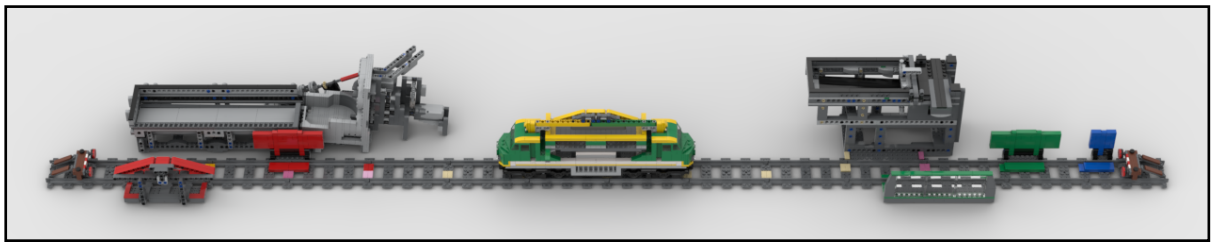
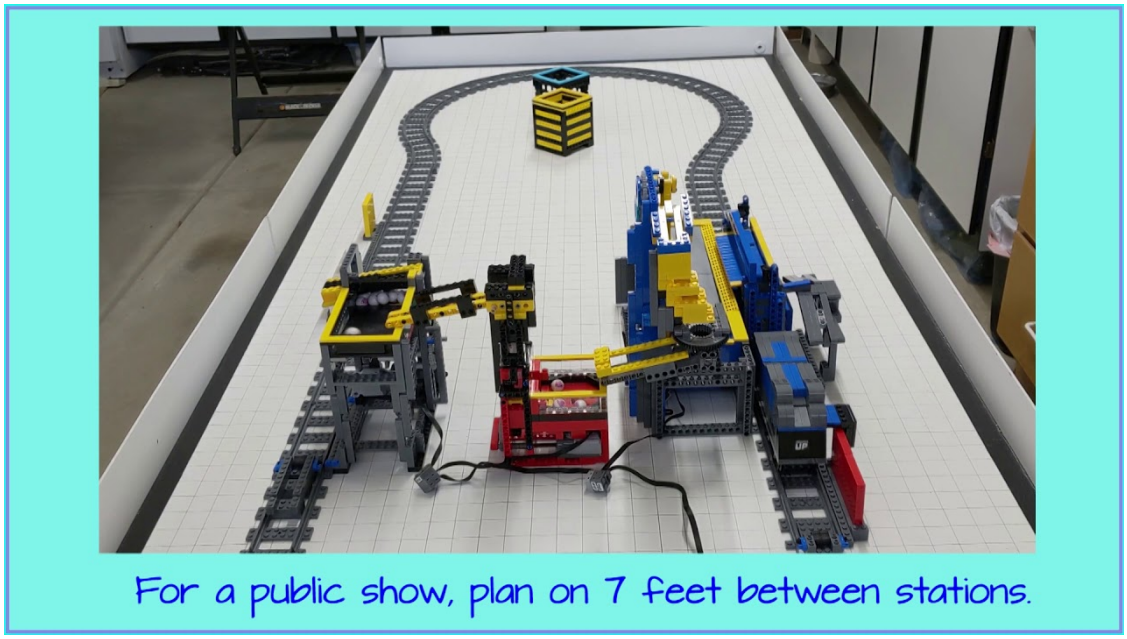


MOD08 Cargo Train

Cargo Train for GBC is a PoweredUp-based train for transporting balls between a *Loader* and an *Unloader* platform. The accompanying *Bricklink Studio* file can be used to recreate each of the components. Although these components are broken down further into logical submodels to guide the builder through recreation *within Studio*, there are no step-by-step instructions available.

Overview

This GBC is derived from Walt White's *PUP Train*. A video of Walt's original design is available on YouTube at



PUP Train Update https://www.youtube.com/watch?v=NFTAKkYS31A

Some of the larger changes from Walt's original design include

* train style based on an official Lego set;

* No requirement for a "curtain" behind the colour sensor

* No requirement for a motor to agitate the *Loader.*

Cargo Train Demo: https://www.youtube.com/watch?v=vCl_7afn0Bk

Configuration

For my purposes, and as laid out here, I am using the Cargo Train as a *ball return* for a standalone GBC display. As such, balls are collected from the right, and returned to the left. For use as a module in-line with a regular loop, the accompanying PoweredUp App code can be altered, changing the motor direction in order to carry the balls left to right, with the *Loader* and *Unloader* order similarly swapped on the table.

Cargo Train GBC as a ball return for the Ferris Wheel GBC

Components

Listed below is a breakdown of each component within the GBC, along with relevant notes you should be aware of.

Cargo Train - Version 1

The original design is based on the 2018 Cargo Train, Set 60198, currently still available and predicted to return mid-2022. This set provides the Hub, Motor, and basic train-styled parts required.

