **circle_key_count** | How many Circular Lock Pieces the player has. Every 3 makes a complete key. |
| **scale_count** | How many Fish Scales the player has in this stage. |
| **cat_coin_count** | How many Cat Coins the player has in this stage. |
| **jewels_flag** | Bitflag that represents which Big Jewels the player has collected in this stage. |

## Lance Data

This segment contains data specific to Lance.

| | |
|---|---|
| **lance_max_hp**<br>**lance_max_mp**<br>**lance_attack** | These values show Lance's stats on the File Select screen, and are recalculated whenever you load the game. They are stored and provided only for convenience. |
| **lance_upg_atk**<br>**lance_upg_hp**<br>**lance_upg_mp** | Similar to above, these are only for showing how many of each Upgrade Lance has, and are only provided for convenience. Changing these values will not change Lance's stats, as they will be recalculated later. |
| **game_completion_percent** | The overall completion rating of the entire game. |
| **pretty_petals** | How many Pretty Petals Lance has |
| **pretty_petals_total** | The total amount of Pretty Petals that Lance has ever collected. |
| **lance_gameover_count** | How many times you have run out of health and restarted at your last save point. |
| **lance_revives_count** | How many times you were revived by Willow after running out of health. |
| **enemies_defeated** | How many enemies you've defeated. |
| **steps_taken** | How many steps you have taken (increments for every 16 game units you walk.) |
| **health_fish_collected** | How many Health Fish you've picked up. |
| **petals_collected_green**<br>**petals_collected_blue**<br>**petals_collected_red**<br>**petals_collected_gold**<br>**petals_collected_rainbow** | How many of each type of Pretty Petal Lance have been collected. |
| **pretty_petals_spent** | How much you've spent in Pretty Petals. |
| **pit_fall_count** | How many times Lance fell into a pit. |
| **hit_by_enemy_count** | How many times an enemy hit Lance successfully. |
| **overall_damage_taken** | How much damage Lance has received from enemies. |
| **friendly_damage_taken** | How much damage Lance has received from his own attacks. |

| | |
|---|---|
| **screens_seen** | How many screens you've stepped into. This increments whenever the screen transitions from one area to the next. |
| **pots_broken** | How many pots you've broken. |
| **secret_spots_found** | How many "secret spots" you've found. |
| **lance_state_flags** | Bitflag value that keeps track of various information about Lance. |
| **fish_scales_total** | How many Fish Scales Lance has throughout the entire game. |
| **big_jewels_total** | How many of each Big Jewel number Lance has. There are 5 separate Big Jewels in each stage, numbered from 0 to 4, and this increments how many of each ID he has. This is leftover from when the game used LANCE letters as collectibles, before they were replaced with Big Jewels. |
| **big_jewels_all_total** | The total amount of Big Jewels Lance has. |
| **fish_scales_all_total** | How many Fish Scales are in the entire game. Used to make stats calculations faster internally. |
| **cat_coins_all_total** | How many Cat Coins are in the entire game. Used to make stats calculations faster internally. |
| **lance_items** | A list of items that Lance has, where the key is the internal ID of the item, and the value is how many he has. |
| **lance_active_ability_1** | What MoveTech Ability Lance has equipped. |
| **lance_active_ability_2** | What Thrown Ability Lance has equipped |
| **lance_abilities_owned** | A list of abilities Lance owns. It is important to note that both Base Abilities as well as their Upgrades are stored in this list. In the case of a Base Ability, "Equipped" is updated to show if it's the active one that the player is using currently. In the case of an upgrade, "Equipped" tells if that upgrade is equipped or not. |
| **lance_passives_owned** | Which Passive Abilities Lance owns and has equipped. The game will validate if Lance actually has enough magic to equip the abilities specified here, so don't lie. |
| **mail_list** | List of mail Lance has in his inventory. |

# Flag Tables

Flag Tables are a binary buffer which has been compressed and converted to Base64 (to prevent data loss when being stored in the save file.)

## Global and Room Flags

These keep track of various events and states in the game, such as talking to NPCs, opening gates, and more. Due to the complexity of the flags, they are not documented here and are simply documented within the game's project file itself. You can use the Debug Console to edit these flags using `gflags` and `rflags`, respectively. However, keep in mind that you can only modify the room flags for the stage you're in. Additionally, you may use the command `resetflags` to reset all the room flags in the current room you're in. Due to how much is stored in the Global Flag Table, there is no way to reset the Global Flags.

## Fish Scale Flags

These keep track of what Fish Scales the player has obtained in the stage, using the Fish Scale ID as the offset for the bit to check to see if the player got it or not. The command `sflags` exists to modify these, but it is generally not necessary to edit these individually. However, you can use the Debug Console command `resetscales` to make it as if you never collected any Fish Scales in the stage.

