

AUTOSPHERE **Level Editor**

Manual

Beta 0.1

August 1, 2011

Contents

1	Introduction	3
2	Basics	4
2.1	Starting the Editor	4
2.2	Where are my custom levels stored?	4
2.3	Contribute to <i>Cubosphere</i> with your levels	4
3	The Editor Menu	5
3.1	Continue	5
3.2	Level Options	5
3.3	New Level	5
3.4	Load Level	5
3.5	Save Level	5
3.6	Overview	6
3.7	Test Level	6
3.8	Quit	6
4	Controls	7
4.1	Camera Movement	7
4.2	Cursor Movement	7
4.3	Creating Blocks	7
4.4	Deleting Blocks	7
4.5	Moving Blocks	7
4.6	Selecting Sides	7
4.7	Selecting Block Types	8
4.8	Applying Block/Side Types	8
4.9	Setting the Player's Starting Position	8
4.10	Spawning Items	8
4.11	Setting Side Properties	9
4.12	Inserting and Modifying Enemies	9

5	Block and Side Variables	10
5.1	Elevator	10
5.2	Exit	11
5.3	Laser	11
5.4	One Way Arrow	11
5.5	Blinking Block	11
5.6	Spikes	12
5.7	Switch	12
5.8	Teleport	12
6	Enemy Variables	14
6.1	Gear, Spiker and Jumper	14

1 Introduction

This is the manual for the internal *Level Editor* of *Cubosphere*. Using this feature allows you to create custom levels, play them and share them with friends. This manual will help you using the editor. But always keep in mind that *Cubosphere* is still under development. Therefore the editor interface could be not perfectly adjusted for an easy usage and your levels might be incompatible with following versions of *Cubosphere*.

2 Basics

2.1 Starting the Editor

Once you are in the main menu of *Cubosphere* you can choose the menu entry *Level Editor*. Afterwards you will have to choose if you want to create a new level or load an existing user level.

2.2 Where are my custom levels stored?

If you have created an user level, it will be stored after saving in the following directory:

Windows version: `<CUBOSPHERE-DIRECTORY>\user\default\levels`

Linux version: `~/.cubosphere/default/levels`

If you want to share your levels with your friends, you can copy levels from this directory into the same of another computer. All level files have the extension *.ldef*. Furthermore, each time you save a level, a JPEG-image is created, too. The picture is only needed for the preview picture in the load level menu and it not necessarily needs to be supplied with the *.ldef*. You might also find a file called *_selection.ldef* inside the aforementioned folder. It is just a temporary file, so you can ignore it.

2.3 Contribute to *Cubosphere* with your levels

When you have created some interesting levels, which are inventive and challenging (but still beatable in below 100 tries, please), you can send them (as *zip/rar/tar*-archive) to

cubosphere@web.de

We will take a look at your work and see, if we can insert your levels into the game's main episode. However, since we have already over 200 levels, it is not always manageable to include your levels; but simply give a try and send your work to us. We are pleased with all kind of user contribution.

3 The Editor Menu

By pressing *ESC* inside the editor you can show and hide the editor menu. In this menu you have several options which are explained in detail in this section.

3.1 Continue

Selecting this entry has the same effect as pressing *ESC* again. The editor menu will close and you can continue editing your level

3.2 Level Options

With this menu you can setup the theme and several other properties of your maze. You can change the initial time and the maximum time for the player to beat the level. Furthermore, you can set the appearance of the ball and the music track.

3.3 New Level

Will cause the editor to clear the current maze and start over with a new level.

3.4 Load Level

Choose an existing custom level to load.

3.5 Save Level

You can enter a name to save your level. Press *ESC* if you don not want to save. If the name you have entered already exists and differs from the filename of the current level, you will have to confirm that the saving process will overwrite the existing level. Press *ENTER* to do so, *ESC* to specify another filename.

3.6 Overview

The camera will fly around the level until you press *ESC*.

3.7 Test Level

Start your level in playing mode. Winning or loosing in this mode results in a restart of the level. To return to the editor press *ESC* and choose Quit.

3.8 Quit

Leave the editor. If you have changed the current level without saving yet, you will be asked, whether you want to save these changes or not.

4 Controls

4.1 Camera Movement

Pressing *up arrow* or *down arrow* will move the camera forward and backwards. Move it to the side by pressing *left arrow* or *right arrow*. Use your mouse to look around.

4.2 Cursor Movement

All editing processes are be done by the cursor. It is a box frame you can move through the level. Use the following keys to move it: *w*, *s*, *d*, *a*, *e*, *q*. Remember that the direction of these movements are static and will not rotate with the camera. So the cursor movement may be inverted or flipped if you look from another direction. Once you have blocks inside the level you can set the cursor on a block by targeting it with the cross-hair in the middle of the screen using your mouse and clicking the *left mouse button*.