LEGO White Battery Box Powered Up Bluetooth HUB NO. 4 (28738)

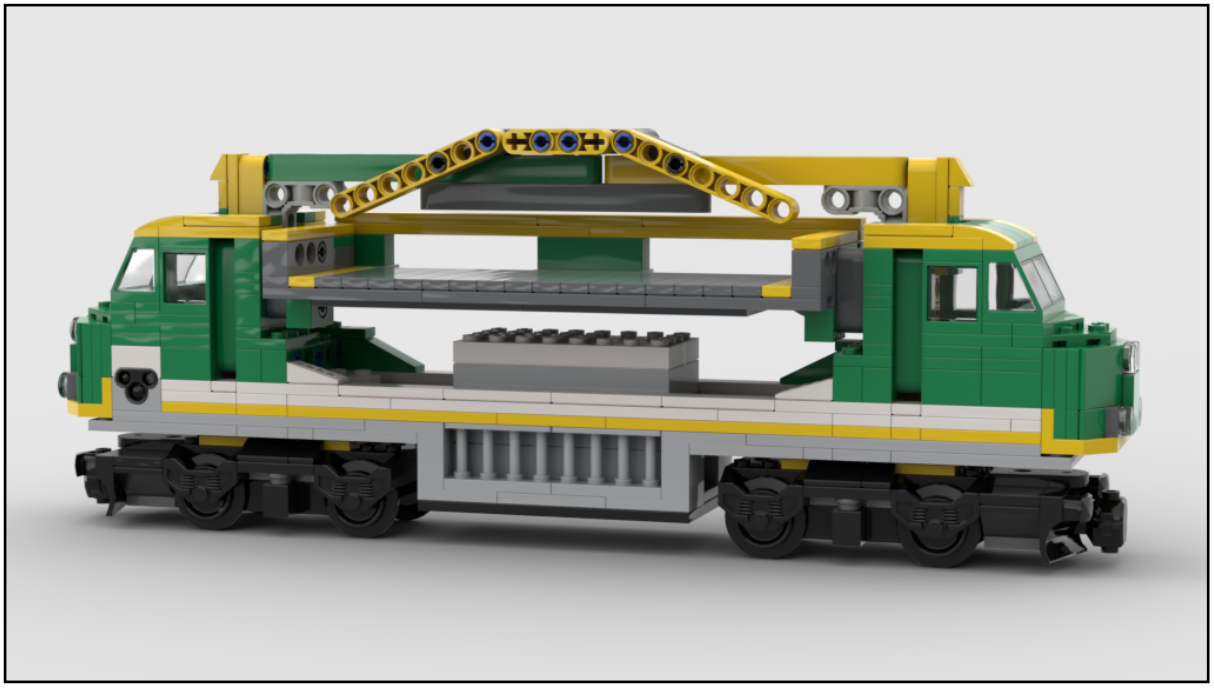
LEGO Black 9V Train Motor with Powered Up Attachment (28740)

This traditional-style of engine and carriage is completely functional, and may be preferred by train purists.

Cargo Train, version 1

Cargo Train - Version 2

The original train design was revised in order to make the system more efficient. Integrating the engine and tipper carriage together reduced the overall minimum length of track required, enabling a single 6ft table display. The additional reduction of weight (less parts, including removal of the steel balls) along with less friction (fewer wheels) provides increased battery life.

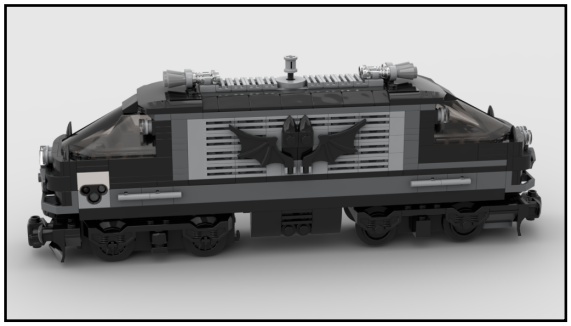


Cargo Train, version 2

Additional Train Designs

The *Studio* file also includes two alternate train designs, both derived from those above. Although these additional designs have not been physically built, they are functionally equivalent to the version 1 or 2 designs that have been tested, so should operate without issues.

