Scripting "Junction Coal to Big Flats Power"

Using the enhanced TrainPlayer Programming Language (TPL)

This document is an explanation of the steps used to add a new set of TPL scripts to Eric Twickler's "Junction Coal to Big Flats Power" layout. It is intended that these step by step notes may prove helpful to users with no prior knowledge of scripting but who may now want to add their own scripts to any TrainPlayer layout.

Eric explained that his layout is designed for unit or block coal trains operating between the mine and the power plant. The cars are loaded one by one as they pass under the tipple, they cross the scale, add a caboose and head out for the Power Co. climbing an imaginary grade to Deer Pass. A day's work involves loading four trains at the mine and unloading two at each plant.

Downgrade from Deer Pass the train enters the power plant yard and heads for the appropriate siding, crossing the scale before going over the unloading trestle where each car is unloaded individually as it passes through the unloading shed.

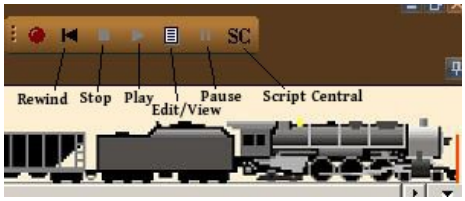
Returning trains must hold at one of three places to allow other trains to cross; Big Flats before the main line switch, either of the Deer Pass sidings, or the mine's lead track after uncoupling the caboose. Trains then back into the mine yard from the lead track passing under the tipple.

In scripting this layout we have sought to maintain slow running speeds, but we have set the trains to run at double the speed that Eric intended. The values used for these speeds can still be adjusted in the Master Script if required.

Preparations for Scripting TrainPlayer layouts with TPL

For writing or playing any TPL script the Script Toolbar is essential, if this is not already displayed on your screen you can activate it through the view menu.

In this project we were using some of the new advanced TPL commands and did not use the Red Script Recorder Button, the purpose of the other icons are labeled here.



Step One - We start with some simple Junction Actions to load and unload the Hoppers

Examination of the loading tipple on the layout track plan reveals one piece of hidden track under each loading bay.

To place a Junction Action on these track sections we need to split each of these pieces of track into two sections so that a new track junction sits dead center under the tipple.

Note: The current version of Junction Coal uses a trigger HT as the hoppers on this layout have been changed to the new 50px Default Steam Carset since this article was originally prepared in April.

RF August 2014

To create the first Junction Action, we selected the White Arrow pointer on the TrackLayer Toolbar (essential step), right clicked on the track at the center point of the top loading track (depicted with a red circle in this image) and selected "Create Action Here" from the context menu. This opens the Junction Action dialogue box depicted.

The drop down boxes were adjusted to set the conditions for triggering this action. In this example the engine will be ignored and the action will be triggered only by a car with an aar code of HM, and only if it is traveling West to East.

The top line in the "Take Action section" was initially filled with an automatically generated comment commencing with a single asterisk (*). We have edited this to commence with two asterisks (**) which also represents a comment; the difference being that text following a single asterisk is echoed to the Schedule Window whereas comments following a double asterisk are not. Comments help you remember what the script does when you look at it again.

load $x_car is a TPL Script instruction to identify the car crossing the junction and load it with its default load.

Any number of TPL script commands can be used in a Junction Action but here we only required two lines, a comment and a single line Command instruction to identify the car and load it with its default load.

Once we had entered this information we clicked Apply at the bottom of the dialogue to add the script to the layout (Note: clicking OK also adds a script to a layout and simultaneously closes the Junction Action dialogue).

At this stage we saved the layout and tested it by driving the top train under the tippler. Once we were satisfied that each car was automatically loading as it passed under the tippler, we returned the trains to their start positions by clicking "Rewind Script" on the Scripts Toolbar.

We have explained the purpose of Junction Scripts first as these are the quickest for a novice to get up and running, and for this layout it is the loading of the hoppers that comes first in the sequence.

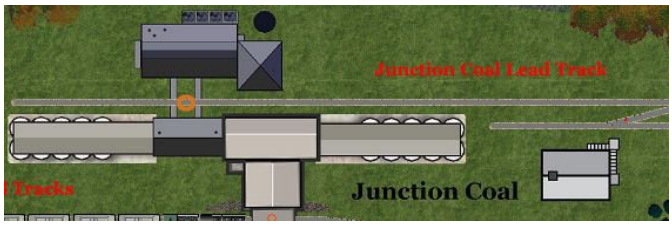
The same process was then used to add Junction Action Scripts to the other three loading tracks. And to the two unloading trestles at the Power Plants. The only difference at the Power Plants being that for unloading we changed the command to **unload $x_car** and changed the right hand drop box to read E>W (East to West).

