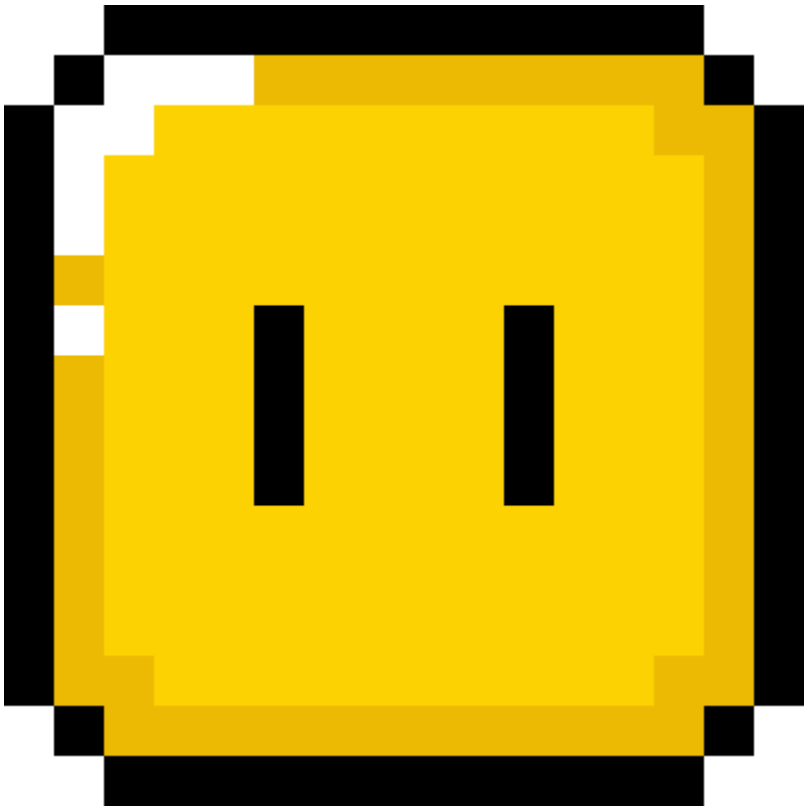


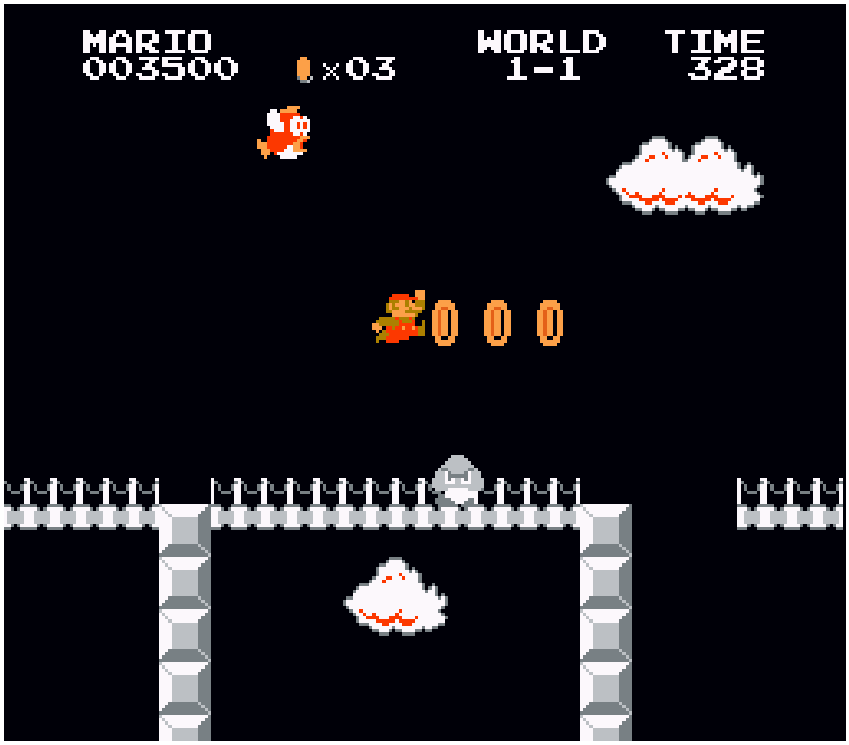
Level-Headed

“Water Vapor” Build

Created by Coolcord

Built on 12/07/14

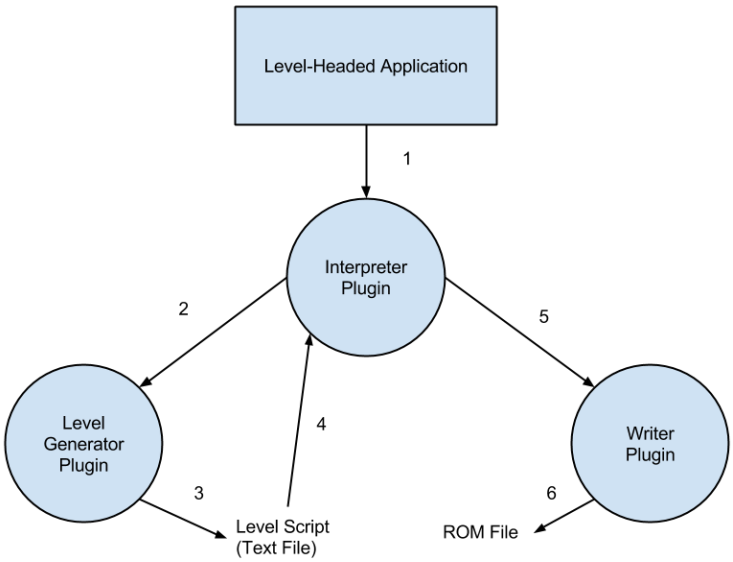
Table of Contents



What is the Goal of Level-Headed?