# Custom Difficulty Files

You will find these in `[SaveDir]/savedata/cstmdiff/`. They correspond with the `ottsav` file with the same number. The INI file expects the following values.

| **PlayerHealthMod** | **Range: -2 thru 6. Default: 0** |
|---|---|
| How much to add or subtract to the player's base health (which is 3.) | |

| **PlayerMagicMod** | **Range: -1 thru 7. Default: 0** |
|---|---|
| How much to add or subtract to the player's base magic (which is 2.) | |

| **PetalDropMult** | **Range: 0 thru 4. Default: 1** |
|---|---|
| Multiplier which determines the chance that Pretty Petals will drop from enemies or when cutting grass, for example. A value of 0 will make Petals never drop, while a value of 4 will make them drop way too much. | |

| **HealthDropMult** | **Range: 0 thru 4. Default: 1** |
|---|---|
| Multiplier which determines the chance that Health Fish will drop from enemies or when cutting grass, for example. A value of 0 will make Fish never drop, while a value of 4 will make them drop way too much. | |

| **EnemyAggression** | **Range: 0 thru 3. Default: 1** |
|---|---|
| This determines how aggressive enemies are in their attack, and is the main decider of difficulty across the standard difficulty modes. It can also affect certain other aspects of the game. The numbers directly correspond to the difficulty modes:<br>- `0`: Chill<br>- `1`: Regular<br>- `2`: Tough<br>- `3`: Wild | |

| **EnemyDamageMult** | **Range: 0 thru 4. Default: 1** |
|---|---|
| Determines how powerful enemy attacks are. A value of 0 means that enemies will not damage the player, while a value of 4 means that every guy in the game will pretty much OHKO you. | |

| `PlayerDamageMult` | `Range: 0 thru 4. Default: 1` |
|---|---|
| Determines how much damage the player deals to enemies. A value of 0 means that the player will deal no damage, while a value of 4 means that the player will do a lot of damage. It should be noted that self-inflicted damage also scales off this modifier, so if for example you throw a bomb too close and hit yourself with it, you will take damage relative to this modifier. | |

# Save Thumbnails

When saving the game, a little screenshot of the game is taken. They are stored in `[SaveDir]/savedata/thumb/` as PNG files, and are shown to the player next time they load up the game so they may see where they had played last time.

# Font Packs

To help make it easier for the player to pick fonts to use within the game, there are predetermined Font Packs included in the game. You can find them in the installation directory under `/fonts/font_packs/`.

```
[PackInfo]
pack_name=Default
menu_header=amaranth_bold
menu_windowtitle=amaranth_bold
menu_regular=amaranth
dialogue_regular=amaranth
dialogue_bold=amaranth_bold
dialogue_italic=amaranth_italic
dialogue_bolditalic=amaranth_bolditali
allow_resize=1
menucount_smaller=16
menucount_small=14
menucount_medium=11
menucount_large=9
menucount_larger=8
```

To the left, an example of the default Font Pack is provided. You may specify the name of the Font Pack to appear in the game. Each option is the name of a folder containing a font for the game to use. For example, the specification `amaranth_bold` would load the font from `/fonts/amaranth_bold/`.

The `menucount` options specify how many menu items can appear on the pause menu. This is important because the menu will otherwise render off screen and be unusable to the player.

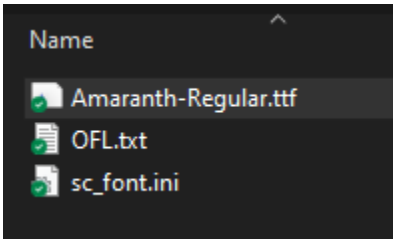
Finally, the option `allow_resize` specifies if this font can be upscaled or not. You need to disable this for pixel art fonts, or they will appear blurry!

It should be noted, if the same font is specified two or more times within the file, the game will only load the font file into memory once, and then use it for any time it is referenced.

# Adding Fonts

You may add fonts by creating a folder for the font in /fonts/ and then placing the file in there.

```
[Font_Information]
font_name=Amaranth
font_file=Amaranth-Regular.ttf
font_about=Copyright 2016 The Amaranth Project Authors
font_base_size_large=17.5
font_base_size_medium=11
font_base_size_small=8
font_is_bold=false
font_is_italic=false
font_range_start=32
font_range_end=255
```




Each font requires the font as a TTF file, as well as the `sc_font.ini` file, which tells the game how to read the font. An example of the `sc_font.ini` file for the Amaranth Regular font is provided above.

The `font_name` appears in the game for the player to choose. The `font_about` may also appear in the game. The `font_file` specifies the name of the font file to load. You can also specify if the font is bold or italic. You **must** specify these because the game can't pick up on this on its own, even if the font has the name "bold" or "italic" in it. The `font_range` specifies the unicode start and end range to load into memory. Please try to only load into memory what you need. Unfortunately, multiple sets of ranges cannot be specified.

The `font_base_size` options tell the game how to resize the font. These values are relative to the base resolution of 360p. Therefore, when the game is running at 1080p, a font base size of 17.5 will become 52.5 when loaded into the game.

When loading the fonts, you will sometimes have to specify the font line height yourself, as there are no options to change this when loading it into the game. You may use a program like FontForge to edit the file and prepare it to be loaded into the game.

Finally, all font folder names **must** be in lowercase, or the game will not read them.

# The Data Folder

The data folder is located in the installation directory under `/data/`, and contains a lot of different information and images that the game uses during runtime. There are built-in editors for much of this data, which can be enabled by creating a symlink to the installation directory of the game. Please see the Debug segment of the Main Configuration file for more information about how to enable these editors. **Please note that enabling these editors can damage the game if you are irresponsible with them!**

It should be noted that many of these files are either automatically generated, or are only designed to be edited using in-game tools. The formats for these files are included for reference below, but you should not modify these files directly unless otherwise noted!

