



简明用户指导  
**The Open Distribution System Simulator™  
(OpenDSS)**

作者: Jason Sexauer, OpenDSS User

翻译: 李慧娟



## License

---

Copyright (c) 2008-2012, Electric Power Research Institute, Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the Electric Power Research Institute, Inc., nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY Electric Power Research Institute, Inc., "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL Electric Power Research Institute, Inc., BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.



## 目录

---

<b>LICENSE</b> .....	<b>3</b>
<b>目录</b> .....	<b>5</b>
<b>图形</b> .....	<b>7</b>
<b>什么是 OPENDSS?</b> .....	<b>8</b>
简介 .....	8
简单的历史 .....	8
这个文档的目的 .....	9
<b>OPENDSS 的电路模型</b> .....	<b>10</b>
母线 .....	10
电力传输元件 .....	10
电力转换元件 .....	11
辅助元件 .....	12
<b>使用 OPENDSS 的图形用户界面(GUI)</b> .....	<b>13</b>
用户界面 .....	14
工作流程 .....	14
<b>OPENDSS 编程基础</b> .....	<b>18</b>
命令语法 .....	18
命令动词 .....	18
参数 .....	18
COMMENTS 注释 .....	20
多行命令 MULTI-LINE COMMANDS .....	20
包含外部文件 .....	20
工作流程 .....	20
代码范例 .....	22
<b>COM 接口介绍</b> .....	<b>24</b>
开始使用 COM 接口 .....	24
TEXT 接口介绍 .....	25
电路接口 (CIRCUIT INTERFACE) 介绍 .....	26
算法和解的接口 (SOLUTION INTERFACE) 介绍 .....	26
VISUAL BASIC 例子 .....	28
MATLAB 例子 .....	30
PYTHON 例子 .....	32
<b>其他资源</b> .....	<b>34</b>
下一步该去哪儿? .....	34
参考资源 .....	35
更多帮助 .....	35



# 图形

---

图 1: 母线 ..... 10

图 2: 电力传输元件 ..... 11

图 3: 电力转换元件 ..... 11

图 4. OPENDSS 用户界面 ..... 14

图 5. 显示线路的电压..... 16

图 6. 电压 剪影（VOLTAGE PROFILE ） ..... 16

图 7. 线路损失 ..... 17

图 8. 电路示范 ..... 22

# 什么是 OpenDSS?

---

## 简介

The Open Distribution System Simulator (OpenDSS 或 DSS) 是一个全面的电力系统配电网仿真工具。OpenDSS 是开放源代码的配电网仿真计算分析软件。它既可以作为一个独立的执行程序使用，又可以作为动态链接库（DLL）被很多其他软件平台调用。作为独立执行程序使用时，OpenDSS 有一个文本用户界面供使用者建立基于文本的模型和察看仿真计算结果。

这个软件支持几乎所有常用的用于配网规划分析的基于有效值（RMS）的稳态分析（即频域分析）。并且它还支持许多有关智能电网的新分析来迎接电网分析的未来需要。这个软件的特性包括：支持分布式发电分析；支持电能传送的效率分析；智能电网应用；谐波分析。这个软件设计的有良好的可扩充性，所以可进行修改升级以适应未来需求。

DSS 的另一个特点是它可以进行连续时间的系统仿真计算, 比如，它可以仿真计算一个电路在一年的情况。这个软件内嵌仿真计算模式，比如：

- 帧计算 (snapshot)
- 天计算(daily)
- 年计算(yearly)
- 谐波分析 (harmonic)
- 动态分析(dynamic)
- 故障（短路）分析(fault)
- Monte Carlo 故障分析 (M1,M2,M3)
- 等等 ...

这些模式因开发者在不同研究项目中需要而被先后加入这个软件。然而，开发者并不能预测使用者的所有需求，所以这个软件还提供了组建对象模型（Component Object Model (COM)）接口，DSS 作为 DLL，让用户编写其他程序调用以满足不同的需要。通过 COM 接口，用户可以用其他程序设计执行不同的客户解决方案 和执行仿真计算，包括可以通过这种方式定义系统模型。这样，DSS 可以独立于任何的数据库或者固定的文本电路模型定义。例如，它可以在 MS Office 中用 VBA 驱动， 或者其他有 COM 功能的第三方分析软件驱动。用户很普遍的用 Mathworks MATLAB，Python, C#, R, 还有其他软件语言来驱动 DSS。这个特性提供了一个强大的外部分析功能和很好的结果图形展示。

## 简单的历史

最早 OpenDSS 的开发始于 1997 年四月在 Electrotek Concepts, Inc. 当时这个程序被称为“DSS”，配网仿真计算器。Roger Dugan 是主要开发者，随后 Tom McDermott 加入。他们俩组成了开发团队直至 2001 年 Tom 离开 Electrotek。Roger 继续维护升级这个程序到最近 Tom 通过 OpenDSS 这个项目又重新回到开发团队。EPRI Solutions 在 2004 年获得了这个软件，在 2007 年它又与 EPRI 合并。在 2008 年，EPRI 发布了该软件带有开放源代码的许可，来与其他智能电网领域正在进行的电网现代化努力相合作。

这个软件上接近于谐波分析软件或者动态软件，而不是一般意义上的潮流计算软件。关于这点的一个最明显的表现是母线（bus）在这个软件中无足轻重，而在其他大部分的潮流



软件中，母线占据中心位置，其他的计算都以它为基础。在 OpenDSS 中，母线是当需要时动态生成的。对一个将被主要当作潮流计算的工具，这最初的设计可能看起来很奇怪。但是,它赋予了这个工具很大的建模灵活,尤其是它能适应各种不同的负荷模型和特殊的电路结构。用一个谐波潮流工具解正常潮流是容易的，反之则不然。。

## 这个文档的目的

这个文档希望给新用户一个快速指导。 它包括很多 OpenDSS 手册上的信息，但它并不全面深度覆盖这个软件的使用。有兴趣的用户可以参阅文中列出的其他资料以获得更多信息。

## OpenDSS 的电路模型

OpenDSS 包含一个基于有效值（RMS）的稳态配网模型和一个连接这些电网元件控制的内部通讯系统。最基本的电路模块是电力传输元件，比如传输线，变压器，电容器，和电力转换元件，例如发电机和负荷。其他一些辅助模块，例如控制，发电或负荷曲线形状，计量器，参数提取，它们可以进一步细化电力传输和转换元件的定义。更多这些模块的信息可以在用户手册中发现。通过这些元件，代表一个相连系统的母线和节点会被动态生成，这和传统的以母线为中心的潮流计算工具根本的不同。

### 母线

母线可以有 [1..N]个节点. 母线是电路元件的连接点。在很多电力系统分析软件中，母线和节点基本是同一概念，但是它们在 OpenDSS 中是完全不同的。母线包含节点，就是说，一个母线可以有多个节点。

Node 0 总是被作为一个母线的参考节点/接地节点/0 V 节点。其余节点传统上是其他相，如， node 2 是 B 相。当指定一个母线的位置时，包含的节点要被包括。（ node 0 除外，它被认为是一直存在）例如，指定一个 3 相母线 *BUSNAME*,