Each of the sections of hidden track that we split needed to be adjusted as only one half of the track remained hidden after splitting the section. To correct this we needed to right click on the visible section and set it as Hidden.

Once this initial set of Junction Actions were in place, we needed to ensure the trains were all back in their starting positions (Rewind Scripts from the Script Toolbar), and again we Saved the file before moving on to the next step.

We also made an additional backup copy of the layout in case we needed to backtrack later. At this stage we had a fully functioning layout. The trains could be selected and driven from the controller but all cars loaded themselves automatically as they passed under the tippler, and unloaded as they passed over the trestles at the Power Plants.

Step Two - Additional track adjustments that were needed before moving on to prepare the Train Scripts

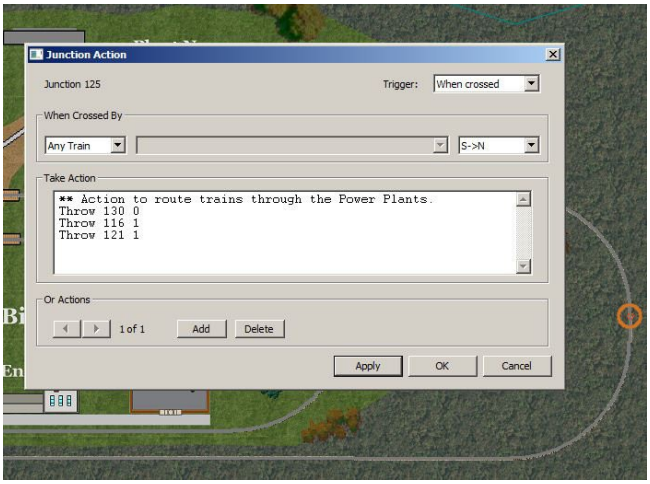


Again we activated the white arrow pointer on the TrackLayer toolbar, performed a right click at the point shown with a red circle in this image on the Lead Track at Junction Coal, and selected the option to "Split Track Here" from the context menu.

The connection point this created was required for future use in the scripts as a reference point to stop the engine for servicing on its return to the mine.

We also used this same technique to split the track dead center on the Scale at Junction Coal.

Our second adjustment was to add a further Junction Action on the approach curve to Big Flats Power.



To do this we identified an existing junction J125 on the approach to Big Flats Power (see red circle on image), we right clicked on this junction and selected "Action ..." from the context menu.

This opened a further blank Junction Action edit dialogue box. We changed the setting of the right hand drop box from "Any Direction" to read "S>N" and we typed in four lines of text into the main part of the window as a basic starting script. (see image)

This was a temporary instruction to ensure that all trains entering Big Flats would take the same route through the Power Plant for unloading.

The three commands are simply instructions to throw switches to provide a fixed route through Plant 2

This script was to be modified later to become an important part of the sequence. The TPL code would be changed to ensure that every alternate train would be routed through the other Power Plant, and that unloaded trains were held back at the Power Plant until the next loaded train had arrived (see Step Six below).

Once again we clicked Rewind on the Script Toolbar and Saved our progress. The layout still functioned from the Train Controller but now we were ready to start adding our Train Scripts.

Step Three - Adding the basic Train Scripts

We commenced this stage by selecting the the engine on the top loading track, and clicking the Edit/View icon on the Script Toolbar. This opened the Script Edit Dialogue box for this train. The dialogue that opened contained some default text in the form of a comment. We could have changed this, but on this occasion we only added an additional asterisk to the first two lines (i.e. a single asterisk is a comment that is echoed to the Schedule Window, whereas two asterisks represent a comment that is hidden from the Schedule Window). See image overleaf.

Next we added a few more lines of text to get the first Train Script started, together with some further comments to explain what the script would be doing.

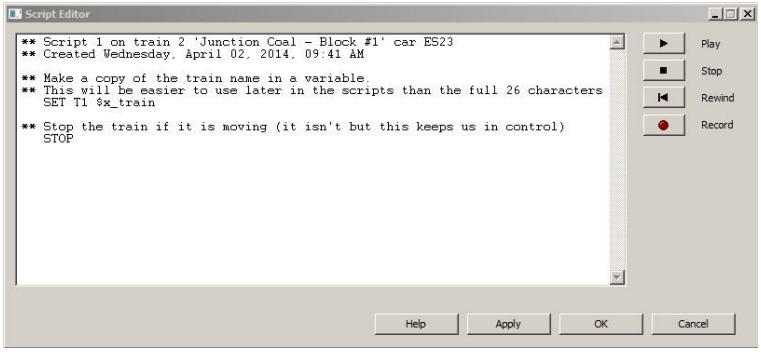
At this stage there were only two active lines in the script.

```
SET T1 $x_train

STOP
```

Indentation is optional, as are comments, but basic commenting does aid readability.