In this section of the manual you will find details about the different files and folders located in the Data folder.

## Overlays

Each screen takes up a 160x90 portion of the overlay. The maximum size that a room can be is 12 screens by 22 screens, or 7680x7920. This results in an overlay that is 1920x1980. Any bigger would be larger than 2048x2048 and risks being automatically downsampled when loaded into the game. Please note that these overlays can be edited using in-game tools, but cannot be changed by the game during regular gameplay. You may use the Debug Console's `gedit`, `aedit`, and `ledit` respectively.

There are three different types of overlays used in the game, located in `overlays`, `alphas`, and `lights` respectively. They are as follows:

**Color Overlays**

These are overlaid onto the game using an "overlay" style blend mode. They allow parts of the screen to be lighter or darker, as well as differently colored. These are the main overlays which help control colors in the game, and to darken edges of the screen in certain areas such as caves.

**Alpha Overlays**

These overlays are drawn directly to the screen using alpha blending. You can use them to make lights bright or obscure part of the game world.

**Light Overlays**

These overlays tell the game where to use lights in rooms that have light sources. They are used in caves and whatnot so we don't have to have individual objects for lighting.

It should be noted that the in-game tools for editing these overlays are rudimentary, and are intended to be used in conjunction with an art program. Each of the in-game editors allows you to load the file and update it at runtime. Using the in-game editor as well as an external editor will help you to create a lovely overlay.

Also, when editing the files in the game, they use Gamemaker Surfaces. These are lost whenever you resize the window, causing you to lose your work. If you save after losing your work, you will lose the entire overlay. Instead, you can simply reload the overlay from disk to continue editing. Therefore, do not resize the window while editing, and if you do have to resize it, save **before** resizing it.

## Minimap Data

This file, located at `/data/minimap_data.ini`, contains information about all the maps in the game. Each key represents a Gamemaker DS Grid, which has been converted to a string, compressed, and then converted to base64 to prevent data loss. Each grid position is a bitflag value representing all map data for the relevant screen within the game.

All of the minimap data is contained within a single file for ease of access for the game during gameplay. The Debug Console command `medit` will allow you to edit this file.

## Color Data

The color data files, located in `/data/colors/`, are a set of `.bin` files that represent color information for all the screens in the respective stages. Their filenames match the internal name for the stage within the game. The Debug Console command `sedit` will allow you to edit these files from within the game.

Each file is simply a Gamemaker DS Grid which was converted to a string and compressed. Below are the details of each column in the DS Grid.

| Col | Internal Variable Name | Value | Default | Notes |
|---|---|---|---|---|
| `0` | `camlock_grid_position` | `0xXXYY` | `N/A` | Bitmath X and Y grid position of the screen. For example, a screen at grid position 5,4 would have the value of `0x0504`. |
| `1` | `surface_blend_color` | `GML Color` | `$FFFFFFFF` | The base blend color for this screen. |
| `2` | `surface_overlay_color` | `GML Color` | `$FF7F7F7F` | Applied to the screen using “overlay” color math. |
| `3` | `surface_bg_clear_color` | `GML Color` | `$FF000000` | When using `shadow_opacity`, this color appears behind the game view. Can be used for fog effects. |
| `4` | `surface_opacity` | `0-1` | `1` | How transparent the entire game view is. Not really used. |

| | | | | |
|---|---|---|---|---|
| 5 | surface_hue | 0-360 | 0 | Perform a hue shift on the game view. |
| 6 | surface_sat | 0-2 | 1 | The saturation of the game view. |
| 7 | surface_val | 0-2 | 1 | How bright or dim the game view is. |
| 8 | use_light_sources | Boolean | false | Puts light sources on the player and other objects. This is not required to be enabled for screens to use the Light Overlay, allowing for really dark or mysterious rooms. |
| 9 | shadow_opacity | 0-1 | 1 | Indicates how “transparent” the game view is. The lower the value, the more transparent it is. |
| 10 | use_perimeter_blur | Boolean | false | This enables a stylistic blur around the edge of the screen for this screen. |
| 11 | retro_pal_override | 0-MaxPal | 2 | Which predefined Retro Palette this screen is to use. Recommended to keep the value at 2 or above. |

## Fish Scale Data

These files, located in `/data/scales/`, keep track of what scale ID belongs to each Fish Scale in the game. To ease in development, the game automatically assigns an ID to each Fish Scale in the room and stores it in this file so that later, even if scales are added or removed, they will still have their ID. This removes the need of having to assign a scale ID to each individual scale within the Gamemaker IDE.

Each file is a DS Map that has been converted to a string and then compressed. Each key is the position of the Fish Scale, floored and then stored in `0xXXXXYYYY` format. Each value is the ID that has been automatically given to this scale by the game. For instance, a Fish Scale that is at 4284, 5930 will be stored as `0x10BC172A`.

It is important to note that Fish Scale IDs are not automatically reassigned to other Fish Scales. In the event you remove Scales from the game using the Gamemaker IDE, you will need to use the Debug Console command `fixscaleissues` to remove references to Scales that have been removed from the game. However, as you are limited to the amount of scale flags available, it is recommended to change the scales as little as possible.