`BUSNAME.1.2.3`

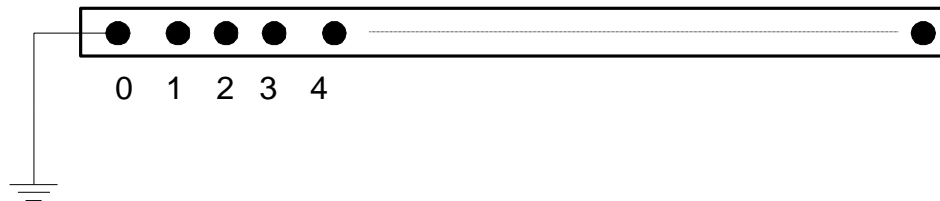


图 1: 母线

### 电力传输元件

电力传输(PD)元件 通常是由两个多更多终端组成。它们的基本功能是从一点向其他点运送电能。电力系统中，最普通的电力传输元件是线路和变压器。所以它们一般含有多个终端（电容器和电感器当并联入电网而不是串联时是个例外）。

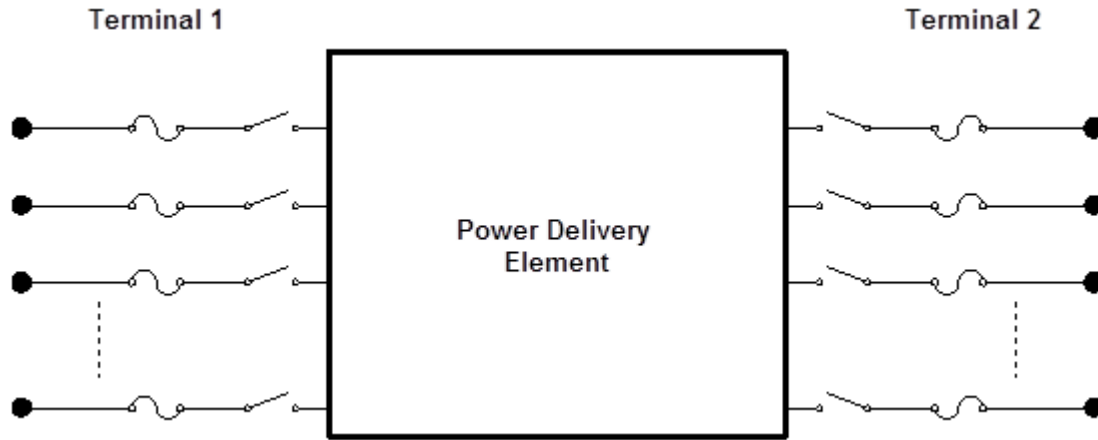


图 2: 电力传输元件

## 电力转换元件

电力转换 (PC) 元件把电能转换为其他能量或反之。有些可能能短时存储能量再提供出来，比如一些无功 (reactive power) 元件。大部分只有一个电网连接点，所以一个多相的终端。最普通的电力转换元件是发电机和负荷。

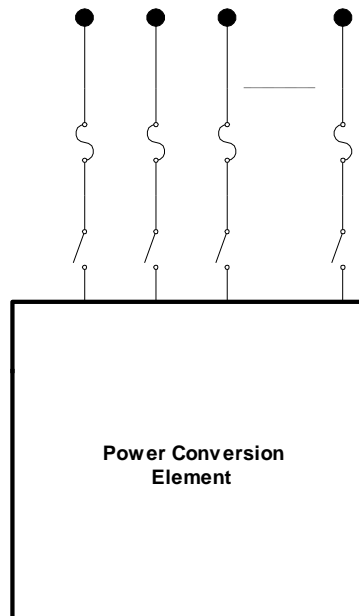


图 3: 电力转换元件

## 辅助元件

辅助元件可以进一步附加在电力传输和转换元件上。辅助元件功能包括：方便的提取系统数据参数，提供控制功能，提供测量，或提供相应的曲线信息以便时间段仿真计算使用。最常用的有：

- **LineCode** – 定义线路的类型，比如线路的阻抗.
- **LineGeometry, LineSpacing, and WireData** – 一起使用，它们共同定义线路的类型基于一些最基本的物理参数，如线路的几何平均半径（ Geometric Mean Radius (GMR) ）。
- **LoadShape** – 定义负荷曲线的相乘因子或实际 kw/kvar 以用于时间段仿真计算
- **Spectrum** – 定义一个电力传输元件发出的谐波的频谱
- **EnergyMeter** – 用于收集一条馈线上的不同计量信息
- **Monitor** – 用于收集具体元件的潮流计算结果
- **CapControl and RegControl** – 模拟可控电容器和可控线路电压调压器的控制

## 使用 OpenDSS 的图形用户界面(GUI)

---

OpenDSS 包含一个图形用户界面(GUI)，它提供了一个编写和分析电力系统案例的结构化的环境。图形用户界面(GUI) 是用户使用 OpenDSS 仿真计算引擎的两种基本方式的一种，另一种是通过组建对象模型（COM）接口。在随后将会介绍这种方式，更多的细节在 OpenDSS 用户手册（OpenDSS Manual）。

应该指出的是 GUI 更是一个辅助电路分析生成，调试代码的工具，而不是取代代码编程。用户传输信息给 DSS 根本上是通过传送给 OpenDSS 命令处理器的文本来实现的。在 DSS 的 GUI 中，代码是这样执行的：

1. 选中要执行的代码
2. 用户然后右击鼠标选择 “Do Selected” 这个选项,它的快捷键组合是 Ctrl-D.
3. 选择的代码也可以通过 “Do” 菜单或者这个菜单下的快速按钮.

所有通过 GUI 执行的命令都有对等的 OpenDSS 编程命令。这些命令都详细的记录在 OpenDSS Manual 中。并且这些命令和元件属性都可以在用户界面的 “help” menu 找到。你也可以使用 “Edit” menu 下的 “Record Script” 工具，把在 GUI 中执行的命令记录到一个 dss 文件中。

仿真计算的结果可以存储在 CSV 文件中。OpenDSS 提供几个标准格式的文本报告文件（参考 Show 和 Export 命令）。用户如果需要更复杂的报告文件，可以使用 Excel 或者其他的应用程序通过 COM 接口来控制 OpenDSS 完成。（随后将介绍 COM 接口）。

## 用户界面

当打开 OpenDSS 时, 你首先看到的是下面的界面:

