

FUNCTION GENERATION TOOL GUIDELINE

Releases	Date	Author	Description
Function_tool_R1.1	23/10/2020	Ha Tien Tai (RBVH-EDA23)	First release

Contents

I. Overall view	3
II. FILE STRUCTURE	5

I. Overall view

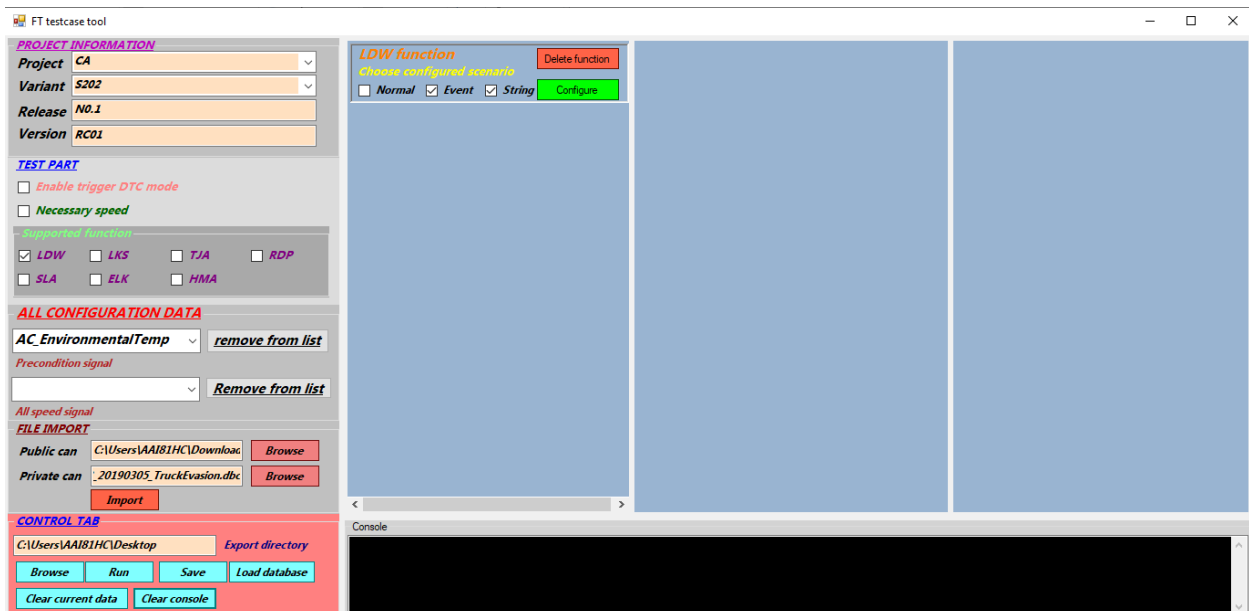
The screenshot shows the 'FT testcase tool' window. It features a sidebar on the left with five main sections, each highlighted with a colored circle and a number: 1 (Project Information), 2 (Test Part), 3 (All Configuration Data), 4 (File Import), and 5 (Control Tab). The main area on the right is divided into three vertical panels, with the top panel highlighted by a cyan circle and the number 6. At the bottom right, there is a 'Console' window, highlighted by a black circle and the number 7. The interface includes various input fields, checkboxes, dropdown menus, and buttons for managing test data.

- **(1) Project Information:** Provide all textbox to input the mandatory information about the project
- **(2) Test Part:** Include all the relevant checkbox when performing function test, content for each checkbox will pop up a small window in section 6 when they are ticked
- **(3) All Configuration Data:** These scroll down lists include all pre-condition signals and speed signals. About function data, they will be display when the relevant pop up window show up
- **(4) File Import:** The location to input the directory of public-can and private-can dbc file
- **(5) Control tab:**
 - Export directory: the output directory of your xml file, the name of the xml will have the format project_variant_release_version_labT.xml
 - Run button will execute the procedure outputting the xml file with all the current configured data
 - Save button will save all current data to the csv file, the file structure for this tool operation will be mentioned in the part II
 - Load Database button will load all the data in csv file that relevant to the current project and variant

- Clear current data button: this one is too obvious 😊
- Clear console: so is this one
- **(6) Content window:** This window contents all the pop up window when you tick a checkbox in Test Part section
 - **Column 1:** Content the function scenario as show below

