

OpenDSS Cheatsheet

Examples of Commonly-Used OpenDSS Commands

Command	Description
New Circuit.MyCircuit	Makes a new circuit with a 115 kV Vsource connected to
	SourceBus
New Circuit.MyCircuit BasekV=69 pu=1.05	Makes a new circuit and changes the voltage to 69 kV, 1.05 pu
New Circuit.MyCircuit	Makes a new circuit with a 345 kV source, 1.05 pu,2000 MVA 3-
~ BasekV=345 pu=1.05	phase fault (default Base MVA for puZ1 property is 100 MVA)
~ puZ1=.05 pu Z0=.08	
Solve	Solves the present circuit with the present solution mode
Solve Mode=Yearly	Sets the mode to Yearly and executes the default 8760-hour
	yearly solution model algorithm
	Note: Changing the Mode resets all Monitor and EnergyMeter
	Object
Solve Mode=Dutycycle Stepsize=2s number=30000	Set the mode to Dutycycle and solve 30000 time steps at a 2 s
	interval.
Solve Mode=Daily Stepsize=15m number=96	Set the mode to Daily and solve 96 steps at a 15 min interval (one
	full day)
Set Mode=dynamic Stepsize=.001 number=2000	In two commands, set the mode to Dynamic and then solve 2000
Solve	steps at a 1 ms interval (the default unit on Stepsize option is sec)
Solve Mode=FaultStudy	Solve the present circuit in FaultStudy mode and display the fault
Show Faultstudy	current. Includes Loads as linear elements if they are defined and
	enabled.
New Fault.F1 Bus1=MyBus.2	Add a SLG fault to phase 2 of MyBus (phases=1 by default).
Solve Mode=Dynamic Number=1 Stepsize=0.00001	Solve 1 timestep in Dynamic mode at a tiny timestep.
	This captures Generator contribution to fault.



Solution Modes:	1. Snapshot, (DEFAULT mode)
	2. Daily,
	3. Yearly),
	4. DIrect,
	5. DUtycycle,
	6. Time, (see LoadShapeClass option)
	7. DYnamic,
	8. Harmonic,
	9. M1 (Monte Carlo 1),
	10. M2 (Monte Carlo 2),
	11. M3 (Monte Carlo 3),
	12. Faultstudy,
	13. MF (monte carlo fault study)
	14. Peakday,
	15. LD1 (load-duration 1)
	16. LD2 (load-duration 2)
	17. AutoAdd (see AddType)
Plot Circuit Power Max=2000 C1=blue 1ph=3	Plots the active circuit (if bus coordinates defined) with line thicknesses
Plot Circuit Power Max=2000 CI=\$00FF0000 1ph=3	proportional to power. $2000 \text{ kW} = \text{width of 7. Colors may be specified as}$
	standard color name or RGB integer number (Hexadecimal format shown:
	00BBGGRR)
Standard Color Names	Black Maroon
	Green Olive
	Navy Purple
	Teal Gray
	Silver Red
	Lime Yellow
	Blue Fuchsia
	Aqua LtGray
	DkGray White