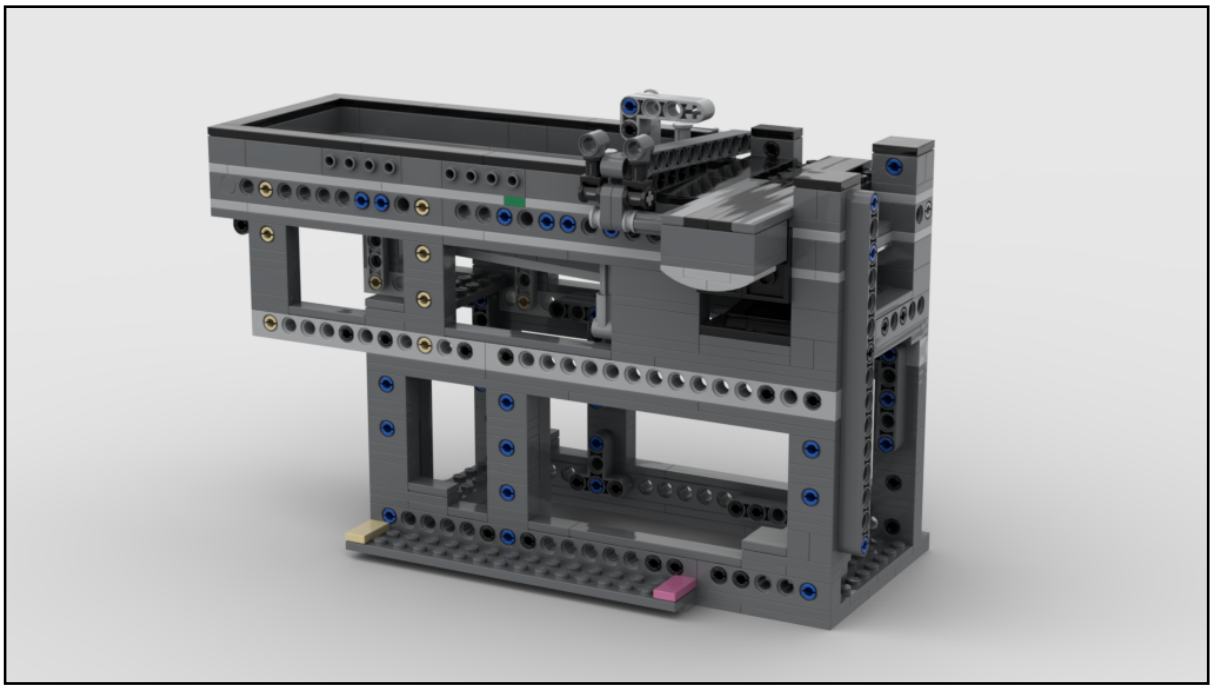
Aero-style modified Version 2



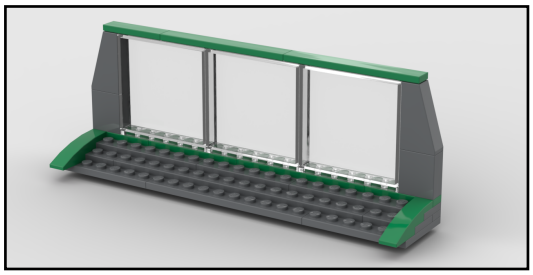
BatTrain modified Version 1

Loader

In order to load the balls onto the train, the input box on the *Loader* is significantly higher than the standard 10-brick height. As the train passes in front of the *Loader*, the gate on the dispensing ramp is raised. The connected lifter is used to agitate the sloping input box ramp. On a very few occasions, balls may not dispense, but upon a return visit by the train the agitation releases them onto the train. This infrequent failure to dispense balls is a well-considered trade-off to remove the need to motorise a constant agitation in the *Loader*.



GBC Train Loader

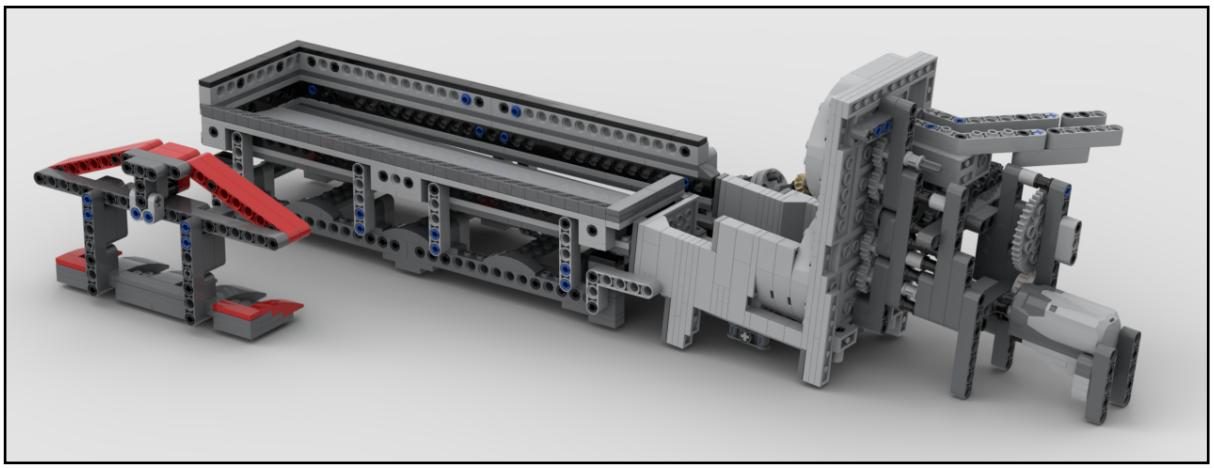


Spill Wall (optional)

As an optional accompanying component to the *Loader*, this green-trimmed transparent wall can be placed on the opposite side of the track from the *Loader*.

