User Manual

LozWare is proud to produce DataNet MKII, possibly the only telnet server that has a user-friendly interface, without sacrificing power.

Installing DataNet

System Requirements

DataNet requires Windows(2000/XP) to run, and additionally requires the MSWINSOCK.OCX in order to function (this file is automatically installed with setup). DataNet is written in Visual Basic and does require the VB runtime files; Windows XP and 2000 come with these files (except the winsocket control, mentioned above), however, if you are receiving runtime problems, you should run the VBRUNTIME.EXE that comes bundled with DataNet (please note that this VB runtime installer was not created by LozWare).
DataNet is not a very demanding program when running idle, nor is it demanding under low traffic, but if you expect heavy traffic aimed at DataNet then you will require a strong processor to do all of the processing in time. Here are the system guidelines:

O.S.:	Windows XP/2000
Processor:	1.0 GHz +
RAM:	128 MB +
Vid. Card:	64 MB +

Installation

To install DataNet, run the executable named SETUP.EXE located in the same directory as this help file, this will start a standard installation wizard which will take you through the installation. After the installation completes, you will be presented with a readme.txt file, which will tell you the basics to get you on your feet. The setup will also launch DataNet upon completion.
In the unlikely event of VB Runtime errors occurring when DataNet is launched, try installing the VBRUNTIME.EXE which can be located in DataNet's home directory: {Program Files}\LozWare\DataNet\Vbruntime.exe

Configuring DataNet

Domains

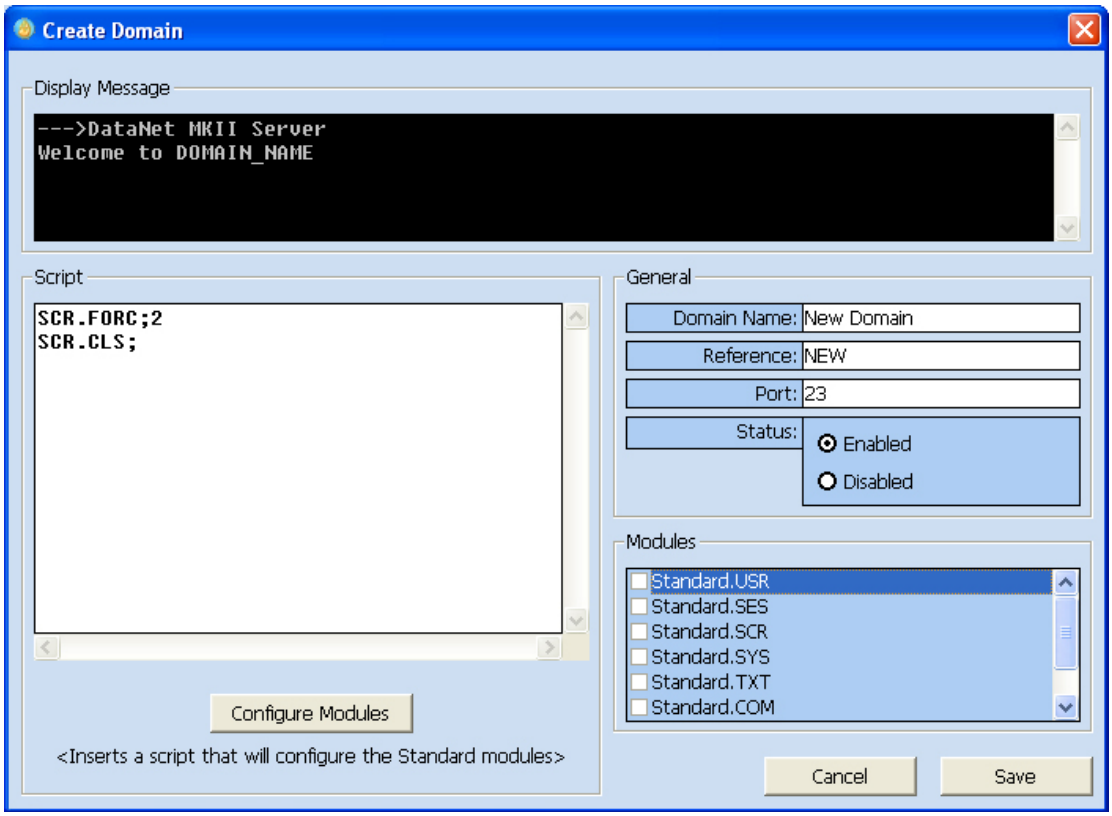
Why Use Multiple Domains?

In some circumstances, you may want to have more than one domain. The main advantages of this are that users from different domains are separated from each other, and can't interact with one another. A further advantage is that domains can be configured so that they only have access to certain modules, which means that you can dedicate certain modules to particular domains. There is also the benefit that you can create start-up scripts for domains, these can personalise a domain tremendously. All these possibilities mean that can have one domain dedicated to one topic, and another dedicated to something different. For example: domain A may be purely built for network communication, and have the chatroom initiated in its start-up script, and not allow users access to any other modules other than the communications module; whereas domain B could be solely dedicated to manipulating text files (like creating invoices) – which results in the file contents of domain B (which could be potentially

confidential) being separate from domain A, and also ensures that clients working in domain B are not interrupted by communications from domain A.