My ultimate goal with Level-Headed is for it to be a random level generator that will work across a handful of 2D platforming games. Some of the games that I have in mind are Super Mario Bros. (NES), Super Mario Bros. 2 (or Lost Levels on NES), Super Mario Bros. 3 (NES), Super Mario World (SNES), Super Mario Bros. X (PC), etc. The main focus will be Mario games, but I intend to add support for other titles in the future. Obviously, this is a very ambitious goal, but I am well aware of what is required to make this piece of software a reality.

In order to accomplish this, Level-Headed is broken up into 4 key components. First is the base application, which is essentially just a minimal interface and a plugin loader. The other 3 components are plugins. Generator plugins create the random levels and output them as a universal readable format of some kind. Interpreter plugins read this format and work with the associated writer plugin to compile the level script. Finally, writer plugins handle this binary data and hack / create the game accordingly.

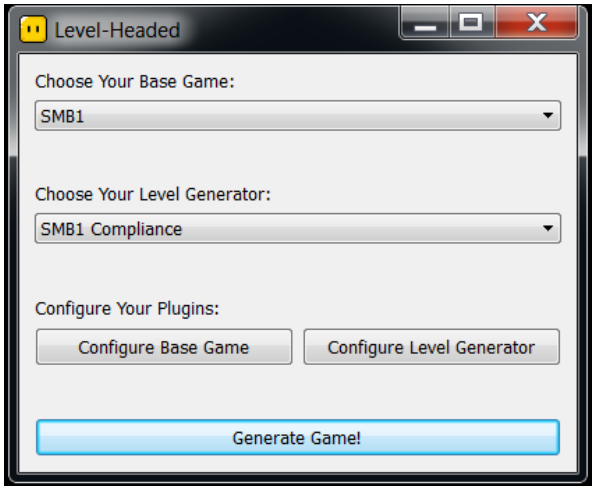


What is Level-Headed in its Current Form?

As of now, my focus is only on Super Mario Bros. (SMB1) for NES. As of this build, Level-Headed has two uses. The first use is intended for gamers. Level-Headed can generate a new Mario game based off of SMB1 in just a few clicks. These games are hacked NES ROMs, and they will work on real NES hardware or an NES emulator. The second use is aimed at scripters. Using the “SMB1 Compliance” language, scripters can writer levels for SMB1 in readable text and use Level-Headed as a compiler to create their game.

Gamer's Tutorial

Level-Headed was designed to be very simple and straightforward to use. The interface is very minimal to keep things easy.



To create a new game, select "SMB1" as the base game and "SMB1 Compliance" as the level generator. Notice that these are the only options as of now, since SMB1 is currently the focus of the project. From here, all you are required to do is click "Generate Game!" to create a new ROM.

The first time running Level-Headed, a base ROM will need to be provided. The 5 following ROMs are supported as of now:

- Super Mario Bros. (JU) (PRG0) [!].nes
- Super Mario Bros. (JU) (PRG1).nes
- Super Mario Bros. (J).fds
- Super Mario Bros. + Duck Hunt (U).nes
- Super Mario Bros. + Duck Hunt + World Class Track Meet (U).nes

Once the ROM is installed, you can remove the ROM you extracted. After clicking generate, Level-Headed will ask you where to save the game, so save it wherever you'd like.

Now you can use this ROM on a real NES or on an emulator of your choosing. Have fun!

Gamer's FAQ

Q: As a gamer, what am I looking for when testing Level-Headed?

A: First off, Level-Headed should not crash. Secondly, all levels created by Level-Headed should be completable. If either of these things is not the case, please contact me, as it may be a bug. Make sure to not delete the level scripts created by Level-Headed, as these will be very useful for me if you find a bug. Here's a list of things to keep in mind:

These are bugs:

- Objects or holes that are too high or long to jump over
- After dying, you should respawn safely
- Enemies should not spawn inside of objects
- Objects should not overlap (unless they did in SMB1)

These are NOT bugs:

- Difficulty is unbalanced
- Pieces of the background / scenery are cut off from an object
- Levels are too short / too long
- Enemies walk into one block width holes and "spaz out"

Q: What types of levels can Level-Headed create?

A: Level-Headed can make Standard Overworld, Underground, Underwater, Bridge, Island, and Castle levels. Every world ends with a Castle level.

Q: Why are levels sometimes short / long?