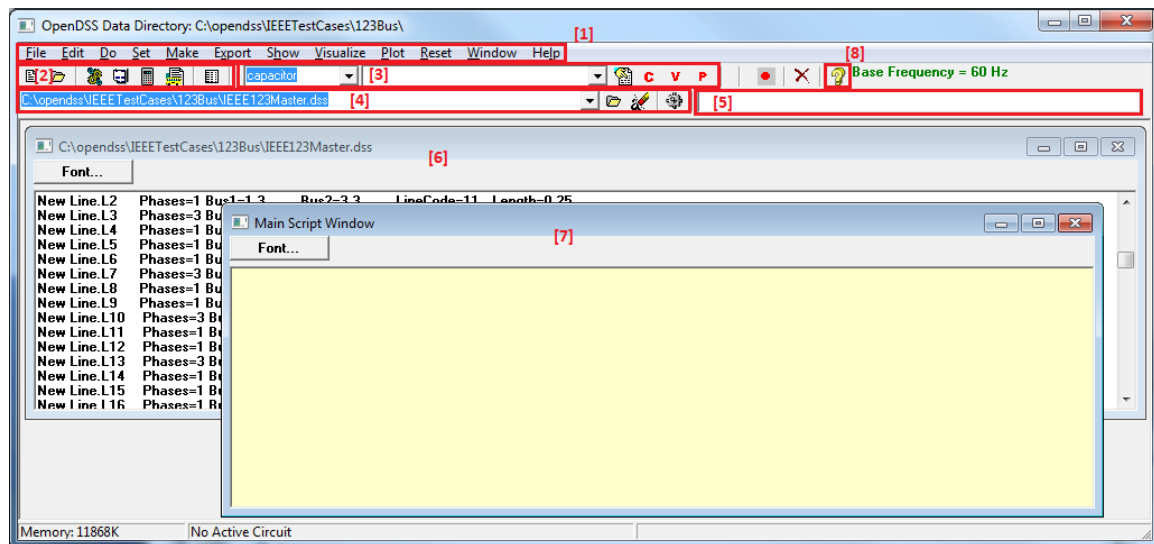





图 4. OpenDSS 用户界面

主要组件包括: (序号和上图号码相对应)

1. 菜单结构, 它来驱动 OpenDSS 中主要的工作流。菜单包括:
  - **Set** 菜单, 这里你可以设置仿真计算参数, 这些参数也可以通过编程设置 options 中的参数 来完成
  - **Export** 菜单, 这个命令用来保存各种输出报告到 CSV 文件。
  - **Show** 菜单, 它和 Export 菜单命令输出的文件信息基本相同, 不同的是它直接在 GUI 中显示报告结果。
  - **Visualize** 菜单, 它显示选中元件 (图 4 中的 “3”) 的图像结果。
  - **Plot** 菜单, 它提供有关仿真计算电路的图像结果。
2. 工具条, 这里你可以直接使用很多常用的 OpenDSS 命令, 比如“Solve,” “Summary,” 和 “Do Command.”
3. 元件工具, 这里你可以选择电路元件以便进行编辑或者显示它的图像结果
4. 编码工具, 这里你可以从已打开的代码文件选择要执行的文件。
5. 结果栏, 这是个缩小版的 result widow。Result widow 可以通过 show 菜单下的 Result Form 显示。
6. 代码 Windows, 这里显示电路模型的 dss 文件。
7. 主代码 Window 是 OpenDSS 的交互窗口。这里用户可以键入命令行, 然后通过选中它们, 用“Do Command” 的快捷组合 (Ctrl-D) 来执行. 这里输入的命令会被保留在窗口中。
8. **Help** 按钮, 这里你可以看到 OpenDSS 的命令和元件属性的帮助参考信息。

## 工作流程

一般来说, 用户需要按以下步骤:

1. **定义电路** 定义线路，变压器，负荷，发电机，等...
  - a. 定义电路是通过编写 dss 文件实现。下一章将介绍更多关于编程的信息。
  - b. 一旦代码编写完成，可以使用  按钮运行选定的代码。
2. **设定电路的仿真计算选项**, 比如，仿真计算模式 (snapshot, daily, harmonic, 等)
  - a. 这可以通过“Set”菜单的命令来实现。最基本的仿真计算模式是 帧计算 (snapshot)，它类似于传统的潮流计算。关于更多的电路仿真计算选项，参阅 OpenDSS 手册。
3. **解潮流计算问题**
  - a. 首先，通过运行 Do 菜单下的“Calc Voltage bases”来确保生成母线节点列表和寻找到基本电压。
  - b. 然后使用  按钮来解电路。
4. **分析已解电路** – 不同的分析包括不同的使用细节。但是一些通常的任务包括：
  - a. **检查线路，变压器，负荷，等元件**– 首先，从**元件工具**中选定元件类型然后元件，使用 C, V, or P 按钮来查看电流，电压和功率。使用  按钮来编辑元件属性。图 5 显示一个查看线路电压的例子。
  - b. **图形化显示电压-距离**– 在主程序窗口中键入 plot profile, 并选中它然后使用 Ctrl-D 组合键来执行这个命令。Voltage profile 将显示沿着这个电路离开变电站电压如何变化; 图 6 给出了一个例子。Plot 命令还有其他参数，参阅 OpenDSS Manual 以获得更多详情。
  - c. **在线路连接图上 (one-line diagram) 图形化显示数据**– 如果你通过 buscoords 命令提供了母线的位置信息 (参阅 OpenDSS Manual), 已可以在这个电路的地图 (one-line diagram) 上显示潮流数据。在菜单上选择 Plot > Circuit Plots > Circuit Plot. 图 7 显示了一个例子。
  - d. **输出数据以使用其他软件分析**– 所有 OpenDSS 的结果可以通过在“Export”菜单中不同的命令来输出。结果存储为 .csv 文件。