How to Make a Domain

To create a domain, you must first open the Domains window inside DataNet. To do this, you must click Management>Domains. With the domains window open, you must now click Add Domain, this will bring you to a new window…



To setup a new domain should take less than 4 minutes if you know what you are doing, but presuming you don't, then here are the basics:
Display Message: What ever is typed in this text box will be displayed before the login procedure for the domain.
General: Allows you to specify basic information for the domain, the Domain Name, Reference, and Port must be unique (the port number must be between 1 and 30000), otherwise DataNet may not function correctly.
Modules: This area indicates which modules will be loaded into your domain. Modules are basically sets of commands, if this is the first time you have created a domain, tick all of the boxes, then you can remove some later if you wish. To understand more about modules, refer to the 'What Are Modules' section.
Script: You may choose to write a script in the Script field, basically, what ever commands you type in the script section will be carried out every time someone successfully logs into this domain. These scripts can significantly customise a domain.
Configure Modules: This button will automatically insert commands into the script field that will configure all of the modules for you, so that your clients will have less work to do each time they use your domain.

How to Modify a Domain

To modify a domain, you must first bring up the ‘Domains’ window (see above). With this window in focus, you must select the domain you want to modify, and click the ‘Modify Domain’ button. This will bring you to a similar screen to the ‘Create Domain’ function, but anything you change in this window will just replace the existing configurations of the domain you selected to modify. It is normally recommended that you close, and re-open the server after modifying or creating a domain, because in the unlikely event of DataNet closing incorrectly, all of the new information about that domain will be lost.

How to Delete a Domain

Deleting a domain is one of the simpler tasks to perform on DataNet; you simply bring up the Domains window (explained above), and select a domain, then click the ‘Remove Domain’ button.

User Accounts

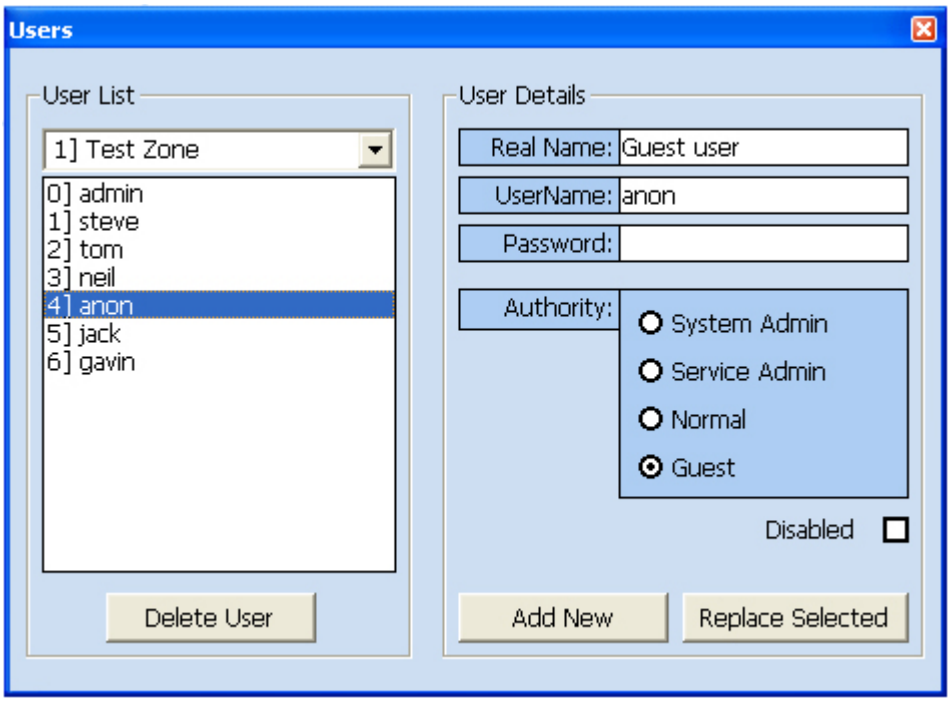
Authority Levels

Authority levels help to create order in your domain, by restricting certain groups of users from carrying out certain commands; this helps you to create a secure domain. Here is a list of the four authority levels:

- **System Administrator**
  can carry out any command on the server, and has access to the whole system.
- **Service Administrator**
  has the power to carry out most commands on the server, but cannot view the whole system, as his/her root path is set to that of their domain (this means that they may access other users’ work area). The service administrator cannot interrupt system admin activities (e.g. terminate their session, or modify their user account).
- **Standard**
  may carry out nearly every one of the commands that the service admin can, but with the exception that the same hierarchical rule applies, they may not interrupt service, or system admin activities. Standard accounts are limited to their own work area on the server, as their root path is a designated work area within their domain.
- **Guest**
  have little power, as they are at the bottom of the hierarchy, and cannot interrupt system admin, service admin, or standard user activities. Like the Standard authority level, guests have their own designated work area and cannot move out of it. Some commands may not be available to the guest user.

Creating an Account

NOTE: You must have at least one domain before you can start to create user accounts.
To create a new account, you must first open up the ‘Users’ window, this can be located under Management>Users.



With this window in focus you may begin to create user accounts, follow these steps:

1. Choose the domain which you would like to add user accounts to (in the example, 'Test Zone' is the selected domain).
2. Fill in the user details on the right side of the window.
3. Click on the 'Add New' button to insert the new user account into the domain.
4. Close the window once you are finished.