When the script looked like this, we clicked the Apply and/or OK button.



We then repeated this procedure for the other three trains waiting to be loaded. To do this we needed to change the reference in the SET command from T1 to read T2, T3 and T4 in each of the other scripts.

Then it was time to Save the layout again, if we had moved any trains then we would have needed to click the Rewind Button on the Script Toolbar before saving.

At this stage we had four train scripts that didn't do very much other than record the train names and ensure the trains were standing still. But we did have all the basic building blocks in place for fully automating this layout. We could still drive our trains using the Train Controller and the Junction Actions, that were introduced in Step One, would still load and unload the hoppers automatically.

Take Time Out to examine Script Central

Before moving on to explain the next stage, this might be a good time to draw your attention to the **Script Central** dialogue, this is where most of the subsequent work on our scripting was done. The purpose of Script Central will be easier to understand now that we have covered the preparation of Junction Scripts and Basic Train Scripts.

If you are looking at the scripted "Junction Coal to Big Flats Power" layout, start by clicking the SC icon on the Script Toolbar to open a new dialogue which has several separate tabs which we will take a very quick tour of for now.

The Scripts Tab is the active tab when the dialogue opens. Here you will see a list of the four Train Scripts which we started in Step Three, but they will now contain a full set of instructions which we will be explaining in the following sections. Click on any of these in the left hand list and you will see the appropriate script text in the right hand pane. Scripts can be edited directly in the right hand pane, or you can double click an item on the list to adjust it in the Script Editor, but for now we are just checking to see that the scripts are present.

The Junction Actions Tab is next, click on this to see a list of the Junction Action Scripts we introduced in Step One. Again clicking on a Script in the list will show its contents in the right hand pane. These scripts can also be edited directly in this pane, but if you need to change the trigger conditions you will have to double click the item in the list and make the changes to the drop down boxes using the original Junction Action editor.

The Subroutines Tab presents a similar list of external routines that may or may not already exist on your TrainPlayer installation. Subroutines are scripts that are stored in standard windows text files and can be called up from within any embedded script type using the CALL command. We have not used any subroutines in scripting this example.

The Reference Tab provides a detailed list of all the commands available to you for writing scripts in TPL. Just take a quick look for now but don't worry at this stage about trying to understand them. TPL is very powerful but it can also function well at a lower level, with a small Command set, as shown in this example layout.

The Settings Tab is for saving the positions of your trains and switches, overriding the defaults which are already set to the positions when the file was first loaded. There is also a facility to save your train and switch positions to a separate file, and to reload them. We will not be using the Settings Tab in this exercise.

Now you have seen what you can access from Script Central we are ready to continue explaining how we scripted this project.

Step Four - Adding a Master Script

A Master Script is different from any other type of script in that it is not attached to any train or to any junction. What makes it special is that it will execute automatically as soon as the layout is opened. A layout does not have to include a Master Script, but if it does in can only have one of them.

To insert our Master Script we reopened Script Central by clicking the SC button on the Script Toolbar, selected the **Scripts** tab (if not already selected). Right clicked on the Scripts list and selected the option to "Add Master Script".

This immediately added a new script to our list, including a default opening comment which we chose to leave in place although we did add an additional asterisk to prevent this line being passed to the Schedule Window when the script was run. We updated the Script by clicking Apply.