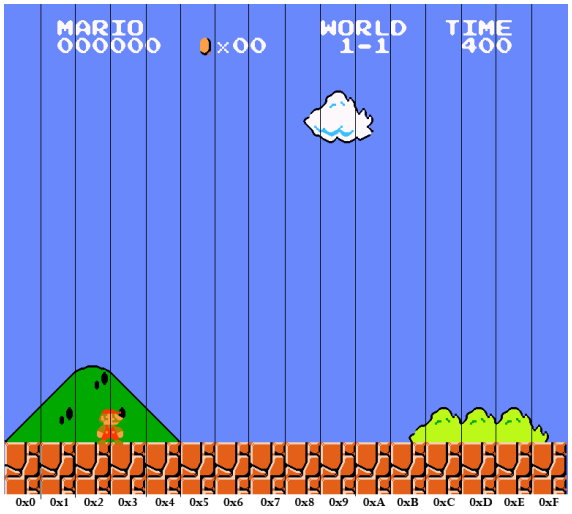
A: Level-Headed is hacking a binary file, so it's important to note that there is only so much space allocated in the ROM. Each level has a certain number of objects that Level-Headed can work with, and this cannot be easily changed. I have a plan that will allow Level-Headed to take objects from one level to another, but it hasn't been implemented in this build. As of now, I'd recommend creating games with **world size 4 or under** for a more balanced game in size.

Q: Levels sometimes feel redundant!

A: Level-Headed uses randomized patterns to create random levels. As of this build, there's only about 5 patterns per level type, and some level types share patterns. More patterns will be implemented in later builds to fix this and add significantly more replayability to the game. If you have any level design ideas, let me know, and I'll add them as a pattern in a later build.

Scripter's Tutorial

Using "SMB1 Compliance" scripts, scripters can make their own levels for SMB1, using Level-Headed as a compiler. There are two different types of scripts that this version of Level-Headed makes use of. The first and main scripts are the ".lvl" scripts. These will be compatible across multiple games in the future. A single ".lvl" script is simply a level for Level-Headed. The second type of scripts are the ".map" scripts. These scripts will be specific to each game. ".map" scripts are used to tell Level-Headed where to put the levels created from the ".lvl" files into the new game. It should be noted that using the "Clear All Random Level Scripts" in Level-Headed will **ONLY** delete levels that start with the name "Random ", so custom level scripts won't be deleted during this process.



Creating Levels With “SMB1 Compliance .lvl” Scripts

To start off, here is a sample “SMB1 Compliance .lvl” script. Notice that each section is separated by a series of “===”. Every “SMB1 Compliance .lvl” script starts with the line “SMB1_Compliance_Level”, followed by 4 sections.

```
SMB1_Compliance_Level
==========================================================================
This block is the notes section. Feel free to put whatever comments you want about the level here.
SMB1_Compliance_Level format created by Coolcord.
Randomly Generated on Wednesday, November 19, 2014, at 02:08:51 PM.
==========================================================================
# Comments can be to any section like so. In-line comments are not supported.
# This block is the header section. The order is important here, so don’t mix it up!
Attribute: Overworld
Brick_Pattern: Surface
Background: Blank_Background
Scenery: Fences
Compliment: Trees
Time: 400
Midpoint: 12
==========================================================================
# This block is the object section. Place any objects you want to be in the level here.
Question_Block_With_Mushroom 20 7
Vertical_Blocks 11 8 3
Horizontal_Blocks 1 8 9
Vertical_Blocks 9 8 3
Horizontal_Blocks 1 10 9
Vertical_Blocks 4 9 2
Steps 8 6
Pipe 7 7 4
Pipe 7 9 2
Horizontal_Coins 8 8 9
Vertical_Blocks 16 8 3
End_Steps 4
Flagpole 17
Castle 4
==========================================================================
# This block is the enemy section. The format is very similar to the objects section.
Goomba_Group 21 10 3 Normal
Green_Koopa 11 10 Moving Normal
# Notice how each enemy is followed by a difficulty
Green_Koopa 11 9 Moving Hard
Red_Koopa 11 10 Normal
Goomba 11 10 Normal
==========================================================================
```

“SMB1_Compliance .lvl” Header Section

Attribute: Overworld

Attributes are the most important piece of the header, as they control a large amount about the level, including music, palette, types of objects and enemies that can spawn, etc.

Possible Attributes: Overworld, Underground, Underwater, Castle

Brick_Pattern: Surface

The brick pattern setting controls which brick pattern the level will start with. To explain brick patterns in short, the SMB1 engine will infinitely generate “bricks” or blocks in a certain pattern as the player continues to move to the right. In the case of “Surface”, this will create the ground that the player walks on.

Possible Brick_Patterns: No_Bricks, Surface, Surface_And_Ceiling, Surface_And_Ceiling_3, Surface_And_Ceiling_4, Surface_And_Ceiling_8, Surface_4_And_Ceiling, Surface_4_And_Ceiling_3, Surface_4_And_Ceiling_4, Surface_5_And_Ceiling, Ceiling, Surface_5_And_Ceiling_4, Surface_8_And_Ceiling, Surface_And_Ceiling_And_Middle_5, Surface_And_Ceiling_And_Middle_4, All

Background: Blank_Background

The background setting simply determines what the background is. As far as the SMB1 engine is concerned, this can affect the time of day as well as the palette.

Possible Backgrounds: Blank_Background, In_Water, Castle_Wall, Over_Water, Night, Snow, Night_And_Snow, Night_And_Freeze

Scenery: Fences

Sceneries help compliment the background. As far as the SMB1 engine is concerned, normally only levels with the Attribute setting of “Overworld” use a scenery. It’s possible to use sceneries on levels with other attributes, but it may look rather silly.