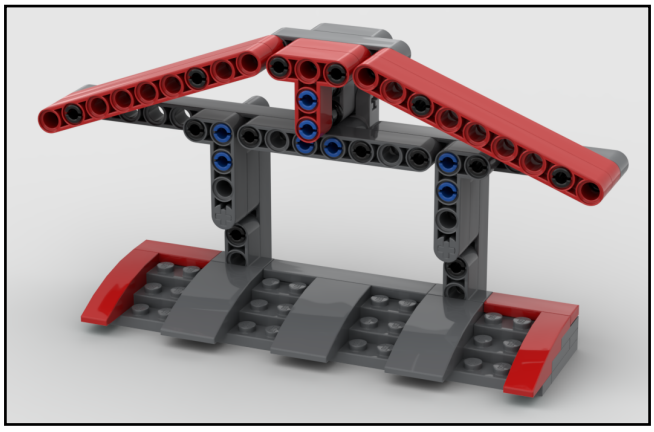
Under all the right circumstances, it is possible (though rare) for balls to spill out of the loader without being captured on the train. This can occur when there are far too many balls in the *Loader* when the train comes to collect them, and they don't all fit in the tipper. If this occurs frequently, the speed of the train or length of track between stations can be adjusted to unload the balls more often.

Unloading Components



Tipper

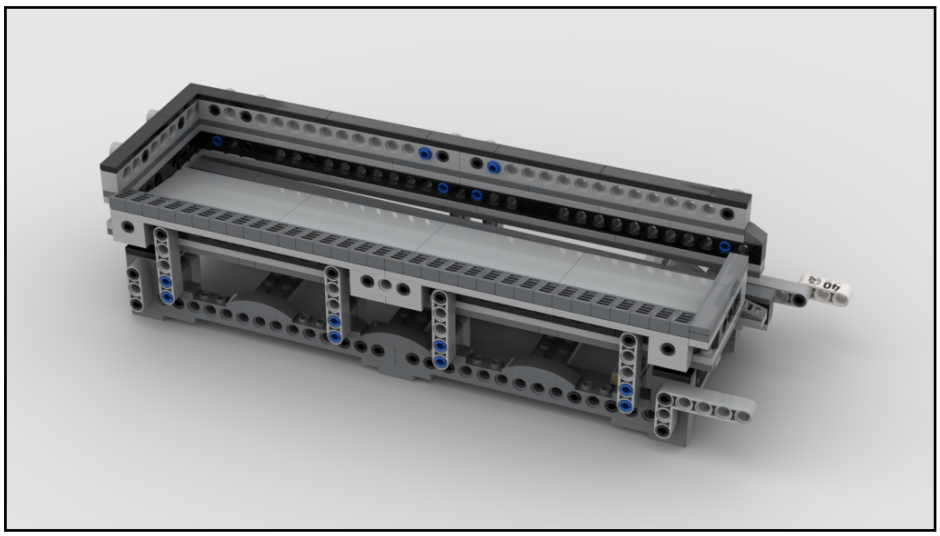
As the train approaches the *Unloader*, the protruding green arm will ride over this red *Tipper*, rotating the ball tray to empty the balls from the train.



Unloader Tipper

Platform

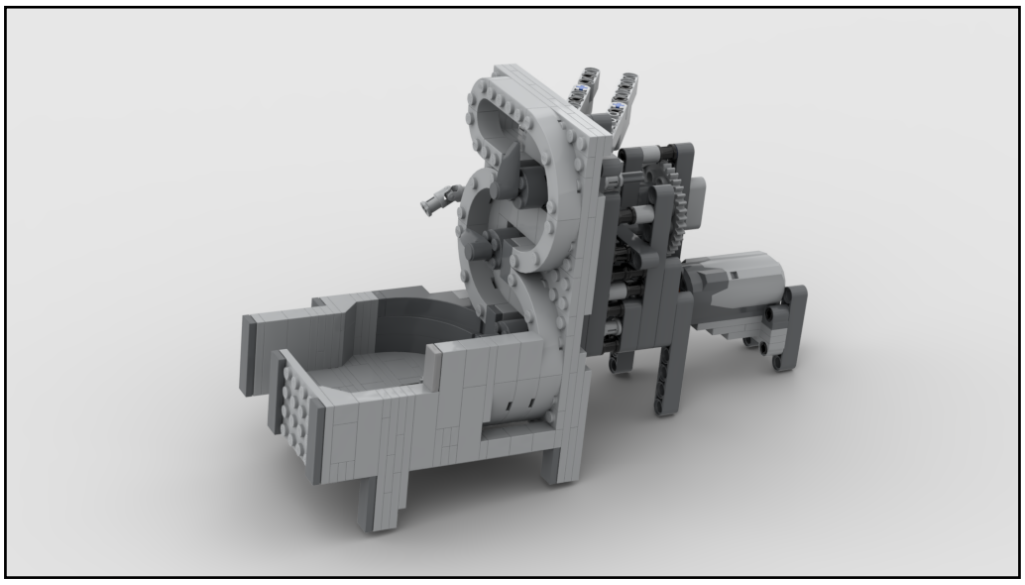
As balls are dumped out of the train, they will roll down to the rear of the *Platform*, where the beam at the rear is gently agitated to keep the balls flowing on their way out of the *Platform* and into the *Serpentine*.



Unloader Platform

Serpentine (modified)

As a result of the train geometry, the unloading *Platform* is lower than the standard 10-brick GBC requirement, so the balls are fed into a modified *Serpentine* module to raise them back up and feed them at standard height to the next module. This *Serpentine* variant is derived from Lawrie George's version, itself based upon Brian Alano's original design.