plot profile phases=primary plot profile phases=all plot profile phases=1	Various Plot Profile command examples. Requires an Energymeter object at the head of the feeder.
New Line.MyLine Bus1=FromBus.1.2.3	Typical Line object definition. A good idea to define units on both the
Bus2=ToBus.1.2.3	LineCode and Line objects. Units don't have to match. OpenDSS will take
~ LineCode= 336ACSR Length=2.5 Units=Mi	care of conversions
New Load.MyLoad Bus1=LoadBus.1.2.3 kW=250	Typical Load definition (3-phase is default). Don't forget kV property.
PF=.95	For phases > 1 , specify L-L kV.
~ kV=12.47	
New Load.My1phLoad Phases=1 Bus1=SecBus.1	25 kW 240-V single-phase load
kW=25	
~ kV=0.24 PF=0.99	
New Load.MyLLLoad Phases=1 kV=12.47	1-phase L-L connected load at 12.47kV level
~ Bus1=MyLoadBus.1.2	
~ kW=25 kvar=5	
New Monitor.M1 Element=Line.MyLine Terminal=1	Equivalent syntax for defining a default Monitor object connected to
New Monitor.M1 Line.MyLine 1	terminal 1 of the Line object named MyLine. The default mode (0)
New Monitor.M1 Line.MyLine	captures Voltages and Currents in Magnitude/Angle at the terminal.
New Monitor.M1 Line.MyLine 1 VIPolar=No	Voltages and Currents are saved as complex numbers instead of the polar
	form of magnitude/angle
New Monitor.M1 Line.MyLine 1 Mode=1 PPolar=No	Monitor saves power at the terminal in (kW, kvar) instead of polar form of
	(kVA, angle).
Monitor Modes	0: Voltages and Currents
	1: Powers
	2: Transformer taps (Transformer elements only)
	3: State Variables (PCElements only)
	4: Flicker (Pst) of Voltages only (10 min simulation req'd)



Modified Monitor Voltage and Power Modes	Monitor Objects can capture sequence quantities directly by a bitmask adder to the Mode number. Combine with adders below to achieve other results for terminal quantities: +16 = Sequence quantities +32 = Magnitude only +64 = Positive sequence only or avg of all phases
	e.g., Mode=17 will return sequence powers
Scripting Variables (var command) <i>Introduced at version 7.6.3.32</i>	Scripting variable names must begin with "@".
var (shows all scripting variables)	When encountered as a token in the script, the variable name is replaced with the present value of the variable.
var @lastfile (reports value in Result form/interface)	Example: plot Circuit 6 max=0 dots=@dots labels=@labels subs=n object=@lastexportfile C1=\$00FF0000 C2=\$000000FF
var @dots=no @labels=yesetc	Some intrinsic variables:
(see example \rightarrow)	 @lastfile (generally set to the name of the last output file written) @lastExportfile (set to the name of the last file written by the Export command) @lastShowfile (set to the name of the last file written by the Show command) @lastPlotfile (set to the name of the last file written by the Plot command)



One-Timers

A couple of things you need to do one time. OpenDSS stores these values in the Windows registry so you don't need to do them more than once.

Command	Description
Set Editor=MyFavoriteEditorFullPathName.exe	The default text editor is Notepad, which is expected on all Windows
	installations. Use this command to change it to another editor. All text
	file reports will be automatically opened with this editor. Note that for
	creating scripts, the editor must be capable of producing plain text files
	in ANSI/ASCII encoding. It is nice to have a more powerful editor for
	working on large scripts and reports. Column selection capability is a
	great help.
Set DefaultBaseFrequency=60	Set Default Base Frequency, Hz. Sets solution Frequency and default
Set DefaultBaseFrequency=50	Circuit Base Frequency. VSOURCE objects will default to this
	frequency and impedances specified for LINE, REACTOR,
	TRANSFORMER, etc objects are assumed to be on this frequency base
	unless otherwise specified.



COM Interface Tips

Starting the In-Process COM Server

Language	Command		
VBA	Requires setting References to OpenDSSEngine.DSS		
	Public DSSObj as OpenDSSEngine.DSS		
	Set DSSObj= New OpenDSSEngine.DSS		
	DSSObj.Start(0)		
	-or-		
	Dim DSSObj as Variant		
	Set DSSObj = CreateObject("OpenDSSEngine.DSS")		
	DSSObj.Start(0)		
Matlab	DSSObj = actxserver('OpenDSSEngine.DSS');		
	Start=DSSObj.Start(0);		
Python	import win32com.client		
	Class DSS:		
	self.engine = win32com.client.Dispatch("OpenDSSEngine.DSS")		
	self.engine.Start("0")		
Delphi	(Import OpenDSSEngine Type Library into IDE)		
	DSSObject: IDSS;		
	DSSObject := coDSS.Create;		
	If DSSObject.Start(0) then;		



