Generic Data Screens

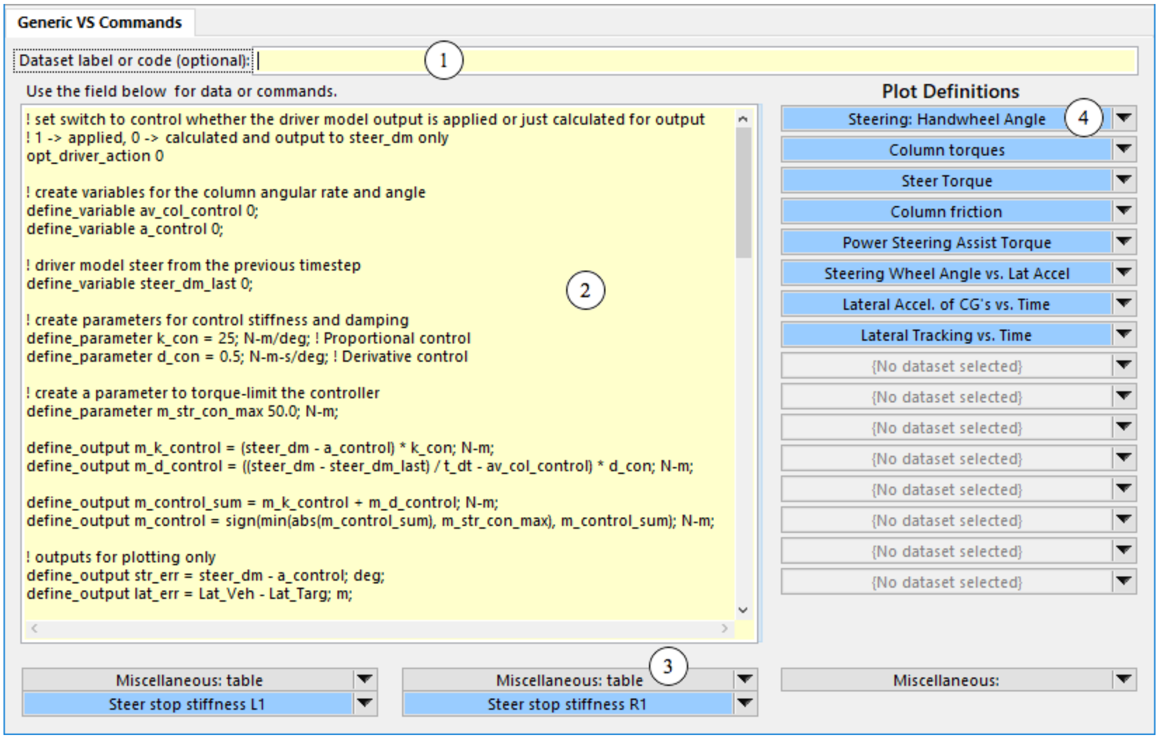
VS browsers include screens that are intended for advanced users to assemble groups of datasets or parameter values in a way that is not provided with the regular GUI screens (Table 1).

Table 1. Generic screens for assembling custom datasets.

Library Name	Typical Applications
Generic Data Group	Nine sets of data links and miscellaneous data fields.
Generic Data Group (More)	Extend Generic Data Group (if more links or fields are needed).
Generic Data Links	Assembly of many data links with a few misc. data fields.
Generic Data Lists	Three lists of variables or simple lines of input.
Generic VS Commands	A listing of VS Commands with associated plot definitions.

User Settings

The **Generic VS Commands** screen (Figure 1) has four kinds of user settings. The other screens listed in Table 1 only have three kinds of settings (they do not have the plot definition links ④).