Unloader Serpentine

Signposts

Train behaviour is based upon the detection of coloured signposts along the track:

RED Indicates the train is at the *Unloader*, triggering a stop and return.

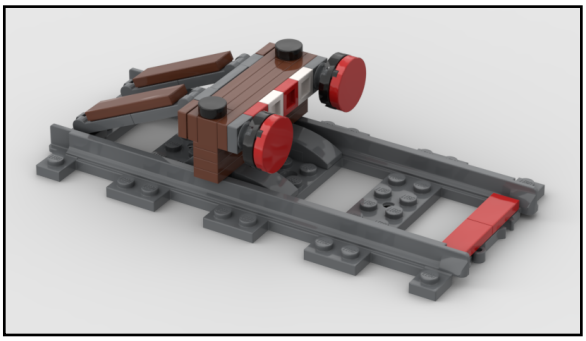
GREEN - The train is at the *Loader*. Stop and return.

BLUE - The RED sign wasn't detected. STOP and go back. This is an infrequent failure of the *colour sensor (train going too fast typically.)* This sign allows us to error-detect and respond automatically.

YELLOW - Not shown. In a configuration where the *Unloader* is at the end of the line, the *Buffer Stop* will stop the train with the *colour sensor* pointing directly at the *RED* sign, so a YELLOW counterpart to the BLUE sign is not necessary. An additional YELLOW sign can be constructed (and corresponding Powered Up code) to self-correct in a similar manner to the BLUE sign, if the unloader is not located at the end of the line.

