# System Requirements

## Windows, Mac, Linux

|  |  |  |
| --- | --- | --- |
|  | Minimum | Recommended |
| Memory | 2 GB | 4 GB |
| Graphics | Intel 620 UHD or equivalent | Intel Iris Xe or equivalent |
| Storage | 200 MB free space | 200 MB free space |
| display resolution | 1280 x 720 px | 1920 x 1080 px |
| refresh rate\* | 60 Hz only | 60 Hz only |
| periphrals | Mouse or equivalent  Physical keyboard with letter and arrow keys | Mouse or equivalent  Mechanical keyboard with letter and arrow keys, and able to process 3+ simultaneous keystrokes |

\*While systems with other refresh rates can theoretically run the game, the game was designed solely for 60 Hz displays and unwanted effects may arise from other refresh rates.

## Browsers

### Supported

* Google Chrome
* Firefox
* Safari
* Microsoft Edge

### Untested (may not be functional for older versions)

* Opera
* Brave
* Vivaldi
* And others that are not explicitly mentioned

### Unsupported

* Internet Explorer

# Installation

1. Download the zip folder “LI\_SHAWN \_SDDAss3.zip”.
2. Locate the downloaded zip folder. If no directory was specified, this is most likely in the downloads folder.
3. Unzip the folder. Depending on your operating system, this may be right clicking and locating an “extract” or “unzip” option in the menu or double clicking on the folder. This should create a new, unzipped folder.
4. Within the new folder, locate the index.html file. Double click this to open this in your default browser. (See “Supported Browsers” for a list of browsers that can be used).
5. Alternatively, right click and select “open with” in the menu to select non-default browsers to run the program.
6. The default browser may also be changed in the system’s settings.
7. Enjoy the game.

# Gameplay Instructions

## Setting up the screen

Before starting the game, zoom in or out (“ctrl” + “=”, ”ctrl” + “-“) such that the game screen is of suitable size within the browser’s viewport. You may also choose to full screen the browser window (F11).

After finding a suitable screen size, refresh the page (“ctrl” + “R”) for the highest possible resolution.

## Menu

When opening the html file or pausing the game, a menu as such will appear.

To bring up this menu, press the escape key.

A screen shot of a game

Description automatically generated

### Start game/resume/demo complete

This serves as the pause button for the game. Clicking on the button or pressing the escape key will resume the game.

### Wireframes

This button enables and disables wireframes.

When wireframes is turned on, only the white outline of objects will be visible. When wireframes is turned off, all outline and fill colours will be visible.

The remainder of this guide will be illustrated assuming wireframes are turned off.

A black and white drawing of a rectangular object

Description automatically generatedA black rectangular frame with a black border and a yellow square with a black square with a red square with a black square with a yellow square with a black square with a black square with a white

Description automatically generated with medium confidence

## The box (i.e. the player)

You, the player, will be represented by this beautiful yellow box.

A yellow square in a grey room

Description automatically generated

Depending on the state of the box (see “Basic Movement”, “Death”), the box can change colours.

A yellow square on a gray background

Description automatically generated A square in a grey background

Description automatically generated ![A blue square on a gray background

Description automatically generated]() A group of squares on a grey background

Description automatically generated

The goal of the game is to utilise various movement options on the box such that it can reach the end of each level.

## basic movement

All of the following movement options assume that gravity pulls downwards.

### Left & right

By pressing the left arrow key on ground or in air, the box will move leftwards.

Similarly, pressing the right arrow key on ground or in air will result in the box moving rightwards.

### Jumping

When on solid ground (see “Obstacles), pressing the up arrow key will allow the box to jump upwards.

### fast fall

When in the air, holding the down arrow key will allow the box to fall downwards faster.

### Wall jumps

When in contact with a wall on the left or right of the box, a wall jump can be performed by holding both the up arrow key and the arrow key pointing away from the wall.

A screenshot of a computer

Description automatically generated

In the above example, a wall jump would be achieved by pressing both the up and left arrow keys.

## Gravity

This is the central mechanic of this game.

### Temporary gravity change

By pressing WASD keys after the box is in air for some time, gravity will change in the pressed direction for some time before reverting back to the previous direction. This achieves movement similar to a double jump or a dash in the specified direction.

When gravity is changed in the pressed direction, the box will be the stronger shade of blue:

![A blue square on a gray background

Description automatically generated]()

When gravity is reverted afterwards, the box will be the dimmer shade of blue:

A square in a grey background

Description automatically generated

The temporary gravity change is regained by landing on solid ground (see “Obstacles”), indicated by the dimmer shade of yellow:

A yellow square in a grey room

Description automatically generated

A temporary gravity change can be used after being in air for a certain period of time. When a temporary gravity change is available for use, the block will be the stronger shade of yellow:

A yellow square on a gray background

Description automatically generated

### Permanent gravity change

A permanent gravity change is executed by performing a temporary gravity change into a solid wall.

A yellow square with a red arrow pointing to a yellow square

Description automatically generated

More specifically, this is pressing the WASD that corresponds with the direction of the wall, and coming in contact with said solid wall. In the above example, pressing “A” and hitting the leftwards wall would enable a permanent gravity change toward the left.