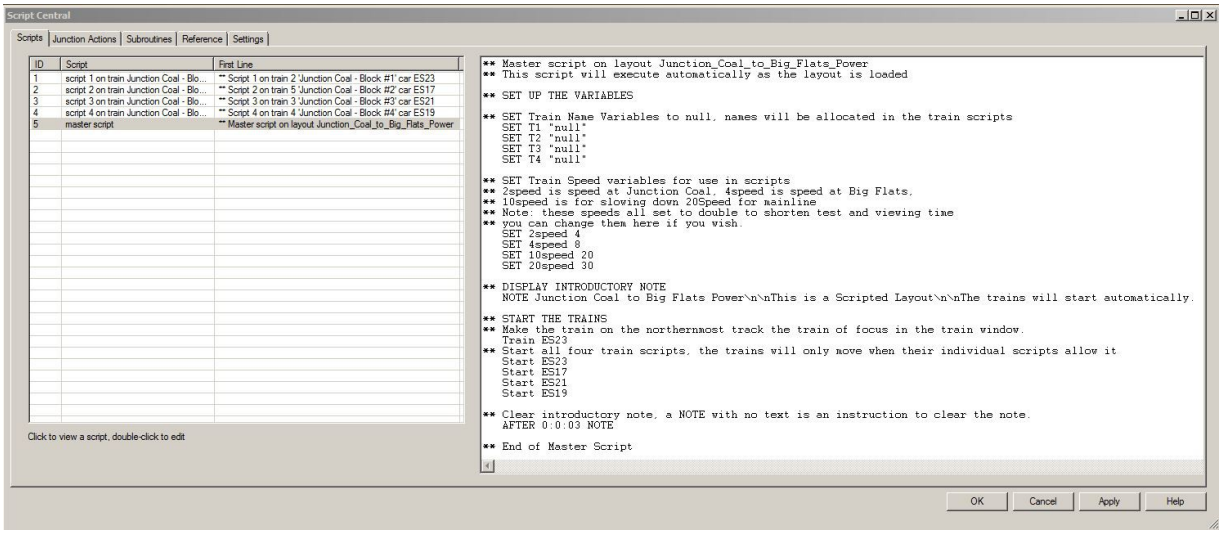
On this layout we used our Master Script to automatically perform three specific tasks as the layout opens:

1. Set Up the Variables
2. Display an Introductory Note
3. Start the Train Scripts running

The Master Script can also be used to define specific Procedures which we can CALL repeatedly from within our other Scripts like an extension to the language. These are very powerful but are not needed in this introduction.

Here is a copy of our completed Master Script, this has been extensively commented to explain exactly what each command is doing. Remember when looking at this that any line starting with ** is merely a comment and is not essential to the functioning of the script. The other lines (indented only for ease of reading) are all that is necessary for the script to function. Commenting scripts is good practice as it helps you to follow your own logic.

The variables T1 to T4, and the four speed variables are simply repositories for storing values that can be used elsewhere in the Scripts. We chose to Set the speed values here where they can be easily amended if we want to change them without having to seek out every occurrence of their use.



Once again we saved our file before moving on to Step Five. If we had moved any trains we would have needed to click the Rewind Button before saving. The next time we opened our file this Master Script executed automatically to set the variables, display the introductory note and start all four of the train scripts running.

Unfortunately we didn't immediately see the benefit of this because the Train Scripts we prepared in Step Two only Set a name in a variable and finish with a STOP command. So next we had a little more work to do with the Train Scripts.

Step Five - Completing the first Train Script

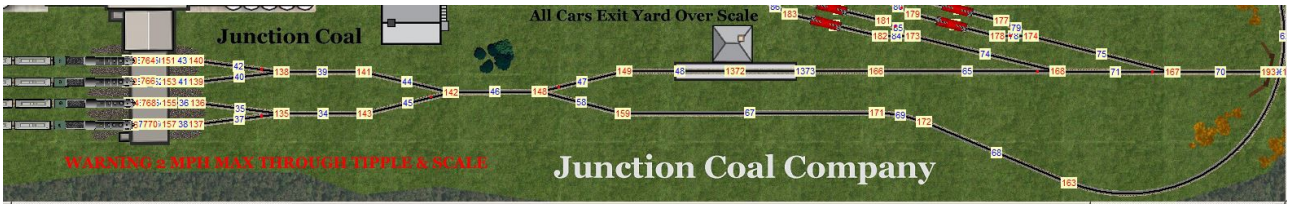
We started this step by opening the layout, or by selecting "Revert to Saved" from the File Menu if it is already open.

Because this layout now has a Master Script it is no longer sufficient just to rewind the scripts (because Rewind Scripts does NOT rewind the Master Script). In many situations the Train Scripts are dependent on settings made in the Master Script, and unless the Master Script is also rewound the Train Scripts could fail. So as the layout included a Master Script we performed a "Revert to Saved" from the file menu (instead of Rewind from the Scripts Toolbar) to ensure everything was positioned for the scripts to run properly.

When we clicked "Revert to Saved" our layout reopened and the Pop Up Note we included in the Master Script was displayed. However we were able to ignore this initially because our Train Scripts were still not set up to run the trains.

The next step was to add more TPL commands to our first Train Script to start the train moving under the tippler, route it across the scale, proceed to the end of the yard, and reverse back to collect a Caboose for the onward journey.