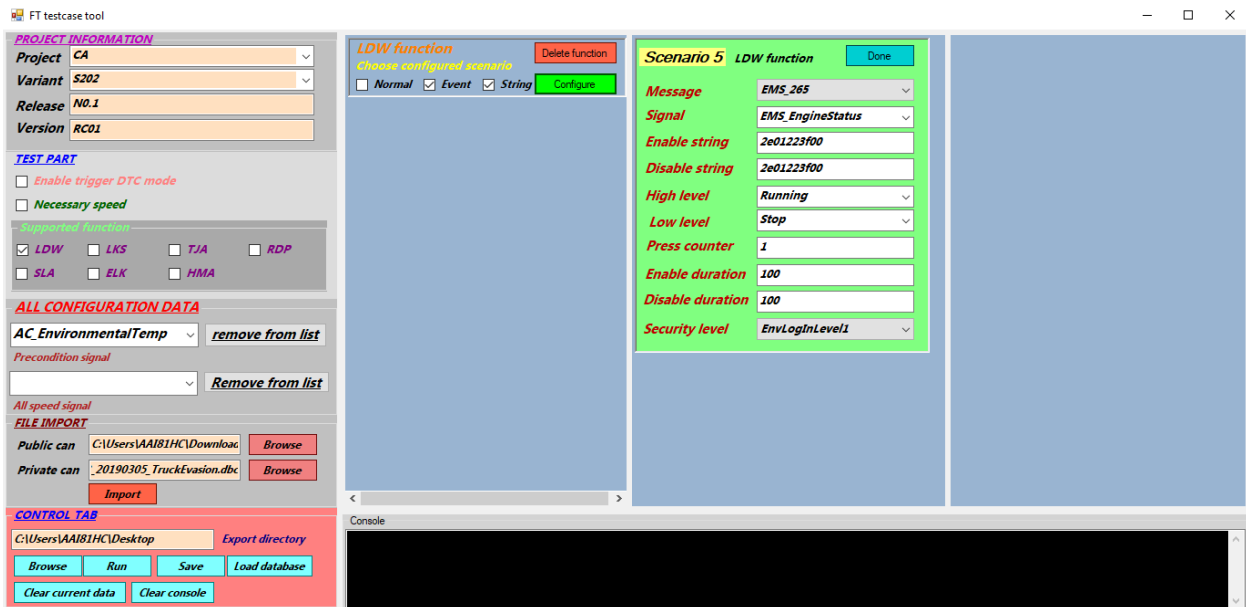


- If there's any pre-configured scenario for that function, the relevant checkbox will be ticked

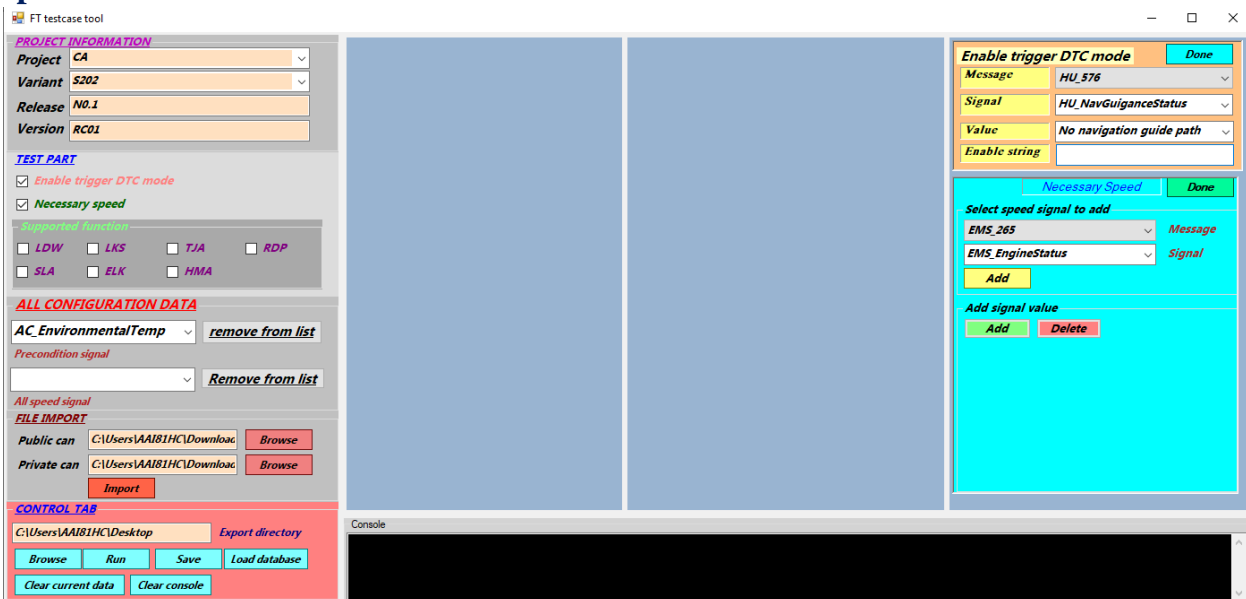


Example for LDW function with scenario 5 (event and string)