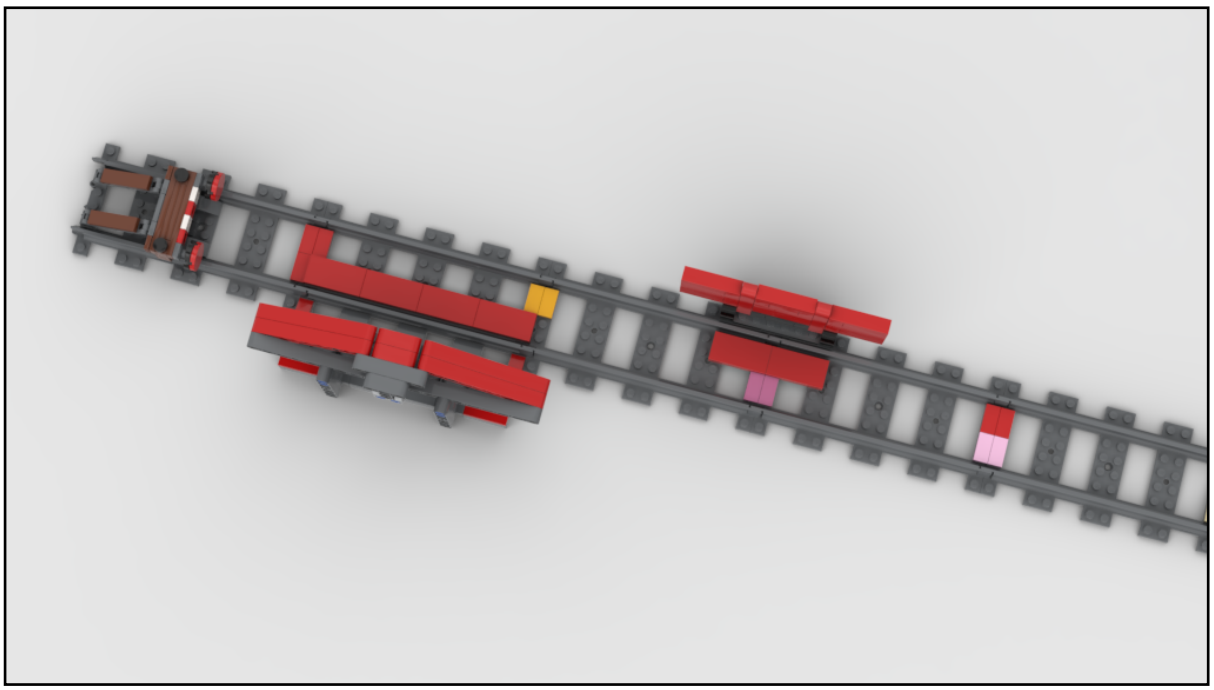
Buffer Stop

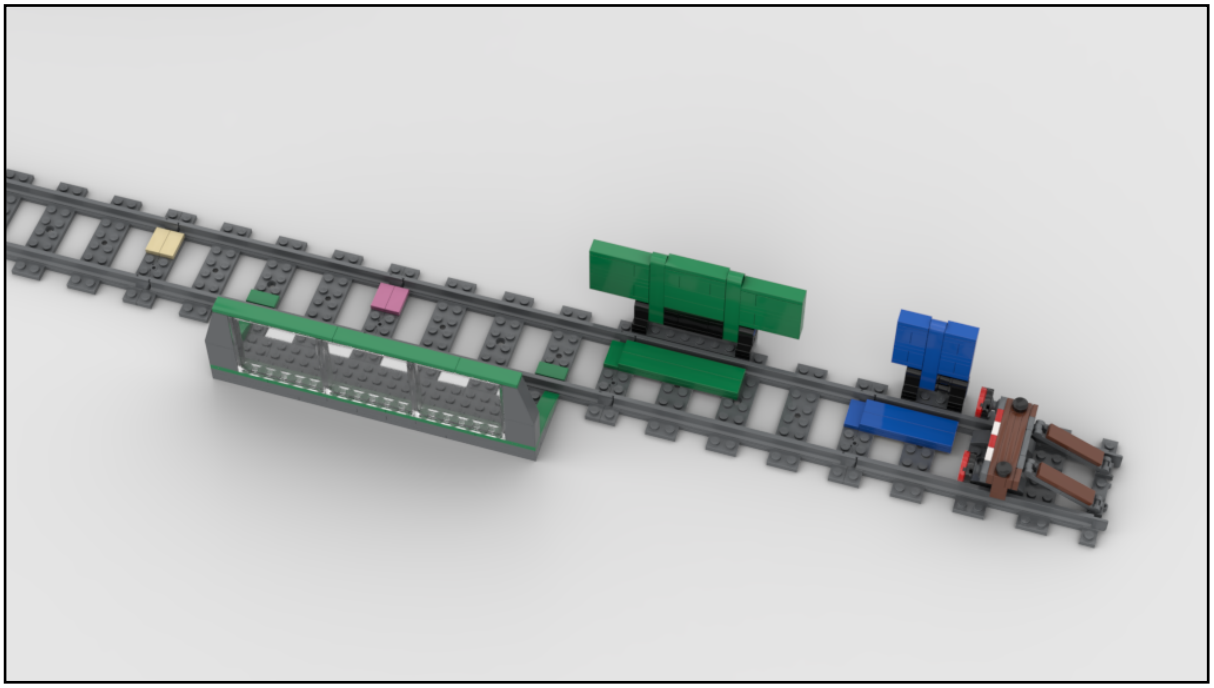
Located at the end of the train tracks, each *Buffer Stop* is strong enough to physically stop the train without derailing it. Apart from looking pretty cool, they provide a fallback in case of sensor failure.



Track Tiles

Located along the tracks, tiles of varying colours are used to provide visual cues when setting up the module, indicating which tracks connect to each other, and where track accessories such as sign posts, ball walls and ramps should be located. In between these two track ends shown below, a varying number of tracks can be connected, indicated by track segments with *Tan* tiles at each end.

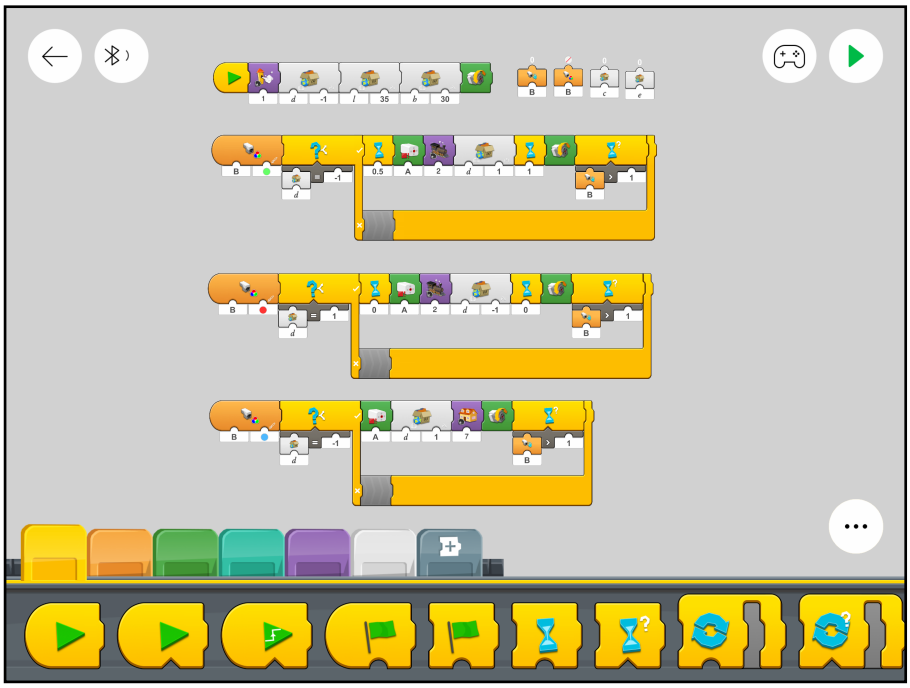




PoweredUp Source Code

Screenshots of code which controls the train is shown below, and subsequently followed by a brief explanation of each code block.

It should be noted that the *PoweredUp App* is particularly power hungry. For the purposes of running the app for multiple hours, your device should be connected to a power source capable of charging at a rate that doesn't allow the device battery to be depleted.



Cargo Train for GBC - Main Source Code

Cargo Train for GBC - Motor Control Subroutine Code

Values

Real-time display of the sensor readings along with display of some variables (*c* and *e*) used during development. Not required for normal operation.

Initialisation

This block is activated when the code is first started. It sets

* the train direction (*d*) to -1 (left-to-right)
* the motor speed (*l*) to 35
* the *no longer necessary* timer variable (*h*) to 30
* calls the *Motor Control Subroutine* which starts the train moving, in this case, toward the *GREEN* sign.