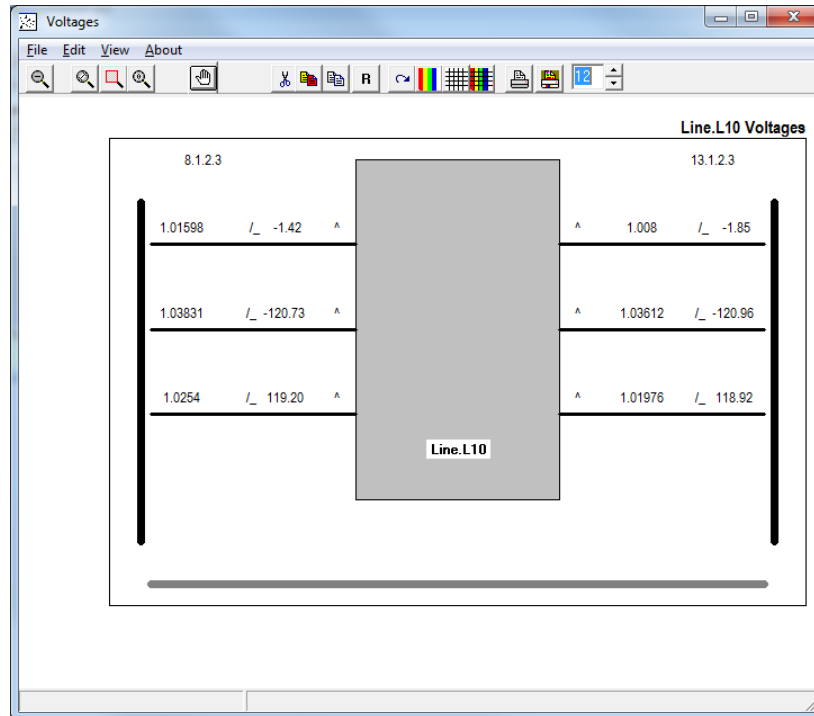


图 5. 显示线路的电压

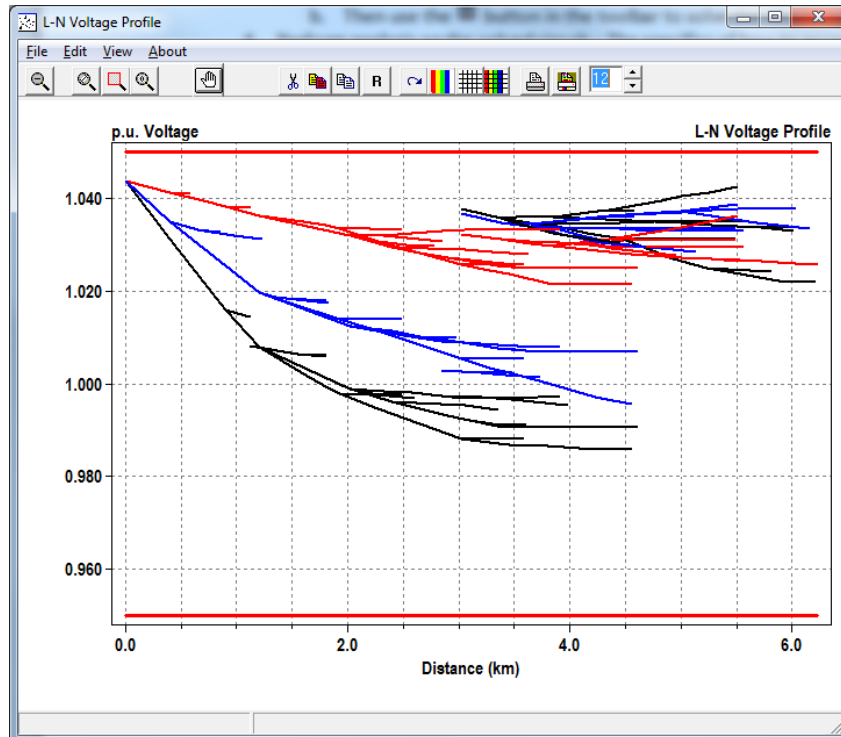


图 6. 电压 剪影 (Voltage Profile)



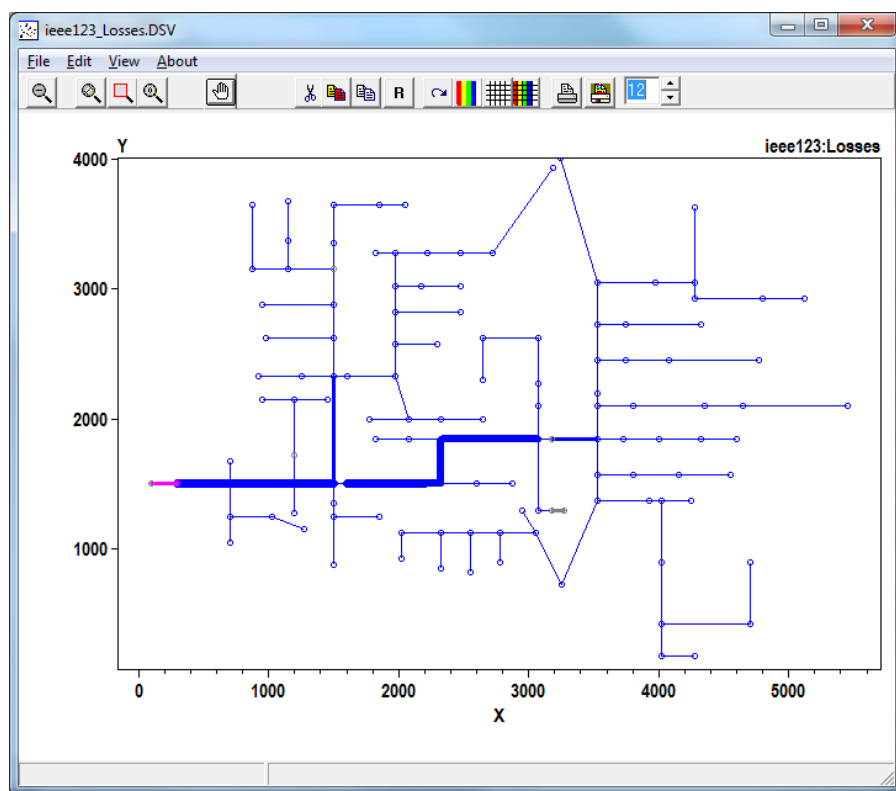


图 7. 线路损失

## OpenDSS 编程基础


---

DSS 的所有功能都能用文本编程命令来实现。这些文本可以通过以下方式:

1. 在程序窗口选择并执行,
2. 通过 COM 接口,
3. 标准文本文件, 通过 **Compile or Redirect** 命令

这些方式令使用 DSS 很方便, 无论是对一个小型简单的电路分析还是一个复杂的分析。这同时也为使用其他软件的用户提供了便利。

OpenDSS 的编程语言是区分大小写的 ( case insensitive ) 。

请查看 **Help** 命令来获得当前最新的命令和属性 (通过菜单上的  button)。

### 命令语法

命令是一个单行的编程文本. 有些命令动词直接作用于电路元件(比如 **New** and **Edit** 命令)。这类型的命令 格式如下:

```
CommandVerb (命令动词) ElementClass.Element.Param1=Val1, Param2=Val2
```

有的命令动词不直接作用于电路元件 (比如 **Plot** 和 **Export** 命令)。这类型的命令有如下格式:

```
CommandVerb Param1=Val1, Param2='Value 2', Param3=(1 2 3)
```

### 命令动词

一个完整的命令动词列表可以通过界面上的  按钮获得。一些常用命令动词包括:

- **New** – 生成新的电路元件。
- **Edit** – 编辑指定的电路元件
- **Set** – 定义仿真计算计算的参数, 比如 **mode** (仿真计算模式) . 参见 “options” 在命令帮助文件中
- **Solve** – 在当前电路上进行仿真计算
- **Show** – 用文本显示计算的潮流结果。
- **Export** – 输出计算的潮流结果到 **text** 或者 **csv** 文件。
- **Plot** – 图形化潮流结果。

如果没有指明具体命令, 默认 **Edit** 命令

### 参数

参数/值 可以用逗号 (,) 或者空白 (blank, tab) 分开. 如果值要求分割符的话, 下列格式可被接受。尽管实际上可以互换, 还是鼓励遵循下面的标准格式:

- 双引号        "... "        字符串
- 单引号        '...'        字符串

- 小括号 (...) 数组
- 中括号 [...] 数组
- 大括号 {...} 嵌入数学式子 (参阅 OpenDSS Manual)

类成员或者元件的参数（也叫属性）的接口可以通过点 “.” 获得。所用的元件都应通过它们的全名来指定，除非通过上下文可以推断出它的类<sup>1</sup>。全名有如下格式：

```
ElementClass.ElementName
```

元件的 属性的全名格式如下：

```
ElementClass.ElementName.PropertyName
```

当使用同一命令作俑于同一元件的不同属性时，除第一个外，其他的属性不需要全名。比如<sup>2</sup>：

```
Line.L1.Bus1=1, Bus2=5
```

除了指定属性然后赋值外,属性赋值还可以按默认顺序进行。按默认顺序和指定属性赋值可以混合使用，例如：

```
New EnergyMeter.Feeder Line.L115, terminal=1, enabled=false
```

当生成一个新元件时，许多属性有默认值。在帮助文件中有完整的记录有关于什么属性有默认值及默认值是多少，什么必须被指定。一般来说 element, object, terminal, bus1, 和 bus2 属性没有默认值，它们必须被指定。

---

<sup>1</sup> 比如，定义电容器控制(capcontrol)的 element 属性时，element 是 capacitor 的事实可以很容易得出，这种情况，ElementClass 可以不包括在名字中。