- **Column 2:** Content the configuration window relevant to that scenario when you press Configure



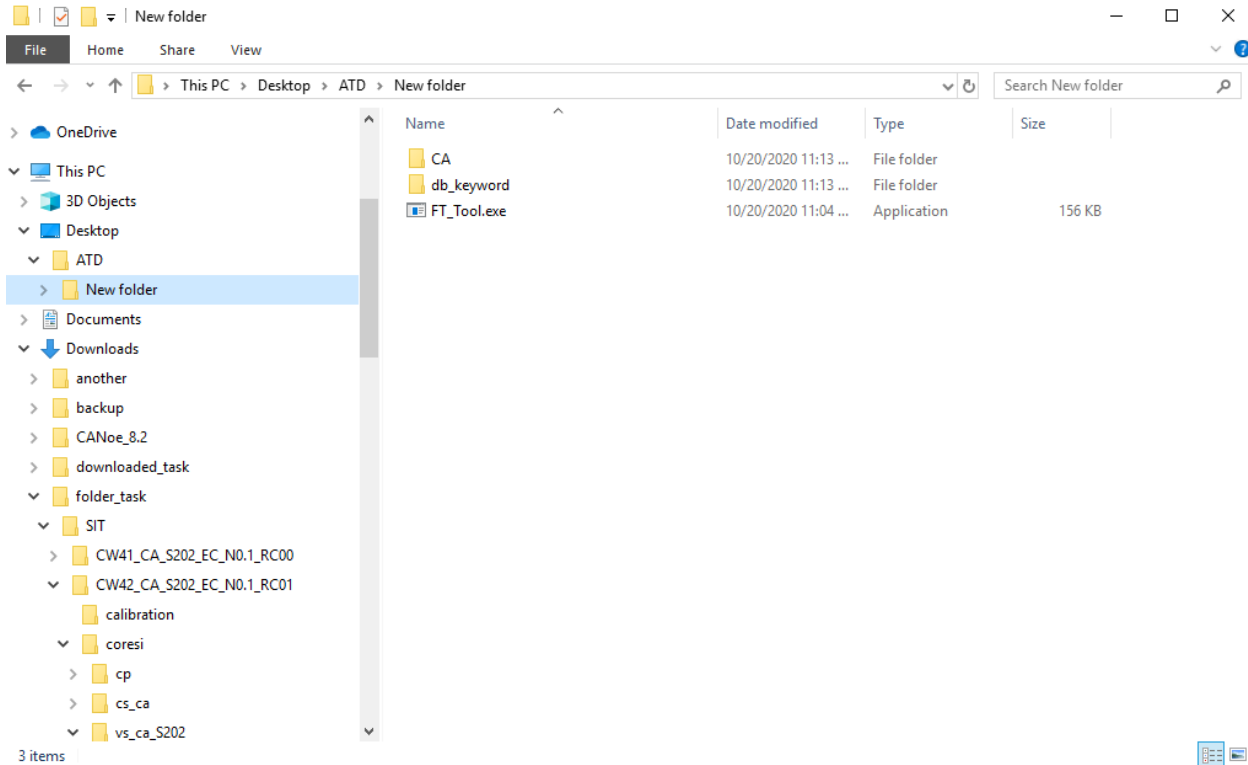
- **Column 3:** Content the configuration window for trigger dtc mode and necessary speed