4.3 Creating Blocks

If your cursor is at a free position a new block is spawned by pressing *SPACE*.

4.4 Deleting Blocks

If your cursor is on a block you can press *BACKSPACE* to remove this block.

4.5 Moving Blocks

If a block is at the cursor position, you can press *SPACE* to select and deselect the block. When your whole cursor appears red the block inside will move with your cursor when you will press *w*, *s*, *d*, *a*, *e* or *q*.

4.6 Selecting Sides

To select a single side of a block simply point at it with the cross-hair and click with the *left mouse button*. The cursor will highlight the selected side

red. If you want to select the parent block of the side you can press *SPACE*.

4.7 Selecting Block Types

Press *b* and a menu for block type selection will occur. Once you select a block type it will be shown in the lower left of your screen. Every new block you create will be of the current selected block type. Another way to change the current block type is done by pointing with the cross-hair on a existing block side and pressing *TAB*. The type of the side will be selected as current block type.

4.8 Applying Block/Side Types

If you want to change the type of an existing block, first select the block new type in a way described above. After that move the cursor over the block and press *ENTER* to apply the new type. If you have a side selected the current side will change otherwise the whole block. However there are block types which are only allowed to be applied to a whole block and not to a side (for instance invisible and breaking blocks). If you have a side selected and the old or the new block type is a block-only type, the whole block will be changed. There is a faster way to change side types: Simply *click right* on a side to apply the new type directly. This has the same effect as *clicking left* on the side and pressing *ENTER* afterwards.

4.9 Setting the Player's Starting Position

Each level needs a player start position. Without this, you are not allowed to save the level. To set a player start position point at a side and press *p*. An arrow will indicate the direction the player will be looking in when you start the level. By pressing *p* again the arrow is rotated.

4.10 Spawning Items

To spawn an item first select a side (by *left clicking* on it). Then press *i* to call the item selection menu. To remove the item simply choose *<none>* in

this menu.

4.11 Setting Side Properties

Depending on the side type you can change various properties of the side. To change them point at a side and press *v* to pop up this menu. A detailed explanation of the variables you can change is given in section 5.

4.12 Inserting and Modifying Enemies

Select a side and press the *o* key to open the enemy menu. When you first call this menu, you will first have to select an enemy to create. After selection your new enemy is spawned. However, you will often have to adjust the properties of the enemy. Therefore, simply press *o* on the side again. This time, another menu will pop up. You can change the variables of the enemy (which is explained in detail in section 6), rotate the enemy by 90 degrees and change or delete the enemy. The later is simply done by changing the enemy to *<none>*.

5 Block and Side Variables

In section 4.11 is explained, how to access the side/block-variables menu. In the following lines we give insight to the purpose of every single variable of the special blocks.

5.1 Elevator

The Elevator is the only block in game, which is able to move around. Therefore it has quite a lot of variables to configure the elevator. If you want to build an elevator platform consisting of more than one block, please make sure, that all blocks belonging to the same elevator unit have the same values in all variables. Furthermore, make sure, that between elevators and static blocks is always the distance of one block of free space.

Amplitude: The total distance between both turnaround points measured in block sizes

Speed: The speed of the elevator

Phase: The Phase determines, at which position the elevator is initialized when the level starts. 0 means central between both turnaround points, whereas 0.25 and 0.75 sets the initial position into one of the the turnaround points. 0.5 is also centralized, but starting in the opposite direction as if Phase is set to 0. However, you can specify any floating point number to adjust the starting point of the elevator even more freely.

Direction: If Direction is set to 0, the elevator will move left and right. Whereas 1 causes an up-down movement, 2 stands for a motion into the front-back direction. However, as you know the game, absolute directions are quite senseless in *Cubosphere*.

DelayTime: The Delay Time determines, how long the elevator stops in the turnaround points. You can also set it to any number below zero

to change the linear stop-motion movement of the elevator into a sinusoidal motion (like a harmonic oscillator).

5.2 Exit

Next Level: When you complete the level, the next level specified by the contents of this variable is loaded (no file extension needed). Please make sure, that the next level exists, because otherwise *Cubosphere* might crash. There is a special value *win* for the Next Level variable. By setting Next Level to *win* a score screen is showed instead of loading a next level.

5.3 Laser

When building lasers, you must ensure, that the laser hits another block. Although it is not necessary, that the block hit by the laser is also a laser tile, we recommend it for a proper design. The level editor changes simultaneously the variables on both sides connected by the laser.

Start Active: Determines whether the laser is activated upon starting the level or not.