There is no in-game editor for this file, as the game will automatically update it when the extra debug tools are enabled.

## Collectible Data

This file, located at `/data/collectible_data.ini`, contains automatically generated totals for each stage in the game, as well as totals for all collectibles in the game. It is referenced from within the game to determine how "complete" the game is. It is strongly recommended that you do not modify this file, as the game automatically updates it as needed when the extra debug tools are enabled.

## Supporters

These CSV files simply contain the names of supporters of the game, arranged by how much they're supporting the game.

## Flags File

This file is used to name the different types of flags available in the game. It is updated through a function that reads the game's source code for enums representing the flag values. Because it uses the game's source code to update, it's not really something you can set up to update yourself.

# The Language Folder

Shield Cat has the ability to be translated into many different languages through the use of Language CSV files included with the game. Each language is contained in a subfolder in the `/lang/` folder, with the default English language being located at `/lang/en_us/`.

# Translating the Game

**IMPORTANT! Shield Cat is currently a game in development! This means that language strings will be added, changed, or removed without notice. It is *strongly* recommended not to attempt to create a translation for the game until after it has been released!**

When creating a language file for the game, the first step is to create a folder in the `/lang/` directory for it. Each language must have a `lang.ini` file for it to be picked up by the game. Here is an example of the default English file:

```
[Language_Info]
name="English (USA)"
author="CyanSorcery"
font_packs="default,opendyslexic,playfairdisplay,cabin,amaranth,retro"
font_default="default"
```

The name of the language will appear in the game when the player chooses it. You can also specify which Font Packs are available to this language, in addition to the default. When the player picks a different language, their Font preferences are reset. Please see the segment about Font Packs for more information.

There are several things that you need to keep in mind when working with or adding translation files:

- When loading a language that is not English, the game will first load the English strings, and then load the desired language on top of that. This means that missing strings will always be presented in English.

- In the event that the game cannot find a string associated with the requested language string, the game will simply display the requested language string.

- If any language strings or files are missing, the game will let you know when you run it. In the Save Directory for the game, you'll find a `missing_lang` folder. This will contain a lot of useful information about what strings weren't found when loading the translation in. **WARNING:** These files are regenerated when you restart the game, so don't put any notes or anything else into these files or they will be lost!

- This comparison function can only let you know when a string is not present in the language file. It cannot let you know if a string has been updated.

- **IMPORTANT:** CSV files **MUST** be in "UTF-8 with BOM" format with the newline format as "CLRF" or they will **NOT** be read by the game. On top of this, the game will **NOT** give you any kind of error indicating that it's gone wrong.

- Finally, the game only reads language strings from disk when it first starts up. This means that you will have to close and re-open the game to see any changes you've made be applied.

## Language File Format

Each Language file is a CSV that contains all the data the game needs. It uses the following format:

| Language Key | Language String | Comments |
|---|---|---|

**Language Key**

This is a string that the game uses to look up the string to use.

**Language String**

This is the actual language string.

**Comments**

This is where you can put translator notes. Please note that Multi Line language strings can also have comments within them, so this is mostly used for Single Line language strings.

**It is strongly recommended that you thoroughly read this segment of the manual before attempting to modify the language files in any way.**

# Dialogue Instructions

Shield Cat uses a built-in command interpreter for dialogue processing that allows very advanced and complex dialogue to take place. It also allows for styling of text, as well as custom effects, conversation branching, and more.

**IMPORTANT:** As this game is currently in development, more commands may be added and some commands may be removed at any time. Depreciated commands will be reported in the debug console.

If you put an invalid command in the dialogue, the Dialogue Parser will warn you at runtime with both a message in the debug output, as well as with a message on the screen.

## Single Line vs Multi Line

There are two types of dialogue that can be read by the dialogue parser. In Single Line mode, all instructions are in one line. This is used for signs that only have one line of dialogue, but otherwise is mostly depreciated in favor of the more powerful Multi Line mode. In Multi Line mode, you get the full power of the dialogue parser. Whether an object uses Single Line or Multi Line is determined by the game itself, and cannot be changed by the language files.

It is important to note that, in Multi Line mode, the dialogue parser will go line by line, processing each command or language string in turn. There are functions that allow you to jump around in dialogue, so please be aware of how the dialogue parser works.

It is also important to note that this **only** applies to dialogue which is intended to appear in Dialogue Sequences. Some language strings are meant to show on the menu, in item descriptions, etc, and will **not** support most of the commands present here! Most of these language strings are in the General CSV.

## Standard Dialogue

A standard dialogue sequence will look something like this:

```
;Any line that starts with a semicolon is considered a
;comment and will be ignored by the dialogue parser.

>actor Roxy as Actor0
>voice Roxy set roxy

Roxy: Hey Lance, I'm talking to you from within the manual.
Roxy: Isn't that [wave]neat?[/wave]
Lance: The documentation? What on earth are you talking about?
Roxy: Haha, nevermind.

?Roxy: Anyways, what do you think about shields?
- continue<default>: They're good
- dont_like<cancel>: Don't like them
?end

;If the player likes shields (note the use of "continue"
;which simply continues dialogue parsing below)
Roxy: I'm really glad that you like shields!
;We must use this command or we'll go into
;the dialogue where the player doesn't like shields.
>end

;If the player doesn't like shields
>anchor dont_like
Roxy: Well, the game is called [b]Shield[/b] Cat…
Roxy: …so you better get used to shields being in it!
;While it's not necessary to use "end" here, it's still
;good practice in case you add more dialogue later.
>end
```

At first glance, this may seem super complicated! However, once you understand it, it's not that complicated at all. So, let's run through it.