When creating user accounts, it is recommended that you only have one system administrator for the whole server, and about 2 – 4 service administrators for each domain.

Modifying an Account

To modify an account, you must open the 'Users' window (explained above), then follow these steps:

1. Select the domain which holds the user account you wish to edit.
2. Select the user account by left clicking on it (in the example, the 'anon' account is selected).
3. Now adjust the user details.
4. Now click the 'Replace Selected' button to update the user account.
5. Close the window once you are finished.

Deleting an Account

Open the 'Users' window, and select the domain that the user account is located in, then select the user account you wish to delete, then click the 'Delete User' button.

Disabling an Account

Follow steps 1-2 of how to modify an account, then, in the user details box to the right, confirm the 'Disabled' checkbox and click then 'Replace Selected' button to update the user account. This account will be disabled.

Modules

What Are Modules?

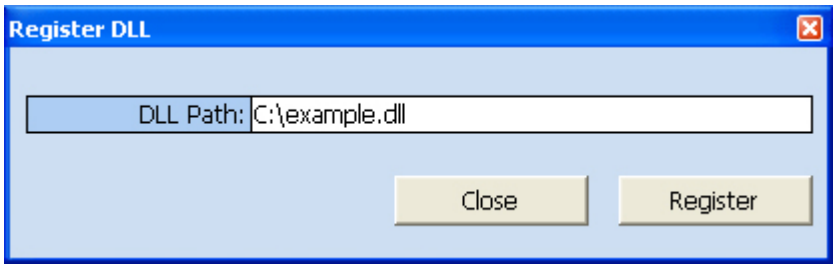
In terms of DataNet, a module is a group of commands that all have the same purpose/goal. For example, the COM module that belongs to the STANDARD family is built to supply the user with a range of commands that lets them communicate with other users.
DataNet's framework is written in a way that makes the user command library entirely dependant upon 'plug-in' DLLs, this means that the more advanced user/programmer can write their own modules for the server, thus, making DataNet a very adaptable service. A single DLL file may contain multiple modules, and each of these modules can contain many different commands.
Bottom Line:
A module is a home for a group of commands with similar functionality, so when you call upon a command, you must specify which module it belongs to (explained in the 'Command Syntaxes' section).

Adding a New Module

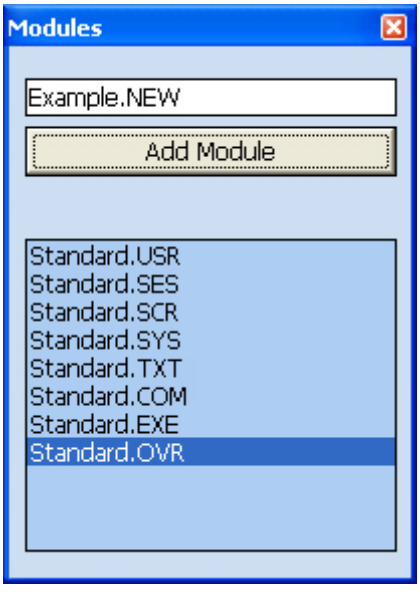
Adding a new module to DataNet is probably about as hard as it gets, but with DataNet's user-friendly interface, you should be able to accomplish your goals within minutes. As you may or may not know, modules come in the form of a 'Dynamic Link Library' (often referred to by its extension, .DLL), you may find that one single .DLL file will contain multiple modules (for example, the Standard.DLL that comes with DataNet houses about eight different modules). If you want to add a new family of modules, then you must know two things, that is:

1. The correct Family name for the DLL (not always the same as the filename)
2. The correct module names of each module that you want to add.