Possible Sceneries: No_Scenery, Only_Clouds, Mountains, Fences

Compliment: Trees

Compliments allow for certain items to be used in a level. For example, bullet bill turrets require the “Bullet_Bill_Turrets” compliment. Islands require either the “Trees” or “Mushrooms” compliment. As far as the SMB1 engine is concerned, the “Clouds” compliment will make the level look like a beanstalk bonus area.

Possible Compliments: Trees, Mushrooms, Bullet_Bill_Turrets, Clouds

Time: 400

Time simply sets the time limit in a level. As far as the SMB1 engine is concerned, it is important to note that these are not in units of seconds. A value of 0 should only be set to bonus areas, as setting it to 0 on a normal level will be an instant kill.

Possible Times: 0, 200, 300, 400

Midpoint: 12

Midpoints determine at what page the player will spawn when they die. Keep in mind that 1 page = 16 blocks. It is also important to note that the player will spawn between 3 and 4 blocks into this page at y coordinate 10 (ground level).

Possible Midpoints: Any value from 0 to 15.

“SMB1_Compliance .lvl” Object Section

The following are all possible objects that can be written in the object section of a level.

Key:

x: This is a relative coordinate from the previously spawned object. Any value from 0 to 16 is considered “safe.” It’s technically valid for the x to be as high as 31, but the scripter must keep the absolute coordinates of each object in mind when using such a distance. In other words, each page (every 16 blocks) must have at least one object. Keep in mind that this coordinate is calculated separately from the enemies x coordinates.

y: A low value of 0 is at the top of the screen. Most objects cannot be spawned lower than y coordinate 11.

length: The size of the object, usually ranging from 1 to 16 unless otherwise specified.

height: The height of the object, usually ranging from 1 to 16 unless otherwise specified.

Question_Block_With_Mushroom x y

Question_Block_With_Coin x y

Hidden_Block_With_Coin x y

Hidden_Block_With_1up x y

Note: In the SMB1 engine, if the level does not have the “Overworld” attribute, this object will behave exactly like the “Brick_With_1up”.

Brick_With_Mushroom x y

Brick_With_Vine x y

Note: For the vine to work properly, a pipe pointer will be necessary. Review the “Pipe_Pointer” enemy for more information.

Brick_With_Star x y

Brick_With_10_Coins x y

Note: In the SMB1 engine, placing multiple 10 blocks within about 2 pages of distance will cause the timers on all 10 coin blocks in the area to be shared. This means that hitting a 10 coin block will cause the other 10 coin blocks in a short distance to be activated as well.

Brick_With_1up x y

Underwater_Sideways_Pipe x y

Note: These pipes are always enterable, meaning that a pipe pointer will be necessary. Review the “Pipe_Pointer” enemy for more information.

Used_Block x y

Trampoline x y

Bullet_Bill_Turret x y length
Note: Bullet Bill turrets can only be spawned if the level compliment is set to “Bullet_Bill_Turrets”.

Island x y length
Note: Islands will not work if the level compliment is set to “Bullet_Bill_Turrets”.

Horizontal_Bricks x y length
Note: In order for any of these to contain items, a brick with some kind of item must be spawned on top of the horizontal brick at the desired x coordinate.

Horizontal_Blocks x y length

Horizontal_Coins x y length

Vertical_Bricks x y length

Vertical_Blocks x y length

Corral x y height
Note: In the SMB1 engine, these only work if the level has the “Underwater” attribute.

Pipe x y height
Note: This object must have a height that ranges from 2 to 8.

Enterable_Pipe x y height
Note: This object must have a height that ranges from 2 to 8. For these to function properly, a pipe pointer will be necessary. Review the “Pipe_Pointer” enemy for more information.

Hole x length

Hole_With_Water x length

Bridge x y length
Note: In the SMB1 engine, the y values of these are hard-coded. Consequently, they can only be spawned at the y coordinates 7, 8, and 10. This object cannot be spawned at absolute x coordinate 15.

Horizontal_Question_Blocks_With_Coins x y length
Note: In the SMB1 engine, the y values of these are hard-coded. Consequently, they can only be spawned at the y coordinates 3 and 7. In order for any of these to contain items, a question block with some kind of item must be spawned on top of the question blocks at the desired x coordinate.

Page_Change page
Note: This will jump to another page, allowing objects to be spawned at large distances away from each other. While it is possible to use this to go backwards, I would advise against it. The page value can be anywhere between 0 and 63. Keep in mind that 1 page = 16 blocks.

Reverse_L_Pipe x
Note: These pipes behave differently from normal pipes. For these to function properly, a pipe pointer will be necessary. Review the "Pipe_Pointer" enemy for more information. In vanilla SMB1, these are only seen in the "cutscenes" where Mario is walking to a pipe between levels. This object cannot be spawned at absolute x coordinate 15.

Flagpole x
Note: These should be followed by either a "Castle" or "Big_Castle", otherwise in the SMB1 engine, the player will lose control of Mario and Mario will infinitely walk forward. This object cannot be spawned at absolute x coordinate 15.

Castle x

Big_Castle x
Note: In the SMB1 engine, the background should be changed to "Castle_Wall" 3 blocks before if this is used at the end of the level. If this castle is at the beginning of the level, set the background to "Castle_Wall" in the header and change the background to something else 6 blocks after the "Big_Castle". Review the "Change_Background" object to change the level's background.

Axe x
Note: The y coordinate of axes is hard-coded to 6. This cannot be changed. This object cannot be spawned at absolute x coordinate 15.