<sup>2</sup> 注意此处命令没有指定，默认是 Edit

## COMMENTS 注释

代码的注释可以用 // 或者 !， 如下:

```
// Edit the voltage regulator control
RegControl.Ctrl1. maxtapchange=1 ! Limit to one tap change
```

多行注释可以通过 /\* ... \*/。 注意 /\*必须在第一列。 比如:

```
/*    comment out the next two monitors
New Monitor.Source-PQ  Vsource.source 1  mode=1 ppolar=no
New Monitor.source-VI  Vsource.source 1  mode=0 VIpolar=Yes
*/
```

## 多行命令 MULTI-LINE COMMANDS

如果一个命令超过一行， ~ 符号 应被使用， 比如:

```
New Line.L1  Bus1=1, Bus2=2, Length=1
~  units=mi, geometry=3PH_3/0_Horiz
```

注意，上面的代码实际上是两个命令。 所以，任何没有默认值的属性在生成这个元件时必须在第一行中定义。下面的代码会发生错误:

```
// This will error because Bus1 and Bus2 are not set in the first line
New Line.L1  Length=1, units=mi
~  Bus1=1, Bus2=2, geometry=3PH_3/0_Horiz
```

## 包含外部文件

有两种方式来包含外部文件

1. Compile 命令,它会嵌入另一个 OpenDSS 代码在当前处。  
Compile 命令会改变路径到外部文件的路径。
2. Redirect 命令,同样， 它会嵌入另一个 OpenDSS 代码在当前处。  
不同的是， redirect 命令会保留原有的路径。

一些属性，例如 LoadShapes 的 mult,可能会需要大量的数据，不方便直接放入主代码中或者来源于其他各式。File ,可以读入一个文件并使用它的内容作为一个元件的值。比如:

```
LoadShape.LS1 mult=(File='Example.csv')
```

参阅 OpenDSS Manual 关于 File 和他的姐妹功能 sngFile and dblFile.

## 工作流程

前面已经提及，任何通过 GUI 完成的任务都可通过代码完成。下面描述的工作流程和上节的工作流程类似。

1. 定义电路: 定义线路，变压器，负荷，发电机，等...

- a. 使用 `Clear` 命令来清除任何以前载入的电路  
ex: `Clear`
  - b. 使用 `New c` 命令定义新电路  
ex: `New object=circuit.ExampleCircuit`
  - c. 使用 `New` 命令定义新元件  
ex: `New line.L1 bus1=1 bus2=2 linecode=336ACSR length=1.25`
2. 设定电路的仿真计算选项： 比如， 仿真计算模式 (snapshot, daily, harmonic, 等)
  - a. 使用 `Set` 命令  
ex: `Set mode=snapshot voltagebases=(115, 12.47, .14)`
3. 解潮流计算问题
  - a. 首先，使用 `CalcVoltageBases` 命令确保母线列表生成，发现相对应的电压。  
ex: `CalcVoltageBases`
  - b. 使用 `solve` 命令来进行仿真计算。  
ex: `Solve`
4. 分析已解电路—不同的分析包括不同的使用细节。但是一些通常的任务包括：
  - a. 检查线路，变压器，负荷，等元件—使用 `Visualize` 命令  
ex: `Visualize element=Line.L1 what=powers`
  - b. 图形化显示电压-距离—使用 `plot` 命令  
ex: `Plot profile`
  - c. *Visualize data onto a one-line of the feeder – Use the Buscords and Plot commands*  
ex: `Buscoords BusCordsFile.dat`  
`Plot circuit`
  - d. 输出数据以使用其他软件分析—使用 `Export` 命令  
ex: `Export voltages`

## 代码范例

以下面电路为例子进行示范:

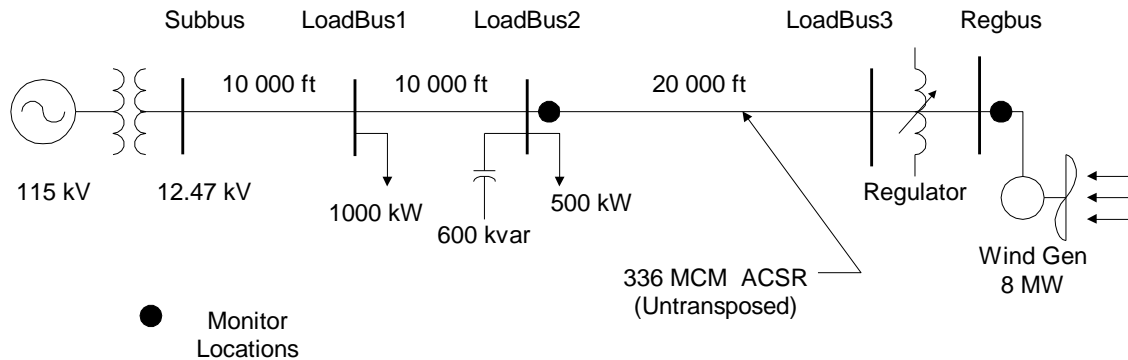


图 8. 电路示范

下面的代码生成上述电路

```
// The first step is always to clear the DSS and instantiate a new circuit
clear
New object=circuit.ExampleCircuit
~ basekv=115 1.00 0.0 60.0 3 20000 21000 4.0 3.0 ! edit the voltage source

// Define some load shapes for the loads and wind
! This is an example of defining parameters via their default order
! in this case, the num of points and interval
New loadshape.day 8 3.0
~ mult=(.3 .36 .48 .62 .87 .95 .94 .60)
! This is an example of an inline calculation, see OpenDSS Manual for more info
New loadshape.wind 2400 {1 24 /} ! unit must be hours
~ mult=(file=zavwind.csv) action=normalize ! wind turbine characteristic

// Define a linecode for the lines - unbalanced 336 MCM ACSR connection
New linecode.336matrix nphases=3 ! horizontal flat construction
! rmatrix, xmatrix, and ymatrix are in lower triangular matrix format.
! see the OpenDSS Manul for more details on how to specify matrixes.
! In ohms per 1000 ft
~ rmatrix=(0.0868455 | 0.0298305 0.0887966 | 0.0288883 0.0298305 0.0868455)
! In ohms per 1000 ft
~ xmatrix=(0.2025449 | 0.0847210 0.1961452 | 0.0719161 0.0847210 0.2025449)
! In nf per 1000 ft
~ cmatrix=(2.74 | -0.70 2.96 | -0.34 -0.71 2.74)
~ Normamps = 400 Emergamps=600

// Define Substation transformer
! Note that the buses property provides an alternate way to specify the
! buses beyond bus1= and bus2=
New transformer.subxfm phases=3 windings=2 buses=(SourceBus subbus)
~ conns='delta wye' kvs=(115 12.47) kvas=(20000 20000) XHL=7

// Define the lines
New line.line1 bus1=subbus bus2=loadbus1 linecode=336matrix length=10
New line.line2 loadbus1 loadbus2 336matrix 10
New line.line3 Loadbus2 loadbus3 336matrix 20
```

```
// Define the loads
New load.load1 bus1=loadbus1 phases=3 kv=12.47 kw=1000.0 pf=0.88 model=1
~ class=1 duty=day
New load.load2 bus1=loadbus2 phases=3 kv=12.47 kw=500.0 pf=0.88 model=1
~ class=1 duty=day conn=delta

// Capacitor with control
New capacitor.Cap1 bus1=loadbus2 phases=3 kvar=600 kv=12.47
New capcontrol.Cap1Ctrl element=line.line3 terminal=1 capacitor=Cap1
~ type=current ctratio=1 ONsetting=60 OFFsetting=55 delay=2

// Regulated transformer to DG bus
New transformer.reg1 phases=3 windings=2
~ buses=(loadbus3 regbus)
~ conns='wye wye'
~ kvs=(12.47 12.47)
~ kvas=(8000 8000)
~ XHL=1 ! tiny reactance for a regulator

// Regulator Control definitions
New regcontrol.subxfmCtrl transformer=subxfm winding=2 vreg=125
~ band=3 ptratio=60 delay=10
New regcontrol.reg1Ctrl transformer=reg1 winding=2 vreg=122 band=3
~ ptratio=60 delay=15

// Define a wind generator of 8MW
New generator.gen1 bus1=regbus kv=12.47 kW=8000 pf=1 conn=delta
~ duty=wind Model=1

// Define some monitors so we can see what's happening
// (See documentation on how the mode parameter works)
! Monitor the power output of the wind turbine
New Monitor.gen1 element=generator.gen1 terminal=1 mode=1
! Monitor the voltage and currents at the second load bus
New Monitor.loadbus2 load.load2 1 mode=0
! Monitor sequence voltages and currents magnitudes of line 3, terminal 1
New Monitor.line3 line.line3 1 mode=48
! You need an energy meter in order to get line distances for a profile plot
New Energymeter.em1 line.line1

// Define voltage bases so voltage reports come out in per unit
Set voltagebases=(115 12.47 .48)

// Generate the bus list and figure out the voltage bases
Calcvoltagebases

// Simulation options to make the cap and reg controllers operate in sync
// with the rest of the simulation
Set controlmode=time
// Simulation options to do a time based simulation for 24 hours (86400 sec)
// with a time step of 1 sec starting at hour 0, second 0
Set mode=duty number=86400 hour=0 stepsize=1 sec=0

// Conduct the simulation
Solve

// Show some results
! Plot how the voltage at loadbus1 looked during the day
Plot monitor, object=loadbus2, Channels=(1,3,5)
! Visualize the line's flow as it appear at the last timestep
Visualize element=Line.line1 what=powers
! Show the voltage profile on the feeder as it appeared at the last timestep
Plot profile
```

## COM 接口介绍

---

尽管在标准用户界面中可完成很多工作，但是用户可以通过 COM 接口获得更多的灵活性。COM 接口可以使用户在其他软件中开发算法来驱动 OpenDSS 引擎以实现一些 DSS 中尚未开发的功能。比如优化算法。目前 DSS 仅有一些简单算法,但是可以外部开发复杂算法。这些外部算法将依靠 DSS 来仿真配网的相应状态，以此来调节优化有关控制变量。

一个使用 COM 接口的例子是循环执行代码。OpenDSS 没有循环执行代码的能力。最接近的是 `Next` 命令，它循环增加时间。而在其他语言中，循环相对很容易实现并且运行相对较快。

详细的 COM 接口的记录可以在 OpenDSS manual 中找到。但是最新版的记录是在 COM 接口中。使用一个类库浏览器（type library browser (TLB)） – 如 Microsoft Office 的 Visual Basic Editor – 在 Excel 的 VBA 的编辑器中可以浏览 DSS 的 COM 接口所包含的库（library）。在 Excel 的 VBA 的 Editor 中，在 reference (在 Tools 菜单中) 中填加 OpenDSSEngine。然后你可以使用"Object Browser"（在 View 菜单中）浏览 OpenDSSEngine 的库。

在这个简介中，Visual Basic 的代码将被作为例子，因为它相对其他编程语言较容易被读懂并且 较易获得。你可以使用 Microsoft Excel 的 macro editor 或者免费版的 [Visual Studio Express](#) 的 Visual Basic 来尝试下面的例子。在 Mathwork's MATLAB 和 Python 的例子在笨重的最后也会提到。OpenDSS 用户还有在其他编程语言成功使用 COM 接口的经验，比如 C++, C#, and R 等。

## 开始使用 COM 接口

首先，必须确认你的编程环境和 OpenDSS 的 COM 接口相连。在 Visual Basic 中，是这样实现的：Tools > References or Project > Add Reference。从这里选择 OpenDSSEngine COM 类型库。



使用下面的代码初始化一个 OpenDSS 对象并且生成一个链接：

```
' Declare the OpenDSS related variables
Dim DSSObj As OpenDSSEngine.DSS
Dim DSSText As OpenDSSEngine.Text
Dim DSSCircuit As OpenDSSEngine.Circuit
Dim DSSSolution As OpenDSSEngine.Solution

' Instantiate the OpenDSS Object
DSSObj = New OpenDSSEngine.DSS

' Start up the Solver
If Not DSSObj.Start(0) Then
    MsgBox("Unable to start the OpenDSS Engine")
    Return
End If

' Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text
DSSCircuit = DSSObj.ActiveCircuit
DSSSolution = DSSCircuit.Solution
```

可以看出，从 DSS parent object 中存在几个子类（children classes）：

- **Text**, 它提供对命令编辑器的接口（command line interpreter interface）。使用它，你可以直接使用在前面章节中提到的 OpenDSS 命令。
- **Circuit**, 它提供了对组成电路元件的接口。使用它的成员，你可以编辑，监视不同电路元件。
- **Solution**, 它提供了对算法和解的接口。使用它的成员，你可以设定算法参数，解电路，查看解的属性，比如解所花费的循环次数。

下面将详细介绍这几个接口。

## TEXT 接口介绍

Text 接口是最简单的，但是也是最有用的接口。你可以借助它在 COM 接口中执行 OpenDSS 命令。这样，COM 用户可以使用任何一个 OpenDSS 命令。

下述 是使用 text 接口的例子。

```
' Load in an example circuit
DSSText.Command = "Compile 'C:\example\IEEE123Master.dss'"
' Create a new capacitor
DSSText.Command = "New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200"
DSSText.Command = "~ Enabled=false" ' You can even use ~
' Change the bus for Line L1
DSSText.Command = "Line.L1.Bus1 = 5"
```

如果执行的命令是显示一个结果，这个结果可以用 Text 的 Result 属性来获得：

```
' Export voltage to a csv file
Dim Filename As String
DSSText.Command = "Export Voltages"
```

```
Filename = DSSText.Result
MsgBox("File saved to: " & Filename)
```

## 电路接口（CIRCUIT INTERFACE）介绍

电路接口（circuit interface）可以被用来编辑电路中不同元件的属性。几乎所有 OpenDSS 的元件类（比如 Lines, Loads, Capacitors, CapControls, 等）在电路接口中都有一个子对象。这些子对象还有一些便利的函数可以使用户在循环中遍历所有的成员元件。例如，下面的代码会遍历所有电路中的负荷（loads）把 kW 增加 20%:

```
' Step through every load and scale it up
Dim iLoads As Integer ' Track what load we're on

iLoads = DSSCircuit.Loads.First
While iLoads
    ' Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2
    ' Move to next load
    iLoads = DSSCircuit.Loads.Next
End While
```