## Commands

Dialogue commands will always start with `>`. There are a lot of commands available for you to use, which are all detailed further on in this manual. The dialogue parser assumes that any line that starts with `>` is a command, and any line that doesn't is part of the dialogue sequence. Please see the segment on Commands for more information.

## Comments

You can put comments directly into the language strings. Any line that starts with a `;` will be considered a comment. It is highly encouraged that you leave comments for anything that may be confusing.

## Dialogue

The Dialogue Parser is designed for one thing, and that's to process dialogue and put it to the screen! Here, you can see a conversation between Roxy and Lance. When switching to another character, the Dialogue Parser will automatically close the dialogue box over the first character, and then open a new one over the second one. You can also have multiple lines of dialogue for the same NPC in a row. In this instance, the game will reuse the same message box and put each new line in a new message.

In order to have characters talk to each other, we use the concept of "Actors." Please see the segment on Actors for more information.

## Other Features

The Dialogue Parser also allows you to ask Questions to the player, with branching paths depending on the player's response. In order to branch the dialogue, we use Anchors to specify places we can jump in the dialogue. There are also Tags that you can use to specify **bold** and *italic* text, as well as many other things.

# Actors and Name Assignment

"Actors" are any NPC or player character in the game world. NPCs can also be investigatable objects like signs. Some things to note about Actors:

- The calling instance for the dialogue sequence will always be defined as `Actor0`. For instance, if you are talking to the Gazelle, a reference to his in-game instance will automatically be assigned to `Actor0`. Please see below for how to change this.

- Lance is automatically assigned as `Lance` so you can always refer to Lance's player object in this way. Please note however that this is based on the language string in the General Language file. So, if you're translating to a different language, for example Japanese, where Lance's name is transliterated as Ransu or ランス, then you **must** change `lang_names_lance` to ランス, and then reference Lance as ランス throughout all other dialogue sequences. Otherwise, your reference will fail!

- Some dialogue sequences will have many NPCs. In this case, `Actor1`, `Actor2`, `Actor3` etc. will be available to use. These are defined on a per-dialogue basis.

The command actor will set the name for each actor that can be in the scene, which will correspond with the internally assigned data for that NPC. For instance, you can use the following command to be able to refer to the Gazelle in the cutscene as `Gazelle` instead of `Actor0`:

```
>actor Gazelle as Actor0
```

Using this command will allow you to do the following:

```
Gazelle: What, now you're using me in the language documentation?
```

If you wanted to translate the game to Japanese, you would do the following:

```
>actor ガゼル as Actor0
```

You would then be able to do this:

```
ガゼル： 明らかに、ロキシーは翻訳プログラムを使用しています。 彼女は日本語さえ知らない！
```

This functionality allows you to reference the NPC across multiple languages without having to use the English name for that NPC.

If an unknown Actor is specified, the dialogue box will show in the center of the screen, and an error message will appear on the screen. If you intentionally want to center the dialogue box in the screen, you can do the following:

```
noone: If no one says a message, does that mean anyone even said it?
```

This may be used even if a named NPC is talking, in the event that you wish for the dialogue to be centered for some reason.

It should also be noted that you may use spaces in actor names by using backticks. For example, if you wanted to refer to a character called Mr. Railway, you'd do the following:

```
>actor `Mr. Railway` as Actor0
Mr. Railway: Trains are very good.
```

# Dialogue Tags

These are commands that you can use within a dialogue sequence. Some of them are standalone tags, while others require a terminating tag. While it is not necessary to order the terminating tags (or even to put the tags at all) it is very good practice to make sure all the tags are in place and organized to prevent issues from arising. Some of these tags also accept arguments, which will be detailed below.

| **Emotes** | `[emote:emote_type]` |
|---|---|
| Cause the actor who's currently speaking to perform the specified emote. Valid emotes differ between scenes and may not be valid for the specified cutscene sequence. However, all emotes have fallback sequences in the event they're not valid for the current sequence. Please see the segment about Emotes for more information. | |

| **Emotes (for other actor)** | `[emote:emote_type_actor]` |
|---|---|
| This extends the Emote command to work on an actor who is not the current actor who is speaking. You can use this to cause other actors to react to something said immediately when it is printed on the screen. When targeting another actor, you must use the name you've given them in your translation. Please see the segment on Actors for more information. | |

| **New Line** | `[nl]` |
|---|---|
| Forces a line break here. If this is the third line in the current dialogue set, this will make the player have to press the button to continue the dialogue. However, please be aware that different fonts show different amounts of words per line, so you can't rely on this always working. | |

| **New Message** | `[nm]` |
|---|---|
| When using the Single Line mode, this causes any text following this to be placed in a new message. Does nothing in Multi Line mode. | |

| **Color** | `[col:color][/col]` |
|---|---|
| Any text rendered after this command will use the specified pre-set color. Due to different dialogue sequences having different color schemes, it is not possible to define your own color. Here is a list of values you can use:<br><br>`item, ability, location, name, action, button, hint, important` | |

