Notation of building instructions for created GraviTrax® marble runs

Document version Aug 20, 2023

new notation for balconies and walls since version 1.10

new notation for unknown elements since version 1.11

Summary

GraviTrax® is a marble run building system from Ravensburger. To start with you need at least one starter set. There is an official app to register own runs. That app focuses on graphics effects and visualizing runs while the notation proposed here aims at a compact text format and can be used even without support from a program. However a program can be useful to check the recipes for errors, output human readable instructions and more. The descriptions of such marble runs can be noted in files. A program **gravi** has been written to store many such descriptions in a database. This allows to quickly determine the number of different elements for a run, register owned construction sets, visualize marble runs, output build instructions for runs, export stored runs and more.

Contents

Introduction

If you are reading this description, you probably already have a GraviTrax® kit, a marble run building system from Ravensburger, or at least heard of it. Designing a marble run is quick and easy by using the base planes, the hexagonal tiles, the rails and other elements from the sets. Soon you will be happy about every functioning track and later on you may wish to be able repeating the construction. With so many ways to build tracks out of the parts, it is almost impossible to remember the layout so that you can build the same track again later.

There is an app from Ravensburger that you can use to model such tracks and then save them. These tracks can then be sent to a central computer, from where your friends can then pick up the building instructions. But for that you need the app and you have to have been registered on that computer. There is another way, and if you want, even without an app, just with pencil and paper.

In the following you will be shown how you can write down the tracks you have built so that you can rebuild them later. Of course you can write long sentences, for example: "Take a base plane, place it so that the top left corner is a green hexagon. Then put a gray tile on the second top green field..." This will definitely take a long time and is not very practical. Therefore, it is suggested here how you can use as few characters as possible to write down how your marble runs were built.

For this purpose, the following chapters will gradually describe how the parts of GraviTrax® need to be noted to form recipes for rebuilding tracks. That's a lot of text to let you understand how everything is meant. Later, if you just want to remember how certain parts were named, you don't have to go through the whole text again. In the appendix, tables and pictures summarize the rules for writing down the building instructions.

On writing down your building instructions, it can always happen that you make mistakes. It is therefore very useful if you have checked them by a computer. There is a program for these checks, that tests whether you have written down all the parts correctly and whether the tiles and rails fit together. Then, from your written rules, a picture will be created of what the track looks like, if there were no errors. Otherwise the program reports where you made mistakes. You can compare with the picture to see if you wrote down the path correctly. The program just mentioned is called gravi, but as I said, it is not absolutely necessary.
Where you can get the program from and how you can install it is described at the end of this text. The program is written in Perl and runs on Windows, Linux and MacOSX computers. You can do a lot more with the program than has been said here, but more on that later. Now let's get started.

Base planes and positioning tiles

After all, as few characters as possible should be used to write down the recipes to build tracks. Therefore each part of GraviTrax® is noted with only one or two characters. In order to better remember the characters, they were chosen so that either a letter from the English name of the part is used or the character is visually reminiscent of the part. The underscore character _ was chosen for the base plane.
Since you can put many base planes together, you first indicate which of the planes is to be described. The first base plane is put on the top left and a green hexagon has to be in the top left corner. Since the plane forms the first row and the first column, you write

_ 1 1

If you place further planes to the right, you write **_ 1 2, _1 3** and so on. If you start another row under the first row, these planes are then called **_2 1**, **_ 2 2** and so on. The first number indicates the row, the second the column. You can also separate the numbers with commas, e.g. **_ 1,1** instead of **_ 1 1**.
Now that you can write down which base plane you are currently on, you also want to place parts on it. But how do you note where the parts go? Please look at a base plane (picture to the right). It has 6 columns with hexagonal holes. In the first, third and fifth columns there are 5 ways to insert a piece, in the others only four.

Like for the base planes, you number the positions. The first position at the top left is called 1.1, so you simply write **11**. Below that is position **21** and to the right of 11, slightly lower, you will find position **12**.
If you place two base planes one below the other, the half hexagons in rows 2, 4 and 6 become a complete hexagon on which there can also be parts. These positions are then called **52**, **54** and **56** in the upper base plane. If you want, you can also name the positions **02**, **04** and **06** in the lower base plane, but you must not omit the zero, since a position always consists of a row (here the zero) and a column.
Now you have already learned the most important part, the naming of positions on the base planes. If you place two base planes next to each other and want to write down the third field from the top on the second base plane in the fifth column, this is how it works:

```
      # An example
_ 1 2 # this is the 2nd base plane in the first row
35    # this is in the 5th column the 3rd field from the top
```

Here you can also see that you can add arbitrary text (comments) on a line that will help you understand the notation. Anything after the **#** character up to the end of the line is such a comment. Since there are typically only few characters on most lines, you could also separate the descriptions with a semicolon instead of starting a new line each time. However, this quickly becomes confusing and should therefore not be used.

Further down it is described how you can write the positions in a different way. Instead of numbering base planes, all rows and columns are then counted consecutively. This saves you a few lines in the description, but you have to be more careful when writing down the line and column numbers.

Base plane arrangements that do not form rectangles

Most of the time you will use a rectangular area for your tracks. But you can decide to build tracks where the ground plane is shaped differently, for example in the form of an S, T or U. In order to take this into account in the display, you must specify which planes are missing in the rectangle. Instead of the **_** you use **!** to say that this record is missing. For a T-shape made from a 3x3 rectangle, note the missing base planes like this:

```
!   2 1
!   3 1
!   2 3
!   3 3
```

Height tiles

Height tiles allow you to arrange other elements of GraviTrax® above the base planes. This can give the marbles speed and these tiles are normally the most frequent elements in a marble run. The starter set contains gray height tiles and half as high black height tiles. There is also a "vertical starter set" that contains other height tiles. These will only be described in a later section.

The large gray height tiles are assigned the character **1**, the black half-height tiles the **+** sign. Therefore it can also be said that a gray tile is one **height unit** high, a black one only a half. Height tiles can be stacked on top of each other. To note this, you simply write the chars one after the other, for example **+1+1**. This is a nice pattern of alternating black and gray tiles. For a track, however, it does not matter how the height is achieved. You could have taken only 3 gray tiles and no black ones, because **111** gives the same height. Since such tiles are used very often, you can add up the heights and write down the sum instead. So **+1+1** becomes a **3** if you don't value the beautiful pattern. But if you have a small height tile more, for example 111+, then you can also write **3+**.

But how are you supposed to write down a height of 12 tiles? 12 does not mean a height of 12 but 1 height tile and two more, that's only 3. You can write 12 ones in a row or digits that add up to 12, so for a height of 12 you write **93** or **66** or **9111**. You also have to write down where the height tiles should be placed. First comes the position (**35**) and then the height tiles (**+1+1** or **3**):

```
      # placing height tiles
_ 1 2 # the 2nd base plane in the first row
35 3  # 3 large height tiles in the 5th column, 3rd field from the top
```

Now that you know where and at what height you want to place tiles, it's time to design the tracks.