Color: Just choose one of the colors. You can toggle the lasers by rolling over a switch of the same color.

5.4 One Way Arrow

Rotation: A number between 0 and 3 to rotate the arrow indicating the only allowed direction

5.5 Blinking Block

ActiveTime: How long the Blinking Block is visible (and solid)

DeactiveTime: And how long is it invisible (permeable)

BlendTime: Transition time between both phases. The block is solid during this time.

Phase: A floating point number between 0 and 1 defining the current state, when the level starts (cf. Phase in section 5.1)

5.6 Spikes

Phase: A floating point number between 0 and 1 defining the current state, when the level starts (cf. Phase in section 5.1)

Speed: Determines the speed and frequency of the spikes

5.7 Switch

StartActive: Is the button active, when the level starts. This is not important for the game mechanics. It just determines, if the button is glowing or not.

Color: Sets the color the switch is acting on

5.8 Teleport

StartActive: Is it possible, to use the teleporter when starting the level, or must we find a switch to activate the teleporter before.

Color: The color has two meaning for teleporters. On the one hand, it determines, which switches can be used to activate and deactivate the Teleporter. On the other hand, the color also constrains the possible target Teleporters, as they have to have the same color, too.

DestSide: When the ball rolls over an active Teleporter, *Cubosphere* have to find the destination of the teleport. This is done by the following procedure: A list of all active and same-colored Teleporters is created. The order of this list can be checked by selecting the compatible Teleporter sides in the editor and sorting it by the shown *side IDs*. Now,

the game has to choose one of these target Teleporters. This is, where the DestSide variable comes into play. Beginning with the list position of the starting Teleporter (indexed by DestSide= 0), DestSide states the offset of the target Teleporter in the list. Therefore, the next Teleporter is indexed by DestSide= 1 whereas the previous is accessed by DestSide= -1. You can also skip Teleporters in the list by setting DestSide to numbers of higher amount. However, in many cases it will be faster to try some values of DestSide, until the right target Teleporter is addressed, than following the aforementioned explanation.

DestRotation: A number between 0 and 3 representing the direction the ball is looking after the teleport.

6 Enemy Variables

In this section the variables of the enemies are explained. In section 4.12 is explained, how you can spawn enemies and how you get to the dialog to edit the variables.

6.1 Gear, Spiker and Jumper

All three enemies have the same four variables:

Movement: The Movement string determines, how the enemy moves around.

It simply consists of a sequence of concatenated letters, whereby each letter represents a movement step of the enemy. Following move command letters are allowed:

f – The enemy moves one step forward

l – Rotate left by 90 degrees

r – Rotate right by 90 degrees

j – Causes a normal forward jump (two blocks long)

u – A high jump without moving forward

d – The enemy will perform a long jump (three blocks long)

All letters in the Movement strings are sequentially executed. Whenever the Movement string is completely processed, the whole procedure starts over again beginning with the first letter in the string. To give an example: a simple Movement string reads

ffrjr

An enemy with this movement string will firstly move two block forward continued by a rotation to the right. It is followed by a normal jump and another right rotation. After that, the enemy will move two steps forward again, and so on. Note that this movement will take place on a square with an edge length of three blocks.

Move index: The Move index indicates the letter of the Movement string, which is first to be executed when the level starts. The first letter is indexed by 0, the second by 1 and so on. In the aforementioned example Movement string *ffrjr* a Move index of 2 e.g. causes, that the enemy starts with the rotation to the right, followed by a jump, another rotation to the right and two steps forward.

Speed: The Speed variable determines the velocity of the enemy. You can set it to any value between 0 and infinity, but we recommend not to exaggerate with the values of the Speed variable.

Interaction: Sometimes in *Cubosphere* it is necessary, to kill an enemy with a laser, while it is sometimes important, that an enemy is not vulnerable to lasers. In order to give the level designer the control over the interaction between enemy and hazards initiated by special blocks, we have introduced the Interaction string. Similar to the Movement string, the interaction with the environment is defined by the letters the Interaction string. You can concatenate the letters listed below. Hereby, the order of appearance is not important. When one of the following letters is not a part of the Interaction string, the enemy can pass the hazard without interacting.

b – The enemy cracks breaking blocks, when moving over them

p – Static and piercing spikes can do harm to the enemy

i – The enemy slides on icy surfaces

l – Lasers can kill the enemy

s – When moving over a switch, the enemy toggles it

t – The enemy is catapulted by trampolines

For example the Interaction string

pst

causes following behavior: The enemy can cross breaking blocks and

lasers. It is able to turn around on an icy surface. However, when colliding with a spike, the enemy is killed. Furthermore, it will use trampolines and is able to toggle switches.