| **Bold** | `[b][/b]` |
|---|---|
| Any text in these tags will be bold (if the font supports it.) | |

| **Italic** | `[i][/i]` |
|---|---|
| Any text in these tags will be italic (if the font supports it.) | |

| **Wavy Text** | `[wave][/wave]` |
|---|---|
| Text will be animated in a wave-like fashion. Cannot be combined with other text effects. | |

| **Shaky Text** | `[shake][/shake]` |
|---|---|
| Text will be animated in a shaky fashion. Cannot be combined with other text effects. | |

| **Full Worder Rendering** | `[full][/full]` |
|---|---|
| Causes text to appear per-word instead of per-letter. Has no effect if the player's text speed is really high. | |

| **Zoom In** | `[zoomin][/zoomin]` |
|---|---|
| Will cause text to appear by starting small and then "growing" to its full size. | |

| **Zoom Out** | `[zoomout][/zoomout]` |
|---|---|
| Text will appear to "drop in" to the dialogue box by starting large and then shrinking to its regular size. | |

| **Automatic Text** | `[auto][/auto]` |
|---|---|
| Control is taken away from the player and the text will be rendered at the player's chosen speed until the ending tag is reached. If you use this without the terminating tag, it will cause the text to automatically go to the next dialogue (at which time the auto function will be canceled.) You can use this to have text display from one actor and then quickly swap to another actor to have them "cut in" to the dialogue. | |

**Voice** `[vo:style]`

This specifies the text blip used for the NPC that's talking. This is only useful in Single Line mode, as Multi Line mode has a different (and better) way to specify voices. Valid arguments are as follows:

```
none, deep, lower, mid, mid2, high, high2
```

Please note, some characters such as Lance or Willow have their own special voice blips. These are registered internally under their **English** names, so you will have to use the command as follows, regardless of what language the translation you're working on is in:

```
[vo:lance]
```

**Wait** `[wait:frames]`

Specifies how long the game should wait here (in game frames) before the next letter is shown. This can be any integer between 0 and 600, though it should be noted that 600 frames comes out to 10 seconds, and the player is not able to progress dialogue or otherwise take any action during this time.

**Wait For Var** `[waitfor:$var]`

This halts dialogue processing until the variable specified becomes true. This variable is predetermined per dialogue sequence by the game itself.

**Stop** `[stop]`

This stops dialogue processing completely. The dialogue will only resume when the player presses a button. Can be used for dramatic effect, but please don't overuse it.

**Show Variable** `[var:$var]`

This allows showing one of the built in variables for the dialogue sequence to be shown in the dialogue itself. This is usually used to show the price for items or minigames where the price has the chance to vary. These are pre-programmed per dialogue sequence.

## Depreciated Tags

These are depreciated commands and **SHOULD NOT BE USED.** They are only documented here as some of them may be remaining within the language files. They will cause an error to be shown in the debug output of the game, and eventually will not be recognized at all.

- `[td:style]` Was used to specify if text was bold or italic (or both) but was replaced by dedicated bold and italic tags.

- `[col:default]` Color was specified as a single tag, and so this was for resetting the color. The preferred method is to use `[/col]` to reset the color.

- [fl:bool] This was used to enable Full Word rendering. However, it is replaced by the `[full][/full]` tags.

- `[auto:bool]` This was used to enable Automatic Text Rendering. However, it is replaced by the `[auto][/auto]` tags.

- `[ti:style]` This was used to determine how text would initially appear on the screen. However, it has been replaced by the tags like `[zoomin][/zoomin]` and `[zoomout][/zoomout]`.

- `[te:effect]` This was used to specify text effects. However, it has been replaced by tags such as `[wave][/wave]` and `[shake][/shake]`.

# Variables

Variables start with $, and are predefined in code. This means that most variables are only available in the dialogue sequence they're being used in, and CANNOT be used in other dialogue sequences. Do not rely on variables to be available that aren't already present in the dialogue sequence!

Some variables, however, are consistent across all dialogue sequences. Here are a list of them:

**`$TalkLevel`**

How much the player has talked to this NPC, from 0-7. This allows the player to continue talking to a single NPC and get more dialogue out of them each visit. Some dialogue sequences may need more than 8, in which there will be a branch and (usually) two different dialogue sequence entries. Please note, some dialogue sequences may not have this variable defined ie. the player is reading a sign whose dialogue does not change.

You can set some variables using the following command:

```
>set $TalkLevel = 1
```

Please note that not all variables can be set, and are read only. Also, please note that some built in variables are actually Functions.

# Conditionals

You can check the value of a predefined variable by using the following command:

```
>if $TalkLevel == 1 goto talk1
```

In this situation, the value of the variable `$TalkLevel` will be compared to `1`. If it is equal to `1`, then we will go to the anchor `talk1` and continue text processing from there. If they are not equal, then text processing will continue on the next line. Please see the segment about Anchors for more information.

# Emotes

"Emotes" refer to an action an NPC can take to express some kind of emotion, such as anger or shock. Emotes can be specified either in between dialogue sequences or during dialogue sequences. Please remember that not all NPCs have all emotes available! They are predefined in the code, and if you use an unspecified one, it will just reset the NPC to their initial state. Emotes that are specified between dialogue sequences will happen immediately when that point of the dialogue is reached, while emotes that happen at the start of a string of dialogue will wait until the box is open to apply. Some emote sound effects also take priority over others.