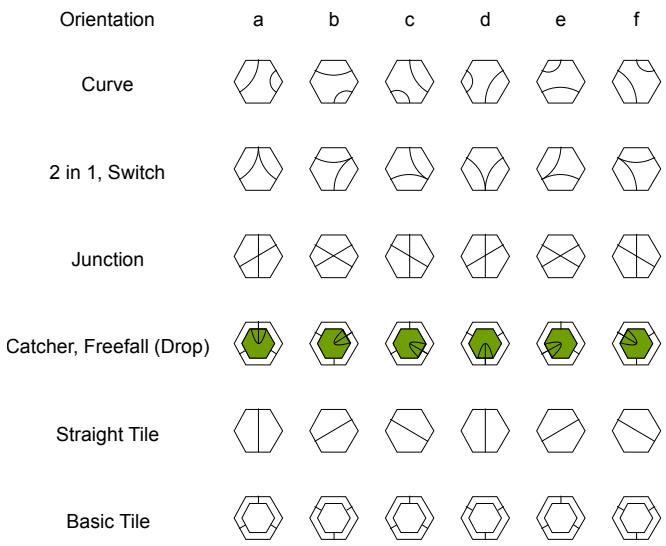
Basic elements (hexagonal bricks)

The starter set contains several hexagonal tiles and rails. Let's start with the tiles. A capital letter is assigned to each of the different tiles. Since the 26 letters are not enough in the long run, tiles (like the base tile) can also be named with a prepended x, y or z. The base tiles are assigned the character combination **xG**, but are not noted in tracks, since they only become a landing tile, free fall, catcher or splash with a green insert. You can see the assigned symbols in the following table:

Symbol	Tile	Comments
A	Launch Pad	The alphabet starts with A
Z	Landing	and finishes with Z
C	Curve	
X	Junction	The symbol of a junction
Y	2 in 1	Two tracks go into one, looks like an Y
W	3 in 1	Similar to Y, 3 tracks go into one
S	Switch	The same tile as 2 in one, but with a green switch
M	Magnetic Cannon	
V	Vortex	
D	Freefall	Freefall (Drop)
G	Catcher	C is already taken by “Curve”, G looks similar
P	Splash	S is already taken, the 2nd character is used instead
xG	Base tile	Not used in tracks

More tiles from extension sets will be discussed later. You can find a list of all currently known elements in the table in the appendix.

For most tiles it matters how they are oriented. Take a curve, for example. Depending on how you turn it, a marble takes a different path. In order to describe the correct marble path without discontinuations, you have to specify for each tile from the table above how it should be oriented. This is achieved with the letters **a**, **b**, **c**, **d**, **e** and **f**. You can see how the letters are assigned in the image to the right.



The left column shows tiles with the orientation **a,** which is chosen in such a way that (almost always) a marble can leave the tile at its upper edge. Where this is not the case (e.g. for the **freefall**), it is mentioned separately. The orientation of a tile can best be seen at the **catcher**, al this tile has only one outgoing path. With orientation **a**, the ball can only run upwards. If you turn the tile further clockwise, you get the orientation **b** etc. As for the freefall there is also only one path, that incoming path defines the direction (as an exception to the rule above).

You can also see that the tile 2 in 1 and the switch look the same apart from the green element. In the 2 in 1 tile, the ball leaves the tile where the two tracks come together. On the switch a ball can only get out of there if the seesaw is in the correct position, otherwise the seesaw will block the ball. Therefore, for most tracks that is the incoming direction to the switch. However, the orientation is the same as for the 2 in 1 tile.

With the **switch**, it can be important which position the seesaw has. You can indicate this in the description of the tile by placing a **+** or **-** between the **S** for switch and the letter for the orientation. Such additional information is also referred to as a **detail**. The **+** indicates that the seesaw has been rotated clockwise. If the tile orientation is a**,** the narrow end of the seesaw then points to the left. You then write, for example, **S+b** or **S-b** instead of just **Sb**.

For some tiles, some orientations have the same image. Orientation **a**, **c** and **e** look the same for a base tile. You can choose one of the orientations at will. However, orientation is important for some tiles that otherwise look symmetric. As an example the Gaussian shoots up only with orientation **a** and down with orientation **d**. With these rules you can now place elements (tiles) on the base plane or on height tiles. A curve on 3 height tiles in the example above then looks like this:

```
35 3Cf # three height tiles and one curve on top
```

Now tiles are to be arranged on a total of 4 base planes. Can you recreate that?

```
# An example with 4 base planes
_ 1 1 # top left
43 5Ac # Start on 5 gray height tiles
55 3Cf # Curve on 3 height tiles
_ 2 1 # bottom left
35 3Cf
_ 1 2 # top right (empty, you can also omit the line)
_ 2 2 # bottom right
13 Ce
14 cf
24 Za # Landing
```

It already almost looks like a marble run. The start could have been orientation **a** instead of **c**. Since only one ball will start to the bottom right, the **c** looks a bit nicer, but a would have been just as correct. How the rails connecting the tiles are noted is the subject of the next section.

Rails and other connection elements

If you have built the previous example, you can easily see which connecting elements are still needed. In the starter set there are three types of rails, short, medium and long. In addition, there is the finish line, which, like the rails, connects to a hexagonal tile. All connecting elements are given **lower-case letters** as abbreviations, whereby a prefix **x**, **y** or **z** is also possible here.

There are other connecting elements (rails and walls) included in additional sets. These will be described later. You can find a list of all currently known elements in the table in the appendix

The following assignments were made for the aforementioned rail types:

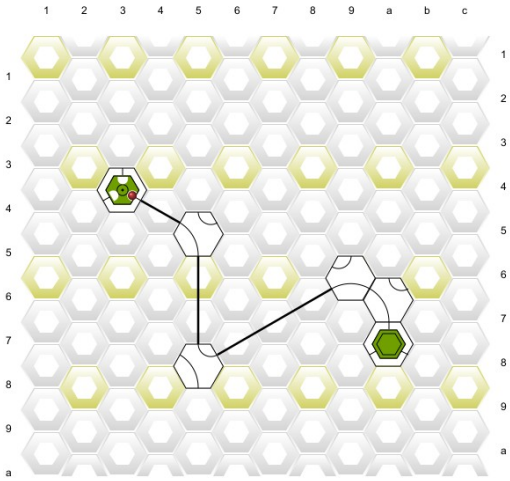
Symbol	Element	Comment
s	Rail Short	
m	Rail Medium	
l	Rail Long	This is the lower-case „el“, not the capital letter I
e	Finish Line	

If you now want to describe which two tiles a rail should connect, you first write down the position, tile symbol and orientation from where the rail starts. Then after a space you enter the type of rail (e.g. **s** for the short rail) and finally the direction in which the rail is pointing. As with the orientation of the tiles, you choose the letters **a** to **f** again, where **a** means the direction upwards, **b** means top right, **c** bottom right, d bottom, **e** bottom left and **f** top left.

Thus the description of a rail consists only of two (or three with **x**,**y**,**z**) lower case letters, the type and the direction. Each tile may have as many such descriptions as there are rail attachment points, e.g. three for a curve and four for a crossing.

Since a track (except for the finish line) always connects two tiles, it doesn't matter which tile you choose to describe the rail. But depending on the rail, the direction is then a different one. Now you know all the rules to complete the previous example. It then looks like the picture on the next page. How the red ball comes into the picture is described later.

```
 # Track A from the starter set booklet
_ 1 1     # top left
43 5Ac sc # Start
55 3Cf md
_ 2 1     # bottom left
35 3Cf lb
_ 2 2     # bottom right
13 Ce
14 Cf
24 Za     # Landing
```



There are no elements on the upper right base plane, so the line _ 1 2 from the previous example has been omitted. You can use the program (gravi) mentioned above to create building instructions. The following text is created from the example above (somewhat shortened):

Instructions for track A, board size 2x2
At plane 1,1 pos 43 5 x Height Tile large, Launch Pad Orientation (c)
At plane 1,1 pos 55 3 x Height Tile large, Curve Orientation (f)
At plane 2,1 pos 35 3 x Height Tile large, Curve Orientation (f)
At plane 2,2 pos 13 Curve Orientation (e)
At plane 2,2 pos 14 Curve Orientation (f)
At plane 2,2 pos 24 Landing Orientation ↑ (a)
From plane 1,1 pos 43 to plane 1,1 pos 55 Rail Short Direction (c)
From plane 2,1 pos 35 to plane 2,2 pos 13 Rail Long Direction (b)
From plane 1,1 pos 55 to plane 2,1 pos 35 Rail Medium Direction ↓ (d)

First the tiles on the first plane are described, then on the plane 2,1 below, then on the last plane, regardless of the order in which you wrote down the tiles and rails. When the tiles have been placed, then you can put the rails on them, therefore these lines come last. The program also can create the image of the noted marble run. In the picture you can see how the ball would travel from A to Z. Except for the transparent level, all starter set parts are now described.