Starting the In-Process COM Server, cont'd

Language	Command
C#	(Project>Add Reference (Select OpenDSSEngine)
	 using OpenDSSEngine; public DSS DSSObj;
	DSSObj = new DSS(); if (!(DSSObj.Start(0)))
Python	<pre>import win32com.client Class DSS: self.engine = win32com.client.Dispatch("OpenDSSEngine.DSS") self.engine.Start("0")</pre>



Setting Some Commonly-Used Interface Variables

When the Start(0) function is called, the OpenDSS COM server creates all the interfaces. You can assign variables to this after starting the server to make access more efficient and your code more readable. VBA will be used as the example language, but it looks pretty much the same in the other languages after deleting the Set command. The Set command is fairly unique to VB and is how you instantiate *Objects*. Other languages are able to figure this out in the normal language syntax. Also, see examples for setting DSSText in the table below.

Public DSSText as Text	This variable will allow you easy access to the text interface
Set DSSText = DSSObj.Text	Command and Result properties, for example:
Var DSSText: IText;	DSSText.Command = "? Line.Myline.Bus1"
DSSText : OpenDSSEngine.Text; // Delphi	Bus1Name = DSSText.Result
DSSText := DSSObj.Text;	
	You can execute any of the OpenDSS commands with this
Public DSSText as Text; // C#	variable.
DSSText = DSSObj.Text;	
class DSS: # Python	
self.text = self.engine.Text	
Set DSSCircuit = DSSObj.ActiveCircuit	The ActiveCircuit interface is one of the most important for
	getting to various quantities via the COM interface.
Set DSSCktElement = DSSCircuit.ActiveCktElement	When any action in the program sets a circuit element active, this
	variable will give you access to the properties of this element.



Set DSSSolution = DSSCircuit.Solution	A common use of this Solution interface is to solve the circuit:
	DSSSolution.Solve
	This invokes the appropriate solve function in the program directly and should generally be the fastest way to execute a
Set DSSBus = DSSCircuit. ActiveBus	I his variable will give you rne properties of the active bus.
Set DSSControlQueue = DSSCircuit.CtrlQueue	This variable will give you access to the Control Queue in the
	program. You can see what is on the queue and push messages
	onto it.
Set DSSLines = DSSCircuit.Lines	This variable allows access to the Lines collection in the active
	circuit. You can use it to cycle through all the lines in the circuit,
	for example:.
	iLine = DSSLines.First
	While iLine>0 Do
	Do something
	iLine = DSSLines Next
	Loon
Cat DCCI and DCCCircuit Lands	This mariable allows assess to the Londs collection in the active
Set DSSLoads = DSSCircuit.Loads	This variable allows access to the Loads collection in the active
	cırcuit.
Set DSSGenerators = DSSCircuit.Generators	This variable allows access to the Generators collection in the
	active circuit.

While many users choose to drive OpenDSS with Matlab programs, the Object Browser in Microsoft tools, including Office, are generally better for exploring the contents of the OpenDSS type library and learning how to use the COM interface. The code completion feature in VBA and Visual Studio is particularly useful in showing what options are available after typing the name of the variable. Some users draft their code in VBA or C# and then translate to Matlab. There are also 3rd party Type Library or IDL



browsers/documenters that are helpful. If you are familiar with IDL, see the OpenDSSEngine.ridl file in the DLL folder under the SOURCE folder in the code.

<u>DSS</u> Interface Properties and Methods (from Excel)