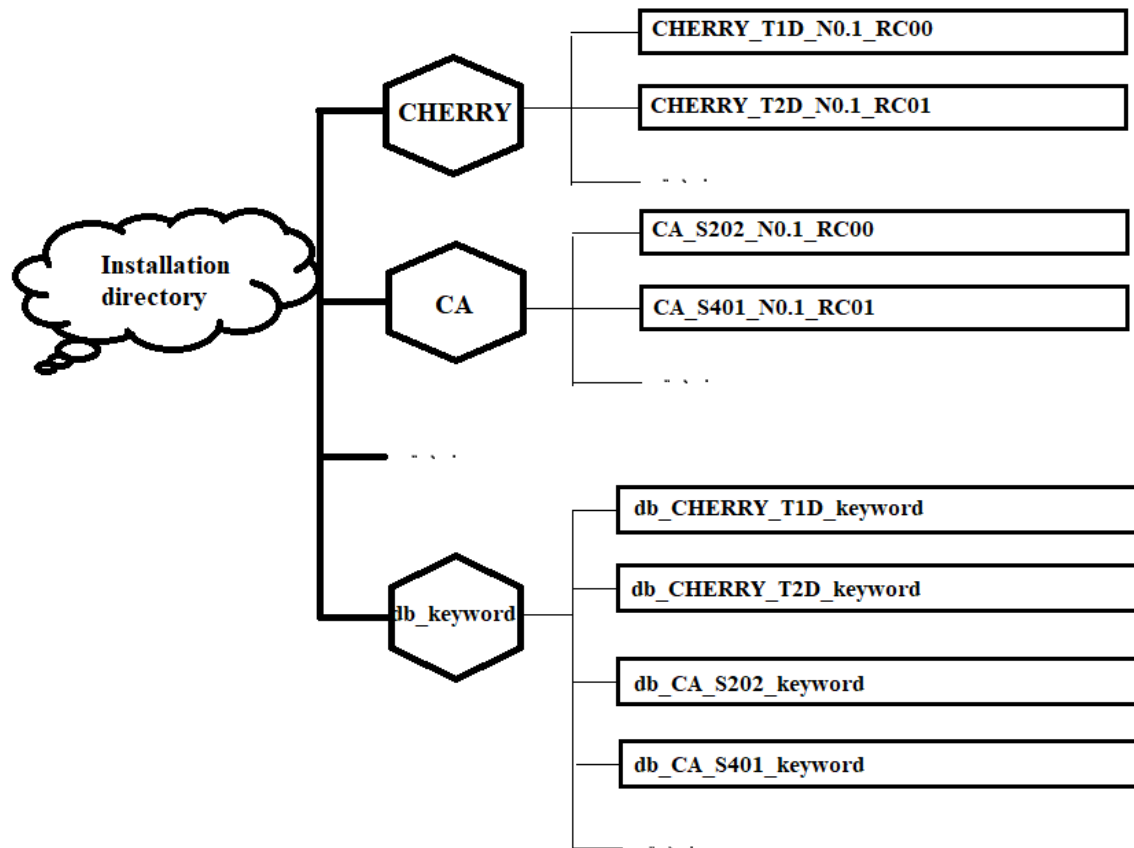
- **(7) Console window:** print out all error and configuration status throughout setting procedure

II. FILE STRUCTURE

- For each particular project and variant, this tool will generate two csv file. One is for all the setting data, the second is for the necessary keyword when importing dbc file
- All the file is available in the same directory with exe file



- The example file structure can be summarized in the below picture



III. EXAMPLE SETTING

- In this part we will perform the example setting to get the general idea of this tool
- We use project CA_S202_N0.1_RC01 for example
- First fill in all the textbox in project information section

FT testcase tool

PROJECT INFORMATION

Project: CA
Variant: S202
Release: N0.1
Version: RC01

TEST PART

☐ Enable trigger DTC mode
☐ Necessary speed

Expected functions

☐ LDW ☐ LKS ☐ TJA ☐ RDP
☐ SLA ☐ ELK ☐ HMA

ALL CONFIGURATION DATA

Precondition signal: [dropdown] [remove from list](#)
All speed signal: [dropdown] [Remove from list](#)

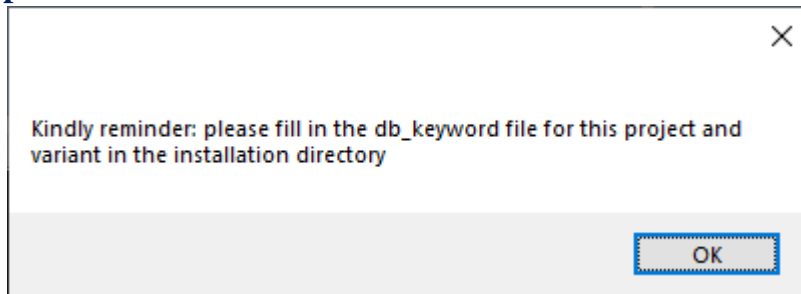
FILE IMPORT

Public can: [text] [Browse](#)
Private can: [text] [Browse](#)
[Import](#)

