Train Status Client User Manual

Abstract

This document is a user manual for the Train Status Client program. The Train Status Client is an application that listens to changes in a model railroad layout controlled by the Computer Automated Traffic System (CATS) and summarizes the status of trains in a tabular form.

1. Introduction

The **Train Status Client** (TSC) application pushes the suite of Computer Automated Traffic System (**CATS**) applications further into model railroad operations. **CATS**, itself, focuses primarily on dispatching trains. It also includes some operations features (for example, crew assignment). In the prototype world, railroads have departments and personnel for providing operations support. They do the behind the scenes legwork of deciding what power to put on a train, what crew to run the train, when a train leaves, and even what cars go into a train. Except for switch lists, many model railroads completely ignore these activities, handle them on an ad-hoc basis, or fold them into the dispatcher job.

Though **CATS** focuses on dispatching trains, it does include some of these operations activities. For example, **CATS** contains capabilities of describing jobs (operator roles), crew assignments, hours on the job monitoring, and other operations oriented tasks not normally done by a prototype dispatcher. On a small to medium sized layout, the dispatcher may have time to handle the operations bookkeeping. But, on larger layouts, the dispatcher is often too busy orchestrating train movements. On those larger layouts, a dedicated job often manages the operations tasks. TSC is written to provide assistance in the operations tasks not directly involved with moving trains, but are affected by trains moving. As will be seen, it accepts information from **CATS** and formats it in a tabular form. TSC runs on another computer (or multiple computers, if desired), networked to the **CATS** computer.

TSC is an application for monitoring the status of trains on a model railroad. It is also a model of a prototype train status application. Most good models have an example to inspire them. This is the inspiration for TSC:

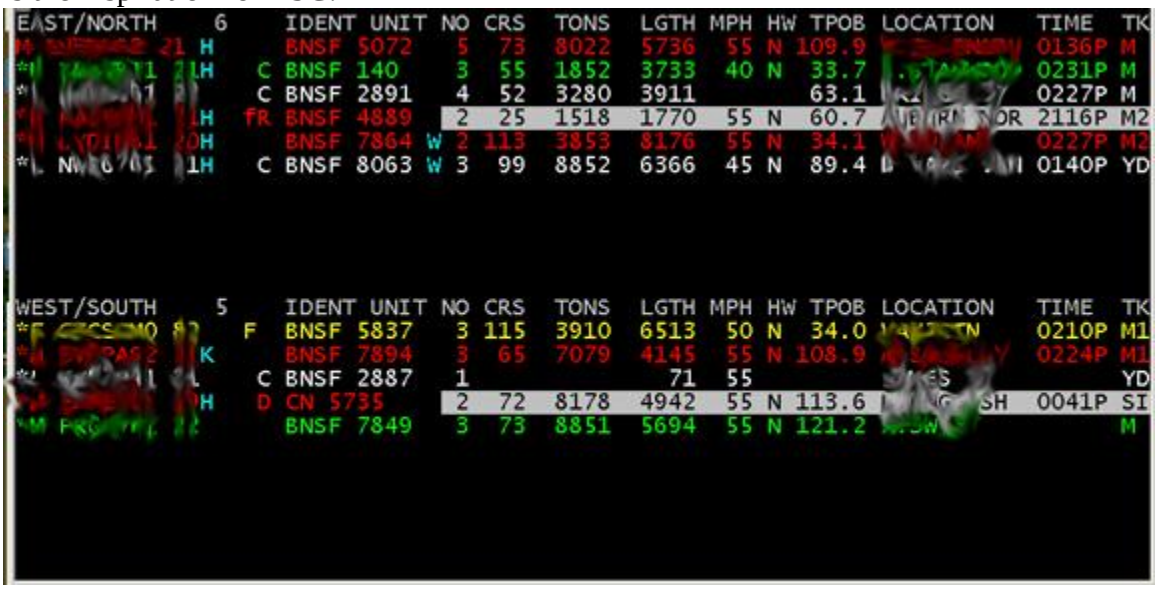


Figure 1

Some column values have been blurred out, but the exact content is not important. The look of the screen and kind of information presented are important.

The first thing to notice is that the screen is divided into two sections depending upon the direction of a train. The second thing to notice is that each row corresponds to a single train. Each column is some property of that train. The third thing to notice is the use of color. I don't know if there is any significance to the particular color or not, but making all of one row the same color and making rows different colors aids in following the eye across a row.

The screen conveys the following kinds of information:

- train symbol (train identifier, which should be unique)
- special instructions (high and wide, expedite, hazardous materials, etc.)
- the kind of train (coal drag, intermodal, reefer, manifest, etc.)
- the lead engine markings
- the number of engines
- the number of cars
- the train's tonnage
- the train's length
- the train's last reported speed
- the train's last reported location (station)
- the time of the last report

It is not hard to dream up other things that might be shown: engineer, conductor, dead on law time, etc. Just as every model railroad is different, every model railroad would have its own possibly unique pieces of information that should be included.

Some of this information is purely for operations. Some of this is generated by **CATS**. All of it is provided by **CATS**. A later section (2.3) will describe how to configure **CATS** to provide the information.

So, the intended users of TSC are:

- a Train Master – someone whose job is to oversee which trains are on the layout at any time
- the Crew Master – someone whose job is to manage assigning crew to trains
- a Yard Master – someone who runs the yard and wants to know when trains will be descending upon him
- the dispatcher – for a quick glance as to the problems a train will or will not cause him

1.1. What is New and Different

Version	Date	Changes	Section
1.00	12/15/08	First version	
1.01	2/2/09	Initial cleanup	
1.02	7/5/09	• Added Window height and width as command line options. • Fixed a bug so changes to bands take affect immediately	5.4

		• Added a column alignment for the trains table in **CATS**; TSC will follow the alignment. • Fixed a timing bug in TSC where it would receive data from **CATS** before it completed its initialization. Henceforth, the network connection is initialized last.	
1.03	9/20/09	• Reworked the color/font hierarchy by eliminating the "bindings" level.	3.2
1.04	11/20/09	• Packaged a different version of log4j for logging • Fixed up calls to methods inside log4j	
1.05	12/05/09	• Fixed up color handling	
1.06	1/22/11	• Added the ability to edit the table contents • Added a crew status window • Save saves the size and position of the train and crew status windows	

1.2. Information Exchange Compatibility

The following table records compatibility between **CATS** and **TrainStat**.

Format	**CATS**	**TrainStat**	**Comments**
	Before 2.13		Does not support **TrainStat**
2	2.13	0.1-0.2	Prototype – not really released
3	2.14 -2.16	0.3, 1.02	First production
3	2.17-2.20, 2.30	1.03-1.05	**CATS** sends train colors
4	2.21 or 2.31	1.06 -	**CATS** receives requests for changes from **TrainStat**

1.3. Capabilities

TSC was originally written to be only a status display – it received information from **CATS**, formatted it, and displayed it in a tabular form. Later (version 1.6), editing was added. The user can change the information on the screen and send the changes back to **CATS**. Only the requested changes are sent. The requested changes appear only on the local TSC displays – other TSC displays do not see them. If **CATS** accepts the changes, it broadcasts those updates to all TSC displays, which change accordingly.

In order to support crew assignments, TSC maintains two windows – one with train status and one with crew status. The latter is hidden when TSC launches, but can easily be unveiled (see Section 3.3.2.5). The crew window can be hidden (unhidden) by changing the Show Crew checkbox, minimizing the window, or clicking on the window closer.

Some fields cannot be changed through TSC:

- The train symbol or crew name (they can be changed on a newly added train/crew, but are locked after **CATS** accepts them).
- The station where a train is at.
- The time a train reported at the station.

Note that some fields involving time are frozen (e.g. the crew time left to going dead on the law). Though time ticks continue, the fields do not change. This will be fixed in the future.

2. Setting Up

TSC is a Java application, packaged as a .jar file. It is bundled into a zip file for distribution. Included in the zip file are this document, a default property file, a sample configuration file, and scripts for starting TSC under various computer operating systems. The property file is used to customize the text for a specific model railroad (see Section 5.3 for details).