如果你想要编辑一个具体的元件，使用 SetActiveElement 方法，ActiveDSSElement 的接口如下：

```
' Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement("Capacitor.C83")
DSSCircuit.ActiveDSSElement.Properties("kVAR").Val = 1200
```

这种方法和简单的 text 接口可以取得相同的结果，如下,根据自己的需要选择方法。

```
' Does the same thing as the previous snippet
DSSText.Command = "Capacitor.C83.kVAR = 1200"
```

电路接口的另外一个有用的特性是可以获得潮流计算（power flow）的结果，比如电压，损耗等。下面的例子获得母线的名字和电压：

```
' Get bus voltages
Dim BusNames As String()
Dim Voltages As Double()
BusNames = DSSCircuit.AllBusNames
Voltages = DSSCircuit.AllBusVmagPu

' See what an arbitrary bus's voltage is
MsgBox(BusNames(5) & "'s voltage mag in per unit is: " & Voltages(5))
```

更多细节在 OpenDSS manual 的 COM section 的 “All commands”中 (比如 AllElementLoses, AllNodeVmagByPhase)

## 算法和解的接口（SOLUTION INTERFACE）介绍

算法和解的接口（Solution Interface）被用来查看和控制 OpenDSS 的计算过程和控制过程。这包括解电路，设定解的模式，查看是否收敛，报告 time 和 duty 模式中的时间段，等等功能。

最基本的功能是,这个接口可以让我们解电路, 如下:

```
' Solve the Circuit
DSSSolution.Solve()
If DSSSolution.Converged Then
    MsgBox("The Circuit Solved Successfully")
End If
```

你也可以在更多细节上控制解电路。比如, 下面的例子仿真 30 秒电路反应在加入一个大负荷后。

```
' Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = _
    "New Monitor.Mon1 element=Line.L100 mode=0"
DSSSolution.StepSize = 1          ' Set step size to 1 sec
DSSSolution.Number = 30           ' Solve 30 seconds of the simulation
' Set the solution mode to duty cycle, which forces loads to use their
' "duty cycle" loadshape and allows time based simulation
DSSSolution.Mode = OpenDSSengine.SolveModes.dssDutyCycle
DSSSolution.Solve()

DSSCircuit.Enable("Load.L1")      ' Enable the load
DSSSolution.Number = 30           ' Solve another 30 seconds of simulation
DSSSolution.Solve()

MsgBox("Seconds Elapsed: " & DSSSolution.Seconds)
' Plot the voltage for the 60 seconds of simulation
DSSText.Command = "Plot monitor object=Mon1 Channels=(1,3,5)"
```

电路接口 (**circuit interface**) 提供许多很具体的解电路背后的方法和控制信息接口。使用不同的解命令 (比如, Solve, SolveDirect, SolveNoControl, 等...) 和其他函数如 SystemYChanged 和 MostIterationsDone, 可以获得解的细节或者在细节上控制解的过程。使用 CheckControls 及其他方法, 连同 DSSCircuit.CtrlQueue 接口, 可以实现用户定义的控制策略。更多细节可以参阅 [OpenDSS TechNotes](#)。

## VISUAL BASIC 例子

前面提到的 VB 例子在此处汇总。

```
' *****
' * Initialize OpenDSS
' *****
' Declare the OpenDSS related variables
Dim DSSObj As OpenDSSEngine.DSS
Dim DSSText As OpenDSSEngine.Text
Dim DSSCircuit As OpenDSSEngine.Circuit
Dim DSSSolution As OpenDSSEngine.Solution

' Instantiate the OpenDSS Object
DSSObj = New OpenDSSEngine.DSS

' Start up the Solver
If Not DSSObj.Start(0) Then
    MsgBox("Unable to start the OpenDSS Engine")
    Return
End If

' Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text
DSSCircuit = DSSObj.ActiveCircuit
DSSSolution = DSSCircuit.Solution

' *****
' * Examples Using the DSSText Object
' *****
' Load in an example circuit
DSSText.Command = "Compile 'C:\example\IEEE123Master.dss'"
' Create a new capacitor
DSSText.Command = "New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200"
DSSText.Command = "~ Enabled=false" ' You can even use ~
' Change the bus for Line L1
DSSText.Command = "Line.L1.Bus1 = 5"

' Export voltage to a csv file
Dim Filename As String
DSSText.Command = "Export Voltages"
Filename = DSSText.Result
MsgBox("File saved to: " & Filename)

' *****
' * Examples Using the DSSCircuit Object
' *****
' Step through every load and scale it up
Dim iLoads As Integer ' Track what load we're on

iLoads = DSSCircuit.Loads.First
While iLoads
    ' Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2
    ' Move to next load
    iLoads = DSSCircuit.Loads.Next
End While

' Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement("Capacitor.C83")
DSSCircuit.ActiveDSElement.Properties("kVAR").Val = 1200
' Get bus voltages
```

```

Dim BusNames As String()
Dim Voltages As Double()
BusNames = DSSCircuit.AllBusNames
Voltages = DSSCircuit.AllBusVmagPu

' See what an arbitrary bus's voltage is
MsgBox(BusNames(5) & "'s voltage mag in per unit is: " & Voltages(5))

' *****
' * Examples Using the DSSSolution Object
' *****
' Solve the Circuit
DSSSolution.Solve()
If DSSSolution.Converged Then
    MsgBox("The Circuit Solved Successfully")
End If

' Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = _
    "New Monitor.Mon1 element=Line.L100 mode=0"
DSSSolution.StepSize = 1          ' Set step size to 1 sec
DSSSolution.Number = 30           ' Solve 30 seconds of the simulation
' Set the solution mode to duty cycle, which forces loads to use their
' "duty cycle" loadshape and allows time based simulation
DSSSolution.Mode = OpenDSSengine.SolveModes.dssDutyCycle
DSSSolution.Solve()

DSSCircuit.Enable("Load.L1")      ' Enable the load
DSSSolution.Number = 30           ' Solve another 30 seconds of simulation
DSSSolution.Solve()

MsgBox("Seconds Elapsed: " & DSSSolution.Seconds)
' Plot the voltage for the 60 seconds of simulation
DSSText.Command = "Plot monitor object=Mon1 Channels=(1,3,5)"