Dataset label or code (optional):

Use the field below for data or commands.

```
! set switch to control whether the driver model output is applied or just calculated for output
! 1 -> applied, 0 -> calculated and output to steer_dm only
opt_driver_action 0

! create variables for the column angular rate and angle
define_variable av_col_control 0;
define_variable a_control 0;

! driver model steer from the previous timestep
define_variable steer_dm_last 0;

! create parameters for control stiffness and damping
define_parameter k_con = 25; N-m/deg; ! Proportional control
define_parameter d_con = 0.5; N-m-s/deg; ! Derivative control

! create a parameter to torque-limit the controller
define_parameter m_str_con_max 50.0; N-m;

define_output m_k_control = (steer_dm - a_control) * k_con; N-m;
define_output m_d_control = ((steer_dm - steer_dm_last) / t_dt - av_col_control) * d_con; N-m;

define_output m_control_sum = m_k_control + m_d_control; N-m;
define_output m_control = sign(min(abs(m_control_sum), m_str_con_max), m_control_sum); N-m;

! outputs for plotting only
define_output str_err = steer_dm - a_control; deg;
define_output lat_err = Lat_Veh - Lat_Targ; m;
```

Plot Definitions:
- Steering: Handwheel Angle
- Column torques
- Steer Torque
- Column friction
- Power Steering Assist Torque
- Steering Wheel Angle vs. Lat Accel
- Lateral Accel. of CG's vs. Time
- Lateral Tracking vs. Time
- {No dataset selected}
- {No dataset selected}
- {No dataset selected}
- {No dataset selected}
- {No dataset selected}
- {No dataset selected}
- {No dataset selected}
- {No dataset selected}

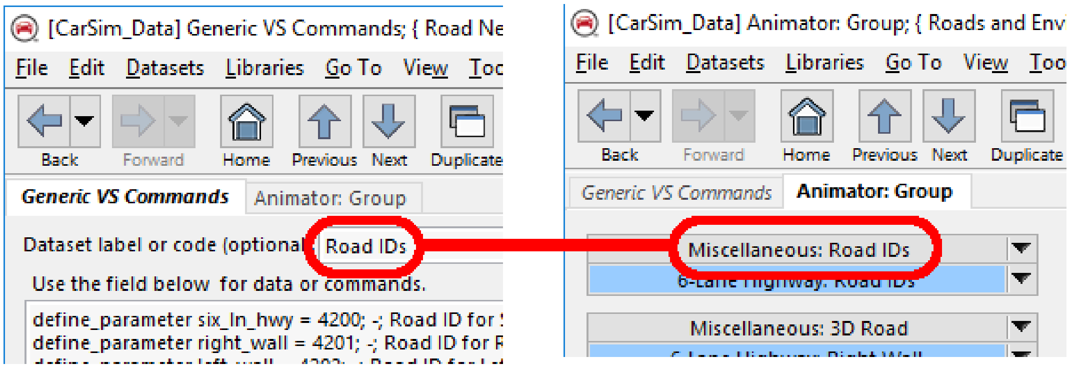
Miscellaneous: table
Steer stop stiffness L1

Miscellaneous: table
Steer stop stiffness R1

Miscellaneous:

Figure 1. Example generic screen: VS Commands.

① Typically, this field is used to (optionally) add a custom label for the dataset. When linked from another screen, the library name will display as “Miscellaneous:”, followed by the contents of this field.


[CarSim_Data] Generic VS Commands; { Road Ne

Dataset label or code (optional Road IDs

Use the field below for data or commands.

```
define_parameter six_ln_hwy = 4200; -; Road ID for :
define_parameter right_wall = 4201; -; Road ID for R
```