2.1. Installing the Application

First, create a folder (directory) on the computer that is to run TSC. Into that folder, copy the TSC .zip file. Finally, run your favorite de-zip program to extract the contents of the .zip file[1]. After the .zip file has been unzipped, the folder should contain

1. TrainStat.jar – the Train Status Client application
2. TrainStat.doc – this documentation file
3. TrainStatus.properties – the file containing the text strings used by TSC
4. TrainStat.bat – the execution script for Windows
5. TrainStat.app – the execution script for MacOS
6. TrainStat.csh – the execution script for Linux
7. demo.xml – a sample configuration file
8. jdom.jar – a JMRI library needed for reading and writing XML files
9. log4j.jar – a JMRI library needed for logging errors
10. COPYING – a copy of the GNU Public License

A nice touch is to create a "shortcut" for the appropriate execution script and place it on the user's desktop. Except for MacOS (in which the shortcut is included), that chore is operating system dependent and left to the user.

2.2. System Requirements

The system that TSC runs on needs the following:

- Java 1.5 or greater (it may work on something older, but has not been tested)
- log4j.jar (found in the TSC distribution)
- jdom.jar (found in the TSC distribution)
- a network connection to a computer running **CATS**

[1] Windows users can simply double click on the .zip file and Windows will ask if the user wants to extract the contents of the file. If the answer is "yes", Windows will prompt the user for where to place the contents and also provide an option for creating a folder to hold the contents.

2.3. Configuring CATS

There are two aspects to configuring **CATS**:

- Specifying the information presented in TSC
- Specifying how the TSC computers talk to the **CATS** computer

2.3.1. Defining the Kinds of Information

CATS contains a mini-database about the jobs on a model railroad, the crew who are working an operating session, and the trains, themselves. The kinds of information are defined in the **CATS designer** program (see Sections 10, 11 and 12 of the **designer** manual). For example, **CATS** cannot show the train's length if it has not been told that it needs to hold the train's length. The way to tell it that it needs to hold the train's length is through **designer**.

The kinds of information fall into two categories: active information that **CATS** holds in response to things happening on the layout and passive information that a user enters into **CATS** (either through **designer -** when the CTC panel is created - or in real time during the operating session). There is very little of the first category (train location). Most of the rest is of the second category. However, followers of JMRI development should see that future enhancements exist in integrating TSC (and **CATS**) into JMRI operations, such as car routing.

As an example, let's walk through adding something, say train length, to **CATS**.

First, **designer** is used to add a new column to the train table to hold the train length information. The menu navigation is *Train->Edit Train Fields* and click on a row. The editing table is rotated 90 degrees clockwise from when the train table is shown, so train length field will appear between the row selected and the one below (and if it is in the wrong place it can be moved by clicking on the *Move Record Up* and *Move Record Down* buttons). Click on *Add Record*. Something like the following will appear:

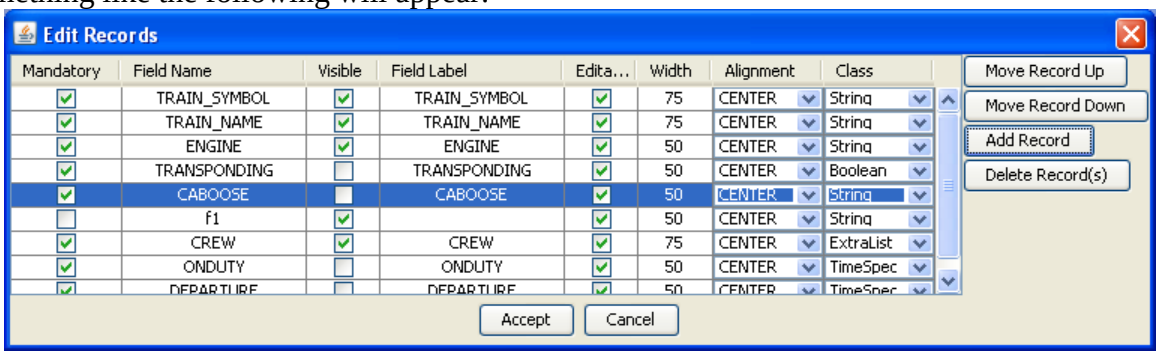


Figure 2

The new row (where the train length column will be described) is just below the highlighted row, with *f1* in the *Field Name* column. The *Mandatory* and *Field Name* boxes cannot be changed.

Figure 4 Train Edit Screen

All the rest can be changed. Here is what the above window might look like after the changes are made:

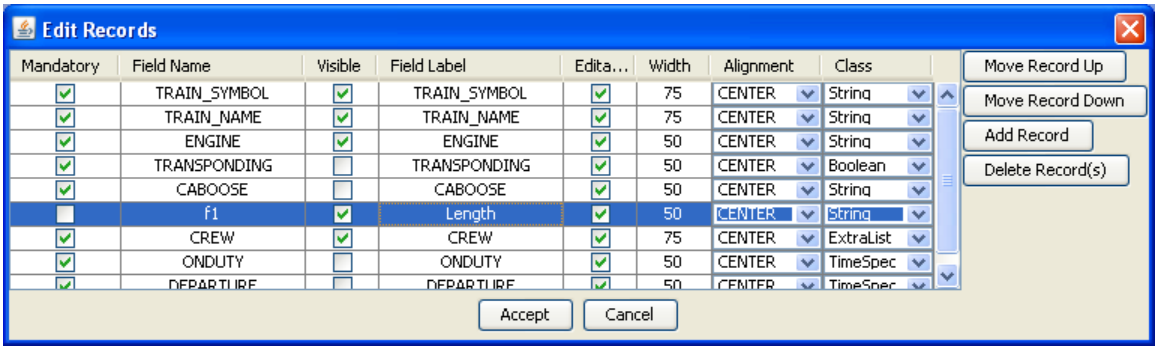
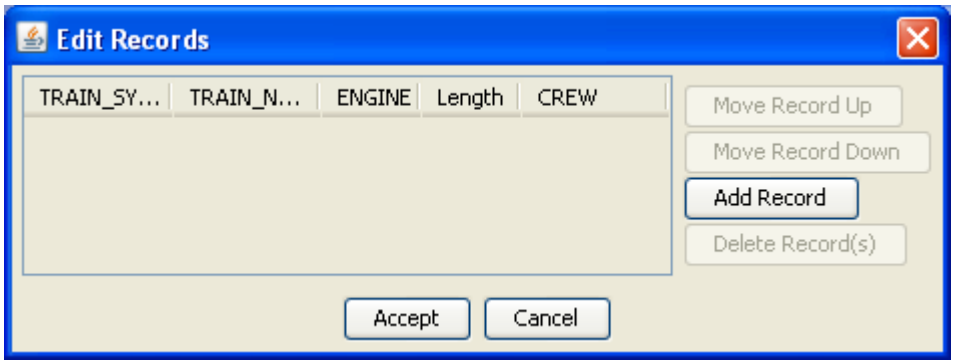


Figure 3



- *Visible* is checked so that the train length column will be shown
- *Field Label* is the title on the column ("Length")
- *Editable* is checked to allow the dispatcher to change a train's length
- *Width* is the initial number of pixels wide the column will be
- *Alignment* tells TSC where to position the length information in the column
- *Integer* tells **CATS** that a number will be put in the column

After clicking on *Accept* the resulting train screen will look like:

At this time, trains can be added and their lengths filled in. After the train information is saved, it can be picked up by **CATS**. Please remember that changes made through **designer** are permanent (until changed again through **designer**) and changes made through **CATS** (during an operating session) are lost when **CATS** is ended.

2.3.2. Defining Stations

The main focus of TSC is for showing the station that a train was last reported at and the time it arrived there. Stations are defined in **designer**. Here is how.