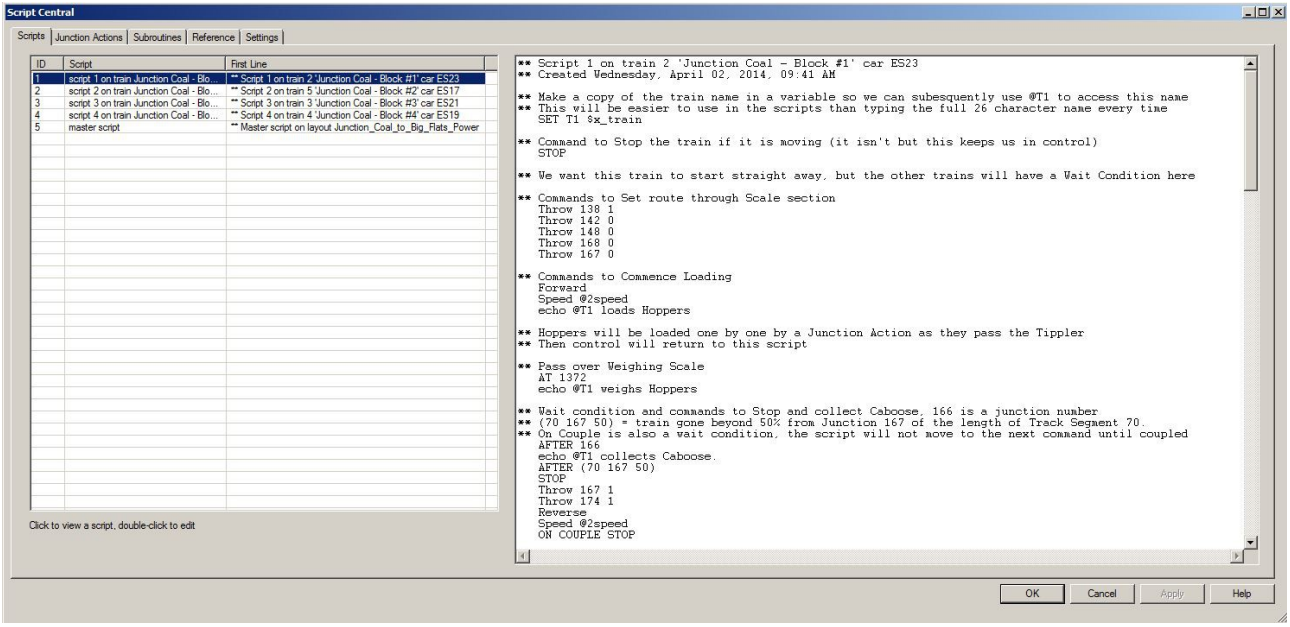
Before we could do this we needed to identify some switch numbers from the track diagram. So we used the Tools Menu and select "Show Numbers" to obtain a view that resembles this.



We then examined the switches this train would cross before it could reverse for its Caboose, and as the diagram shows we needed to set a route through Switches 138, 142, 148, 168, 167 and then back through 167 and 174. Each required switch position was obtained by hovering the mouse pointer over the switch and noting its current value from the tool-tip, we need to observe the current visual position to decide if we want the current value or the alternative value in our script.

Next we opened Script Central (by clicking the SC icon on the Scripts Toolbar) and selected the entry for Script 1.

We then typed into the right hand pane the TPL text to complete this first movement, the resulting script is depicted in this image from Script Central.



Remember that the lines commencing with ** are merely comments to aid in reading the script, the indentations are optional but all indented lines are TPL instructions for the train to follow. @ means -Extract the value you have already stored in this variable. If you need further information to help you understand the Commands used in this code you can look up the specific item in the Script Central Reference Tab.

It is important to be aware that a Wait Condition such as AT, AFTER or ON COUPLE refers to making the Script wait before moving on to the next command. The train will not wait unless an appropriate STOP command is in the Script.

We then continued to use this same technique for writing the rest of the Train Script for this first train, not a job for the faint hearted but very satisfying when it all starts to come together. While writing this code we periodically clicked Apply at the bottom of the Scripts Tab.

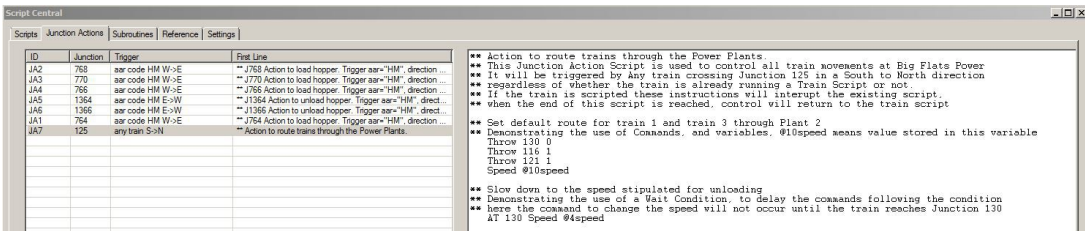
We were able to Save our progress at any time so long as we first selected "Rewind Script" from the Script Toolbar to reposition the trains before saving, and then took the option from the File Menu to "Revert to Saved" to ensure we were also rerunning the Master Script.