[CarSim_Data] Animator: Group; { Roads and Env

Miscellaneous: Road IDs

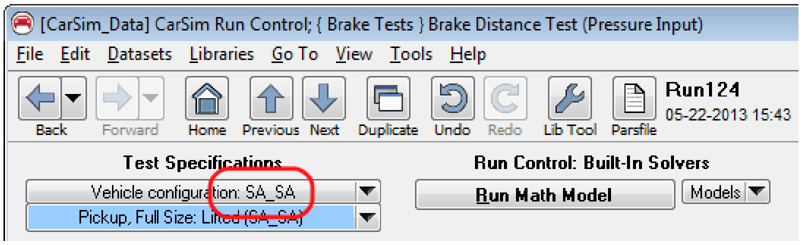
6-Lane Highway: Road IDs

Miscellaneous: 3D Road

6-Lane Highway: Right Wall

Figure 2. Labeling a Dataset

However, when the generic screen is used in place of a vehicle library dataset, this field must contain a vehicle code to enable a VS Math Model to be constructed to run a simulation. When you make a run from the **Run Control** screen, the VS browser automatically constructs a VS Math Model based on a “vehicle code” associated with the vehicle dataset selected in the upper-left link of the **Run** screen. Some examples: in CarSim, a vehicle with solid-axle front and rear suspensions has the code `"SA_SA"`; in TruckSim, a similar vehicle has the shorter code `"S_S"`. You can see the code for a given type of vehicle by linking to a dataset for that vehicle type on the **Run Control** screen and viewing the message that appears in the top part of the link (Figure 3).


[CarSim_Data] CarSim Run Control; { Brake Tests } Brake Distance Test (Pressure Input)

Run124
05-22-2013 15:43

Test Specifications

Vehicle configuration: SA_SA

Pickup, Full Size: Lifted (SA_SA)

Run Control: Built-In Solvers

Figure 3. Vehicle code "SA_SA" for a vehicle in CarSim.

If you use one of the generic screens to assemble vehicle data that will be linked from the upper-left link on the **Run Control** screen, then you must provide the code that identifies the type of vehicle to be simulated by using this field.

② Miscellaneous data field. Enter information in the form that is expected either by the VS Math Model or VS Visualizer.

The format is typically such that each line has a keyword and value or equation, separated with white space (at least a single space). For more details on syntax, see the documentation in the document *VS Math Models Reference Manual*.

Most of the keywords used to start an existing run are written in the echo file created by the VS Math Model when the run is made. An easy way to view this file is via the button **View** on the **Run Control** screen.

③ Links are used to include datasets from other libraries available in the VS browser. Some datasets may apply to more than one place in the model. For example, a tire dataset can be applied to any of the wheels. See the next subsection for notes on specifying the context of a tire or other dataset that can by applied in different locations.

④ Links to **Plot: Setup** datasets that define sets of variables to be plotted. These are typically new variables added with VS Commands in the data field ② for the dataset.

Applications

There are at least two primary applications for these screens. First, they can be used to assemble large datasets that are made of other VS datasets. For example, advanced scenarios and procedures can be assembled using new variables added with VS Commands and activated using multiple events.

Warning Three VS Commands (DEFINE_EVENT, EVENT_SET_GT, and EVENT_SET_LT) have the option of specifying Parsfile names. If any of these commands are used with the *pathname* argument used to specify a Parsfile, then the dataset cannot be exported using the normal **File** menu items of the VS Browser: **Export Consolidated Parsfile** or **Export Expanded Parsfile**. (A file will be generated, but it will not import properly.)

The VS Browser has an **Events** screen that automatically generates proper DEFINE_EVENT commands. Mechanical Simulation recommends that you always use this screen to define events, rather than attempting to type them in a miscellaneous scrollable field in a generic screen (or any other screen with a miscellaneous scrollable yellow field).

The second application is to specify special parameter values that do not fit in existing data screens, such as nonlinear suspension properties that can be used instead of the linear coefficients shown on the standard CarSim and TruckSim suspension screens.

Specifying the Context or Location of a Linked Dataset

When combining parameters and other datasets, be aware that the data are sent to the math models in the same order they are numbered on the screen: top to bottom then left to right.

Many of the components are used more than once (tires, dampers, springs, etc.), and a keyword is used to determine where the description of the component should be applied. For example, the keywords IAXLE and ISIDE are used to specify the current axle (suspension) and side, respectively. Put the lines IAXLE 2 and ISIDE 1 in a data field to specify that any links that follow will be associated with axle 2 on the left side.