Transparent planes

Transparent planes (**^**) (and **=**, see below) have to be described by a separate line giving the center position xy on the ground plane followed by the plane symbol. Anything after the line with the plane symbol is on the transparent plane. If there is a second level above the first transparent plane, this is again written down with a line that only contains the position of the center and the character **^** (or **=**). In the picture on the right, a transparent plane with the center at position 33 is drawn in light blue. Similar to height tiles you can place other tiles on the plane. Therefore it can be considered a large height element that is just as high as a black height tile. As an example, a level at a height of 5 should be created on the base plane 1.2 at position 33 with a start and a curve. To do this, place 5 height tiles each on positions 21, 25 and 53 and then on top the transparent plane. You write it like this:

```
_ 1 2 # on this plane is the transparent layer
21 5 # 5 height tiles each
25 5
53 5
33 ^ # on top of the height tiles the transparent plane at position 33
24 3Ce # curve on the transparent plane with 3 height tiles
25 3Af # Start on the transparent level with 3 height tiles
```

In the following example, the transparent plane extends over two base planes. The supporting pillars for the transparent plane are then also distributed over two planes. On the base plane is a curve, just above it on the first transparent plane is a catcher and on the second is a vortex. Since the positions on the transparent plane are differently numbered than on the base plane, the positions are called 31 and 33 respectively.

To make it easier for you to find the beginning of a transparent plane, you can also add a line with the word **level** and a number that indicates that a transparent plane with parts on being described after that line. Each transparent plane gets a new number. Then the line for the transparent plane (**^**) doesn't have to follow immediately after it. This allows you to sort the rows, for example, according to their position on this level, which is not always possible otherwise.

```
_ 1 1 # the base plane
16 5 # 3 times 5 height tiles on the base plane
46 5 #   for the first transparent plane
_ 1 2
33 5 # the third pillar is on the second base plane
level 1
12 5 # again 5 height tiles for the second level
42 5 # The positions are relative to the center at 33
35 5
31 ^ # Center of first transparent plane (on ground plane)
33 Ga # a catcher in the middle of the first level
level 2
31 ^ # the second level, the center is again on the 2nd base plane
33 Va # here lies a vortex in the middle of the second plane
_ 1 2
31 ca# a curve on the base plane
```

If there are other layers above a transparent plane, they must be given a higher number. You achieve this by first describing the lower level or by giving the higher level a larger number with a **level** line. Using such lines has other advantages as well. If you print building instructions using the program `gravi` mentioned before, the number of the level is always printed as well. It will then be much easier for you to find the places in the description above that belong to a certain level.

With each line that contains the _ character, you switch back to the base planes. You can also do or emphasize this more by a **level 0** line. So you could write:

```
_ 1 2 # the base plane
21 5 # 5 height tiles on the base plane each
25 5
53 5
33 ^ # the center of the first transparent plane
21 5 # 5 height tiles on the transparent plane each
25 5
53 5
33 Ga # a catcher in the middle of the first transparent plane
33 ^ # this is now the second level
33 Va # a vortex in the middle of the second plane
level 0
33 Ca # a curve, now back on the base plane
```

A new, smaller, transparent plane is included in the Speed Starter Set and the 2021 Advent Calendar. The symbol **=** has ben chosen for it. You can see the position numbering in the picture on the right. Everything else from this section also applies to the small transparent layer. Here is the notation for levels summarized again:

Symbol	Plane	Comment
_	Base plane	A green hexagon is in the top left corner
^	Transparent Plane	Its position is the center of the plane
=	Small Transparent Plane	Its position is the center of the plane

So far you can't yet write down how many balls you need and where they should placed. This is the topic of the following section.

Marbles

The marbles get sent from start tiles (start, lift, spinner, ...). You can see how many balls you need by looking at whether only one or more lanes are possible from there. But maybe you want to describe the color of the balls that are supposed to start running or that should be loaded into the magnetic cannon. To do this, you add marble descriptions to the tile on which the balls should lie.

The balls are written with the letter o, after which you can specify a color - **R** for red, **G** for green, **B** for blue, **S** for silver and **A** for gold (chemical symbol Au) - and a placement (**a** is at the top again , then clockwise **b** through **f**) . Up to 3 balls fit on the start tile, up to 6 on the spinner and many more in the lift. For the start tile from the previous example, you could write:

```
34 5^Ac sc oRc # Start tile with a red ball at the bottom right
```

Describing marbles is optional, and if you do it only the **o** (lower-case **o**) matters, you can also omit the color **RGBSA** or placement **abcdef**.
Where balls are required, the correct number of balls with the color silver will be added automatically if you did not specify anything. Three balls (**RGB**) are placed at the start tile initially. You can also put the number of balls of the same type in front of the o, for example 3**o**S for 3 silver balls.

If you want to launch several balls one after the other from the same position, you can also describe it by adding more marble descriptions to the tile from where the balls should start, for example

```
34 5^Ac sc oRc oRc oRc # Start with 3 red balls at c, 2 will be launched later
```

In order to build more complicated tracks and use more new parts, you need more rules, which are explained in the following sections. Before that there is a section on how you can use a different position notation. This is the more advantageous the more complicated the tracks become. The determination of a position is then easier, especially when using walls and balconies (from vertical starter set).

Alternate position notation

Instead of starting again with position 11 for each base plane, you could also number the positions continuously. The numbers 1 to 9 won't get you very far. If you continue with a to z then you can describe a fairly large marble track on 6 x 7 ground planes. Since it is already clear that in the first row there are positions 11 to 16 on the first base plane and 17 to 1c on the plane next to it, all lines with the _ are omitted. Then track A from the starter set booklet looks like this:

```
43 5Ac sc oc
55 3Cf md
85 3cf lb
69 Ce
6a Cf
7a Za
```

This is a little shorter than before, but you may not see where field 6a is as quickly. You could put a piece of paper under the base planes and write the positions in the holes. Then you don't always have to count where the field is. Of course, this also applies to the transparent layers. A transparent plane on ground plane 1, 2 with a curve would then look like this:

```
level 1
19 ^Ca # Curve tile on the transparent plane above center
39 ^ # a transparent plane at position 39, that's on plane 1,2
```

You can also number the rows and columns consecutively. Then you always have to insert a comma (or other separator character) between the two numbers. The last line in the example above is then `7,10 Za`

If base planes without tiles are to be omitted, you must write lines starting with **!** as described above.

New and unknown extensions

For GraviTrax® new extensions are issued regularly, which can contain new tiles, rails and other elements. The new elements need to be assigned to symbols so that you can write them down according to the rules presented in the previous sections. As already mentioned above, the 26 capital letters are not sufficient for all known tiles and some tiles get therefore prefixed with the characters **x**, **y** and **z**. The tiles starting with **z** belong to the GraviTrax® Power Series and are not currently checked for incorrect orientations.

Some extensions require additional information to describe everything exactly. If this is the case, the detail information is always placed after the symbol and before the direction. You have seen this on the switch (**S**), where the position of the seesaw can be described as a detail with **+** or **-**. This information is written as a **detail** between the symbol for the tile and the character for the orientation, as explained above. Where detail information is important is described alongside the element descriptions.

It also can happen that new tiles are not yet registered in the `gravi` program and have not yet associated with a symbol. New tiles that have been invented by users and produced with a 3D printer can also not be not known to the program. In such cases you can write the name of the tile between two vertical bars and use it wherever tiles are otherwise noted. The orientation of the tile can be specified, but it is not checked by the program. Here is an example:

```
11 3|Elevator| ma mc
```

This tile is drawn in the track picture as a hexagon with the name |Elevator| in the middle. If the name matches an element name (see table in the appendix), the corresponding symbol is used instead. Therefore, when noting an element, you could write **|curve|**. instead of **C**, for example.

