Every now and then I'm faced with the need to have a command line interface in one of my programs. The first time I mashed up a primitive parser myself. The second time I just grabbed the first library off the net but now I finally want to get a good overview. So as a first step I'm going to collect whatever library I can find on the net. The only conditions are: it has to be build for .NET and free to use. For my own use there will be a few more criteria but that can wait for now.

The List

Library Name	License	Version	Release	
BizArk	Microsoft Public License	2.0.1 stable	2011-11-20	4.0 174 kB
Command Line Parser Library	Microsoft Public License	2.6 stable	2010-06-29	
CommonLibrary.NET	MIT License	0.9.8 alpha	2011-11-27	4.
C# Command-Line Option Parsing Library	Apache License 2.0	1.0.1	2005-(	
c# test.net	Apache License 2.0	1.11.924.348	2011-09-24	2.0, 3.5, 4.0 484 kB
Genghis	Genghis License	0.8	2007-08-24	2.0?
Getopt .NET	LGPL	0.9.1	2004-05-25	?
NDesk Options	MIT X11	0.2.1 unstable	2008-10-20	?
.NET CLI	BSD License	0.1.1	2008-05-30	2.0
TestAPI	Microsoft Public License	0.6 stable	2011-02-07	3.5

So that's already quite a list. There were a lot more options out there in the form of code snippets on forums or contributions on programming sites but I'm deliberately leaving those out. I don't have the time to evaluate and compare every little snippet out there. Also I want to know that a particular solution was developed and improved over time based on the feedback from users and is not just the child of a one night stand with the latest C# compiler...

Argument Format

Next I'm going to look at the supported argument formats. As with everything computery there are plenty of different styles on how CL arguments are supposed to be formatted.

Slash vs Dash

One of the main differences is what character is used to start an argument. On Windows environments programs generally use slashes to start their arguments like /help and so on. On Unix and Linux or generally with GNU software the dash variant like -v or --verbose is prevalent.

Short & Long Versions / Multiple Aliases / Auto completion

GNU software often allows options to be specified with short (-v) and long identifiers (--verbose). A more generic approach used by some libraries is to allow multiple aliases for the same option to be defined or to auto complete partial identifiers.

Option Grouping

Another common feature isthe ability of multiple flags to be specified in compact form. Instead of writing "/a /b /c" one can achieve the same effect with "/abc" .

Here is an overview of what the different projects support in that regard:

Library Name	First char	Separators	Aliases
BizArk	/	?	Multiple
Command Line Parser Library		?	Long / Short

CommonLibrary.NET	Any	Any	No?
C# Command-Line Option? Parsing Library ? ?	?		
c# test.net	/ -	?	?
Genghis	/	?	No
Getopt .NET	-	?	Long / Short
NDesk Options	/ -	?	Multiple
.NET CLI - ? Long / Short No?			
TestAPI	Any	Any	?

Library Reviews

BizArk

BizArk seems to use the / variant exclusively, supports multiple aliases for every argument but does not allow for multiple flags to be specified in compact form.

Command Line Parser Library

The Command Line Parser Library supports the dash variant exclusively. Both long and short versions can be used and option grouping is supported as well.

CommonLibrary.NET

CommonLibrary.NET seems to allow the programmer to freely define the characters to use for initiating an option and separating it from the value. On the first glance multiple aliases and compact form are not supported.

c# test.net

C# test.net appears to allow slash, dash and numbered parameters but is severely lacking in terms of examples. I've found a post by the developer on StackOverflow and one on his blog .

The library is interesting though in it's approach since it allows completely different set of parameters for different commands.

Genghis

Genghis seems to support only the slash variant. Multiple aliases and grouping are not supported.

Getopt .NET

Getopt .NET is a port of the GNU getopt function and therefore supports the dash arguments.

NDesk Options

Formerly known as Mono.Options this library supports both slash and dash options, allows grouping and multiple aliases.

TestAPI

TestAPI is a collection of classes for various tasks. The command line parser supports a customizable character for initiating an argument and for separating it from its value.

Ok. That's enough for now. There are a few more things I need to look at - such as how comfortable they are to work with and if they are compatible with a commercial software product. But I need a break now.