The grid cursor needs to be positioned on a block boundary:

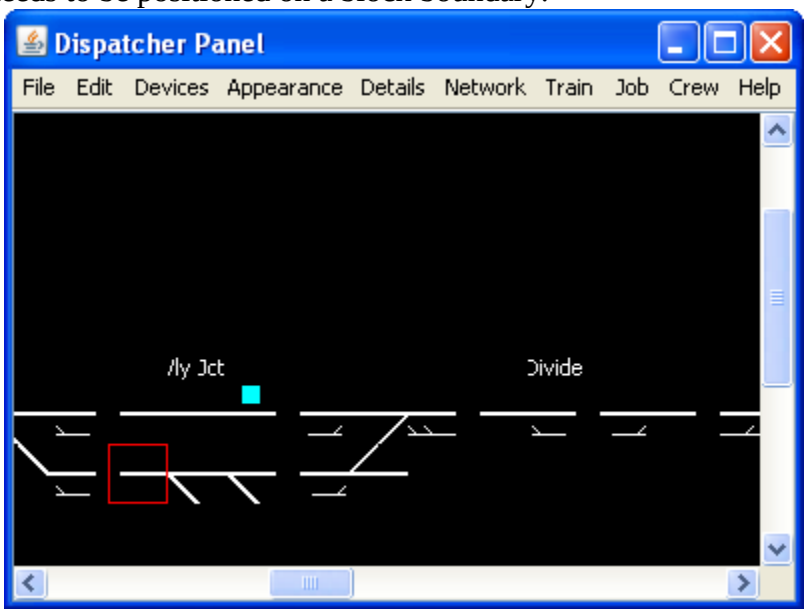


Figure 5

Then, *Details->Track Ends->Define Block.* That pops up a window like:

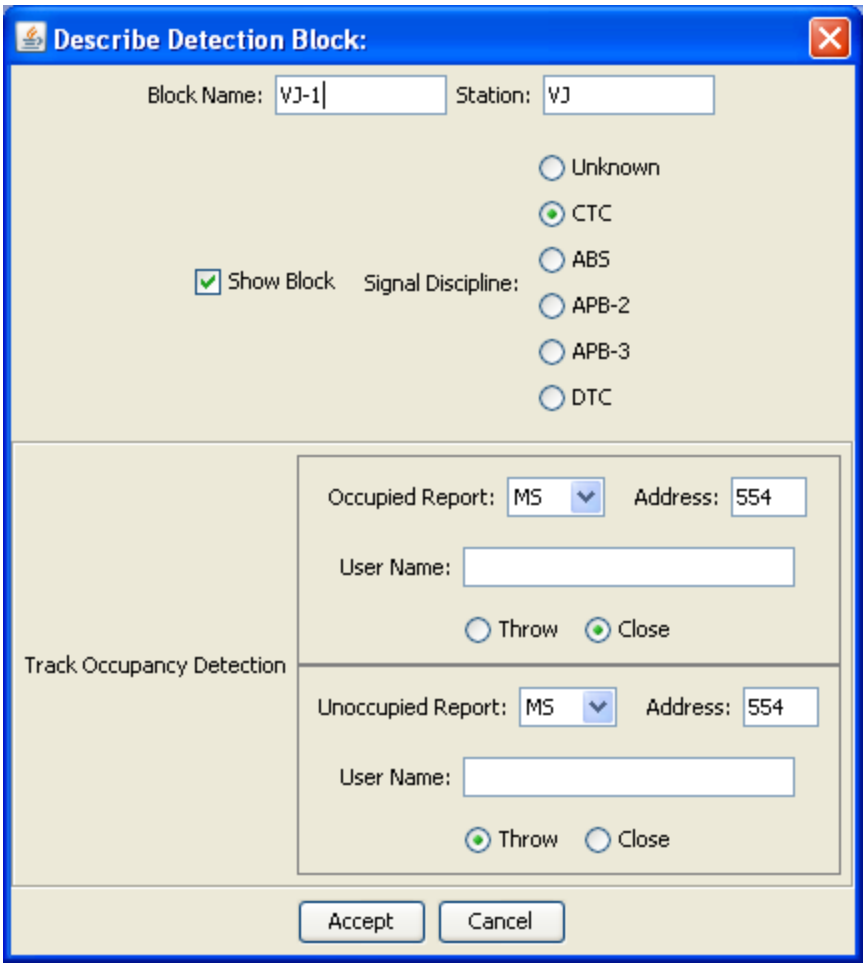


Figure 6

The station name goes in the box labeled *Station*, "VJ" in this example. Unlike with block names, the same station name can appear in multiple blocks. In the above example, Vly Jct is a passing siding, with some industry tracks. The Traffic Manager does not care which track a train is on, only that it arrived at Vly Jct. Thus, both the block on the main and the block on the siding have "VJ" in the *Station* box. Of course nothing enforces this, so the owner could just as easily given the blocks separate station names.

As a point of information, TSC will update a train's location only when the *Station* box is filled in. If a train occupies a block without a station name, TSC will ignore the update.

2.3.3. Saving the Network Settings

TSC needs to know how to connect to the **CATS** computer. This means that TSC and **CATS** need to know something about the computer protocols that make the Internet work. In general, there is little (if anything) that is needed to change in **CATS**. The procedure to make permanent changes to the **CATS** layout file (so that the network settings load into **CATS** whenever the layout file is loaded), is to start up **designer**. Click on *Network* and two boxes will appear:

- *Server Port*
- *Start Train Stat Server*

The first can be ignored unless it is necessary to change the network port being used by TSC to communicate with **CATS** (see Section 6). The second should be checked if **CATS** should start up ready to service TSC computers.

When the layout file is saved through **designer**, the network settings will be saved.

2.3.4. Entering the Network Settings Once

As is consistent with the rest of the way **CATS** handles configuration information, the network settings can be set in real time, during an operating session (though the settings will not be saved for any subsequent operating sessions).

After the layout has been loaded into **CATS** clinking on *Network* brings up a screen such as:

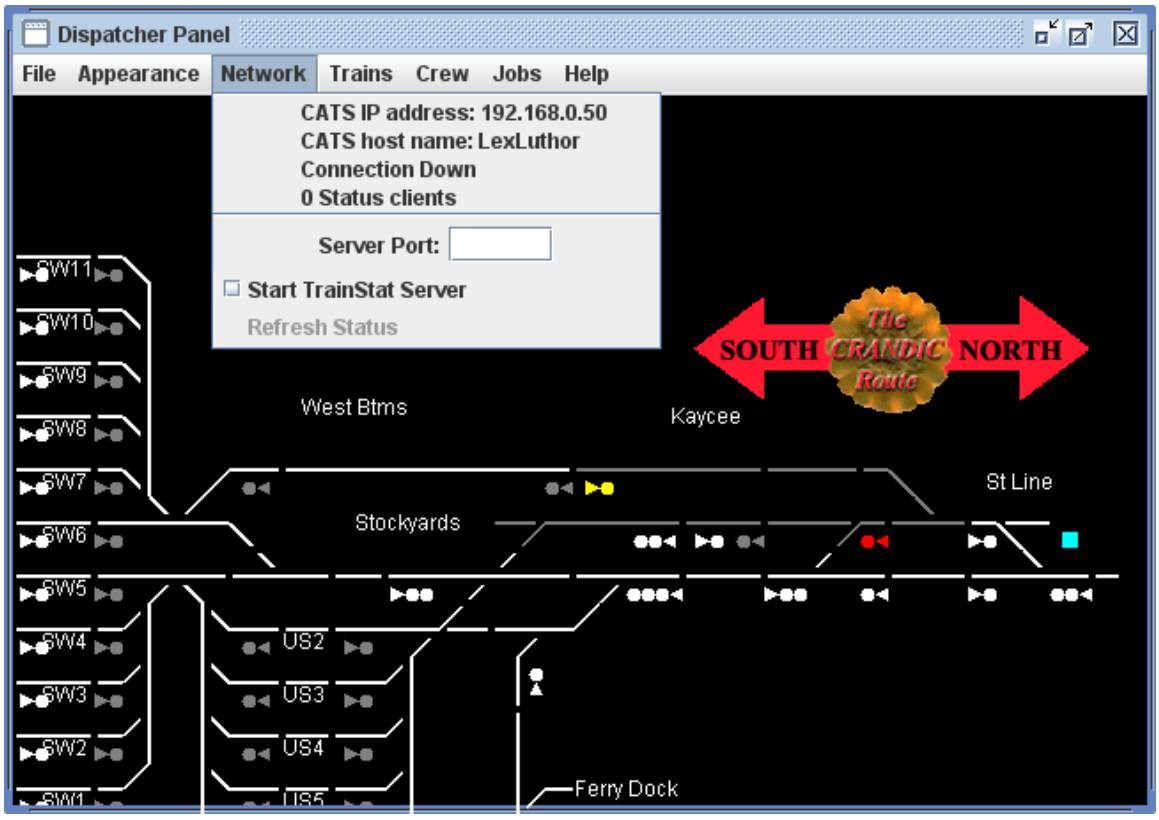


Figure 7

Line by line, this pull down window shows:

1. the IP address of the computer running **CATS**. This is useful to know when configuring the TSC computers.