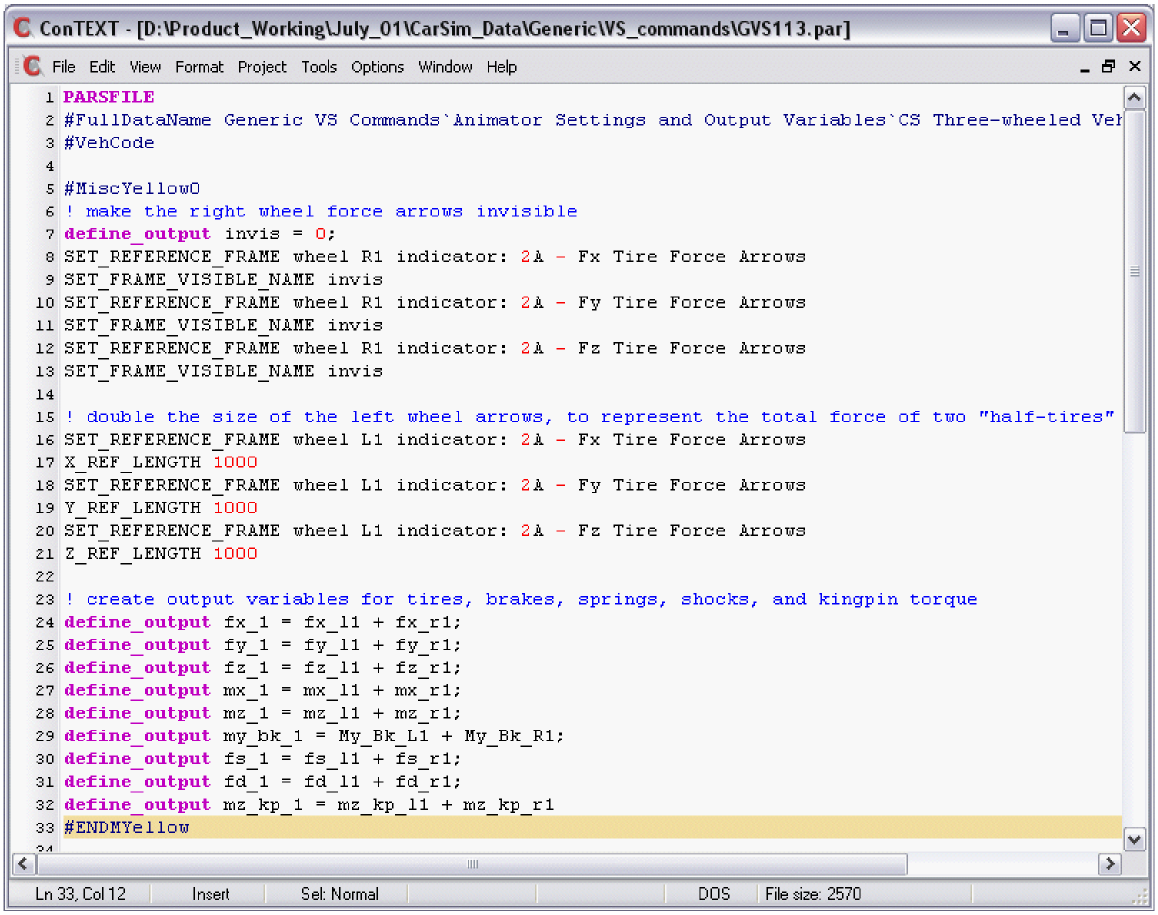
A quick way to determine how keywords are used to locate datasets is by viewing an echo file for a run of interest. (Use the button **View** on the **Run Control** screen.)

The VS Commands Screen

VS Commands can be written in any miscellaneous data field in the VS browser, including the **Run Control** screen, all of the generic screens listed in Table 1, and many other screens in the browser. The **Generic VS Commands** screen is intended to provide a convenient screen for this purpose: the data field is wide, to accommodate the length of most commands.