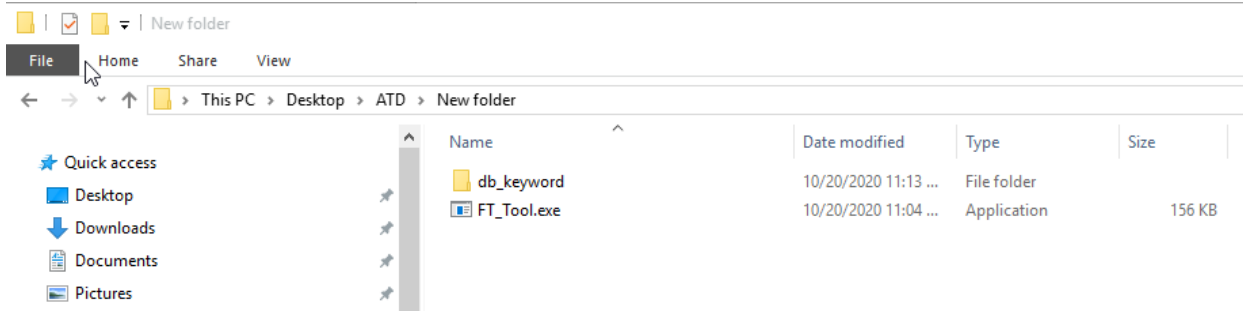
CONTROL TAB

[Export directory](#)
[Run](#) [Save](#) [Load database](#)
[Clear current data](#) [Clear console](#)

- Next we choose the directory of the public and private can, you can use browse button to set the link or just paste the link if you have it already. Notice the small popup window like this will appear



- After this state, the tool has generated the keyword file for this project and variant



- Go in db_keyword folder and open file db_CA_S202_keyword.csv. It has the default format like this

	A	B	C	D	E	F	G	H	I
1	Node keyword (format: private first-public second)								
2	keyword to search for name***	keyword to search for unit***	physical value***						
3									
4									
5									
6									
7									
8									
9									

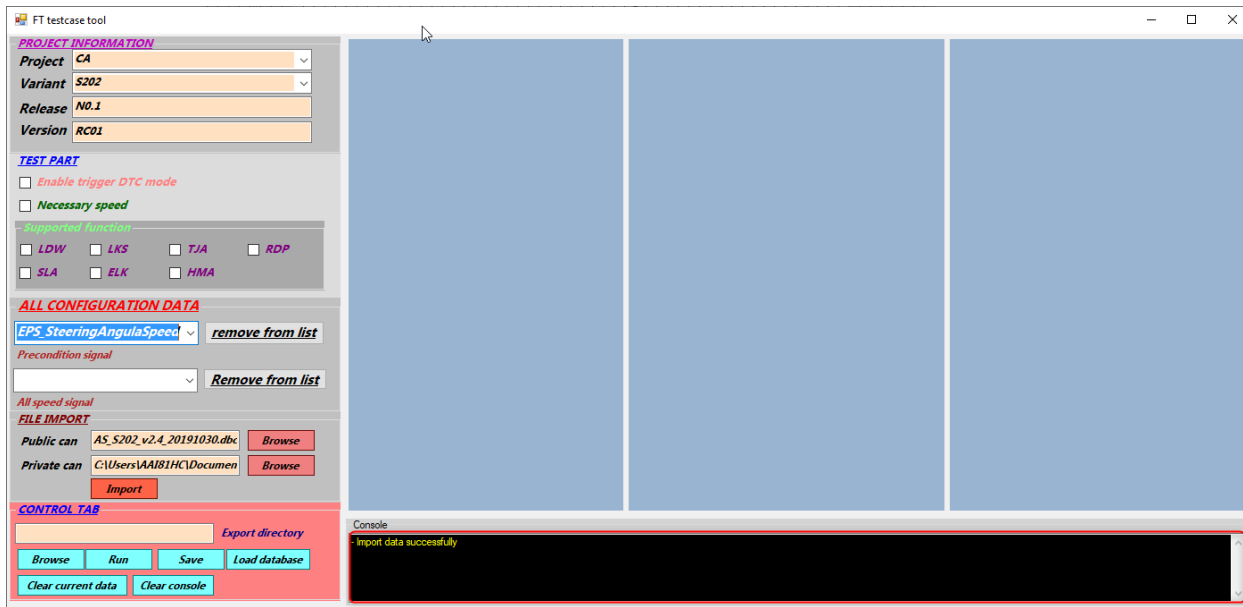
- The main purpose of this file is to list out which ECU node we want to read in the dbc file and automatically select some precondition signal. Because the name of the MPC node may be different for each project and variant so these node keyword is mandatory
- At B1 we fill in the node in the private-can dbc and C2 for the public one