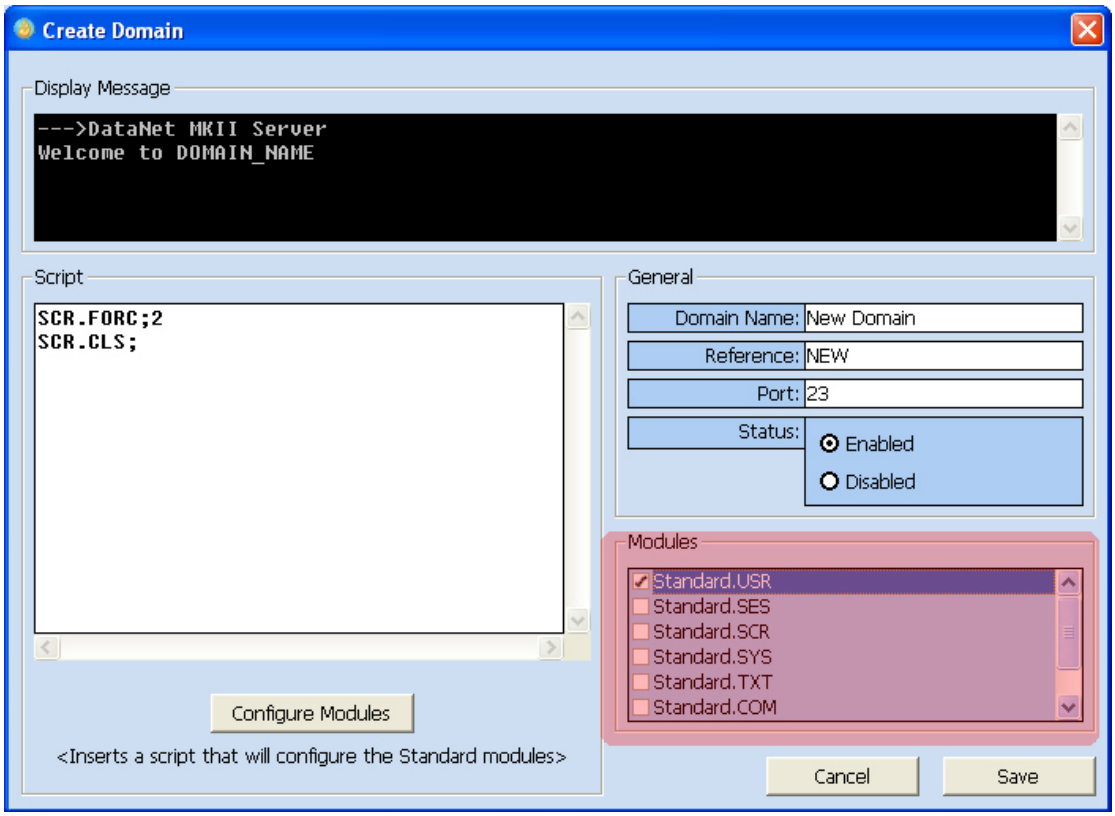
All of this information should be supplied with the DLL file (we will assume that you are entering modules from a new DLL file, if this is not the case, then you won't need to worry about registering the file, just skip to the next section). Firstly you will need to 'register' the DLL file onto your system so that it is detectable by Windows. To do this the easy way, you can open the 'Register DLL' window in DataNet by clicking Tools>Register DLL.



You can then type in the full path of the DLL into the textbox, then click the 'Register' button. A message will be displayed if the DLL file has been registered successfully.
With the new DLL registered, you can start to add modules to DataNet. To do this you must open the 'Modules' window, which can be located under Management>Modules.



In this new window, you must type in the full name of the module, the format for this is: FAMILY.MODULE. You have to replace the word FAMILY for the family name of the DLL (like I said before, this should be supplied with the DLL in a help file, and the family name is not necessarily the same as the DLL filename), then replace the word MODULE with the module name, which, like the family name, should be given to you inside the DLL's help file. After you type in the module name, click 'Add Module', and repeat the process if there is more than one module that comes with the DLL.
Congratulations! You have now added a module to DataNet's command library, for this module to be accessible by your domain(s), you must go and 'Modify' each of them by enabling your new module in the 'Modules' list box:



How to Create a Module

If you are an advanced user who has a strong understanding in programming (preferably VB), then you may like to find out how to write your own command 'family' by writing your own DLL for the server. Included in DataNet's home directory there is a folder called 'Module Template', inside this folder there is a VB project called 'Module'. This code is fully commented and should be fairly easy to understand. Basically the class modules are your command modules, so if you create a new class module called 'EXM', and set the application title to 'EXAMPLE' then when it comes to registering the new module you should refer to it as: EXAMPLE.EXM.
The only code that you should edit when writing your modules is the code below the designated line inside the class modules. You will notice that there is a normal module named BASE; you should not modify the contents of this file, as it contains a set of pre-made basic commands that allows you to interact with the domain easily.

To setup your new module (class module), you must do the following things:

1. In the public Title function, set the return value to the desired name of your module, 3 letters is fine, any longer and users will start to get annoyed!
2. In the public Description function, set the return value to a brief description of your module and what it does, try and make it 2 – 3 lines, and bear in mind how wide a telnet console is, as it has to fit inside one!
3. If you are using any public variables in your module to store details on a client's session, you will want them to be reset each time a different client uses your module. This is because DataNet reuses pre-loaded winsockets if they are not being used. This means that clients that connect to the server can sometimes have the same index as a previous client, so it is important that your modules reset any array items (with the given index) back to a default value. If you do not include this in your programming, you module will cause great security threats.
4. In the public RunCMD function, you can add your own commands for your module. Every time a user directs a command to your module, this function will be called with the arguments: Command (CMD), Argument, Index. These arguments are self explanatory, but I will explain them anyway:
    - The CMD variable is the command that the user typed
    - The Argument variable is the argument that the user typed for the command, sometimes an argument is not always required.
    - The Index variable indicates which winsocket index the command came from, allowing you to use the SendReply function to send a message back to the user who sent the message. And apart from anything else, the Index variable allows you to distinguish between different clients.

The easiest way (and as far as I know, the only way) to create a command is to create a series of IF statements to tell apart different commands. EG:

```
If UCase(CMD) = "TEST" Then
SendReply vbCrLf & "Your argument was:", Index
SendReply Argument, Index
Exit Function
End If
```

Inside the IF statement, write the outcome of your command. As you can see, the example command above refers to all three of the given variables (CMD, Argument, and Index). The way I write my modules (as you will see if you look at the module template) is to include the Exit Function statement at the end of each IF statement, and then include a SendReply command at the very end of the RunCMD function that sends the message 'Unkown Command' back to the client; this means that if the CMD variable doesn't match any of the IF statements, then an Exit Function will not be called, thus resulting in the 'Unkown Command' message being sent out when the processor reaches the bottom of the script.