Although the screen supports basic editing and display of the field contents, it is often easier to work with VS Commands with a full text-editing program, such as the ConTEXT program provided with VehicleSim products. To view the data in ConTEXT, click the Parsfile button in the top part of the screen, or use the **File** menu command **View Text Parsfile** (Ctrl+P). This opens the parsfile (Figure 4).

The contents of the miscellaneous data field are identified in the text file with the keywords `#MiscYellow0` (line 5) and `#ENDMYellow` (line 33, highlighted). The contents of the text file between these keywords (lines 6 – 32) contain the information shown in the screen (Figure 1).



```
 1 PARSFILE
 2 #FullDataName Generic VS Commands`Animator Settings and Output Variables`CS Three-wheeled Veh
 3 #VehCode
 4
 5 #MiscYellow0
 6 ! make the right wheel force arrows invisible
 7 define_output invis = 0;
 8 SET_REFERENCE_FRAME wheel R1 indicator: 2A - Fx Tire Force Arrows
 9 SET_FRAME_VISIBLE_NAME invis
10 SET_REFERENCE_FRAME wheel R1 indicator: 2A - Fy Tire Force Arrows
11 SET_FRAME_VISIBLE_NAME invis
12 SET_REFERENCE_FRAME wheel R1 indicator: 2A - Fz Tire Force Arrows
13 SET_FRAME_VISIBLE_NAME invis
14
15 ! double the size of the left wheel arrows, to represent the total force of two "half-tires"
16 SET_REFERENCE_FRAME wheel L1 indicator: 2A - Fx Tire Force Arrows
17 X_REF_LENGTH 1000
18 SET_REFERENCE_FRAME wheel L1 indicator: 2A - Fy Tire Force Arrows
19 Y_REF_LENGTH 1000
20 SET_REFERENCE_FRAME wheel L1 indicator: 2A - Fz Tire Force Arrows
21 Z_REF_LENGTH 1000
22
23 ! create output variables for tires, brakes, springs, shocks, and kingpin torque
24 define_output fx_1 = fx_l1 + fx_r1;
25 define_output fy_1 = fy_l1 + fy_r1;
26 define_output fz_1 = fz_l1 + fz_r1;
27 define_output mx_1 = mx_l1 + mx_r1;
28 define_output mz_1 = mz_l1 + mz_r1;
29 define_output my_bk_1 = My_Bk_L1 + My_Bk_R1;
30 define_output fs_1 = fs_l1 + fs_r1;
31 define_output fd_1 = fd_l1 + fd_r1;
32 define_output mz_kp_1 = mz_kp_l1 + mz_kp_r1
33 #ENDMYellow
34
```

Figure 4. Parsfile for a VS Commands dataset.

Besides offering the basic capabilities of any text editor (line wrapping, find/replace, etc.), ConTEXT has been configured to use syntax coloring for show VS keywords such as **`PARSFILE`** and **`define_output`**. Also, it recognizes the VS comment character '!' and shows all comments in a different color (blue).

Other Generic Data Screens

Other than the **Generic VS Commands** screen, all the screens listed in Table 1 have the same general purpose: they exist to associate arbitrary lines of inputs from miscellaneous data fields with information from other datasets.

The **Generic Data Lists** screen has three large data fields suitable for placing many short commands. This screen was used in older versions of VehicleSim products to list import and export variables. Current versions of the products have specific screens for activating import, export, and output (written) variables, so this screen is no longer as useful as it once was. It remains in the browsers mainly for backward compatibility.

The **Generic Data Group** and **Generic Data Group (More)** screens have nine links interleaved with miscellaneous data fields, and are recommended when it is necessary to alternate between lines of input such as `IAXLE 1` and links to other datasets. It is not possible make a dataset in the **Generic Data Group** that links to another dataset in the same library, so the nearly identical library **Generic Data Group (More)** is provided to expand the number of linked datasets.

A more convenient way to link many datasets together is provided with the **Generic Data Links** library, which includes 23 links and six miscellaneous links.