The completed TPL Script Code for the first train can now be seen in Script Central, this is heavily commented in an effort to explain exactly what the code is doing, but if anything is not cleared the instruction can be looked up on the Reference Tab. *A complete copy of Script 1, fully commented is attached as Appendix A to this document.*

When the Script was completed we Clicked Apply, Rewound the Script, Saved the Layout and used Revert to Save to test it. Once we were satisfied we were ready to move on to the next step which was to update the basic Junction Action Script at Big Flats before completing the other three Train Scripts.

<u>Step Six - Completing the Junction Action Script for Big Flats</u>

You will recall that back in Step Two we introduced an additional Junction Action Script at Junction 125 to control the passage of trains through Big Flats. Our next step was to make substantial alterations to this Script to identify which train had picked up the script and to control its movements accordingly.

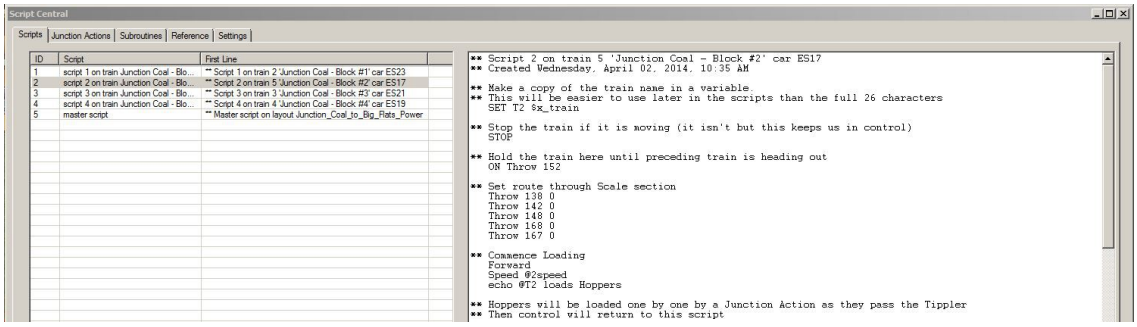


This required us to open Script Central, select the Junction Actions tab, highlight the Script for Junction 125, and make our changes in the right hand pane of the dialogue box. *A commented copy of this full script is included in Appendix B of this document.*

With this Script completed we tested, Rewound the Train Scripts, Saved the File and selected Revert to Save before proceeding with the remaining Train Scripts.

Step Seven - Completing the remaining Train Scripts

Essentially this step was just a matter of repeating what we had done for the first Train Script (Step Five above), but we needed to bear in mind that for each of these three additional scripts the route taken and references to the Train ID were slightly different in each case.



The first major difference between these three Scripts and Script 1 is that we didn't want these trains to start running immediately. This was achieved by inserting an ON THROW wait condition immediately after the STOP command that appeared in the original basic script. In this example for Script 2, the Junction 152 is not a switch, it is the track connection point immediately ahead of the engine at the point where the track joins the hidden section under the tippler. We were taking advantage of the fact that TPL allows us to use a command in another script to throw this Junction even though there is no turnout at this point. You can see this command to start this train if you examine Train Script 1 (appendix A) where the Command to Throw 152 appears immediately after the Caboose is coupled.

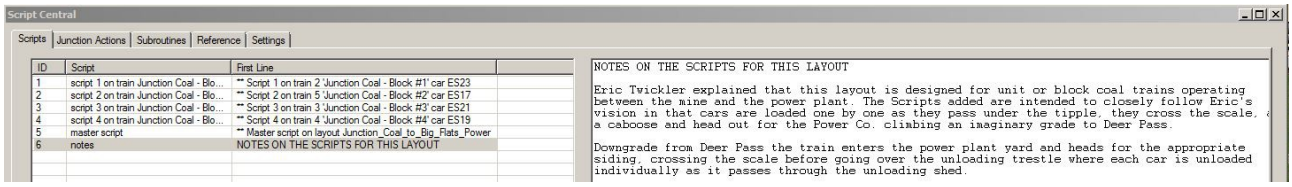
These additional Train Scripts were put together quite quickly by Copying and Pasting text from one script to another but great care was needed in respect of modifying the switch settings, train references etc. (So all references to @T1 in Script 1, needed to be @T2 in Script 2, @T3 in Script 3 and @T4 in Script 4.)