2. the network name of the computer running **CATS**. This is also useful to know when configuring the TSC computers.
3. an indication of the network status. If it says "Connection Down", then the *Start TrainStat Server* checkbox should be checked to tell **CATS** to listen for other computers.
4. *Status clients* is the number of computers that have told **CATS** they want to be informed of changes to trains.
5. *Server Port* is the network port that **CATS** is using to talk to the other computers. Unless network problems are encountered (see Section 6), it can be left blank. If it is left blank, **CATS** will use port "54321".
6. *Start TrainStat Server* is a checkbox that tells **CATS** it should or should not listen for other computers.
7. *Refresh Status* is a button that makes **CATS** send the current status of all trains to all TSC computers. It is intended to bring the other computers up to date if things do not appear to be consistent.

In summary, for **CATS**, the important thing is to be sure that the third line says "Connection Up". The checkbox on the sixth line controls whether **CATS** should broadcast train status changes or not. With one caveat, the user should notice little difference in performance if the checkbox is checked or not. The caveat is that if the box is checked and the **CATS** computer is having network problems (for example, it cannot obtain an address from a DHCP server), **CATS** will be slow initially.

This feature has not affected the performance of **CATS** in any way that I could measure. However, if there are no TSC computers, then the box should not be checked.

3. Starting TSC

TSC is started by running the script appropriate for the operating system of the TSC computer. For example, double clicking on tsc.bat (or a shortcut to tsc.bat) under Windows launches TSC.

Initially, the TSC window will look like

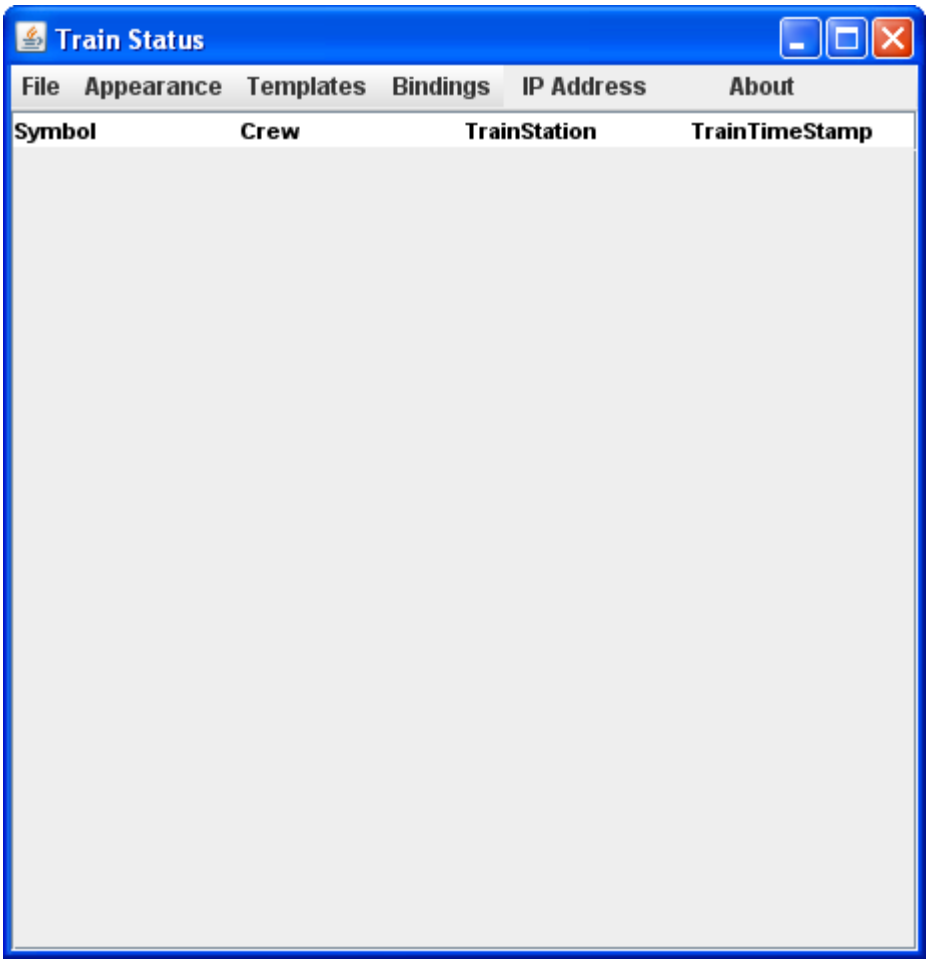


Figure 8

That is a long way from the screen shown in Figure 1. This document will next describe how to change (customize) TSC to look more like Figure 1.

3.1. Screen Layout

The TSC screen layout looks like

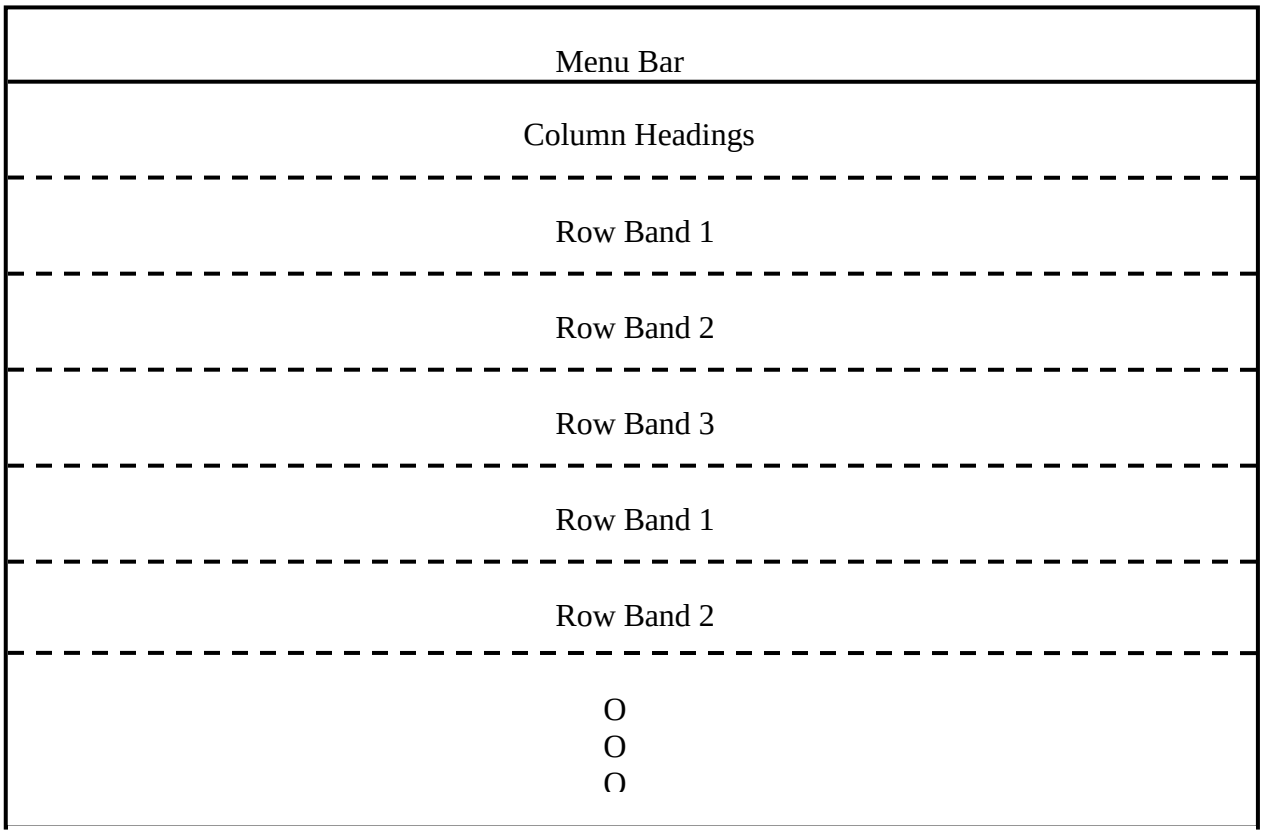


Figure 9

In words, the screen is split into horizontal stripes. The top stripe holds the menu bar. The next lower stripe holds the column headings. The rest of the stripes hold the rows, with the information on one train in each row.