General Tweaking

Traffic Monitoring

The traffic monitor can be located under the tools menu; it displays a range of stats that tells you a bit about the server's traffic. By default bandwidth monitoring will be disabled, this is because under intense traffic tests the server did suffer from a stack

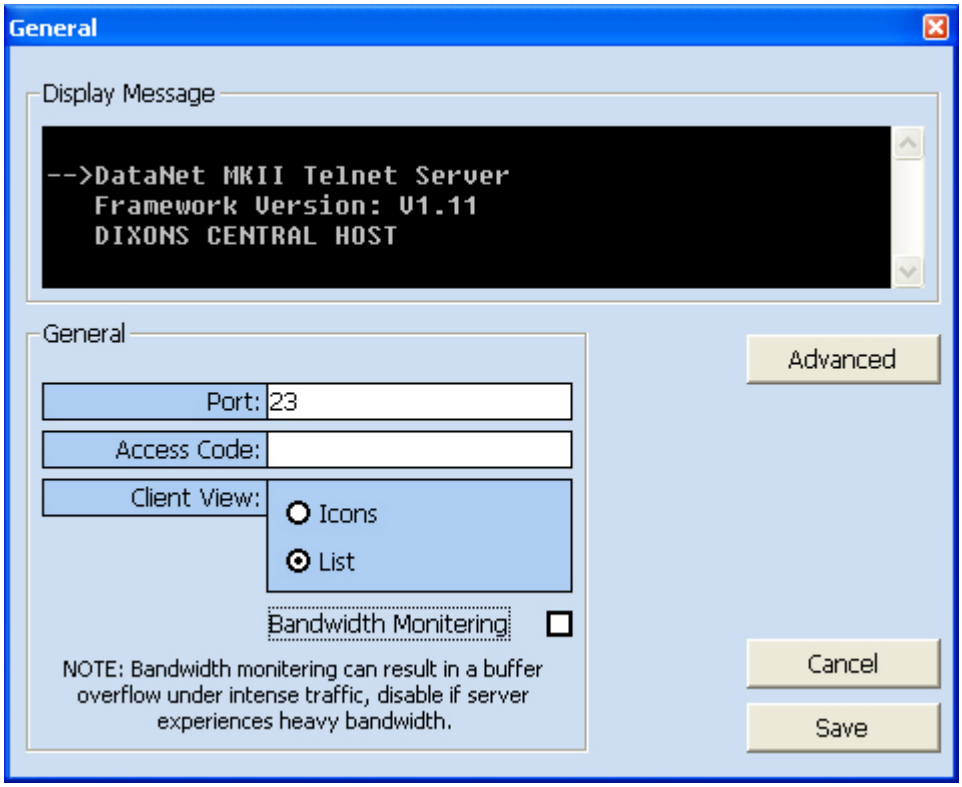
overflow, but if you are sure that your not going to have more than 30 users at a time, then enable the feature.

Display Message

As well as creating custom display messages for the server's domains, you can also create a display message for the initial start-up screen that displays the domain choice menu. This is normally a more generalise message, and tends to state who the server belongs to.

Port

By default this is set to 23, as the standard port for telnet servers is 23. But you may find yourself in a circumstance where you require DataNet to use a different port, at which point you can simply change this value in the General window and click 'Save'



Access Code

For extra security, DataNet allows you to create a password that uses must type before they have access to the Domain list.

Client View

To personalise DataNet, you can choose whether to have the client list (the left column of the main window) display clients as either list items, or icons. For situations where you are expecting little traffic, then use the icon view, however, if you are expecting heavy traffic, it would be more practical to display clients as a list.

Advanced

This button simply opens DataNet's home directory up in a new explorer window.