Flip, Hammer, Jumper, Cascade, Catapult, Tiptube, Volcano, Looping, Transfer, Dipper Spinner and Color Swap

The extensions consist of one action tile, with the exception of transfer where there are up to three. The orientation **a** is as usual given by the outgoing marble running upwards there. This means that when the marble enters the tile from below e.g. in the cascade this is orientation a.

Volcano and Dipper make an exception. For the **Volcano**, orientation a is where the release button is pointing up, i.e. the marble arrives from above and releases the volcano. It's similar with the **Dipper**. With orientation a, the green part points to the top. This means that the orientation of the dipper is determined by the incoming ball and not by the outgoing one. The dipper has a moving part similar to the switch. Depending on its position, the ball is steered in different directions. As with the switch, the position of the seesaw is noted as a detail with + or -. The following letters are used for the extensions:

Symbol	Tile	Comment
F	Flip	
H	Hammer	
J	Jumper	
K	Cascade	
xK	Catapult	Similar to Cascade, works with rubber bands instead of swinging marbles
xT	Tiptube	Balls already in the Tiptube can be noted
N	Volcano	Orientation defined differently
Q	Looping	The Q resembles a Looping
xR	Transfer	
xD	Dipper	+ means, the green part is turned clockwise, - for anti-clockwise
xS	Spinner	Orientation is optional
xP	Color Swap	Marbles in the Color Swap can be noted

From the **Spinner** up to 6 balls can be launched at the same time. An orientation can be specified

Color Swap is a fairly simple extension. An incoming ball remains in this element. If there is already a ball in it, this ball will continue to run instead of the one that arrived. If there are already balls in the **Tiptube** or in the color swap at the start, you can note this as described in the Marbles section.

A track containing a **Catapult** behaves differently depending on the tension in the rubber band, which ages over time. It is therefore possible that such a track does not work as designed for you. A possible improvement is shown in the pictures on the right. A spring (from a ballpoint pen) is fixed with a stable wire in place of the rubber bands. This has the effect that the ball's jump distance is fairly constant even after prolonged use.

Spiral

The **Spiral** is a tile that you can place as usual. You can adjust the height yourself by using more or less curve parts in the spiral. It also works without curve parts. With each curve part, the direction from which the ball can arrive changes if the ball exit remains the same. The orientation of the spiral is determined by the direction of the outgoing ball. **xH** is chosen as the symbol for the spiral (H for helix). Each green piece (minimum two, beginning and end are required) is half a unit high. The total height of the spiral and thus also the direction from which the ball runs into the spiral is determined by the number of green parts. You have to write this number as additional information between the symbol **xH** and the character for the orientation of the spiral, for example **xH4b.** If you don't specify anything, it will be assumed that you use two green parts.

Symbol	Element	Comment
xH	Spiral	
h	Spiral Curve	Not used in track descriptions
i	Spiral In	Not used in track descriptions
j	Spiral Out	Not used in track descriptions

Lifter

The **Lift** needs detail information similar to the spiral. It is given the symbol **xF**. You can vary the height of the lift by using a different number of transparent parts. There must be at least two, since the start and end of the lift are always used. The direction of the outgoing ball can be freely selected here. As an exception the orientation of the lift is defined by the side where the ball enters the lift at the bottom. As detail information, not only the number of transparent parts, but also the direction of the outgoing ball needs to be written down. An example of a lift with 4 transparent elements is **xF4be**.

Symbol	Element	Comment
xF	Lifter	
f	Lift Tube	Not used in track descriptions
xi	Lift In	Not used in track descriptions
xj	Lift Out	Not used in track descriptions

Extension Tunnel

The Tunnel extension brings several new tiles and rails. **Tunnel Curve** (**T**) and **Tunnel Switch** (**U**) work exactly like curve and switch, the **Straight** (**I**) and **Vertical Tunnel (t)** is new. The vertical tunnel is a rail because it is hooked into a tile like a normal rail and connects two tiles. You will find also 3 other new rails, the curved **Bernoulli Rail (b)**. a Drop Rail Convex (**u**) with a hole and a Drop Rail Concave (**v**). A ball can fall through the hole, which then lands in a (catch) **Open Basket (O).**. The basket has to be placed on another rail. This basket position is a position like for a tile. Therefore, a capital letter was chosen for the open basket. You can see all elements from the Tunnel extension and their abbreviations in the following table:

Symbol	Element	Comment
T	Tunnel Curve	
U	Tunnel Switch	The seesaw cannot be removed, therefore no 2 in 1 tunnel
I	Tunnel Straight	Capital letter I, not the lower-case „el“
O	Open Basket	Capital letter O, not the zero
t	Tunnel vertical	
u	Drop Rail Convex	
v	Drop Rail Concave	
b	Bernoulli Rail	

With the tunnel switch, you have to look carefully where the ball runs into. As with the normal switch, this is important for the orientation of the tile. Since the seesaw is covered, you see it harder than with the normal switch. As with the normal switch, you can also enter a **+** or **-** for the initial position of the seesaw for the tunnel switch. The open basket sits on a rail. For a complete description you have to specify the position of the basket, the track symbol and the outgoing direction of the ball on the track, although the track is already described elsewhere.

Zipline

The zipline consists of the starting tile **Zipline Begin** (**xA**) with the marble transport element, the **Zipline End** (**xZ**) and the **Zipline Rail** (xa). The orientation of the tiles is again given by the running direction of the ball, i.e. xA and xZ always have the same orientation. For the zipline elements you use:

Symbol	Element	Comment
xA	Zipline Begin	
xZ	Zipline End	
xa	Zipline Rail	

Flextube

The **Flextube** (**xt**) is a tube very similar to the vertical tunnel. This is why the Flextube is also given the **xt** symbol. In contrast to the vertical tunnel, the direction of the exiting ball can be chosen freely. Therefore you have to write the direction of the outgoing ball (**a** .. **f**) as a detail between the **xt** and the orientation.
As with rails, the orientation is where the Flextube is attached. For example

```
23 6Cb xtfd
```

The flextube hangs on a curve tile with orientation b, the flextube is attached to the tight curve (points downwards) and the ball runs out of the flextube in direction f.

Trampoline

In the trampoline extension you will find two Angled Base (**r**) tiles in addition to the trampoline tile (**R**) with the elastic mat. No rails fit on the trampoline, but you can put one or two angled tiles under the trampoline in addition to normal height tiles. This way you can cause the ball to exit the trampoline in a slightly different direction. The angled tiles have a high and a low edge.
Therefore, orientation is again important for the angled tiles. For orientation **a**, the high side points up. If angled tiles are used, only their orientation is written directly after the R symbol as a detail, the symbol (**r**) is not used. The trampoline tile itself does not need any information about orientation, only for the angled tiles the orientation is important. An example line for a trampoline with two of it might look like this:

```
 43 2Rbb # Trampoline on two height tiles and two angled tiles
```

You have to try how far the balls jump when they hit a trampoline. The mats may become less elastic over time or the tiles may have been manufactured differently. Therefore tracks with a trampoline do not always work as described.

Extension bridges

The Bridges extension comes with a new tile on which two balls are placed as a weight. The green parts for the **Bridge Tile** (**xB**) are attached to the side where the ball leaves the tile. The bridge works best with 4 parts, but two or more than four are also possible. Only with 4 elements unfolding bridges are possible. The bridge tile combines a tile and a rail, because it lets the balls roll from one bridge tile to another. In order for a connection to another tile to be possible, the number of bridge parts must be even. The moving parts (the rail) are always attached to the bridge tile, therefore a rail description is not used. Instead, you write the number of bridge parts directly after **xB** and before the character for the orientation of the bridge tile. With 4 parts you can also omit the number. Although **xb** doesn't appear in a description of tracks, it is used to count how many draw bridge elements are needed and to check if you have enough such pieces. An example:

```
22 xB6c # a bridge tile at position 22 with 6 parts for a draw bridge
```

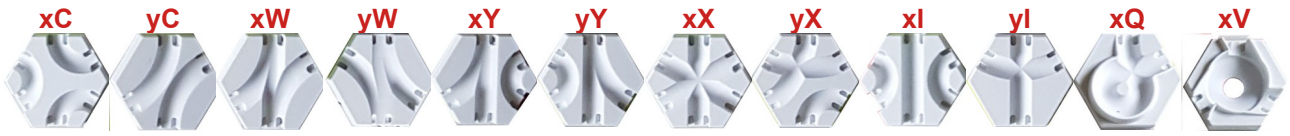
The bridge with 6 elements can then only be used when it is folded out, otherwise the marbles cannot roll. The bridges extension set brings also two new very long rails, the overlong rail (**g**) where marbles run fast and an overlong rail slow (**q**) where marbles are slower than on normal rails.