Excluding the menu bar, the colors and fonts on each of the stripes can be customized. Each stripe has:

- a background color
- a foreground color (the color the text characters are shown with)
- a font size for the text
- a number of rows of characters

The column headings stripe is always a single row of text characters. The number of rows of text characters in the "Row Band" stripes can be configured to be between 1 and 10 rows. There are 3 customization settings. The first setting is for the first stripe of train status rows. The second setting is for the second stripe of train status rows. The third setting is for the third stripe of train status rows. After that, the first settings are applied again, then the second, etc. until all the train status rows have been covered.

For example, here is a customized TSC screen.

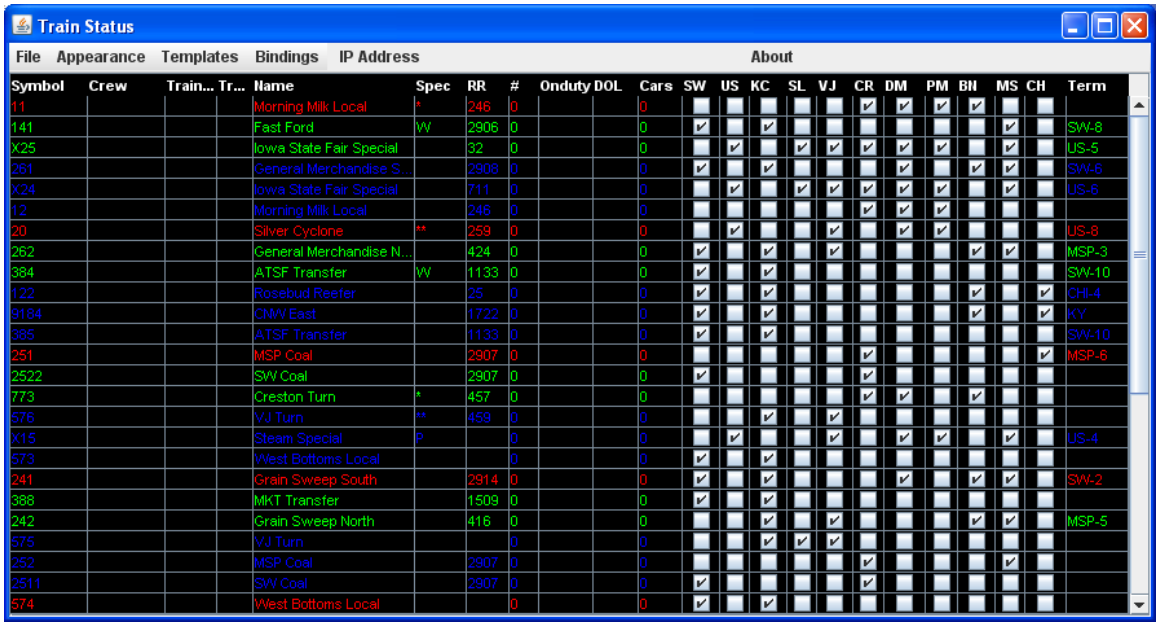


Figure 10 Customized Train Status Screen

It is shown here to illustrate

- The background can be changed.
- The color of the characters of the column headings can be changed.
- The color of the train status information can be changed.
- There is one row of trains in the first “Row Band” (shown in red), 2 in the second (shown in green), and 3 in the third (shown in blue).
- Each “Row Band” uses a different color for the text characters, so that the eye can follow across a row easily.
- The order that the trains appear is the same as they appear in **CATS** (Section 2.3.1).

3.2. Colors and Fonts

As mentioned above, each stripe has a background color and a font. A font has a color (the color for painting the letters), a size, and an emphasis (plain, bold, Italic, or bold and Italic).

3.2.1. Templates

The colors and fonts are selected through a Two-tier structure. Templates are at the top level. These are things that are fixed within TSC. Some are required by TSC and cannot be removed. The user can add new ones and delete ones he added. The color templates (Section 3.3.3.1) are:

- Default color (used when no other color can be found)
- Default font color (the color that characters will be painted in, if no other can be found)
- Background color (active background color)
- Foreground color (which does not seem to do anything)
- Heading background color (the color behind the characters on the column headings)

- Background color of Row Band 1
- Background color of Row Band 2
- Background color of Row Band 3

The font templates (Section 3.3.3.2) are:

- A default (which is used if no other templates are defined or the requested template cannot be found)
- The font for the column headings
- The font for Row Band 1
- The font for Row Band 2
- The font for Row Band 3

These Templates are fixed – none can be removed. Each is tied directly to a stripe on the TSC screen.

TSC uses the following order for determining what font to use for a train (row):

- If the user has changed the font (Section 4.4), then that selection is used.
- If **CATS** tells TSC that a train is a particular color, TSC looks for a font template with the same name and uses it.
- If TSC cannot find a suitably named font template or **CATS** did not provide a name for the font template, TSC will use the template for the band (Section 3.1) that the train is in.
- If the band font template does not exist, TSC uses the default font template.

3.2.2. Using a Template

Templates are names for colors and fonts. Nothing refers to a color or font directly, only to the name of a template. There are several reasons for this level of indirection:

- Typically things fall into groups (e.g. east bound trains or west bound trains; passenger trains or freight trains; priority 1, 2, or 3 trains, etc.). All the things in a group use the name of the same template; thus, with one change (to the template definition), all the colors or fonts of things in a group change simultaneously.
- TSC just presents information created by **CATS**. When **CATS** sends train information to TSC, **CATS** tells TSC the name of the font template to use. Since TSC selects the color for a name, it uses its local definition. Consequently, a train could be red and bold on one TSC display and white and plain on another.

3.3. Menu Items

This section moves through the menu items, describing what each is used for. You may need to refer to the previous section, as some menus describe how to make the customizations.

Before diving into each menu and sub-menu, please note that the text and labels on the menus and pop-ups appear to be "geeky" and "user unfriendly". However, these are all easily changed. See Section 5.3. In the following discussions, the text phrases that can be redefined are underlined.

3.3.1. File

As is convention in most applications, the left most menu holds actions involving files.

3.3.1.1 New

The New sub-menu is for clearing the TSC window to its unadorned, default condition (Figure

Figure 11

8). If any changes have been made to its configuration prior to selecting New, the user will be asked if the changes should be saved. The choices are:

- Yes – will pop up a file selection box in which to enter the name of the file to hold the changes being removed. It is a good idea to add ".xml" to the end of the file name, as that will make it easier to find the file for reading it.
- No – will remove the SaveChanges box and set the TSC configuration back to its default
- Cancel – will remove the SaveChanges box and not alter the existing configuration

3.3.1.2 Load

The Load sub-menu is used for reading in a configuration file. Selecting the Load sub-menu will also trigger a check of the configuration. If it has been changed, a SaveChanges box will pop up, as above (Figure 11).

- Yes creates a file selection box for saving the configuration (remember to append the .xml file type!).
- No does not save the configuration.
- Cancel ends the read activity without reading anything or changing the configuration.

3.3.1.3 Save

The Save menu is the converse of Load. It saves the current configuration. The first time it is used when running TSC, it will present a file selection window to allow the user to select the location of where to save the configuration and its name. Because TSC does not append a ".xml" file type to the file name, it is a good idea to do so. Subsequent uses of Save saves the file to the earlier noted file.