Axe_Rope x
Note: The y coordinate of axe ropes is hard-coded to 7. This cannot be changed. This object cannot be spawned at absolute x coordinate 15.

Bowser_Bridge x
Note: The y coordinate of Bowser bridges is hard-coded to 8. This cannot be changed. Bowser bridges also have a hard-coded length of 13, which also cannot be changed. This object cannot be spawned at absolute x coordinate 15. In order for Bowser bridges to function like they did in vanilla SMB1 (acting as an ending to a Castle level), they will need to be placed in a very specific manner. To make this work as an ending, spawn the "Bowser_Bridge" at the absolute coordinate 0 on an EVEN page (this is important!). Keep in mind that the level starts at absolute coordinate 0 on page 0, 16 blocks into a level is absolute coordinate 0 on page 1, 32 blocks into a level is absolute coordinate 0 on page 2, etc. An "Axe_Rope" will need to be spawned exactly 12 blocks away from the Bowser bridge. An "Axe" will need to be exactly 1 block after the axe rope. Finally, for the ending to work properly, a "Scroll_Stop" will need to be exactly 8 blocks after the axe.

Scroll_Stop x
Note: These objects prevent the screen from scrolling to the current page. This object cannot be spawned at absolute x coordinate 15.

Scroll_Stop_Warp_Zone x
Note: This is a special version of the scroll stop used to trigger the warp zone area. Keep in mind the screen will not move beyond the current page after placing this object. This object cannot be spawned at absolute x coordinate 15.

Flying_Cheep_Cheep_Spawner x
Note: This spawner will cause cheep-cheeps to leap from the bottom of the screen. This object cannot be spawned at absolute x coordinate 15.

Swimming_Cheep_Cheep_Spawner x
Note: This object only works if the level has the “Underwater” attribute. This object cannot be spawned at absolute x coordinate 15.

Bullet_Bill_Spawner x
Note: This object will not work if the level has the “Underwater” attribute. This object cannot be spawned at absolute x coordinate 15.

Cancel_Spawner x
Note: This will stop any of the spawner objects from spawning anything else. This object cannot be spawned at absolute x coordinate 15.

Change_Brick_And_Scenery x Brick_Pattern Scenery
Note: Review the header section for more information about possible brick patterns and sceneries.

Change_Background x Background
Note: Review the header section for more information about possible backgrounds.

Lift_Rope x
Note: These objects will create a vertical rope the spans from the bottom to the top of the screen. These are normally used with the “Lift_Spawner”. Review the “Lift_Spawner” enemy for more information.

Balance_Lift_Vertical_Rope x length
Note: Use this with the “Balance_Lift_Horizontal_Rope”. Review the “Balance_Lift_Horizontal_Rope” for more information.

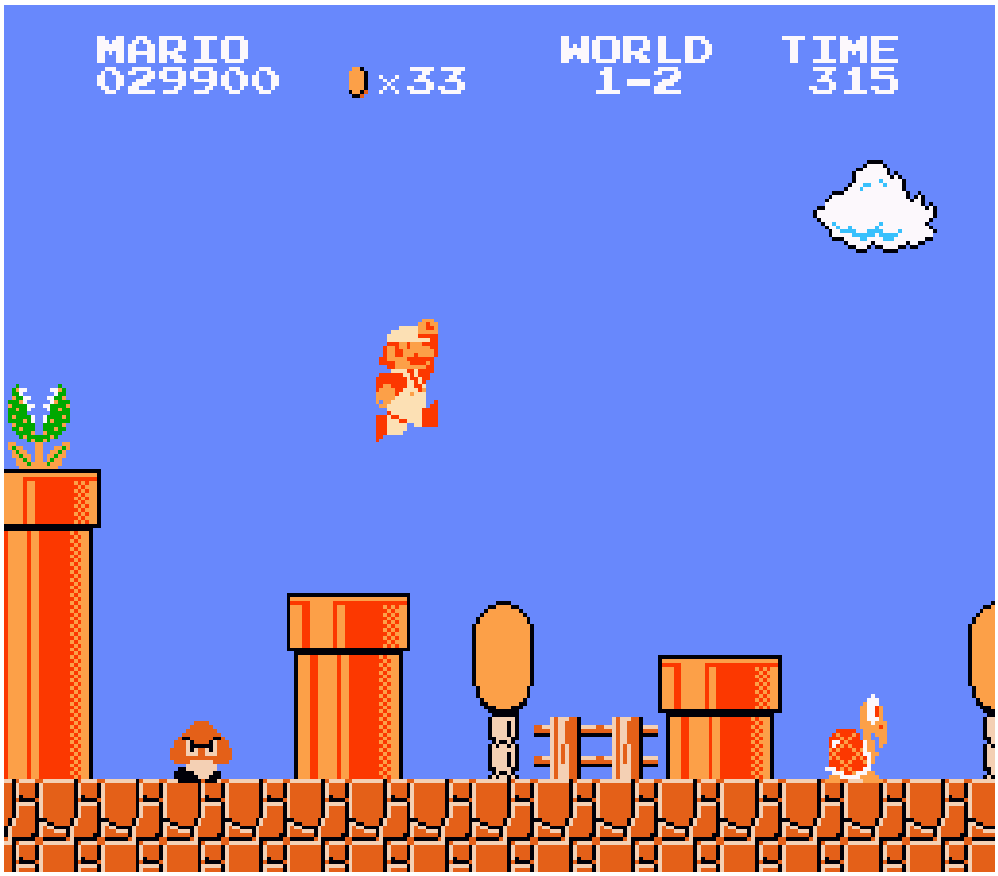
Balance_Lift_Horizontal_Rope x length
Note: The length determines the amount of space between the two balance lifts. The length cannot be less than 3. For balance lifts to work properly, a “Balance_Lift_Vertical_Rope” should be placed on at least one end of the horizontal rope. Next, two “Balance_Lift” enemies should be placed one block before each end of the horizontal rope. Keep in mind that the “Balance_Lift” enemies will need to also be one block below the vertical rope. To find a proper y coordinate for the balance lift, simply take the length of the vertical rope and add 1. This means that if the length of the vertical rope is 1, the balance lift should be placed at y coordinate 2.