To run the train slower, lower the value of *(l)*. If the value is too low, the train will struggle to move when the battery runs low. Raising the value will run the train faster. If the value is too high the *Colour Sensor* will take too long to detect the coloured signs and drive right past them.

Green Block

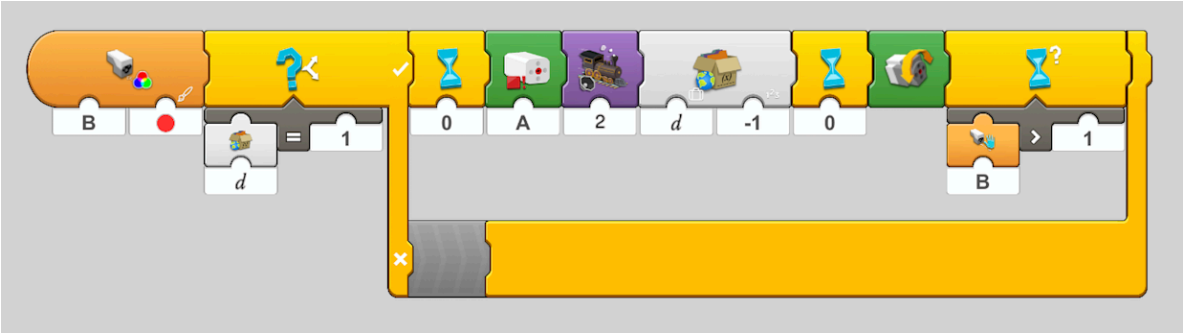
This code is activated when the *Colour Sensor* detects the *GREEN* sign.

If the train is traveling (d) *Left to Right (-1)*

* Wait 0.5 seconds.
* Stop the motor.
* Play a sound.
* Change the direction (*d*) so the next time the motor runs it will go in the opposite direction.
* Wait 1 second (more loading time)
* Call the *Motor Control Subroutine* to start the motor.
* Finally, wait until the *Colour Sensor* (which is also a *Distance Sensor*) determines there is nothing within “1” distance of the sensor. The result of this *waiting* is to avoid re-detecting this same sign until the now moving train has moved away from it.

If the train was not travelling *Left to Right*, then it is travelling toward the RED sign, so we don’t need to do anything. This ensures the train travelling from beyond the GREEN sign (ie, from the BLUE sign) towards the RED sign, is not interrupted.

Red Block



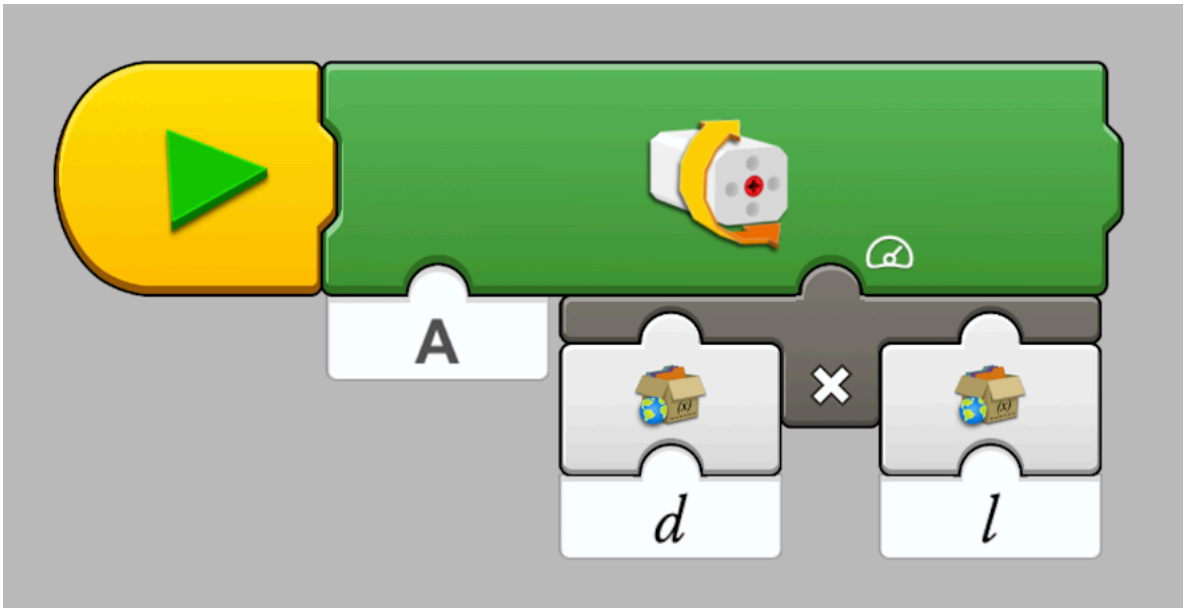
This code is activated when the *Colour Sensor* detects the *RED* sign.