	A	B	C	D	E	F	G	H	I
1	Node keyword (format: private first-public second)	MPC2	LAS						
2	keyword to search for name***	keyword to search for unit***	physical value***						
3									
4									
5									
6									
7									
8									
9									

- Now we need to specify keyword for precondition signal so that the tool can automatically filter it out. Example like this

	A	B	C	D	E	F	G	H	I
1	Node keyword (format: private first-public second)	MPC2	LAS						
2	keyword to search for name***	keyword to search for unit***	physical value***						
3	speed	km/h	85						
4	yaw	degree/s	0						
5	accel	m/s2	0						
6	temp	degree C	23						
7	bartorque	Nm	1						
8	Steeringangle	degree	0						
9	steeringangula	deg/s	0						
10									
11									

- You don't have to remember exactly the signal name, you can fill in a keyword in that signal. This also apply for unit keyword. If a signal fulfill both requirement about name and unit, it will be selected at precondition signal. Save the file and back to the tool
- After you fill in the directory for public and private can press import. If succeed, it will print a small notification on console



- Now drop down the precondition list you can see multiple signals which not quietly what you want. Because the may be some signal that can both fulfill the name and the unit keyword



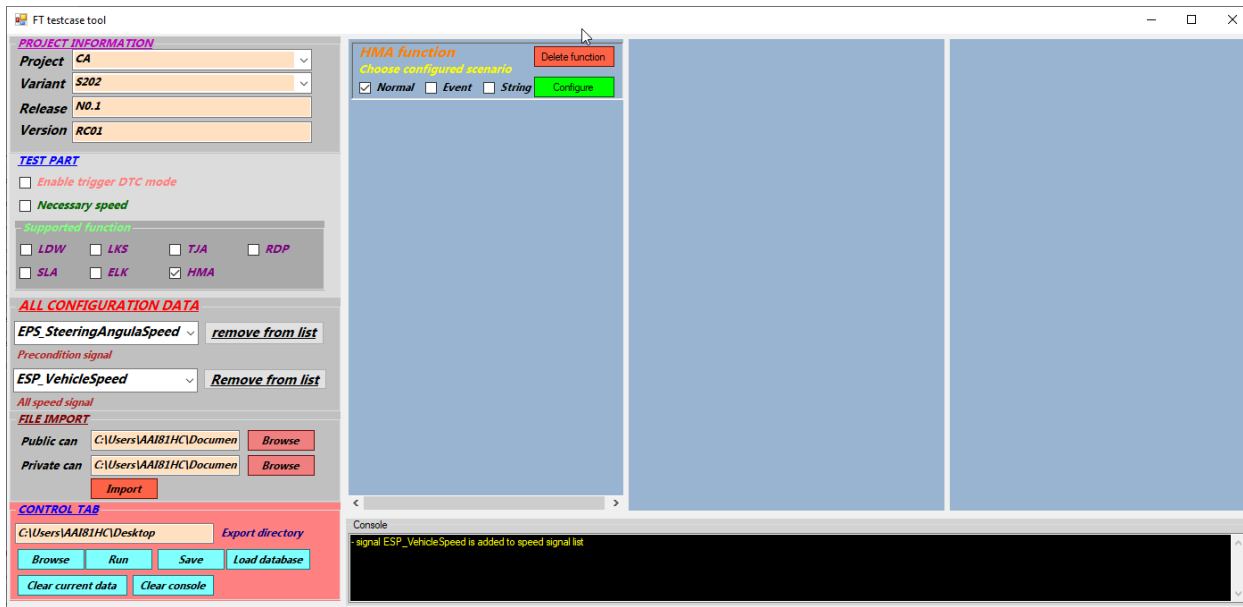
- Here you can select the signal you don't need and remove it from the list with the button on the right. So to get the best result, you need to specify the name and unit at much at possible
- We head to the test part. Tick the enable trigger DTC mode

The screenshot shows the 'FT testcase tool' window. On the left, there's a sidebar with sections: 'PROJECT INFORMATION' (Project: CA, Variant: S202, Release: N0.1, Version: RC01), 'TEST PART' (Enable trigger DTC mode checked, Necessary speed unchecked), 'ALL CONFIGURATION DATA' (EPS_SteeringAngulaSpeed, Precondition signal), 'FILE IMPORT' (Public can, Private can), and 'CONTROL TAB' (Export directory, Browse, Run, Save, Load database, Clear current data, Clear console). The main area is divided into three panels. The rightmost panel is titled 'Enable trigger DTC mode' and contains a 'Done' button, a 'Message' dropdown (GW_514), a 'Signal' dropdown (HU_IACCEnable), a 'Value' dropdown (On), and an 'Enable string' field. The bottom panel is a 'Console' showing 'Import data successfully'.