Repeated testing was used to identify and correct any errors. When everything was functioning properly the Trains were once again rewound and the layout saved.

Step Eight - Adding a Note about the layout to the Scripts Tab in Script Central

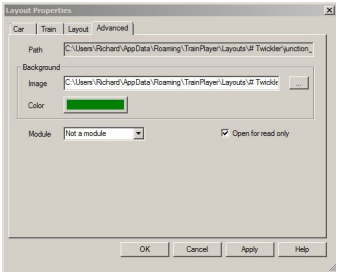
This is an optional step that we took to take advantage of the fact that Script Central allows you to store Notes of any kind under the Scripts Tab of the Script Central dialogue.

Adding a Note uses exactly the same procedure as adding a Master Script. We selected the Scripts Tab in Script Central, right clicked on the list of scripts, and selected "Add Note" from the context menu. We used this note to provide a quick summary about the scripts we had added to this layout.



Once again the layout was saved after ensuring the trains were in the correct starting positions.

Step Nine - Protecting the Scripts from accidental damage by enthusiastic users



Once we were satisfied that the scripting of this layout was complete we selected Layout Properties from the context menu after right clicking on the layout. We selected the Advanced Tab and put a tick in the box which states "Open for Read Only". Then we saved the file again.

When this file is reloaded it is set for Read Only so that if a user accidentally clicks "Yes" when prompted to Save on closing the file, this will not overwrite the original, but offer an incremental file name in the Save As dialogue.

That concludes our explanation of how the Scripts were added to Eric Twickler's "Junction Coal to Big Flats Power"

Richard Fletcher. April 2014

Appendix A - Script 1, the Train Script used to control the first Block train to traverse the layout

```
** Script 1 on train 2 'Junction Coal -
Block #1' car ES23
** Created Wednesday, April 02, 2014, 09:41 AM

** Make a copy of the train name in a variable so we can subsequently use @T1 to access this name
** This will be easier to use in the scripts than typing the full 26 character name every time
   SET T1 $x_train

** Command to Stop the train if it is moving (it isn't but this keeps us in control)
   STOP

** We want this train to start straight away, but the other trains will have a Wait Condition here

** Commands to Set route through Scale section
   Throw 138 1
   Throw 142 0
   Throw 148 0
   Throw 168 0
   Throw 167 0

** Commands to Commence Loading
   Forward
   Speed @2speed
   echo @T1 loads Hoppers

** Hoppers will be loaded one by one by a Junction Action as they pass the Tippler
** Then control will return to this script

** Pass over Weighing Scale
   AT 1372
   echo @T1 weighs Hoppers

** Wait condition and commands to Stop and collect Caboose, 166 is a junction number
** (70 167 50) = train gone beyond 50% from Junction 167 of the length of Track Segment 70.
** On Couple is also a wait condition, the script will not move to the next command until coupled
   AFTER 166
   echo @T1 collects Caboose.
   AFTER (70 167 50)
   STOP
   Throw 167 1
   Throw 174 1
   Reverse
   Speed @2speed
   ON COUPLE STOP

** Release second train to start loading, this throws a non existent switch, 152 is a track connection
** with no junction, however Train Script 2 is waiting for this Throw to occur to release its train
   Throw 152 0

** Head Out from Junction Coal along Main Line
   FORWARD
   Horn
   Speed @2speed

** Report departure of train to Schedule Window
   echo @T1 heads for Deer Pass, Southbound.

** Adjust speed for mainline
   AFTER 167
   Throw 167 0
   Speed @4speed
   AFTER 170
   Speed @10speed

** Approaching Deer Pass Southbound, sound Horn train will not stop.
** Train is authorized to continue to Big Flats
   AT 110
   Horn
   Throw 108 1
   Throw 111 1
   AT 108 Speed @10speed
   echo @T1 at Deer Pass, Southbound
   AFTER 111 Speed @20speed
   echo @T1 heads for Big Flats Power

** Approaching Big Flats Power when we reach Junction 114 sound Horn
   AT 114
   Horn

** All switching at Big Flats is controlled by a separate Junction Action script
** Which is located on the approach curve at Junction 125
** This script will resume control when the Northbound train is approaching Deer Pass
```

```
** Approaching Deer Pass Northbound, sound Horn, train will take the siding and wait
** for the Southbound train to pass on the main line.
  AT 112
  Throw 111 0
  Speed @10speed
  AT 111
  Speed @4speed
  AFTER 111
  Speed @2speed
  AFTER 129 STOP
  echo @T1 at Deer Pass, Northbound, awaiting Southbound train
  Throw 111 1

** Wait until the Southbound train passes and throws 126 then Head out North for Junction Coal
  On Throw 126 0
  Throw 108 0
  Forward
  Horn
  Speed @4speed
  echo @T1 heads for Junction Coal.
  AFTER 108
  Speed @10speed
  Throw 108 1
  AFTER 110
  Speed @20speed

** Approaching JC
  AT 107
  Horn
  Speed @10speed

** Release the fourth and final train to start loading as we pass this point
** The final train has been held back because track ahead was all occupied by first three trains
  Throw 137

** Entering JC
  AT 170
  Speed @4speed

** Report Arrival of train to Schedule Window
  echo @T1 at Junction Coal

** Set Route into JC Lead Track
  Throw 167 1
  Throw 174 1
  Throw 175 1
  Throw 195 0
  Throw 160 1
  Throw 185 1

** Set down Caboose
  After (77 177 50) Stop
  Uncouple 12
  Forward
  Speed @2speed

** Service engine before reversing to yard, using the extra connection point we added as a reference
  AT 1960 STOP
  AFTER 0:0:10

** Reverse to Yard Tracks
  Throw 185 1
  Throw 161 0
  Throw 160 0
  Reverse
  Horn
  Speed @4speed
  AFTER 162
  Throw 161 1
  Throw 160 1
  AT 171
  Throw 148 1
  Throw 142 0
  Throw 138 1
  AFTER 150
  AFTER 0:0:02 STOP

** Switch focus of train window to the train running Script 4
  train @T4

** End of Train Script for Junction Coal - Block #1
```