The following symbols are used for the parts from the bridge extension set.

Symbol	Element	Comment
xB	Bridge Tile	With moving parts (draw bridge)
xb	Bridge Element	Not used in track notations
g	Rail Overlong	
q	Rail Overlong Slow	

New parts in the 2021 advent calendar

In addition to already known parts, 12 new curves and crossings are included in the advent calendar 2021. The illustration shows the tiles and the associated symbols in orientation **a**.



You can also see the orientation of these parts in the attachment. In addition to the tiles shown here, other parts are also included. The rails not yet discussed (**a**, **c**, **d**) are described in the next section. You will also find more rails, a spiral (**xH**), a small transparent plane (symbol **=**) and a golden ball (**A**) in the advent calendar. The following table describes the new tiles.

Symbol	Element	Comment
xC	Curve 3x small	
yC	Curve 2x large	
xW	2x 2 in 1 left	2 large Curves and a straight path, Curve is from top to the left
yW	2x 2 in 1 right	2 large Curves and a straight path, Curve is from top to the right
xY	2 in 1 left with Curve	Large Curve, small Curve and a straight path, large Curve to the left
yY	2 in 1 right with Curve	Large Curve, small Curve and a straight path, large Curve to the right
xX	Straight 3x	3 straight crossing paths
yX	3 Curves, 2 crossing	2 large Curves are crossing, in addition a small Curve
xI	Straight with 2 Curves	2 small Curves to the left and right of a straight path
yI	Cross Straight and Curve	Large Curve crosses a straight path (capital letter I, not lower-case el)
xQ	Loop Curve	
xV	Vortex 3 in	The same element as the top part of the Mixer

Starter set vertical and extension set vertical

The vertical starter set contains many parts that have already been described. Curved short rails are new. There are also pillars, which are height elements that you can stack on top of each other. But they also have openings for mounting transparent walls. Since the walls always connect pillars, they get lowercase letters, like rails. Finally, there are balconies and double balconies. Balconies are hooked into the transparent walls to place height elements and normal tiles on them. How these elements are described will now be shown. First comes the table with the new parts and the corresponding symbols.

Symbol	Element	Comment
L	Pillar	7 height units
xL	Tunnel Pillar	Pillar with a hole
B	Balcony	Attached to walls
E	Double Balcony	Is a height element, the second part needs to be described separately
xs	Wall short	With 10 + 10 hooks for Balconies
xm	Wall medium	With 10 + 13 + 10 hooks
xl	Wall long	With 10 +13 +13 +10 hooks
a	Rail Bernoulli short	Like the Bernoulli rail b, just shorter
c	Rail counter clockwise	Curved rail, Ball turns anti clock wise down the rail
d	Rail clockwise	Curved rail. Ball turns to the right if looked from above (orientation a)

The rails **a**, **c** and **d** are short curved rails. In the rails labeled c and **d**, the ball does not run straight down but makes a counterclockwise or clockwise bend respectively. The picture to the right shows the 4 curved rails **a** to **d**.


bent rails b,c, a and d

New height elements

Pillar, tunnel pillar and double balcony are height elements that can be stacked on top of each other, and also be mixed with the other height tiles. Pillars are as high as 7 gray height tiles, a double balcony is as high as a black height tile.

Previously, no orientations were needed for height elements. Since the tunnel column has a hole, orientation is important here and must be specified. As with many other straight elements, orientation **a** and **d** look the same, as do **b** and **e** or **c** and **f**. The direction of the balcony and double balcony is calculated from the program, it can also be specified, but must then be correct. As always, **a** is the orientation upwards, for the simple balcony slightly at the top right, as the top direction is not possible. For normal pillars no orientation is needed, but if one is specified it will be ignored.

Because these rules are already relatively complicate, here is an example of noting the height elements at position 23, whereby all tiles have the orientation a where needed:

```
23 LEa2Ca   # stacked pillar, double balcony, 2 height tiles and a curve
23 xLaEa2Ca # as before, instead of a pillar a tunnel pillar
23 1L4Ca    # At a height of 1+7+4=12 a curve (pillar is 7 units high)
23 93Ca     # At a height of 9+3=12 again a curve on height elements
```

Walls

Walls connect two pillars or tunnel pillars like a rail. Therefore they get written behind the tile just like rails. However, since simple balconies can hang on the walls, it must be clear to which wall the simple balconies belong. Therefore, each wall is noted once more on a new line with the position from the originating pillar, and then the descriptions of the simple balconies for that wall can follow on further lines. A wall does not always have to be hung in the lowest column. Therefore, a wall is given the number of the pillar in which it is mounted, in addition to the direction. The number starts with one for the bottom pillar (and can be omitted). If it is instead hung in the pillar above, write down the number 2 and so on. Since one wall is as high as two pillars, the second wall can only be hung in the third pillar in the same direction.

Now consider the following example, where at position 13 two walls are hung one above the other in direction b, at position 15 the wall is exactly as high as the second wall at position 15:

```
13 LLLLCc # four pillars one above the other
13 xlb    # Two walls are hung in it, here the first
13 3xlb   # and the second
15 77LLCc # 14 gray height tiles, only two pillars
15 xlb    #: Wall has the same height as the second wall above
```

Double balcony

A **Double Balcony** extends over two fields of the base plane. Both fields must be described. The first part of the double balcony has already been noted in the stacked height elements (see above). The second part of the double balcony is exactly above a hole in the base plane. This position (always the base plane positions, even if the supporting tile or pillar is on a transparent plane) has to be written on a new line, followed by the double balcony symbol (**E**). If the second part of the double balcony is on a different base plane, the base plane must be switched before inserting an **_** line. Therefore, as mentioned above, it is easier to use the consecutive notation of the positions and omit the lines for the base planes. After the **E** you write, what is placed on the double balcony. As always, height tiles come first, if any, then can come another tile at the top. Of course, you can also add rails to the final tile as usual.

The second line for the double balcony, beginning with **E** after the position information, must be written before further double balconies get described. If you want to stack multiple double balconies at one position, all second rows for the double balconies must be written in the order in which they occur from bottom to top.

An example:

```
_ 1 1        # base plane 1
15 LLELLEaCc # four pillars one above the other with two double balconies
15 xlb       # the bottom wall
15 2xlb      # the top wall
_ 2 1        # The lower double balcony points to the second base plane
11 E+Cf md   # Double balcony with a half height tile and a curve with a rail
_ 1 1        # This double balcony originates from the first base plane
14 ECa       # second (upper) double balcony with curve
```

Simple balconies

The simple balconies are hung in a wall. They are then also located over a hole in the base plane. That is used as the position of the balcony. Then you have to say in which opening the balcony was hung. The holes are numbered from bottom to top. Where there are 13 holes, it goes from 1 to 9, then comes a,b,c,d. With 10 holes there are no positions 6, 7 and 8, so you count from the bottom from 1 to 5, then comes 9 to d. Then you write down the **B** for balcony.