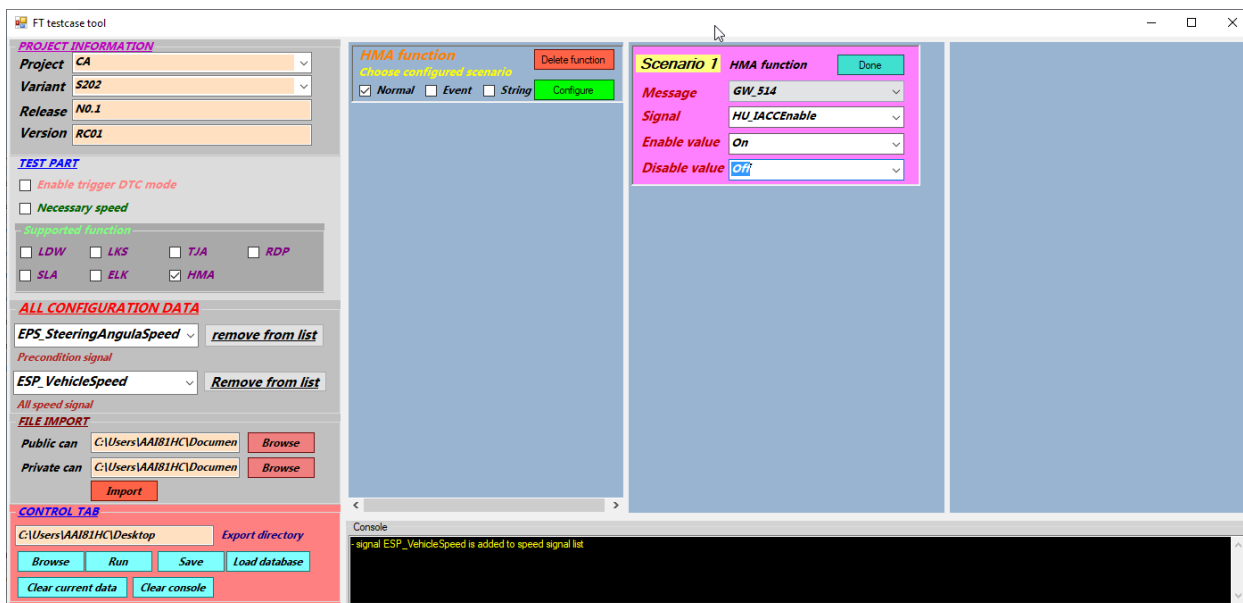
- At here you have two drop down list for message and signal so that you can find the signal easier. *Small tip is that you can paste the signal name in signal drop down list and press enter, it will automatically search that signal for you.*
- After you finish configuring that signal, you can press done
- In the necessary speed tag, you need to add speed signal. It will be updated in all speed signal drop down list. After that you have to specify all the speed value you want to set for that signal

The screenshot shows the 'FT testcase tool' window with the 'Necessary Speed' configuration window open. The sidebar is the same as the previous screenshot. The 'TEST PART' section now shows 'Necessary speed' checked. The 'Necessary Speed' window has a 'Done' button, a 'Select speed signal to add' section with 'ESP_218' and 'ESP_CheckSum' dropdowns, an 'Add' button, and an 'Add signal value' section with 'Add' and 'Delete' buttons and three input fields for speed values (50, 70, 150) labeled 'speed 1', 'speed 2', and 'speed 3'. The 'Console' at the bottom shows 'signal ESP_VehicleSpeed is added to speed signal list'.