After a permanent gravity change, the direction of gravity will be changed indefinitely (that is, until the next gravity change is performed). In this image, gravity is now pulling leftwards instead of downwards originally.

A yellow square on a gray background

Description automatically generated

After a permanent gravity change, the functions of individual arrow keys change slightly. In the above example, the right arrow key would still move the box leftwards on screen, but in the form of a jump off the left wall rather than just moving rightwards.

As a rule of thumb, the arrow keys will move the box in the direction they are pointing towards regardless of the direction of gravity.

## Death, respawn & level completion

### Death

When the box dies, it shatters into 50 small, colourful pieces.

A group of triangles with colorful squares flying out of them

Description automatically generated

Death can occur due to hitting spikes (see “Obstacles”) or being out of bounds, that is, the box not being visible on the screen.

### Respawn

Some time after the death occurs, the box will respawn at a spawn point, and gravity will be reverted to the default direction for that level.

Each level has a different spawn point, and the death and respawn will occur in the same level.

### Level completion

To complete a level, the box must traverse through a gap through the level’s walls.

A screenshot of a computer

Description automatically generated

When a level is completed, the box will not die from being out of bounds (see “Death”). Instead, the camera will shift towards the next level.

As of now, the demo has 3 levels, increasing in difficulty.

## Obstacles

### Walls

The solid black rectangles closing off the majority of the level are walls.



Walls that correspond to the direction of gravity (i.e. walls that the box would land on due to gravity pulling you downwards) are considered as solid ground.

### Platforms

The hollow, black rectangles floating in mid-air are platforms.

A black rectangle with a gray background

Description automatically generated

Platforms are solid ground.

### Spikes

The black, tiled triangles pointing in various directions are spikes.

A group of black triangles

Description automatically generated

Spikes kill the box upon contact.

### Falling platforms

The purple, solid rectangles are falling platforms.

A purple and black rectangle

Description automatically generated

As the name implies, falling platforms are not fixed as platforms are. They fall depending on gravity, which the player can change (see “Gravity”).

Falling platforms are not solid ground. The box cannot jump or regain its gravity change (see “Gravity) from standing on falling platforms.

### Falling spikes

The singular, orange triangles are falling spikes.

A red triangle on a grey background

Description automatically generated

As the name implies, falling spikes are not fixed as spikes are. They fall depending on gravity, which the player can change (see “Gravity”).

Falling spikes still kill the box upon contact.

## Advanced movement & other niche mechanics

### Buffering inputs

Most inputs in this game can be buffered, that is, the inputs will be repeated so long as the keys are held down. For example, continuously holding the up arrow key while gravity is pulling downwards will result in the box jumping continuously.

The main exception to buffering is inputs after permanent gravity changes. Upon a permanent gravity change, all inputs will be cleared.

### Respawn buffering & gravity change

Keystrokes can be buffered (see “Buffering Inputs”) through the death animation, allowing the box to immediately perform actions after respawning.

While the direction of gravity is changed from respawning, the state of gravity change is not. This allows for wavedashes (see “Wavedashing) or permanent gravity changes to be executed swiftly upon respawn.

### Wavedashing

While the maximum velocity that the box can achieve is usually capped, this can be bypassed at times.

For starters, the velocity cap in air is higher than the velocity cap on ground, that is, the box moving faster in air. The velocity cap during gravity change is then even higher than the velocity cap in air, allowing the box to move at its fastest in the first half of a gravity change.

This velocity cap can be bypassed even further with wavedashing. For18 frames after landing from reverted gravity, the velocity cap is not enforced, and acceleration is greatly increased. Jumping within this period will allow the box to gain considerable velocity. Note that this jump input cannot be buffered and must be pressed after landing.

### level transition gravity change

While the direction of gravity will not be changed while going through a level transition, the state of gravity change will be.

This allows for the box to regain its gravity change without touching solid ground upon entering a new level, and will prevent permanent gravity changes to occur across different levels.

### timer manipulation

Pausing the game will result in movement of the box and other obstacles pausing, but will not pause in game timers.

These timers include the timer for reverting gravity, the timer for tracking the amount of time in air before a gravity change can be inputted, and the timer for wavedashes (see “Wavedashing”).

### Wallbouncing

The platform, as implied by its hollow appearance, is composed of four different faces, up, down, left, right, each detecting collisions and gravity changes individually.

Walls, however, can only represent on direction at the time. For example, this may mean that colliding with the floor allows the box to jump, even if the box is underneath the floor, and has many other (im)practical applications.

The orientation of walls may be analysed better when wireframes are turned on.

### lag manipulation

Although the game is designed exclusively for 60 fps displays, certain advantages can be gained through increasing or decreasing the frame rate at which the game runs at.

For lower frame rates, the game will appear slower, allowing more precise movements to be made. However, various timers governed by in game ticks will also last longer, potentially producing unwanted effects such as temporary gravity changes lasting longer than expected. This can be mitigated to an extent when combined with timer manipulation (see “Timer Manipulation”)

For higher frame rates, the game will appear faster, and timers will last for shorter periods of time. As a whole, this does not grant much advantage other than for speedrunning less movement-intensive levels.