Steps x length
Note: This object can only have a length from 1 to 8.

End_Steps x
Note: End Steps have a length of 9 and a height of 8. This is hard-coded and cannot be changed.

Tall_Reverse_L_Pipe x y
Note: This object's y coordinate is limited to a range from 1 to 10. For these to function properly, a pipe pointer will be necessary. Review the "Pipe_Pointer" enemy for more information.
Pipe_Wall x
Note: This object will create a wall from y coordinate 0 all the way into the ground. It is possible to jump over this wall if the player gets high enough.

Nothing x
Note: This is simply a blank object. It can be used to extend the distance between objects if desired, but I'd recommend using a "Page_Change" for large distances. Review the "Page_Change" object for more information.



“SMB1_Compliance .lvl” Enemy Section

The following are all possible enemies that can be written in the enemy section of a level.

Key:

x: This is a relative coordinate from the previously spawned enemy. Any value from 0 to 16 is considered “safe.” It’s technically valid for the x to be as high as 31, but the scripter must keep the absolute coordinates of each object in mind when using such a distance. In other words, each page (every 16 blocks) must have at least one object. Keep in mind that this coordinate is calculated separately from the objects x coordinates. Also note that the first enemy should not be on the first page (so start with a value from 16 to 31).

y: A low value of 0 is at the top of the screen. Most objects cannot be spawned lower than y coordinate 11.

difficulty: This can be set to either “Normal” or “Hard”. On normal, the enemy always spawns. On hard, the enemy only spawns when hard mode is activated. Hard mode is active either after level 5-3 or on a second playthrough of the game.

Green_Koopa x y movement difficulty
Note: Green Koopas have 2 different types of movement. They can be “Moving”, which is their normal behavior, or they can be “Stopped”, which will cause them to stand in place.

Red_Koopa x y difficulty

Buzzy_Beetle x y difficulty

Hammer_Bro x y difficulty

Goomba x y difficulty

Blooper x y difficulty

Bullet_Bill x y difficulty
Note: This is just a single Bullet Bill – not a spawner or a turret. Review either the “Bullet_Bill_Turret” object or the “Bullet_Bill_Spawner” if this is not what you want.

Green_Paratroopa x y movement difficulty
Note: Green Paratroopas have 3 different movement styles. They can be “Leaping”, which is their most common behavior. They can be “Flying”, which involves them flying back and forth from left to right. Finally, they can be “Stopped”, or just hovering in place.

Red_Paratroopa x y difficulty
Note: Red Paratroopas fly up and down. It is recommended to spawn them from y coordinate 0 to 4. Spawning Red Paratroopas lower than 4 will cause them to move slower when going down.

Green_Cheep_Cheep x y difficulty
Note: This is just a single cheep-cheep – not a spawner. Review the "Swimming_Cheep_Cheep_Spawner" object or enemy if this is not what you want.

Red_Cheep_Cheep x y difficulty
Note: This is just a single cheep-cheep – not a spawner. Review the "Swimming_Cheep_Cheep_Spawner" object or enemy if this is not what you want.

Podoboo x difficulty

Pirana_Plant x y difficulty
Note: Pirana plants do not need to be placed in pipes on the SMB1 engine. In the SMB1 engine, the first level has no pirana plants. After that, pirana plants will automatically spawn in pipes. This enemy can be used to spawn a pirana plant outside of a pipe.

Lakitu x y difficulty
Note: Lakitus will stay at whatever y coordinate they are placed at. The SMB1 engine normally places them at y coordinate 1, but any valid y coordinate will work.

Spiny x y difficulty

Bowser_Fire_Spawner x
Note: This enemy obeys the "Cancel_Spawner" object.

Swimming_Cheep_Cheep_Spawner x
Note: This enemy can only be used in levels with the "Underwater" attribute. This behaves in a very similar manner to the "Swimming_Cheep_Cheep_Spawner" object. It also obeys the "Cancel_Spawner" object.

Bullet_Bill_Spawner x
Note: This enemy cannot be used in levels with the "Underwater" attribute. This behaves in a very similar manner to the "Bullet_Bill_Spawner" object. It also obeys the "Cancel_Spawner" object.

Fire_Bar x y direction speed
Note: A fire bar's direction can either be "Clockwise" or "Counter_Clockwise". A fire bar's speed can be either "Fast" or "Slow".

Large_Fire_Bar
Note: Large fire bars always spin "Clockwise" and have a "Slow" speed.