- After you finish press done
- Next select the supported function you want to turn on and off and select the scenario for that function



- Press configure, find the signal you want and set all the necessary data in the textbox. The search feature is also worked here



- After that press done
- At this point, you should press save button so that all you data can be available for the next time. Or it will also save it you press run. Remember that all the necessary information must be filled before saving
- At the export directory after you hit run, you will have your script file, put this to cores and run it



The screenshot shows a Visual Studio Code editor with a file named `CA_S202_N01_RC01_lab1.xml` open. The file contains an XUnit XML test suite. The XML is structured as follows:

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <testwhish title="CA_S202_N01_RC01" version="1.0">
3 <testgroup title="Tests">
4 <externalref type="doors" owner="TAE" - DOORS extension" title="CHERY_Lab1_SIT" />
5 <testgroup title="Precondition">
6 <externalref type="doors" owner="TAE" - DOORS extension" title="CHERY_Lab1_SIT" />
7 <testgroup title="mandatory">
8 <externalref type="doors" owner="TAE" - DOORS extension" title="CHERY_Lab1_SIT" />
9 <testcase title="Tx message:EPS_2AI(ID:0x2A1)::EPS.SteeringAngularSpeed: 0" Ident="-">
10 <set title="E_pubc_EPS_EPS_2AI::EPS.SteeringAngularSpeed_Rv" Ident="-">
11 <envvar name="E_pubc_EPS_EPS_2AI_EPS_StearingAngularSpeed_Rv"/>
12 </set>
13 <wait time="5" title="wait" />
14 <set title="E_pubc_EPS_EPS_2AI_tx" Ident="-">
15 <envvar name="E_pubc_EPS_EPS_2AI_tx"/>
16 </set>
17 <wait time="5" title="wait" />
18 <set title="E_pubc_EPS_EPS_2AI_tx" Ident="-">
19 <envvar name="E_pubc_EPS_EPS_2AI_tx"/>
20 </set>
21 </testcase>
22 <testcase title="Tx message:EPS_2AI(ID:0x2A1)::EPS.MeasuredTorsionBarTorque: 1.00" Ident="-">
23 <set title="E_pubc_EPS_EPS_2AI_EPS_MeasuredTorsionBarTorque_Rv" Ident="-">
24 <envvar name="E_pubc_EPS_EPS_2AI_EPS_MeasuredTorsionBarTorque_Rv"/>
25 </set>
26 <wait time="5" title="wait" />
27 <set title="E_pubc_EPS_EPS_2AI_tx" Ident="-">
28 <envvar name="E_pubc_EPS_EPS_2AI_tx"/>
29 </set>
30 <wait time="5" title="wait" />
31 <set title="E_pubc_EPS_EPS_2AI_tx" Ident="-">
32 <envvar name="E_pubc_EPS_EPS_2AI_tx"/>
33 </set>
34 </testcase>
35 <testcase title="Tx message:EPS_2AI(ID:0x2A1)::EPS.SteeringAngle: 0.00" Ident="-">
36 <set title="E_pubc_EPS_EPS_2AI_EPS_StearingAngle_Rv" Ident="-">
37 <envvar name="E_pubc_EPS_EPS_2AI_EPS_StearingAngle_Rv"/>
38 </set>
39 <wait time="5" title="wait" />
40 <set title="E_pubc_EPS_EPS_2AI_tx" Ident="-">
41 <envvar name="E_pubc_EPS_EPS_2AI_tx"/>
42 </set>
43 <wait time="5" title="wait" />
44 <set title="E_pubc_EPS_EPS_2AI_tx" Ident="-">
45 <envvar name="E_pubc_EPS_EPS_2AI_tx"/>
46 </set>
47 </testcase>
48 <testcase title="Tx message:GM_270(ID:0x270)::AC.EnvironmentalTemp: 23.0" Ident="-">
49 <set title="E_pubc_GM_GM_270_AC_EnvironmentalTemp_Rv" Ident="-">
50 <envvar name="E_pubc_GM_GM_270_AC_EnvironmentalTemp_Rv"/>
51 </set>
52 <wait time="5" title="wait" />
53 <set title="E_pubc_GM_GM_270_tx" Ident="-">
54 <envvar name="E_pubc_GM_GM_270_tx"/>
55 </set>
56 </testcase>

```

The XML file is located at `C:\Users\AAIBTHC\Desktop\CA_S202_N01_RC01_lab1.xml`. The editor shows the file's content with line numbers on the left and a search bar at the top right.

In the next time when you run the tool you just need to select the project and variant, after that press load database and all the data that you previously configure will be available

FT testcase tool

PROJECT INFORMATION

Project CA

Variant S202

Release

Version

TEST PART

☐ Enable trigger DTC mode

☐ Necessary speed

Expected function

☐ LDW ☐ LKS ☐ TJA ☐ RDP

☐ SLA ☐ ELK ☐ HMA

ALL CONFIGURATION DATA

remove from list

Precondition signal

Remove from list

All speed signal

FILE IMPORT

Public can Browse

Private can Browse

Import

CONTROL TAB

Export directory

Browse Run Save Load database

Clear current data Clear console

Console