Using DataNet via Telnet

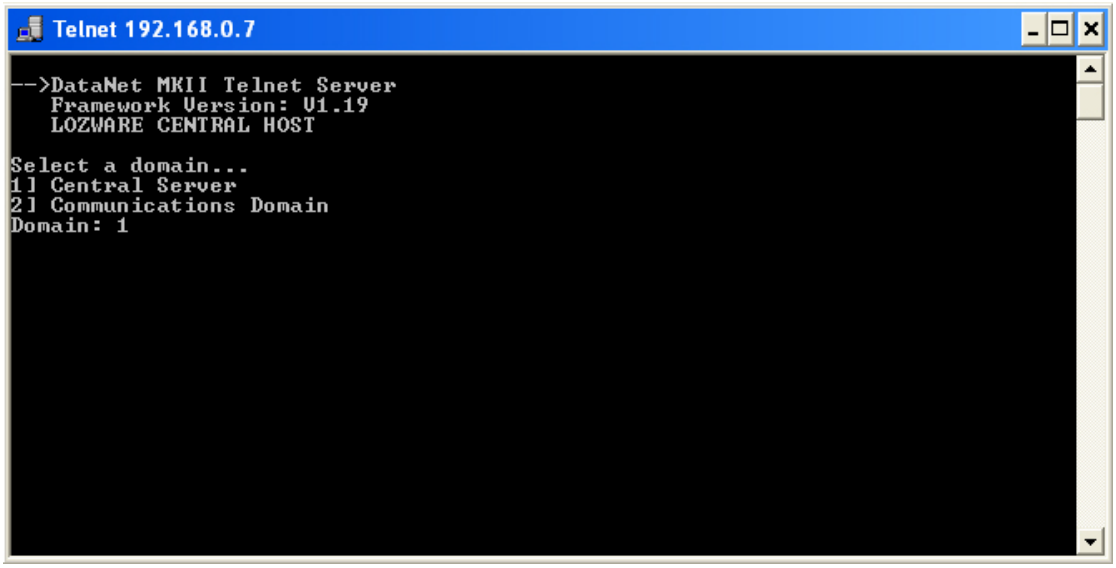
Logging In

Terminals

You can connect to DataNet with practically any telnet client, although DataNet prefers the VT100 terminal, you will find that DataNet will work on most terminal types. DataNet does echo back data, but it is not necessary to disable local echo as DataNet should configure terminals automatically; however, if you are experiencing a double echo, then disable local echo on your telnet terminal.

Access Code & Domain Selection

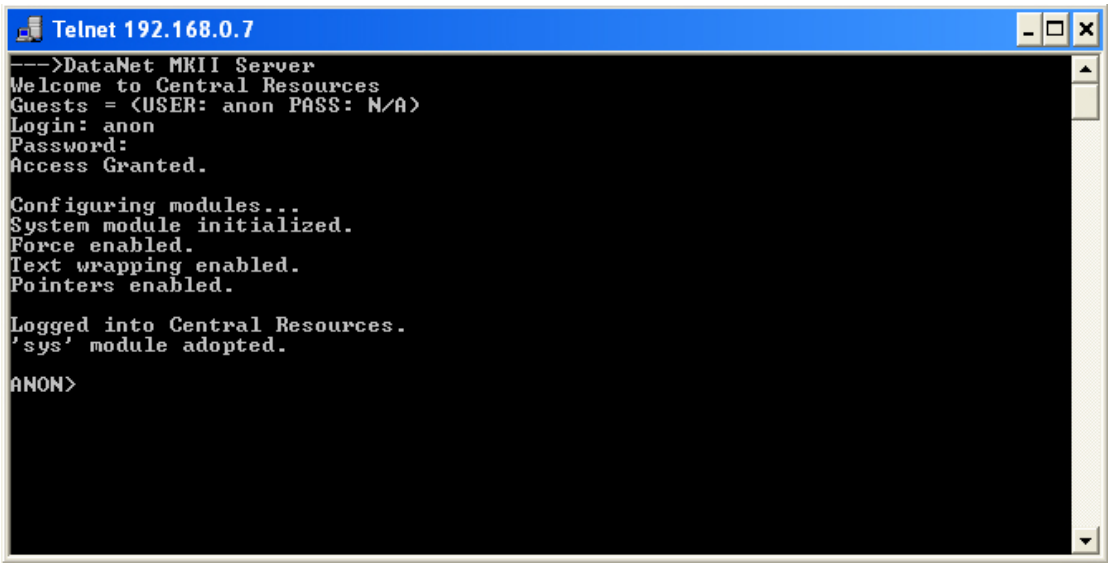
Once connected to the server, you may be prompted for an Access Code, some servers do not employ these security measures and there for the Domain Choice menu will be displayed. You will see all of the server's domains listed, each with a number assigned:



You can specify which domain you wish to connect to by entering in its number, upon pressing enter, you will be connected to the specified domain and prompted to login.

Logging into a Domain

At the domain's login prompt, you must firstly type in your username, press enter, and then enter in your password. Upon a successful login, the domain may run a start-up script (normally to configure modules); you should then be presented with a command prompt:



By default the command prompt should indicate your current path, however, some commands do have the ability to change your command prompt (e.g. the 'com.room' command will change the command prompt so that it reads 'CHAT>'). The command prompt is there to indicate where to type your next command, and generally displays that the server is ready, and all processing for the previous command is complete.

Typing Commands

Command Syntaxes

With DataNet, there are three different ways to type a command, and they are:

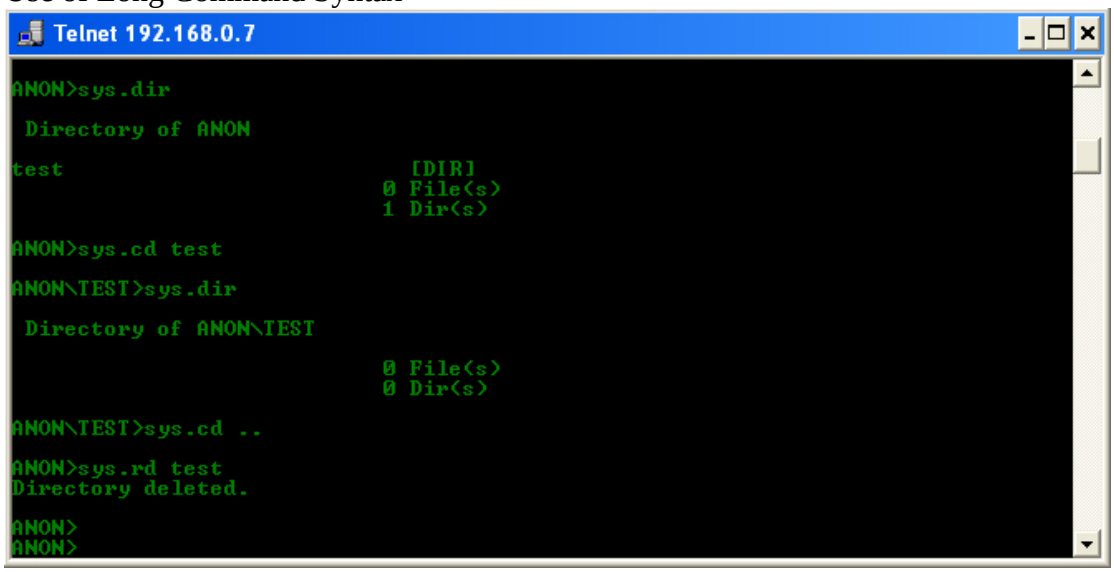
- Long Command Syntax MODULE.COMMAND ARGUMENT
- Short Command Syntax COMMAND ARGUMENT
- Adopted Command Syntax ARGUMENT