Lift x y movement
Note: Lift movment can be either "Vertical" or "Horizontal".

Falling_Lift x y

Balance_Lift x y
Note: Refer to the "Balance_Lift_Horizontal_Rope" object for more information on how to set up this enemy.

Surfing_Lift x y

Lift_Spawner x y movement size
Note: A lift spawner's movement can be either "Vertical" or "Horizontal". A lift spawner's size can be either "Large" or "Small".

Bowser x
Note: Bowser is normally spawned 7 blocks away from the "Bowser_Bridge" object spawn. Bowser does not have to be spawned on a Bowser bridge, but keep in mind that his y coordinate is hard-coded to 8. Bowser also acts as a "Cancel_Spawner", so a "Cancel_Spawner" object is not necessary to stop spawners. For more information on the Castle level end sequence, review the "Bowser_Bridge" object.

Warp_Zone x
Note: This essentially handles all pipe pointers for the pipes within the warp zone. Make sure to use this with a "Scroll_Stop_Warp_Zone" object.

Pipe_Pointer x room page
Note: Pipe pointers take effect once they are on a page. After they are loaded, any object that makes use of them will take the player to the specified page in the specified room slot. The page value can be anywhere between 0 and 63. Keep in mind that the player will spawn between absolute x coordinates 3 and 4 on the specified page. Pipe pointers are the only enemy that take up 3 bytes instead of the usual 2. In other words, a pipe pointer is 1.5 enemies in physical size. For more information on the possible room slots, review "SMB1_Compliance .map" scripts. It is also important to note that pipe pointers do not work with page flags, which means that they must be spawned on the current page, so keep your x value 0 unless you are keeping track of the absolute x value.

Toad x
Note: Toads are normally placed after an axe in a Castle level. If it is the last world, Toad will be the Princess instead.

Goomba_Group x y amount
Note: Enemy groups are hard-coded to spawn at y coordinates 6 or 10. "amount" determines the number of enemies that will spawn in the group. This value can be either 2 or 3.

Koopa_Group x y amount
Note: Enemy groups are hard-coded to spawn at y coordinates 6 or 10. "amount" determines the number of enemies that will spawn in the group. This value can be either 2 or 3.

Page_Change page
Note: This will jump to another page, allowing enemies to be spawned at large distances away from each other. While it is possible to use this to go backwards, I would advise against it. The page value can be anywhere between 0 and 63. Keep in mind that 1 page = 16 blocks.

Nothing x
Note: This is simply a blank enemy. It can be used to extend the distance between objects if desired, but I'd recommend using a "Page_Change" for large distances. Review the "Page_Change" enemy for more information.

Mapping Levels to a Level Slot With “SMB1 Compliance .map” Scripts

To start off, here is a sample “SMB1 Compliance .map” script. Notice that each section is separated by a series of “===”. Every “SMB1 Compliance .map” script starts with the line “SMB1_Compliance_Level_Map”, followed by 3 sections.

```
SMB1_Compliance_Level_Map
========================================================================
This block is the notes section. Feel free to put whatever comments you want about the level here.
SMB1_Compliance_Level format created by Coolcord.
Randomly Generated on Wednesday, November 19, 2014, at 02:08:51 PM.
========================================================================
# Comments can be to any section like so. In-line comments are not supported.
# This block is the base game settings section. The only thing kept here right now is the
# Number of Worlds in the ROM, which determines at which world to end the game.
# This can be any value in the range from 1 to 8
Number_of_Worlds: 2
========================================================================
# Comments can be to any section like so. In-line comments are not supported.
# This block contains the level map section. This determines the level order as well as
# where to put what level scripts into the ROM.
WORLD_6_LEVEL_3 "Level_1_1.lvl"
WORLD_2_LEVEL_1 "Level_1_2.lvl"
WORLD_4_LEVEL_3 "Level_1_3.lvl"
WORLD_8_LEVEL_3 "Level_1_4.lvl"
WORLD_3_LEVEL_4 "Level_2_1.lvl"
# Notice how the following level does not have a script after it.
# This will make use of whatever is in this level slot without overwriting
# what is there.
WORLD_6_LEVEL_1
# It’s also possible to have duplicate levels by listing the room name again like so.
# Consequently, the line below will repeat the level at script "Level_1_4.lvl".
WORLD_8_LEVEL_3
WORLD_8_LEVEL_4 "Level_2_3.lvl"
WORLD_5_LEVEL_2 "Level_2_4.lvl"
========================================================================
```

“SMB1_Compliance .map” files aren’t very complicated. First off, it’s important to note that the filename of the .map file must match the folder name. This folder must then be placed into Level-Headed’s Levels folder under the “SMB1” directory. Each set of levels should only have one .map file.

In the main section of the .map file, there is a series of room slots (also known as level slots) followed by a level script. The order is important, as it determines which order the player will play through the levels. Levels can be repeated by simply declaring the same level slot again as shown. It is important to note that the SMB1 engine can only support up to 36 levels, so keep that in mind when writing a .map file. Worlds don’t need to have any specific number of levels in them, but a world will end

after the player touches an axe. Finally, the element to follow the level slot is the filename of the level script. This can be named anything, but keep in mind that it must be in the same folder as the .map file.