To specify an emote in the middle of a dialogue sequence you would use the [emote] tag. Please see the segment on In-Dialogue commands for more details.

To cause an NPC to use an emote outside of dialogue, you can use the following command

```
>emote Actor emote_type
```

Where Actor is the name of the Actor you'd like to perform the emote. Please see the segment on Actors for more information.

# Anchors

Anchors can be thought of as a “choose your own adventure” book. In such a book, at the end of the chapter it will present you with a choice: To perform one action, turn to page 94. To perform another action, turn to page 69. Anchors can be thought of in the same way, specifying places within the dialogue that we can jump to depending on if certain conditions are met.

To create an anchor, the following command is available:

```
>anchor name_of_anchor
```

Since you are able to skip around anywhere in the dialogue sequence, it is important to know about the following command:

```
>end
```

This command immediately halts dialogue processing and closes the dialogue. It will not process any further dialogue after it reaches this command.

Another command available to use is the GoTo command, which is used like this:

```
>goto name_of_anchor
```

The dialogue parser will find the anchor you specify and continue dialogue processing from there.

## Questions

To cause the game to display a question prompt to the player, you must use the following format:

```
?Gazelle: I'm asking a question, will you answer it?
- yes<default>: Yes
- no<cancel>: No
?end
```

Each choice is prefaced by the `-` character, followed by the name of the anchor this choice will take you to, as well as the text that will be shown to the player in the game. It should be noted that a special anchor called `continue` is available to use. This will simply continue processing dialogue starting with the first command immediately after the question.

You'll also notice the presence of `<default>` and `<cancel>`. The `<default>` tag specifies that this will be the option highlighted when the dialogue box opens, while the `<cancel>` tag tells the game which option to pick in the event that the player presses the Cancel button. You can specify both tags in a single option by doing this:

```
- yes<default><cancel>: Yes
- no: No
```

The final part of the question is the `?end` command, which lets the dialogue parser know that the question has ended. You **must** include this option or the question will not be recognized!

It is possible to put icons by the answers. However, it is not possible to specify these icons from within the language files. These are often set up by an initialization command defined by the game itself, which must be run before the question is to be shown on the screen.

Finally, the game only allows 6 dialogue choices, and it is not possible to make a question with 0 answers.

# Commands

There are several commands available to control the flow of the dialogue sequence. All commands will begin with `>`. It should be noted that these commands are **only** available in Multi Line mode.

Several commands have segments dedicated to them, and are listed:

| **Conditional If** | `>if $TalkLevel == 1 goto cool_anchor` |
|---|---|
| Causes dialogue processing to branch depending on a predetermined condition. Please see the segment on Conditionals for more information. | |

| **Set Variable** | `>set $Var = true` |
|---|---|
| Set a variable to a specified value. Please see the segment on Variables for more information. | |

| **Emote** | `>emote Gazelle surprise` |
|---|---|
| Cause an Actor to perform the specified emote. Please see the segment on Emotes for more information. | |

| **Define Actor** | `>actor Gazelle as Actor0` |
|---|---|
| Assigns a name to a predefined Actor. Please see the segment on Actors for more information. | |

| **Define Anchor** | `>anchor cool_anchor` |
|---|---|
| Defines an Anchor in the dialogue that we can jump to later. Please see the segment on Anchors for more information. | |

| **Goto Anchor** | `>goto cool_anchor` |
|---|---|
| Move dialogue processing to the specified anchor, and continue from there. Please see the segment on Anchors for more information. | |

| **End Dialogue** | `>end` |
|---|---|
| Immediately end the dialogue sequence and close the dialogue. | |

In addition to these commands, the following additional commands are available.

| **Run Function** | `>func $Var` |
|---|---|
| Runs a predetermined function. Used for cutscenes, setting flags, giving the player items, and many other uses. The function's use cannot be changed from within the language files. | |

| **Set Interactable** | `>interactable Actor boolean` |
|---|---|
| Sets if the player can interact with an Actor object or not. Usually used to prevent the player from talking to an NPC again when the NPC has nothing else to say. | |

| **Actor Voice** | `>voice Actor set lower` |
|---|---|

This is similar to the `[vo]` tag, and accepts the same values. However, specifying it this way allows every part of the dialogue where this Actor is speaking to use the specified voice. This should be specified at the beginning of the dialogue, before the Actor speaks.