As you can see, the format for each command is fairly logical (bearing in mind that you understand the concept of 'modules', view 'What Are Modules' if you don't). Originally only the long command syntax existed, but after a while I began to get repetitive strain injury after typing out the module name for each command (joke), so I developed a way of 'adopting' modules. This meant that if you were going to be carrying out a bunch of commands from the same module, then instead of typing MOD.COMMAND each time, you could just skip out the module and type COMMAND. To use this short command syntax you obviously need to tell DataNet which module you want all of your commands to be passed to, the command for this is MOD:MODULE (replacing MODULE with a module name). Upon carrying out this command, you will be able to use the short command syntax for the specified module. If at any point you do need to carry out a command from a different module, you can still use the long command syntax and keep with the adopted module; the

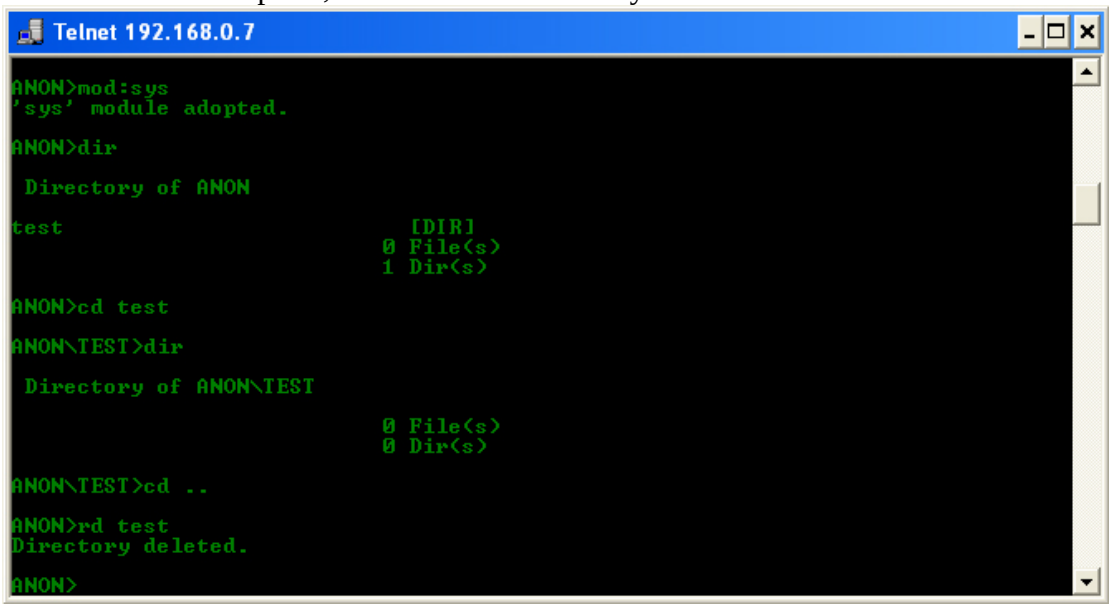
long command syntax will force the command to be sent to the specified module, and not the adopted one.
The adopted command syntax is more extreme than the other command syntaxes as it adopts both a module, and one of its commands. This means that anything typed in will be treated as an argument and sent to the specified module, aimed at the specified command. You will very rarely need to use this function. To use the adopted command syntax, you must adopt a command, the command for this is COM:MODULE.COMMAND. When you are in this adopted state, you will not be able to use the long command syntax, so to carry out any other command apart from the one adopted, you must un-adopt the command, then run your command. To un-adopt either a command or a module, type this:
MOD: (to un-adopt a module)
COM: (to un-adopt a command)

Here are two examples which carry out exactly the same processes, just differently:

Use of Long Command Syntax -

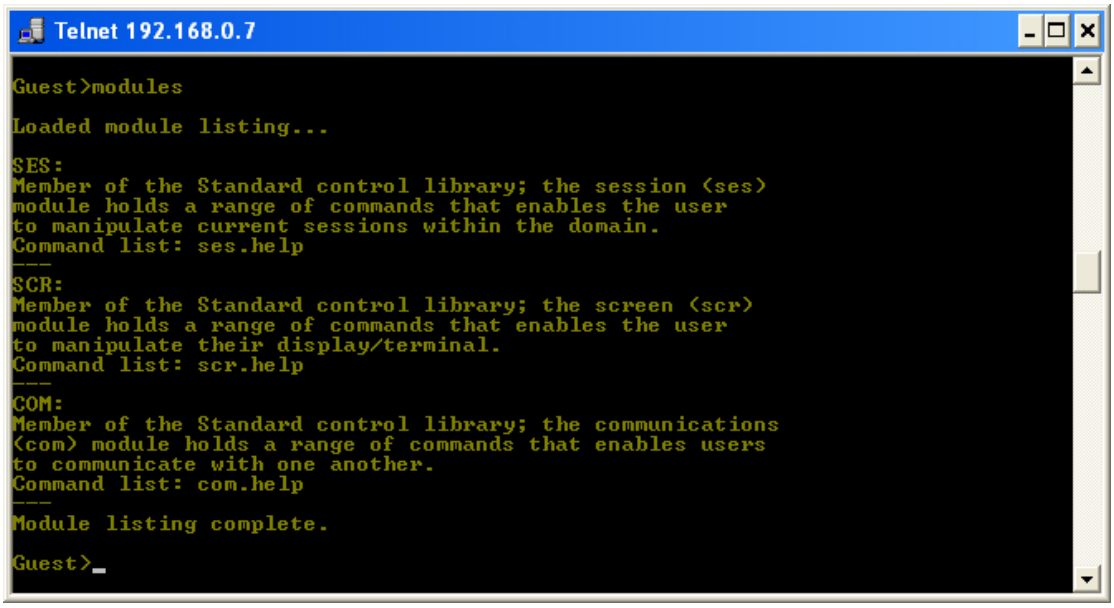


Use of Module Adoption, and Short Command Syntax-

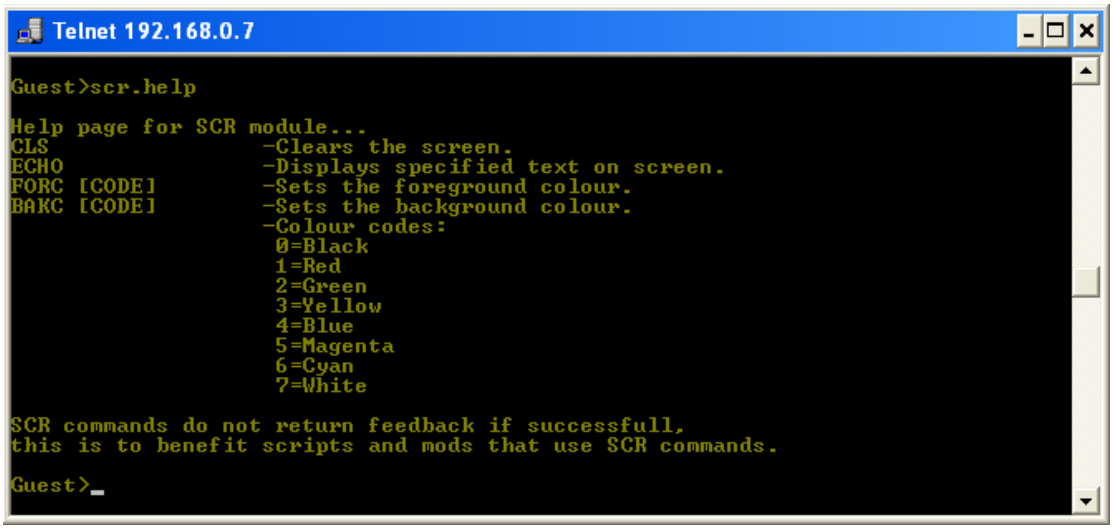


What are the Commands?

The command library for each domain can vary, as the administrators who setup the domain have chosen which modules are to be loaded with the domain, and which aren't. The 'Modules' command was written to help you find out which commands you have access to. As long as you have not adopted a command, you can type 'Modules' at any point in your session. Upon entering this command, DataNet will display which modules are loaded into the current domain:



As you can see, the list contains each module's name, followed by a description of that module, then followed by the command that will bring up further information on that module. So, if we type SCR.HELP, DataNet will present us with in-depth help on the SCR module:



The example above shows us every command that the SCR module has to offer, plus a very brief explanation of that command, and how to use it. So, if we wanted to change the foreground colour of the console to blue using the long command syntax, we would type the following: SCR.FORC 4

Error Handling

How to Solve Errors

DataNet is a fairly shy program in terms of communicating with the administrator, and therefore does not produce many error messages at all. So if DataNet doesn't appear to be working properly, then there is one rule that I have followed throughout the making of DataNet that has worked wonders for me. These simple steps are:

1. Close DataNet via the menu Server>Exit
2. Re-Open DataNet
3. Look out for any error messages (these should explain the problem and how to fix it).

Credits

Creator of DataNet

LozWare - Creator of DataNet and this 15 page user manual. LozWare is not actually a company, as it is more of a name that I use instead of my own (Lawrence Wagerfield), please check out the official LozWare homepage:
http://www.lozware.ndo.co.uk

References

In order to create the installation wizard for DataNet, I used two tools:
ISTool – http://www.istool.org/
Inno Setup 5 – http://www.jrsoftware.org/
I obviously used other programs to create DataNet and this PDF file, but they were created by big corporate companies who not need a mention, as they have enough publicity anyway.

Thanks To

Dave Scarmozzino - who created the .INI functions, and registry commands.
Kroum Grigorov – who tried explaining the concepts of Console APP interaction, but I have yet to master it and include it in the EXE module.

LozWare