Since several walls can be in the same position on top of each other and can point in the same direction, it must always be clear for the balconies to which wall they belong. This is achieved by always describing the simple balconies immediately after the (second separate) line for the wall. Once a new wall is noted, the lines for simple balconies then apply to that wall. Here is another example:

```
35 LLLLCc    # four pillars one above the other, in which three walls are hung
35 xld       # two walls point in the same direction
54 dBCb      # Balcony in the first wall in hole 13, with curve on it
54 5BCe af   # another balcony in hole 5 with curve on it and a rail
35 2xld      # wall above the first wall
44 1B1+Ca aa # Balcony in the second wall in hole 1, with 3 tiles and a rail
35 1xsc      # third wall at the same height between the two previous walls
26 3B        # Balcony in the short wall in hole 3, with no tiles on it
```

As with the double balconies, the positions of the balconies are always for the base plane, never for transparent levels, and here too the consecutive notation of positions is usually easier.

Parts from GraviTrax Pro®

Mixer, Splinter Helix and Turntable

The **Mixer** (**xM**) is a tile that has the same symmetry as the starting tile. There are only two orientations that differ. The specialty is the green moving part which, depending on its position, determines where the ball rolls. If you want, you can write the direction of the outgoing ball as an additional information between **xM** and the orientation of the mixer. For the orientation a, c and e of the mixer, only directions a, c or e are possible, for b, d and f the other three directions. Another assembly of the mixer is possible where the directions are different. However, this is not taken into account in the program.

The **Splinter** (**yS**) is a tile that, depending on the position of the rotating green flap, directs the ball in one or the other direction. The position of the flap therefore decides in which direction the ball runs. As with the junction tile, up to four rails can be connected at the top of the tile, so the orientation is defined the same as for the junction tile. For orientation a, the green flap points upwards slightly to the right. So you don't need any additional information for the initial position of the flap.

At first glance, the **Helix** (**yH**) looks like orientation doesn't matter. The function is also the same, no matter which orientation you choose. However, since the entry points for balls at the top have slightly different heights, orientation **a** should be the one where the track drops less steeply, i.e. the entry point is somewhat higher than the neighboring one..

The element **Turntable** (**yT**) has a straight path on the upper level. For orientation **a** the straight path goes in the upper plane from top to bottom

Carousel

The carousel extension consists of just one tile, the **Carousel** (**yK**). It alternately has ball inlets (as high as a black tile) and slightly lower ball outlets on both the lower and upper levels. If the tile has the orientation **a**, a ball outlet is on the upper side in the lower level.
Then on the upper level there is a ball inlet at the same position. Depending on where a ball arrives at the top or bottom, it can stay where it is or run out in a new direction (inlet **a**, outlet **b**, etc.).

This describes all the rules for the previously known parts. You will find the scattered information summarized again in the appendix. As you can see, the more complicated parts there are, the more complicated the rules. But even with the proposed tracks found in the booklet for the vertical starter set, up to 11 consecutive pictures are needed to show how the elements for the tracks have to be arranged.

Notation of marble tracks

If you have a working marble run that you want to write down, a good technique is to gradually take away the elements and noting what you took as you go. First of all, write down what you want to call the track so that you will remember later what the many letters and numbers mean. Then you might want to remember when you built the track. To do this, you can write “Date: …” in a new line. Exactly how you write the date doesn't matter. If you intend to exchange your tracks with others, you should also add "Name: ...". Further lines starting with a # can contain an additional description of your track. The beginning of your description could then look as follows:

```
Track A from the starter set booklet
Date: December 24, 2020
Name: John Doe
# Now the description of the track begins
```

If you use transparent planes, you start with the top layer. First you write on which base plate (_) the center of the transparent layer is, then the line with the **^** that describes the layer. Then note the first tile with its height and the rails that start from it. Now you can remove the rails and the stone. Do this until the level is empty and continue with the next level. If there are only stones left on the base planes, you first write down a line with the position of the plane (_) from which you want to remove further tiles and rails. Then proceed as with the transparent layers. You can always switch between the base plates by inserting **_** lines. If all base planes are emptied, the track should be completely noted. Of course, you can also do the opposite and write down the parts when doing the track assembly.

Owned GraviTrax® kits

You can also store in a file which GraviTrax® construction kits you have. This is useful in conjunction with the program mentioned. If you have instructions for a track, the program can count how many parts you need to build it. Then the program can check whether the parts in your construction sets are sufficient for this track. The names of the kits are as follows:

Starter Sets	**Extension sets**	**Action tiles**	Looping	Flextube
Starter Set	Building	Zipline	Magnetic Cannon	Helix
XXL Starter Set	Trax	Transfer	Hammer	Turntable
Starter Set vertical	Tunnel	Jumper	Catapult	Color Swap
Starter Set Speed	Lift	Flip	Spiral	Carousel
	Bridges	Tiptube	Mixer	**Game Sets**
	Extension vertical	Volcano	Splitter	Game Flow
	Advent 21	Trampoline	Spinner	Game Impact
		Cascade	Dipper	Game Course

You can write your name in the first line if you also know from others what kind of kits they have and want to store this information as well in separate files. Then you can see which tracks they can build with their parts. After the name line you note how many building sets you have of what type, first the number, then the name. Of course, at least one starter set must be included. You can even add individual parts using the symbols,

not the names of the parts. If you lost a part or it is broken, for example a curve tile, enter it with a minus sign. Your list could look like this

```
John  Doe
 2 Starter Set
 1 Hammer
 1 Spiral
 3 s                # 3 short Rails in addition
-1 C                # a curve tile is missing fehlt
```

If you have such lists for several people and want to save them with the program, it is assumed that the first saved list contains your construction sets.

For the tiles lift, spiral and bridge, parts are needed which are implicitly contained in the tile description. For the construction of the tracks it is important to know how many of these parts are available and to decide, whether there is enough material to build a given track. Therefore symbols were assigned to these elements.

If you build a track, it can happen that you are short of height tiles. There are instructions in the internet on how to build your own height elements, preferably pillars with larger heights. If you already have some and want to note how many and what height you have then you can use the digits 3 to 9 for this, which up until now have only been used as a height, but not for a pillar.

In the picture to the right you can see such a pillar with a height of 7 and a black height stone on top. The hexagon below is a piece from the base plane that fits again perfectly back into the ground plane.

Software for recording the tracks

The above concept has been implemented in a Perl program.Apart from a standard Perl, the database module DBD :: SQLite and SVG are required, in addition the Locale::Maketext::Lexicon module to support the localisation of text in other languages. When installing the modules, other perl modules are also installed, for example DBI. Support for databases other than sqlite3 is currently not planned.

During the first invocation of gravi, a `.gravi.db` sqlite3 file is created in the user's home directory. It can be filled initially with a demo track. When instructions for building a marble run are displayed, an SVG image file (best viewed with web browsers) can be created alongside.

Installation on Linux

Assuming that perl is installed, the installation should be straightforward. The newest Game-MarbleRun-xy.tar.gz has to be downloaded (xy is the version number), unpacked and then you change into the new directory. Then the usual procedure for installing perl modules follows:

```
tar xf Game-MarbleRun-xy.tar.gz
cd Game-MarbleRun-xy
perl Makefile.PL
make
make test
make install
```

If the program should not be installed centrally, a good recipe is to use the local::lib perl module described e.g. at https://metacpan.org/pod/local::lib.