```

## MATLAB 例子

以下是 Matlab 的 COM 接口的例子

```
clc
clear all
close all

% *****
% * Initialize OpenDSS
% *****
% Instantiate the OpenDSS Object
DSSObj = actxserver('OpenDSSEngine.DSS');

% Start up the Solver
if ~DSSObj.Start(0),
    disp('Unable to start the OpenDSS Engine')
    return
end

% Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text;
DSSCircuit = DSSObj.ActiveCircuit;
DSSSolution = DSSCircuit.Solution;

% *****
% * Examples Using the DSSText Object
% *****
% Load in an example circuit
DSSText.Command = 'Compile "C:\example\IEEE123Master.dss"';
% Create a new capacitor
DSSText.Command = 'New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200';
DSSText.Command = '~ Enabled=false'; % You can even use ~
% Change the bus for Line L1
DSSText.Command = 'Line.L1.Bus1 = 5';

% Export voltage to a csv file
DSSText.Command = 'Export Voltages';
Filename = DSSText.Result;
disp(['File saved to: ' Filename])

% *****
% * Examples Using the DSSCircuit Object
% *****
% Step through every load and scale it up

iLoads = DSSCircuit.Loads.First;
while iLoads,
    % Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2;
    % Move to next load
    iLoads = DSSCircuit.Loads.Next;
end
```

```
% Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement('Capacitor.C83');
DSSCircuit.ActiveDSSElement.Properties('kVAR').val = '1200';

% Get bus voltages
BusNames = DSSCircuit.AllBusNames;
Voltages = DSSCircuit.AllBusVmagPu;

% See what an arbitrary bus's voltage is
disp([BusNames{5} ' 's voltage mag in per unit is: '
num2str(Voltages(5))])

% *****
% * Examples Using the DSSSolution Object
% *****
% Solve the Circuit
DSSSolution.Solve();
if DSSSolution.Converged,
    disp('The Circuit Solved Successfully')
end

% Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = ...
    'New Monitor.Mon1 element=Line.L100 mode=0';
DSSSolution.StepSize = 1;          % Set step size to 1 sec
DSSSolution.Number = 30;           % Solve 30 seconds of the simulation
% Set the solution mode to duty cycle, which forces loads to use their
% 'duty cycle' loadshape and allows time based simulation
DSSSolution.Mode = 6;              % Code for duty cycle mode
DSSSolution.Solve();

DSSCircuit.Enable('Load.L1');      % Enable the load
DSSSolution.Number = 30;           % Solve another 30 seconds of simulation
DSSSolution.Solve();

disp(['Seconds Elapsed: ' num2str(DSSSolution.Seconds)])
% Plot the voltage for the 60 seconds of simulation
DSSText.Command = 'Plot monitor object=Mon1 Channels=(1,3,5)';
```

## PYTHON 例子

```

import win32com.client

# *****
# * Initialize OpenDSS
# *****
# Instantiate the OpenDSS Object
try:
    DSSObj = win32com.client.Dispatch("OpenDSSEngine.DSS")
except:
    print "Unable to start the OpenDSS Engine"
    raise SystemExit

# Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text
DSSCircuit = DSSObj.ActiveCircuit
DSSSolution = DSSCircuit.Solution

# *****
# * Examples Using the DSSText Object
# *****
# Load in an example circuit
DSSText.Command = r"Compile 'C:\example\IEEE123Master.dss'"
# Create a new capacitor
DSSText.Command = "New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200"
DSSText.Command = "~ Enabled=false" # You can even use ~
# Change the bus for Line L1
DSSText.Command = "Line.L1.Bus1 = 5"

# Export voltage to a csv file
DSSText.Command = "Export Voltages"
Filename = DSSText.Result
print "File saved to: " + Filename

# *****
# * Examples Using the DSSCircuit Object
# *****
# Step through every load and scale it up
iLoads = DSSCircuit.Loads.First
while iLoads:
    # Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2
    # Move to next load
    iLoads = DSSCircuit.Loads.Next

# Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement("Capacitor.C83")
DSSCircuit.ActiveDSElement.Properties("kVAR").Val = 1200

# Get bus voltages
BusNames = DSSCircuit.AllBusNames
Voltages = DSSCircuit.AllBusVmagPu

```



```
# See what an arbitrary bus's voltage is
print BusNames[5] + "'s voltage mag in per unit is: " + \
      str(Voltages[5])

# *****
# * Examples Using the DSSSolution Object
# *****
# Solve the Circuit
DSSSolution.Solve()
if DSSSolution.Converged:
    print "The Circuit Solved Successfully"

# Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = \
    "New Monitor.Mon1 element=Line.L100 mode=0"
DSSSolution.StepSize = 1          # Set step size to 1 sec
DSSSolution.Number = 30          # Solve 30 seconds of the simulation
# Set the solution mode to duty cycle, which forces loads to use their
#   "duty cycle" loadshape and allows time based simulation
DSSSolution.Mode = 6             # Code for duty cycle mode
DSSSolution.Solve()

DSSCircuit.Enable("Load.L1")      # Enable the load
DSSSolution.Number = 30          # Solve another 30 seconds of simulation
DSSSolution.Solve()

print "Seconds Elapsed: " + str(DSSSolution.Seconds)
# Plot the voltage for the 60 seconds of simulation
DSSText.Command = "Plot monitor object=Mon1 Channels=(1,3,5) "
```

## 其他资源

---

在你探索 OpenDSS 广泛的应用时，你会遇到本简介没有介绍的情况。在这时，你可以参阅 OpenDSS 开发者和用户社区提供的以下资源。

### 下一步该去哪儿？


通过这个入门简介，你应该可以开始使用 OpenDSS。更多的有关 OpenDSS 特性的例子可以在安装目录下找到，包括：

- 更多的 OpenDSS 代码范例, 在 “Examples/Scripts” 目录中。
- 更多的例子有关于：解的模式，比如时间仿真, duty-cycle 仿真, 年能量仿真（annual energy simulations），Monte Carlo analysis, 动态仿真（dynamic analysis），谐波仿真（harmonic analysis），故障仿真（fault studies），电容器位置优化（capacitor placement optimization），地磁感应电流（geomagnetically induced current studies），等
- 关于电路元件的例子, 如
  - 调压器和及其控制，电容器及其控制，集中控制的 volt-VAR 优化控制
  - 各种发电机模型，如传统的 PV（功率电压）发电机，固定功率因数发电机，逆变器，太阳能发电机，感应发电机，储能电源等。
  - 断路器等（Fuses, reclosers, relays, and faults）
  - 负荷曲线（load shapes），增长曲线（growth curves），谐波频谱（harmonic spectrums），线路几何信息（line geometries），温度曲线（temperature curves），能量计量器（energy meters），和监视器（monitors）
- 范例包含以下几种语言的 COM 接口
  - MATLAB
  - Python
  - Visual Basic for Applications through Excel
- 在 EPRITestCircuits 目录中，有一系列真实的配网模型
- 一个简单的输电网模型在/Stevenson 目录中

其他的介绍和培训文件在 OpenDSS 的安装目录中可找到，在 SourceForge 的培训目录中也可找到, 点击[这里](#) [here](#).

## 参考资源

OpenDSS 其他有关编程和 COM 接口的参考资料包括:

- **In-program help interface**, 程序内帮助, 可以通过 Help > DSS Help 或者  按钮打开。简明介绍特性和命令。
- **OpenDSS manual**, 在安装路径目录中。详细介绍特性, COM 接口等。
- **OpenDSS TechNotes**, 在安装路径目录中。一系列介绍 COM 接口, OpenDSS 特性, 及其他具体应用 (如变压器模型, 太阳能发电机) 的文章。
- **OpenDSS wiki**, 点击 [here](#) 它尤其提供 COM 接口的有关资源。
- **Type library browser (TLB)** – 例如 Microsoft Office's Visual Basic Editor – 可以用来查看 COM 对象。 打开 Microsoft Office's Visual Basic Editor, 在 reference (在 Tools 菜单中) 中填加 OpenDSSEngine. 然后你可以使用 "Object Browser" (在 View 菜单中) 浏览 OpenDSSEngine 的库。

## 更多帮助

如果你需要更多帮助, 在 SourceForge 网站的 OpenDSS forums 中, OpenDSS 用户群会为你提供更多解答。论坛点击 [here](#).