It should be noted that you can use the `[vo]` tag to overwrite the Actor's dialogue for a single sentence. You can do this, for instance, if part of the dialogue is meant to be whispered. Such a use would look like this:

```
>actor Roxy as Actor0
>voice Roxy set roxy
Roxy: Here's some regular talking.
Roxy: [vo:none]And here's some talking where no blips happen.
Roxy: [vo:mid]And now here's some slightly lower voice blips.
Roxy: And here this part would be normal again.
```

| **Reset Dialogue Target** | `>reset_target` |
|---|---|

This resets the dialogue target to noone. It can be used to close the dialogue box and then re-open it for the same NPC. Such a use case would look like this:

```
Roxy: I am going to become introspective for a while.
>reset_target
>wait 60
Roxy: I am no longer being introspective.
```

In the given example, the dialogue box would close for a bit, and then eventually re-open. Otherwise, the dialogue box would stay open and the wait would happen, which would look really weird.

| **Destroy Self** | `>destroyself` |
|---|---|
| Causes the calling actor to be removed from the stage entirely. Care should be used when using this function as, if the game doesn't expect the actor to be removed, the game will crash. This is often used for dialogue triggers where the player walks into a zone, a flag is set, and then the dialogue trigger is removed. | |

| **Take Pretty Petals** | `>takepetals amount failure_anchor` |
|---|---|
| Take this amount of Pretty Petals from the player. If the player doesn't have enough, dialogue processing goes to the failure anchor and continues from there. Otherwise, it continues as normal. It should be noted that the amount can be either an amount you specify here, or a `$Variable` defined by the game (preferred.) | |

| **Show or Hide Petals** | `>showpetals boolean` |
|---|---|
| Specify if the game should show the amount of Pretty Petals the player has or not. Useful for dialogues where the player can be charged, ie. for initializing a minigame. The Pretty Petals will be hidden again automatically when the dialogue is over.<br><br>Please note, this command only has an effect on screens where the player's UI isn't shown (such as inside houses) and will not have an effect anywhere where the UI is already shown (as the Petals will always be shown there.) | |

| **Show or Hide Dialogue Renderer** | `>renderer boolean` |
|---|---|
| Sets if the Dialogue Renderer will be visible or not. The Dialogue Renderer is complex, and so some combination of commands can lead to small visual bugs. To allow using these commands together, you can temporarily hide the Dialogue Renderer so the bug is not visible. However, please note that while it is invisible, it still works! If you leave it off, dialogue will be processed, but **it will not be shown to the player!** You should only set the dialogue rendered visibility to resolve visual bugs. | |

| **Dialogue Sizing** | `>sizing auto` |
|---|---|
| Set the dialogue box sizing. By default, it is set to `auto`. If you set it to `top` or `bottom`, the dialogue box will be set to the full width of the screen along the top or bottom of the screen respectively. Useful for long story based cutscenes with pictures and whatnot. When this is changed from the default, no commands specifying actors will work, and dialogue will not be positioned above any actor. | |

| **Heal Player** | `>heal_player` |
|---|---|
| This simply recovers the player's health and magic. | |

| **Open Train Selection Menu** | `>trainselect` |
|---|---|
| This opens the Train Selection menu that allows us to fast travel to a Train Station. It's important to know that the game **only** expects this function to be used with Parker at Train Stations! **Using it in other spots may crash the game.** | |

| **Show Speaker Labels** | `>showlabels boolean` |
|---|---|
| Whether or not to show a label to indicate who is speaking. | |

| **Use History** | `>usehistory boolean` |
|---|---|
| Whether or not we register upcoming dialogue in the dialogue history or not. Please note that this also disallows the player from opening the dialogue history for as long as it's enabled. | |

| **Reset Minigame** | `>resetminigame` |
|---|---|
| When entering into a minigame, certain flags are set so that the game can handle it correctly in case the game ends (player quits out of the minigame, etc) before the minigame ends. This must be used after minigames, except in the case of Marcus' Adventure (which uses different flags.) | |

| **Delay Screen Transition** | `>delaytrans boolean` |
|---|---|
| This prevents screen transitions from occurring. This can be used on a dialogue that starts as soon as the player enters the screen to keep the game world dark and focus only on the dialogue. | |

| **Play Music** | `>music ost_internal_name [vol]` |
|---|---|
| This starts playing the specified music. The `ost_internal_name` would be the name of a song that's included in the game's files (so you can't use this function to play your own music.) The volume is a value between `0` and `1` to indicate how loud it should be, with `0.5` being half volume. You may also put `-1` to play no music. If the specified music is not found, it will also stop playing music. The volume parameter is optional, and will default to `1`. | |

| **Message Box Style** | `>style type` |
|---|---|
| This sets the type of message box used for the dialogue. Often times this is set internally and doesn't need to be changed. However, in certain circumstances it can be necessary to change the style during dialogue. The following is a list of acceptable parameters:<br><br>`Default, Sign, Willow, Dark, Treasure` | |

# Misc. Files

These are files where there's not much to talk about, so don't really need their own section.

# 3D Files

These files are located in the installation directory, inside the folder `/three_d/`. It is **STRONGLY** recommended that you **DO NOT** modify any of these files, as doing so can cause the game to not be able to be started up!

For some of these files, they are loaded as OBJ files at the start, and then the resulting Vertex Buffer and Vertex Format are stored back alongside the OBJ file for faster loading. If you delete the Vertex Buffer and Format files, they will be regenerated when the game starts up next time. However, unless extended debug tools are enabled, the cached files will instead be stored in the game's save directory.

For folders such as room_river_challenge, these contain pre-generated Vertex Buffers for the River Challenge stage. **DO NOT** modify these files, as it can cause the game to become unplayable! If you delete any of these files, the game *should* regenerate them on the next run, but this is not guaranteed.

# Video INIs

These files, located in the installation directory at `/videoini/`, contain all the presets for the game's video settings when not set to custom. Please do not modify or remove these files!