3.3.1.4 Import

The Import menu item is used to extract the column names and widths from a **CATS** layout file. Since it alters the column heading row, it will check to save the configuration before attempting to read in a **CATS** layout file.

This menu is of dubious value at this time, but can be used to pre-load a TSC screen before connecting to **CATS**.

3.3.1.5 Quit

The Quit menu item is used to end TSC. It checks for changes to the configuration (usually there are some) and allows the user to save the changes before ending. Clicking on the "closer" icon on the window has the same affect.

3.3.2. Appearance

Several actions are grouped together under the Appearance menu bar. These do not change the colors or fonts, but do change what is shown.

3.3.2.1 Vertical Lines

Vertical Lines is a check box. When checked, a vertical line is drawn between each column. This is useful for guiding the eye along a column.

3.3.2.2 Horizontal Lines

Horizontal Lines is a check box. When checked, a horizontal line is drawn between each row. This is useful for guiding the eye across a row.

3.3.2.3 All Columns

All Columns is a check box. When checked, all the information about trains is shown. In the **designer** screen used to define columns (Figures 2 and 3), the third column (labeled "Visible") controls whether a field should be shown or not. If not checked, the corresponding column is hidden. If All Columns is checked, then "Visible" is ignored.

3.3.2.4 All Trains

All Trains is a check box. When checked, all trains known to TSC are shown.

A train may be:

- Waiting for a crew.
- Have a crew assigned (assumed to be working).
- Tied down (finished working, but still on the dispatcher panel).
- Terminated (finished working and not on the dispatcher panel).

The first two kinds of trains are always shown. The last two are shown only when All Trains is checked. So, this checkbox provides a way of pruning the number of trains shown to only those that are working or will be working.

3.3.2.5 Show Crew

The crew status window is optional. This check box is used to hide or reveal it.

3.3.2.6 Train Band Widths

There are three sub-menus under Train Band Widths. They correspond to the number of rows in each band (stripe). Each of the sub-menus leads to a selection from 1-10. See Section 3.1 for a discussion on bands of rows.

3.3.3. Templates

As explained in Section 3.2, Templates provide names for colors and fonts. The idea is to make it easy to change groups of things quickly and easily.

Two kinds of Templates exist: colors and fonts.

3.3.3.1 Colors

Selecting the Colors sub-menu pops up the following box:

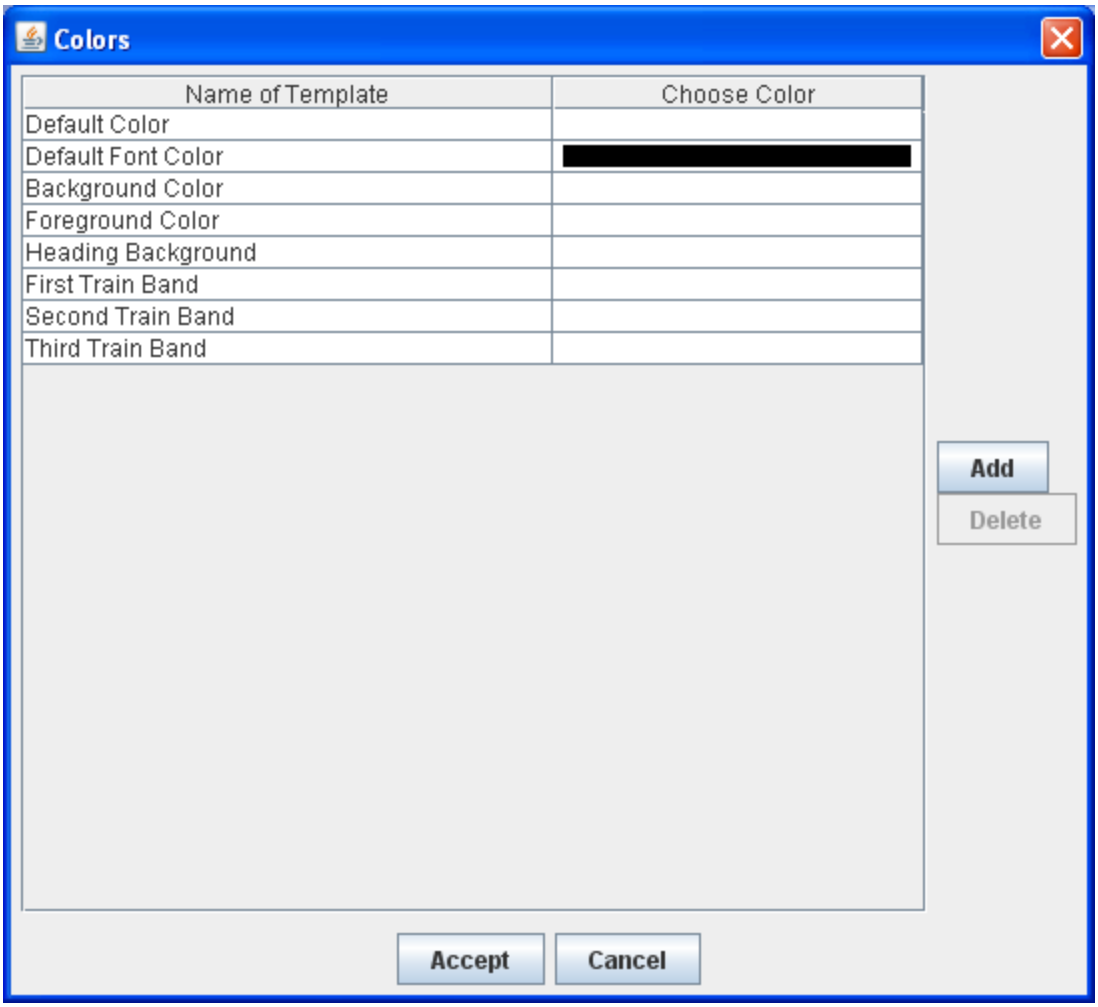


Figure 13 Color Template List

By default the above colors always exist. The left column is the name of the template and the right column shows the current value (color) and is a button. Click and it and you will be presented with a window that lets you change the color.

The AddButton is used to create a new color Template. It simply creates a blank row below the currently selected row.

If the DeleteButton is selected in Figure 12, then the Template is deleted.

Remember that it is necessary to click on the Accept button in Figure 12; otherwise, no changes are made.

3.3.3.2 Fonts

Moving back to the Templates menu item, if the Fonts sub-menu is selected, the following window pops up:

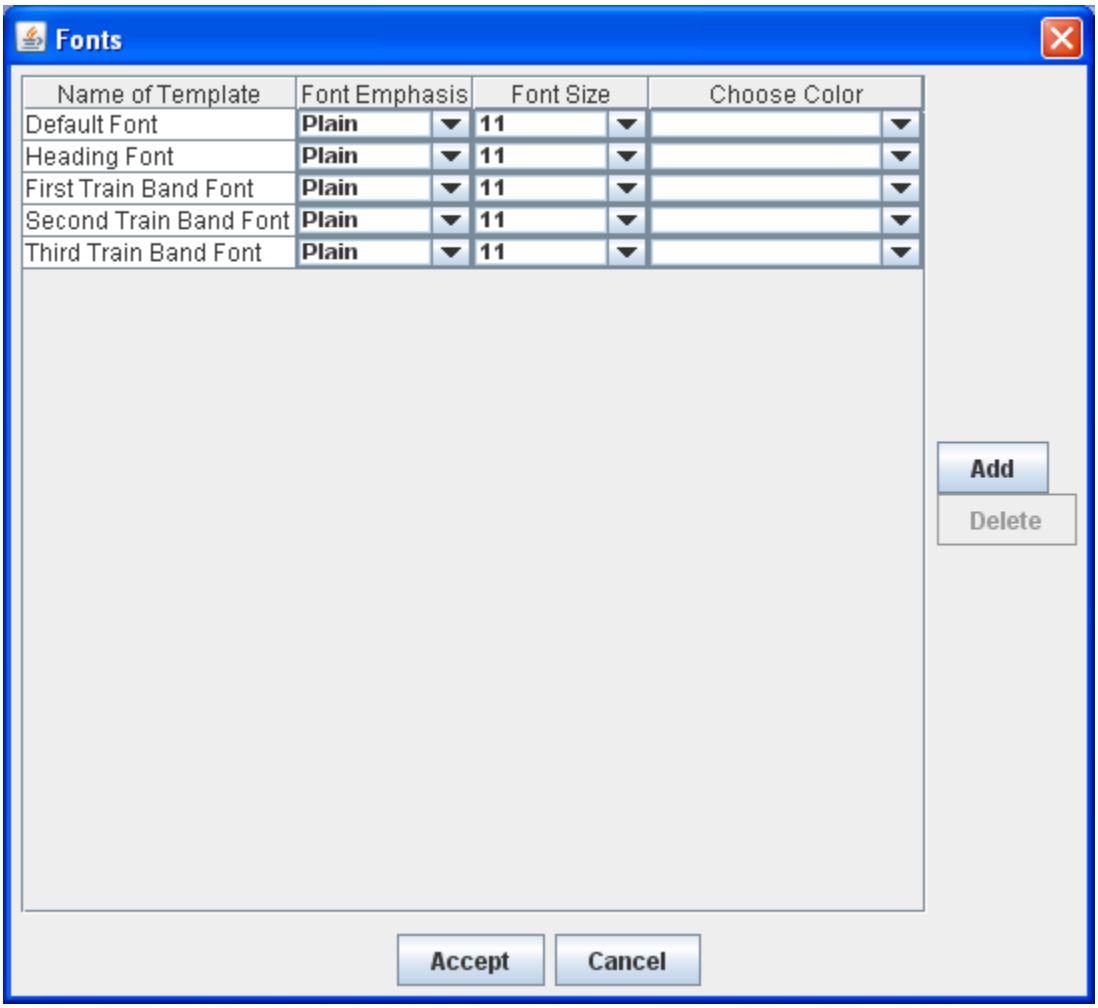


Figure 14 Font Template List

The columns are:

- Left: the name of the template
- Second from left: the font emphasis (plain, bold, Italic, bold and Italic)
- The font size in pixels
- The current color of the font and a button for changing it

New font templates can be added by clicking on the AddButton.

If the DeleteButton is selected in Figure 15, then the Template is deleted.

Remember that it is necessary to click on the Accept button in Figure 15; otherwise, no changes are made.

3.3.4. IP Address

Telling TSC where to find the **CATS** program is the thing most alien to many model railroaders and the most important in making TSC work. Clicking on IP Address on the menu bar brings up the IPTitle window, used for telling TSC how to find **CATS**.

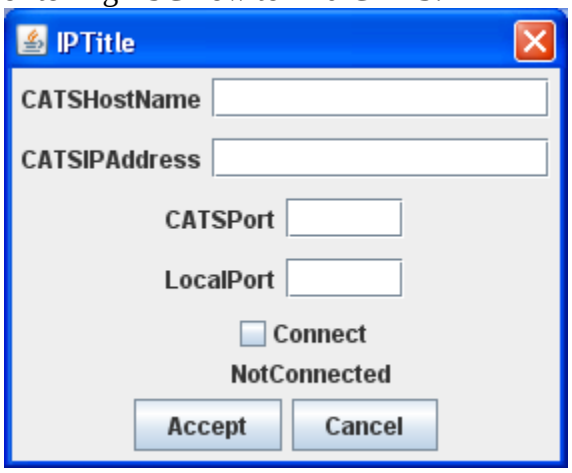


Figure 15

Working from top to bottom, each line is used as follows:

- CATSHostName is the network name of the computer that is running **CATS**. **CATS** can assist a little in determining what to enter. The second line in Figure 7 ("**CATS** host name:") is the name that the **CATS** computer thinks it is known on the network as. That name can be entered here, but also see below.
- CATSIPAddress is an alternative way of telling TSC how to find **CATS**. The IP address of the **CATS** computer is entered here. It should be the multi-digit number from the first line in Figure 7 ("**CATS** IP address:").
- CATSPort is the IP port that **CATS** is using. The value in this box should be the same as on line 5 ("Server Port:") in Figure 7. If this box is blank, TSC will use port number "54321".
- LocalPort is the IP port that TSC will use for talking to the **CATS** computer. It is currently unused, so can be left blank.
- Connect is a check box used for controlling the connection to **CATS**. If it is checked when the Accept button is pushed, TSC will try to contact the **CATS** computer on the IP port described in the rest of the window.
- The last line, NotConnected (in this example), shows if TSC is connected to a **CATS** computer or not. If TSC is connected to a **CATS** computer, this status field will change to Connected.

The way to think about the last two lines is that the check box is what TSC should do and the last line is what it is currently doing. TSC will connect or disconnect from **CATS**, at the whim of the checkbox, whenever the Accept button is pushed. TSC will do nothing if the Cancel button is pushed.

There are two ways of telling TSC how to find the **CATS** computer – by network name (CATSHostName) and IP address (CATSIPAddress). The combinations of values are interpreted as:

- Name=blank and Address=blank: TSC will try to connect to **CATS** on the computer that TSC is running on (so the dispatcher can use TSC, as well as the Traffic Master).
- Name=something and Address=blank: TSC will try to connect to the computer named "something".
- Name=blank and Address=something: TSC will try to connect to **CATS** on the computer with IP address "something".
- Name=something and Address=something else: TSC will try to connect to the **CATS** computer with the name "something".

4. Non-Menu Actions

This section discusses things that can be done inside the body of a TSC window. These things share a common theme in that they result from doing something with the primary mouse button while in the body of the TSC window.

4.1. Changing the Window Size

Dragging one of the sides of the TSC window can make the window wider or thinner. Dragging the top or bottom can make the window taller or shorter.

4.2. Column Width Adjustment

A column's width can be changed by dragging the vertical border on a column heading. This is a local action. Nothing is sent back to **CATS**.

4.3. Column Re-ordering

The left-to-right order of the columns can be changed by dragging a column heading. This is a local action. Nothing is sent back to **CATS**.

4.4. Changing the Color or Font of a Row

If the secondary mouse button (often the right one) is clicked while over a train status row, a BindingPopUpTitle box will appear.

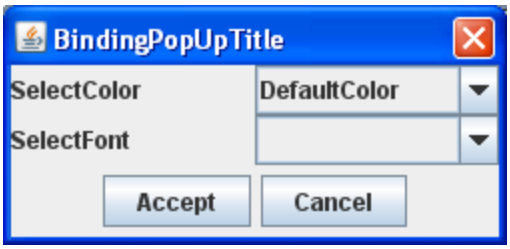


Figure 16

It has two selection lists. SelectColor lists all the color Templates. SelectFont lists all the font Templates. When the Accept button is pushed, the row will be painted in the background color of the first and the letters will be shown in the font of the second. So, in the above example, the background color will be taken from the DefaultColor Template. The SelectFont box has no selection; thus, the row will use the default font Binding for the row band for the characters.

4.5. Selecting a Row or Rows

Clicking the primary mouse button (usually the left one) on a row will select the row. Its background will change color. Holding the shift key down while clicking the primary mouse button will select all rows between the one the mouse cursor is over and any other block of selected rows.

4.6. Editing a Cell

You can change the contents of a cell by clicking the mouse on it twice. The first click selects the row (and the background color of the row changes). The second click selects the cell in the row and places a cursor in it. If the cell (usually the column) is flagged as having editing prohibited, you will not be able to change the contents. Otherwise, enter a new value and touch Enter. TSC checks the format of cell contents, so if you place a string (e.g. "ME") into a cell requiring a number, the string will be rejected.

After touching Enter, the cell's background will change color, indicating that you have made a change, but a request to CATS to also make the change has not been sent. At this point, you can make changes to other cells, click on the Accept button or click on the Cancel button.

4.7. Accept Button

If you make a change to any cell or row, the Accept button will be activated (not grayed out). Clicking on it will trigger TSC to collect all the changes and verify them. If they look good, they are sent to **CATS**. As **CATS** makes the changes internally, it will send updates to all TSC programs. The TSC program requesting the change will see the update, refresh its display, and restore the background color. When no outstanding changes exist, TSC will gray out the Accept button.