Scripts for the second, third and fourth trains are similar to this one, but the switch settings, train references and Wait instructions all differ slightly. The individual scripts can be examined in Script Central when the layout is loaded.

Appendix B - The Junction Action Script used to control all trains passing through Big Flats

```
** Action to route trains through the Power Plants.
** This Junction Action Script is used to control all train movements at Big Flats Power
** It will be triggered by Any train crossing Junction 125 in a South to North direction
** regardless of whether the train is already running a Train Script or not.
** If the train is scripted these instructions will interrupt the existing script,
** when the end of this script is reached, control will return to the train script

** Set default route for train 1 and train 3 through Plant 2
** Demonstrating the use of Commands, and variables, @10speed means value stored in this variable
  Throw 130 0
  Throw 116 1
  Throw 121 1
  Speed @10speed

** Slow down to the speed stipulated for unloading
** Demonstrating the use of a Wait Condition, to delay the commands following the condition
** here the command to change the speed will not occur until the train reaches Junction 130
  AT 130 Speed @4speed

** Report location of train to Schedule Window, using an ECHO statement
** $x_train is the train that has loaded this script which is used by all four trains
  echo $x_train unloads at Big Flats Power.

** Set alternate route for train 2 and train 4 through Plant 1
** Demonstrating use of a simple conditional statement, a system function and a variable value
** If (the train name detected is the same as the train name stored in the variable T2) then do this
  IF ($x_train=@T2)
     Throw 116 0
     Throw 121 0
  ELSEIF ($x_train=@T4)
     Throw 116 0
     Throw 121 0
  ENDIF

** Throw 131 to release any other train already waiting to return to Junction Coal
** If there is a train waiting it will already be running its own copy of this same script
  AFTER 130
  Throw 130 1
  Throw 131 0

** Trains will be unloaded by Junction Action scripts located on the Trestle
** i.e. Control will pass to another Junction Action Script before returning to this script
** Encountering another JA within any script is similar to calling a separate subroutine

** On return to this script, report to Schedule Window that Unloading is Completed

** Adjust speed within yard after unloading
** Demonstrating more wait conditions, commands and taking data from variables
  AFTER 121
  echo $x_train at Big Flats Power, unloading completed.
  speed @10speed
  AT 134 speed @4speed
  AT 133 speed @2speed
  AT 132 STOP

** If this is not the last train wait for arrival of next train before departing
  IF ($x_train<>@T4)
    ON Throw 131
  ENDIF

** Depart Big Flats Power for Deer Pass
  Forward
  Horn
  Throw 130 1
  Speed @2speed

** Report departure of train to Schedule Window, using an ECHO statement
  echo $x_train heads for Deer Pass, Northbound.

** Adjust speed to mainline speed
  AT 131 speed @4speed
  AT 124 speed @10speed
  AFTER 124 Throw 130 0
  Speed @20speed

** Script ends, control is returned to the calling train script at Junction 112
** If the train was not running a script it will continue at the designated speed until you stop it.
```