Here is a list of possible room slots. Keep in mind that these names are based off of the vanilla SMB1 ROM, which had some duplicate levels. Consequently, the naming scheme may seemingly skip some levels. There are a total of 27 main unique slots with an additional 7 slots that are normally used for bonus rooms (totaling at 34 level slots in the game). Note that you will be limited by the maximum number of objects and enemies that each level can hold. Keep in mind that all objects and enemies take up the same amount of space (2 bytes), except for pipe pointers, which take up 1.5 enemies (3 bytes). These restrictions may change in the future.

Level Slot Name	Maximum Objects	Maximum Enemies
World_1_Level_1	49	14.5
World_1_Level_2	80	22
World_1_Level_3	41	14
World_1_Level_4	47	19
World_2_Level_1	49	21
World_2_Level_2	60	20.5
World_2_Level_3	65	10
World_2_Level_4	56	23
World_3_Level_1	57	24
World_3_Level_2	24	18
World_3_Level_3	48	18
World_3_Level_4	53	21
World_4_Level_1	40	6.5
World_4_Level_2	79	22.5
World_4_Level_3	50	18
World_4_Level_4	62	12
World_5_Level_1	30	17.5
World_5_Level_2	56	21
World_6_Level_1	56	4
World_6_Level_2	70	19
World_6_Level_3	49	17
World_7_Level_1	43	13.5
World_7_Level_4	68	10
World_8_Level_1	72	28.5
World_8_Level_2	59	22.5
World_8_Level_3	51	14
World_8_Level_4	55	28.5
Pipe_Intro	3	0
Underground_Bonus	69	22
Cloud_Bonus_1	9	4
Cloud_Bonus_2	20	4
Underwater_Bonus	23	8
Warp_Zone	30	0
Underwater_Castle	12	9.5

Scripter's FAQ

Q: As a scripter, what am I looking for when testing Level-Headed?

A: First off, Level-Headed should not crash. Secondly, the scripts you write should compile into the ROM properly. Levels created with "SMB1 Compliance" scripts should compile the same, regardless of what base ROM is being used. If this isn't the case, or if scripts are clearly not being compiled properly, please contact me, as it may be a bug.

Q: Why should I care about "SMB1 Compliance" scripts?

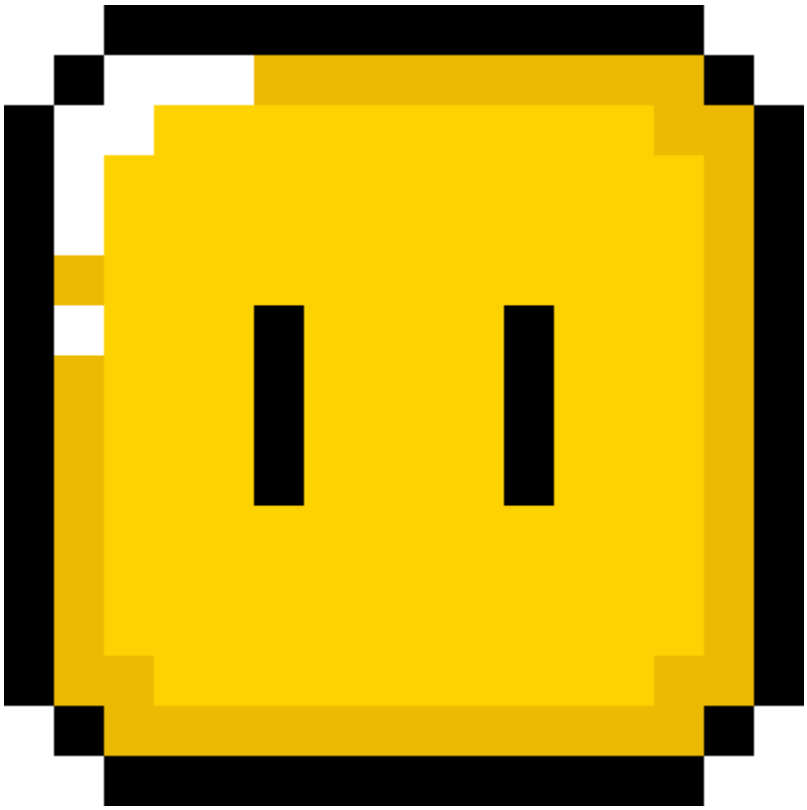
A: As of this build, "SMB1 Compliance" scripts only have one real advantage over other methods of distributing levels for SMB1. Levels created with "SMB1 Compliance" scripts will work across any of the 8 ROMs that Level-Headed supports. This is important to keep in mind, since other formats such as .ips and .ppf will only work with one ROM, meaning that the end user who downloads these patches must get ahold of the exact ROM that the hack creator used. This isn't an issue with Level-Headed, as users can use any of the supported ROMs with the same "SMB1 Compliance" scripts.

Q: What happens if I put two different levels in the same level slot?

A: Level-Headed will parse both levels and compile them, but only the last one will be in the level slot.

Q: What's up with the name "SMB1 Compliance"?

A: Basically, any 2D Mario game that came out after SMB1 can theoretically run levels from SMB1. The reason why I call it "SMB1 Compliance" is that any game that can theoretically run SMB1 levels will be compatible or "compliant" with "SMB1 Compliance" scripts. In the future, there will be other formats besides "SMB1 Compliance," but these will not work with SMB1.



Thanks again

for testing and supporting

Level-Headed!