If an edit is not good, a pop-up window will appear explaining the problem.

4.8. Cancel Button

The opposite of the Accept button is the Cancel button. It is grayed out until a cell or row is changed. Then it is active. Clicking on it will remove any pending changes on the TSC screen.

4.9. Move Up Button

If you select one or more rows with the mouse (Section 4.5), the Move Up button will be enabled. If you click on it, the selected row(s) will move up one row. This action changes only the local screen. Nothing is sent back to **CATS**.

4.10. Move Down Button

If you select one or more rows with the mouse (Section 4.5), the Move Down button will be enabled. If you click on it, the selected row(s) will move down one row. This action changes only the local screen. Nothing is sent back to **CATS**.

4.11. Add Button

The Add button is used to create a new entry in the table. The cells are filled in with default values and the cell with the unique value (e.g. the train symbol or crew name) is left blank. This is the only opportunity that you have to enter something in that cell. After it has been sent to **CATS**, it cannot be changed.

4.12. Delete Button

The Delete button is an option on the crew status screen only after a crew row has been selected. When it is invoked, the row will be highlighted to indicate all of its contents have been changed. Upon clicking the Accept button, a request will be sent to **CATS** to delete the selected crew. When **CATS** sends back an acknowledgement, the row will disappear.

4.13. Rerun Button

The Rerun button is an option on the train status screen when the selected rows are for trains that have been tied down or terminated. When the Accept button is clicked, a request to rerun the selected train(s) is sent to **CATS**. When **CATS** sends back an acknowledgement, the train(s) will appear on the active train screen, again.

4.14. Tie Down Button

The Tie Down button is an option on the train status screen when the selected rows contain active trains. When the Accept button is clicked, a request is sent to **CATS** to tie down the train(s). When **CATS** sends back an acknowledgement, the train(s) will be added back to the active train screen.

4.15. Terminate Button

The Terminate button is an option on the train status screen when the selected rows contain active or tied down trains. When the Accept button is clicked, a request is sent to **CATS** to terminate the train(s). When **CATS** sends back an acknowledgement, the train(s) will be removed from the active train screen.

5. Customizing TSC

As has been seen, TSC is pretty drab and the prototype has some colorful screens. This section attempts to pull together the above into something that can be saved and reused.

5.1. The Division of Labor Between CATS and TCS

A TSC screen consists of two things: content and presentation. A TSC screen is a table with columns and rows. The cells (intersections of a column and row) hold the content. The presentation is how the rows and columns are colored, the font size and color, the column headings, etc. The content is the information. The presentation is the appearance. **CATS** provides the content. TSC simply receives the value of the cells and shows them. Thus, there is nothing in TSC for configuring (or saving) content. It is all done in **CATS** (and **designer**).

One **CATS** instance (the one specified in the network screens) maintains the current database. In the modern world of distributed computing, this architecture is not elegant, but it is simple and it works. Thus, all TSC instances collect edits and send them to that **CATS** instance as requests. It accepts (or rejects) the requests and broadcasts changes back to the TSC (and subordinate **CATS**) instances.

On the other hand, TSC is very flexible in how the presentation can be configured. This section describes how to configure the presentation. Unlike **CATS** and **designer**, presentation changes made while running TSC can be saved. Furthermore, each computer running TSC is independent of any other computer running TSC. They need not share the same configuration.

5.2. Customizations Through the Menus

The following things can be customized and the customizations saved through menu items:

- The vertical lines setting (Section 3.3.2.1).
- The horizontal line setting (Section 3.3.2.2).
- The All Columns setting (Section 3.3.2.3).
- The All Trains setting (Section 3.3.2.4).
- The number of rows of trains in each Row Band (Section 3.3.2.6).
- The ColorTemplates (Section 3.3.3.1).
- The FontTemplates (Section 3.3.3.2).
- The settings for connecting to the **CATS** computer (Section 3.3.4).

The customizations are saved through the Save menu item or when creating a New TSC or Loading a saved one.

5.3. Customizations of Properties

Except for a few error strings, almost all of the labels, window titles, prompts, and error strings can be replaced. In short, almost all of the text that TSC displays can be changed on the computer running TSC.

When TSC starts up, it tries to read in a property file. The property file has one line for each string that can be customized. The first part of the line is the tag that TSC knows about. When TSC needs to display a string, it searches the file for a matching tag. If it finds one, it uses the rest of the line. If it does not find a matching tag, it uses the tag.

The default name for the property file is TrainStatus.properties. It should be placed in the same folder as the tsc.jar file, though even this can be changed (see Section 5.4). A sample file (demo.xml) is packaged in the TSC distribution.

A few tags are not translated directly from what is shown on TSC. The titles on the menus and menu items have phrases (such as ".label" appended to them).

5.4. Script Options

Another way to customize TSC is by adding options in the script (.bat or .csh) file. The main part of the script will have a line that looks like:

```
java -Djava.class.path=".;tsc.jar;log4j.jar;jdom.jar;" -Dsun.java2d.noddraw
TrainStat.trainStat.main
```

TSC looks for options on the end of the line. The options it looks for are:

- *BUNDLE*=something. "Something" is the name of file holding the properties for TSC to use (Section 5.3).
- *CONFIGURATION*=something. "Something" is the name of a TSC configuration file to load when TSC starts up. This file is expected to be an XML file, created through the Save process.
- *TRACE*. This option enables some tracing, allowing the user to see what TSC is receiving from **CATS**.
- *HEIGHT*=integer. The height of the TSC window, in pixels.
- *WIDTH*=integer. The width of the TSC window in pixels.

6. Networking Computers

For the Train Status Client application to work, the computer running the application must be able to communicate with a computer running **CATS**. The communications takes place over an Internet Protocol (IP) network, such as used for connecting to web sites. This means that both the TSC computer and the **CATS** computer must have a Network Interface Card (Ethernet interface), be configured to use it, and physically attached to a network.

Because of the myriad kinds of computers, Network Interface Cards (NICs), operating systems, and other software, this document cannot describe how to set up an arbitrary connection. The user will have to refer to his computer manuals or technical support for that information. However, assuming that both the TSC and **CATS** computers are on a network, there are some things that can be done to assist in identifying each computer on the network. The top two lines of Figure 7 are from **CATS** and show the network name and IP address that the **CATS** computer is using. This information is entered into the corresponding boxes of Figure 20.

Here are a few tips when setting up communication between a TSC computer and **CATS** computer:

- If checking *Start TrainStat Server* on Figure 7 does not change the third line from *Connection Down* to *Connection Up*, then the **CATS** computer may not be connected to a network.
- If other Internet functions (such as a web browser) on the **CATS** computer work, then check the firewall settings. They may be preventing **CATS** from using the IP port.
- Each computer should be able to "ping" the other.
- Start the train status server on the **CATS** computer before attempting to connect to it from the TSC computer.

7. TroubleShooting

This section discusses some common problems and how to solve them.

7.1. Order of Starting Up Computers

In general, it does not matter which computer (TSC or **CATS**) is turned on first. What does matter is that the *Start TrainStat Server* checkbox be checked on the **CATS** computer before the Connect button is checked on the TSC computer. If that order cannot happen, then the solution is to open the IPTitle window (Section 3.3.4) on the TSC computer, make sure that Connect is checked, then click on Accept, after **CATS** is running with the *Start TrainStat Server* box checked.

7.2. Train Locations do not Show Up

First, TSC must be communicating with **CATS**. This should be clear because as soon as TSC has contacted **CATS**, **CATS** will send to TSC everything the former knows about its trains. If communication is occurring, then a likely problem is that the stations were not defined on the **CATS** computer (see Section 2.3.2). TSC will report only when a train arrives at stations defined in **CATS**.

8. Summary

This document has described the capabilities of the Train Status Client application, how to use them, and how to customize them. The Train Status Client application summarizes the information **CATS** knows about trains and presents them in a tabular format.

This is not the final version of TSC. Other capabilities will be added as users figure out new things that would help their operations.