😁 Object Browser			3
OpenDSSengine		▼ • ▶ ■ № ?	
	-		
Classes		Members of 'DSS'	
<pre> <globals> </globals></pre>	_	ActiveCircuit	
ActionCodes		ActiveClass	
ActiveClass		AllowForms	
🛃 Bus		Circuits	
🛃 Capacitors		🖻 Classes	
CapControlModes		ClearAll	
🖾 CapControls		🖻 DataPath	
🖾 Circuit		🔊 DefaultEditor	
😫 CktElement		DSSProgress	
🖾 CtrlQueue		🖻 Error	
🛃 DSS	=	🖻 Events	
🛃 DSS_Executive	-	🖻 Executive	
🖾 DSSElement		NewCircuit	
DSSEvents		MumCircuits	
🛤 DSSProgress		NumClasses	
🖾 DSSProperty		NumUserClasses	
🛤 Error		eset	
Generators			
🛃 Lines		ShowPanel	
LineUnits		Start	
P LoadModels		Text	
🛃 Loads		UserClasses	
LoadStatus		🔊 Version	
🛃 Meters			
P MonitorModes			
Monitors			
P Options			
PDElements			
Reclosers			
B DecControlo	Ŧ]	
Function Start(Code A	s Lo	ng) As Boolean	*
Member of OpenDSSengine.DSS			
Validate the user and s	tart	he DSS. Returns TRUE if successful.	
			*



<u>Circuit</u> Interface Properties and Methods

	OpenDSSengine 🗸 🖌 🖹 🏂 🦹
	· · · · · · · · · · · · · · · · · · ·
	Classes Members of 'Circuit'
Chief Browner	<pre> <pre> </pre> </pre>
	P ActionCodes
OpenDSSengine 🗸 🖌 🖻 🏂 🦉	ActiveClass
	Bus Enters
	📓 Capacitors 🛛 🖻 Loads
Classes Members of 'Circuit'	P CapControlModes M Losses
🛛 <globals> 🔥 🖻 ActiveBus</globals>	A CapControls 🛛 🗗 Meters
P ActionCodes M ActiveCktElement	Circuit 🖻 Monitors
ActiveClass Market ActiveClass	📓 🥵 CktElement 🔤 🖻 Name
Bus 🖻 ActiveDSSElement	🕺 CtriQueue 👘 NextElement
🖾 Capacitors 🛛 🖻 ActiveElement	S DSS NextPCElement
P CapControlModes AllBusDistances	DSS_Executive NextPDEIement
🖾 CapControls 🛛 🖻 AllBusNames	DSSElement MunBuses
💋 Circuit 🔤 🛃 AllBusVmag	E DSSEvents MunCktElements
🖉 CktElement 🛛 🖻 AllBusVmagPu	Boseries Barning Barni
🖾 CtrlQueue 🛛 🖻 AllBusVolts	B DSSProperty E ParentPDElement
🖾 DSS 🗧 🖻 AllElementLosses	Error Delements
DSS_Executive AllElementNames	
🖾 DSSElement 🛛 🖻 AllNodeDistances	e line laite de Rejourne
DSSEvents 🛛 🖻 AllNodeDistancesByPhase	
🖾 DSSProgress 🛛 🖻 AllNodeNames	Loads SaveSample
🖾 DSSProperty 🔤 AllNodeNamesByPhase	P LoadStatus PS Sensors
🖾 Error 🔤 AllNodeVmagByPhase	Meters SetActiveBus
🖾 Generators 🛛 🖻 AllNodeVmagPUByPhase	P MonitorModes
🛃 Lines 🛛 🖻 Buses	Monitors SetActiveClass
P LineUnits Capacitors	P Options
🗗 LoadModels 👘 🗠 Capacity	🗱 PDElements 🔤 Settings
🛃 Loads 🛛 🖻 CapControls	🛃 Reclosers 🛛 🖻 Solution
🗗 LoadStatus 🛛 🔊 CktElements	RegControls 🖻 SubstationLosses
🗱 Meters 🖻 CtrlQueue	Relays 🔊 SwtControls
🗗 MonitorModes 👘 Disable	SystemY 🖄 Sensors
🖏 Monitors 🛛 🗠 Enable	Settings B Topology
P Options S FirstElement	Solution 📷 TotalPower
PDElements SirstPCElement	👹 SolveModes 📷 Iranstormers
🗱 Reclosers 🖉 🔊 FirstPDElement	a swicontrois Supportestorage
M BasCantrala Canaratara	
Class Circuit	Class Circuit
Member of <u>OpenDSSendine</u>	Member of <u>OpenDSSengine</u>

M. Ohiard