Installation on Windows

The installation under Windows requires a working Strawberry-Perl (https://strawberryperl.com/). The security warning must be ignored when installing Strawberry perl. When strawberry perl is installed, you open the command prompt (e.g. the power shell). Then you have to install some required perl modules and

enter the other commands similar to Linux. As DBD :: SQLite should already be included with Stawberry Perl then the cpan command for it does nothing. If Strawberry Perl is installed, ActiveState Perl must not be installed at the same time, otherwise the following commands will not work.

```
cpan install DBD::SQLitecpan install SVG
cpan install Locale::Maketext::Lexicon
tar xf Game-MarbleRun-xy.tar.gz
```

The last command creates a directory with the software. You need now to change into that directory:

```
cd Game-MarbleRun-xy
```

and execute there the following commands:

```
perl Makefile.PL
gmake
gmake test
gmake install
```

If the procedure above is too complicated or strawberry perl should not be installed, a ready to run gravi.exe can be downloaded from the web site https://www.zeuthen.desy.de/~friebel/gravitemp/. Please use it at your own risk.
Since the cmd program or a similar terminal program (e.g. power shell) has to be used to start the program, the characters there are normally printed with the Windows encoding. However, since the gravi program uses Unicode characters, some special characters would be displayed incorrectly. Therefore, the active code page is changed to Unicode (65001) in the program and reset at the end of the program. Since the default font is usually missing some Unicode characters, the font for the prompt should be changed. To do this, right-click on the top bar of the program. In the menu that opens, select the Font tab. There you can either choose the font "DejaVu Sans Mono" or "Source Code Pro". Only these two fonts display the special characters used by gravi correctly. You only have to make this setting once, Windows remembers your choice of font.

Program usage

After the installation you can start the program gravi. You do this in a terminal window under Linux or using the command prompt under Windows.

When you start `gravi` for the first time, if no database has been created yet, you can choose whether a demo track should be included in the database. In this case, this track will be saved as the first track and the build instructions will be printed immediately afterwards. If you want to get a picture of this track, you have to type a name when prompted for a file name, for example `demo`. Then a `demo.svg` file is created containing the image of the track. You can then view this file with a program that understands SVG files. These are, for example, web browsers such as Firefox, Chrome or Microsoft Edge or graphics programs.

The program tries to use the native language for its output and error messages, but falls back to English, if the corresponding language support is not available. Currently only German is implemented. You can force the **switch to a certain language** by the option -l followed by the two letter country code, such as `-lde`.

The program is the center for all functions related to your marble runs. First you can store the list of your owned sets. This is not absolutely necessary, but has many advantages. This way you can see whether you have enough parts to build a certain track or which parts you are missing. Without your list, it's assumed you only have one starter set. **Storing the list of your owned sets** goes with

```
gravi -a file_with_the list
```

This is exactly how you can **save your own noted tracks**, which are stored in files as well.. If you don't have own tracks yet, you can also save the included tracks in the runs subdirectory. This can be done with one call for all files in the runs directory simultaneously:

```
gravi -a runs
```

This way you've saved a whole lot of tracks. If you have noted a new track, you should first **check the track** to see if there are any errors without saving it in the database. You do that with

```
gravi -an my_new_track_file
```

If only one marble run is checked, an image of it can be generated. There you can easily identify errors in your notation, e.g.where a rail may be missing or a tile has the wrong orientation. If the marble run is saved by mistake (in the sqlite database `.gravi.db` in your home directory), you can **overwrite the run** by entering the changed course (again with -a). But please remember that with a new date line or with a changed author, name or source, it will be a new track. Then you can also **delete** the old marble run:

```
gravi -d
```

Then the program asks you to enter the number of the track that you want to delete. If you call the program without any further parameters, you will get a **list of the saved runs** together with the track numbers:

```
gravi

List of registered marble runs
------------------------------------------------------------------------------
 id OK x*y*z title (source)                                          gravi 1.11
------------------------------------------------------------------------------
  1  Y 2x2x0 track A (Construction plan booklet)
  2  Y 2x2x0 track B (Construction plan booklet)
  3  Y 2x2x0 track C (Construction plan booklet)
...
 14  N 2x2x1 track E vertical (Vertical Construction plan booklet)
...
Display instructions for a marble run:
Please enter an id from the list above
```

If you don't enter anything else (Enter), then the program will be terminated. However, if you select a run by entering a number, the **description of the track** will be displayed. If you enter a 1 for the first run, it looks like follows::

```
Instructions for track A, board size 2x2
Please enter file name without .svg (ENTER to continue without SVG output)
```

Now you can choose whether you want to create a **picture of the track** (simplified top view) or just see the assembly instructions. The image has already been shown above, so it is not reproduced here. The web browsers usually display the SVG's correctly. When using graphics programs for the display, then some text are slightly displaced and animation of tracks (see below) will not work.

The horizontal and vertical coordinates are given in the picture around the base plates so that you can quickly determine where a field with a given position is located. The positions can also be output to free positions within the base plate. To do this, you must call gravi with the option -f (for filling in the free fields).

The output of the TRACK A build instructions then look slightly different than shown above:

```
At pos 43 5 x Height Tile large, Launch Pad Orientation    (c)
At pos 55 3 x Height Tile large, Curve Orientation    (f)
At pos 85 3 x Height Tile large, Curve Orientation    (f)
At pos 69 Curve Orientation    (e)
At pos 6a Curve Orientation    (f)
At pos 7a Landing Orientation ↑ (a)
From pos 43 to pos 55 Rail Short Direction    (c)
From pos 55 to pos 85 Rail Medium Direction ↓ (d)
From pos 85 to pos 69 Rail Long Direction    (b)
At pos 43 (Launch Pad) 1 x red Ball Orientation    (c)
```

This is the already mentioned **other notation of the positions**, where the positions of 1..9a..z are given. This makes the text a little shorter, since it no longer has to be said on which base plane a tile is located. For the description with the base planes, you have to specify the -r (relative) option, which means that the positions are relative to the upper left corner of the respective base plane, i.e.:

```
gravi -r
```

In the example above, the track number 14 has an N after it, which means you don't have all the parts to build the track. You can find out which parts you are missing if you specify the option -v (**verbose**), i.e:

```
gravi -v 14
```

If you are only interested in the parts and not the instructions for the run, you can use the -q (**quiet**) option. With this option, however, you will still see what you have to do at the beginning of a marble run, for example loading a cannon or setting the hammer correctly. This is useful if you want the marble run to work properly.

If you are interested in marble **runs with a given element**, say with an extra-long slow rail (symbol **q**), the program will also help you with your search. You just enter the element symbol as argument to gravi:

```
gravi q
```

and get

```
List of registered marble runs
--------------------------------------------------------------------------------
 id OK x*y*z title (source)                                          gravi 1.11
--------------------------------------------------------------------------------
 10  Y 2x2x0 Bridges track (Instruction booklet)
 38  Y 2x2x0 run 6a unfolding bridges (Wolfgang)
...
```

You can also **search for marble runs** by a character string that occurs in the run name or an OK for runs where you own all the required elements. More of such conditions can be combined.If separated by commas, conditions must all be met, those separated by spaces can be met. If you don't have the slow rail and want to know which extension set it is in, you can first display the names of all the **sets that are known** so far:

```
gravi -s
```

To display the **elements contained in these sets** as well you have to add the -v (verbose) option and you can see from the long list that this slow rail is contained (among other sets) in the Bridges extension.

If you specify set numbers or part of the name when calling with -s, only the parts from the corresponding sets are output, as you can see in the following example:

```
gravi -s bridges,10

Bridges Extension (id 9)
12 x Bridge Element                    6 x Ball
 3 x Rail Short                        3 x Bridge Tile
 2 x Rail Medium                       2 x Rail Overlong slow
 2 x Rail Overlong                     1 x Rail Long
Extension Vertical (id 10)
16 x Balcony                           8 x Pillar
 4 x Tunnel Pillar                     2 x Wall Medium
 2 x Wall Long
```

If you don't have the list of abbreviations for the tiles and rails at hand, you can quickly display them.

```
gravi -e
```

If you are no longer sure whether you have registered all your kits, you can check it with -i (inventory):

gravi -i

You can find out what else is possible with the program in the help text

gravi -h

The functions described above are also mentioned there. If you have now become curious and want to know details about how the program works internally or even want to help improve the program, the next section is for you. Otherwise the description ends here

Program details

The program is written in perl5. Almost every version should be usable. The program works on both Linux, MacOSX and Windows. The marble runs are stored in a sqlite database. Because of the small amount of data, sqlite is certainly optimal. However, it should be easy to adapt to another database.
The program gravi uses the newly developed perl module `Game::MarbleRun` and submodules. As usual in perl, you will see a description of procedures in the perl modules by

```
perldoc Game::MarbleRun
perldoc Game::MarbleRun::Store and
perldoc Game::MarbleRun::Draw
```

The storage location for the SQLite database can be changed in the procedure new, as well as correcting the slight SVG text displacement in graphics programs.This is documented in `Game::MarbleRun`.