If the train is traveling (d) *Right to Left (1) then similar to the Green Block,* we are going to stop the train, play a sound, change the direction (d to -1) and restart the motor. Again we wait until the distance sensor determines the train is no longer alongside this sign (B *value* > 1). The *Wait for 0 seconds* blocks allow fine tuning of how long the train should run past the sign before stopping the engine, and then before it starts moving again. The existing values are 0 as the sign posts have been positioned accordingly. The ability to *tweak* these values allows fine tuning of timing for different setups.

Blue Block

This code is activated when the *Colour Sensor* detects the *BLUE* sign. This only occurs if the sensor failed to detect the GREEN sign. Though rare, it is still possible, typically because the train is running too fast. This code does essentially the same as the *Green Block*, (turn the train around) with a different sound played.

Motor Control Subroutine

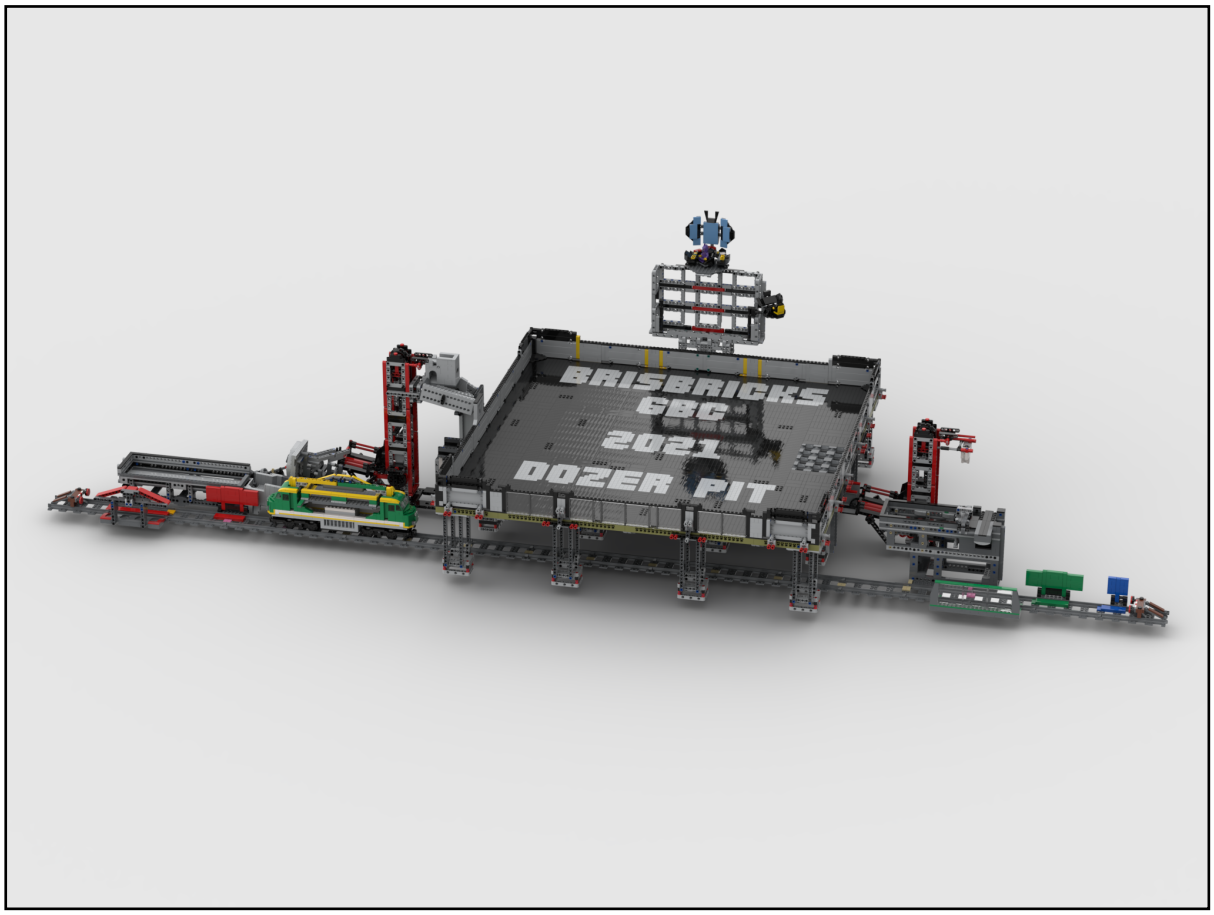


Start the motor, at speed (l) in direction (d). d=1 moves the train Right to Left. d= -1 moves the train Left to Right.

Further Notes

The positioning of the coloured signs in the accompanying *Bricklink Studio* file are suitable for the integrated *Version 2* of the train. If you choose to build *Version 1,* you will need additional track segments and need to adjust the positioning of the colour flags to stop the train at the right location to account for the extra length of the carriage.

Removing and attaching the motor and sensor cables to the hub when it is removed for battery changes can be fiddly for fat-fingered-fellas like me. I use long nose pliers to greatly simplify the process.



Cargo Train GBC as a ball return running underneath the Battle Arena Dozer Pit

Contact Information

Discord - *Great Ball Pit server*

https://discord.gg/YPwm9w7

eMail

rykfield@gmail.com

My GBC and other Lego Projects

http://eucalypt.com/lego/

Bricklink Studio

https://www.bricklink.com/v2/build/studio.page