By default, the program outputs texts in the default language, this is achieved by translation files de.po, fr.po etc. If the files do not exist or are incomplete, English will be used instead. For German the translation is complete, for French there are only most of the element names in the translation. After installing the program, a new .po file can be generated by calling

```
make_gravi_po_file -l lang    # on Windows use gmake instead of make
```

where lang is either an abbreviation for a language such as fr, es, hu etc. or a dialect such as en_GB or en_US or a dialect with specification of the encoding such as de_DE.UTF-8.

You are welcome to write comments, suggestions, error messages, improvement requests or even corrections to the program and translations into other languages to me: wp.friebel@gmail.com

Sources can be found both on
https://www.zeuthen.desy.de/~friebel/gravitemp/ and on github https://github.com/wofr06/marblerun.

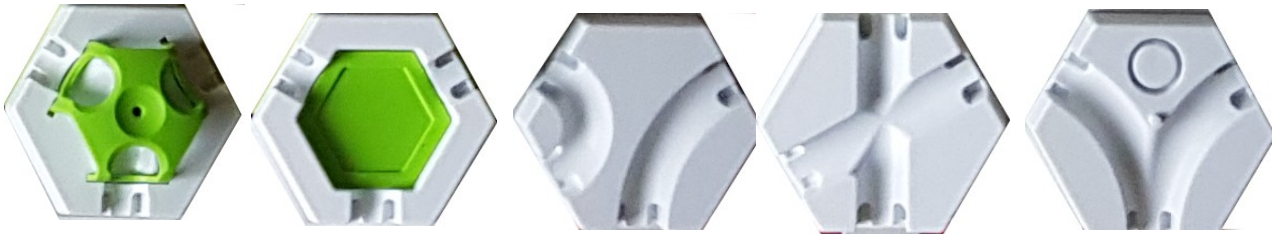
Appendix

Table of all used Elements and its Symbols

Symbol	Element	Detail / Comment
_	Base Plate	Base Plate Row, Column
*	Small Base Plate	Size like Transparent Level
!	Not shown Base Plate	Base Plate Row, Column
^	Transparent Level	Center Position, no Orientation
=	Small Transparent Level	Center Position, no Orientation
o	Ball	Colors ARGB, Direction a-f
+	Height tile small	
1	Height tile large	
2	Height tile 2 units	Not useful
3	Height tile 3 units	Only for self made height tiles
4	Height tile 4 units	Only for self made height tiles
5	Height tile 5 units	Only for self made height tiles
6	Height tile 6 units	Only for self made height tiles
7	Height tile 7 units	Only for self made height tiles
8	Height tile 8 units	Only for self made height tiles
9	Height tile 9 units	Only for self made height tiles
A	Start	Ball descriptions can follow
B	Balcony	
C	Curve	
D	Freefall	
E	Double Balcony	Double Balcony number
F	Flip	
G	Catcher	
H	Hammer	
I	Tunnel Straight	
J	Jumper	
K	Cascade	
L	Pillar	
M	Magnetic Cannon	Ball description
N	Volcano	Ball description
O	Open Basket	No height information, no Orientation
P	Splash	Ball description
Q	Looping	
R	Trampoline	Detail: Orientation of Angled Tiles
S	Switch	Switch Position + or -
T	Tunnel Curve	
U	Tunnel Switch	Switch Position + or -
V	Vortex	
W	3 in 1	
X	Junction	
Y	2 in 1	
Z	Landing	
xA	Zipline Begin	Ball description
xB	Bridge Tile	Number of Bridge Elements (even Number)
xC	Curve 3x small	
xD	Dipper	Switch Position (+ or -)
xF	Lift	Number of Elements (2 or more)
xG	Basic Tile	Only for number of own elements
xH	Spiral	Number of green Elements (2 or more)
xI	Straight with 2 Curves	2 small Curves left and right
xK	Catapult	
xL	Tunnel Pillar	
xM	Mixer	Detail:Direction of outgoing Ball

xP	Color Swap	
xQ	Loop Curve	
xR	Transfer	
xS	Spinner	Ball description
xT	Tiptube	Ball description
xV	Vortex 3 in	
xW	2x 2 in 1 left	2 large Curves and a straight path
xX	Straight 3x	
xY	2 in 1 left with Curve	Large Curve, small Curve and straight path
xZ	Zipline End	
yC	Curve 2x large	
yH	Helix	
yK	Carousel	
yI	Cross Straight and Curve	Large Curve crossing straight path
yR	Releaser	
yS	Splinter	
yT	Turntable	
yW	2x 2 in 1 right	2 large Curves and a straight path
yX	3 Curves, 2 crossing	2 large Curves crossing, a small Curve
yY	2 in 1 right with Curve	Large Curve, small Curve and a straight path
a	Rail Bernoulli short	
b	Rail Bernoulli	
c	Rail counter clockwise	
d	Rail clockwise	
e	Finish Line	
f	Lift Tube Element	Only for number of own elements
g	Rail Overlong	
h	Spiral Curve	Only for number of own elements
i	Spiral in	Only for number of own elements
j	Spiral out	Only for number of own elements
l	Rail Large	
m	Rail Medium	
q	Rail Overlong slow	
r	Angled Base	Only as Detail to R
s	Rail Short	
t	Tunnel Vertical	
u	Drop Rail Concave	
v	Drop Rail Convex	
xa	Zipline Rail	
xb	Bridge Element	Only for number of own elements
xi	Lift in	Only for number of own elements
xj	Lift out	Only for number of own elements
xl	Wall Large	Both as Rail and separate Line with Position
xm	Wall Medlum	Both as Rail and separate Line with Position
xs	Wall Small	Both as Rail and separate Line with Position
xt	Flextube	Direction of outgoing Ball
z+	Light Tile small	
z1	Light Tile large	
z2	Light Tile Base	
zA	Dome Starter	
zE	Elevator	
zF	Finish Trigger	
zL	Lever	
zQ	Queue	
zS	Drop Down Switch	
zT	Trigger	
zZ	Finish Arena	

Tiles (selection)



… .Start (A) Landing (Z) Curve (C) Junction (X) 2 in 1 (Y)

3 in 1 (W) Switch (S) Magnetic Cannon (M) Vortex (V) Freefall (D)

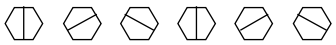
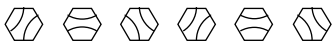
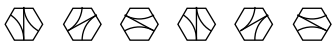
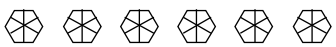
…..Catcher (G) Splash (P) Height tile small (+) Height tile large (1) Base tile (xG)

Flip (F) Hammer (H) Jumper (J) Cascade (K) Catapult (xK)

Tiptube (xT) Volcano (N) Looping (Q) Transfer (xR) Dipper (xD)

. .Spinner (xS) … Spiral (xH) Tunnel straight Tunnel Switch Tunnel Curve

Orientation of Elements

Orientation	a	b	c	d	e	f
Curve						
2 in 1, Switch						
Junction						
Catcher, Freefall (Drop)						
Straight Tile						
Basic Tile						
Balcony						
Curve 3x small						
Curve 2x large						
2x 2 in 1 left						
2x 2 in 1 right						
2 in 1 left with Curve						
2 in 1 right with Curve						
Straight 3x						
3 Curves, 2 cross						
Straight with 2 Curves						
Cross Straight and Curve						
Loop Curve						
Splinter, Vortex 3 in