

Namespace CMD_Parser

Classes

[Option](#)

Houses the Option definitions.

[OptionBuilder](#)

Builder pattern class; used to provide a fluid option defining experience to the user, while not exposing them to the Option objects.

[ParameterBuilder<T>](#)

Builder pattern class used to provide a fluid parameter defining experience to the user, while not exposing them to the Parameter objects.

[Parameter<T>](#)

Generic parameter class which holds the parameter data.

[ParserControl](#)

The loadbearing class, with which the user is able to communicate when defining options and their parameters, as well as when parsing the values obtained from these options during runtime.

Class Option

Namespace: [CMD.Parser](#)

Assembly: CMD_Parser.dll








Houses the Option definitions.

```
public class Option
```

Inheritance

[object](#)  ← Option

Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

Constructors

Option(string)

Parametrized constructor which takes the option name.

```
public Option(string name)
```

Parameters

name [string](#) 

The name of the option.

Class OptionBuilder


Namespace: [CMD.Parser](#)

Assembly: CMD.Parser.dll








Builder pattern class; used to provide a fluid option defining experience to the user, while not exposing them to the Option objects.

```
public class OptionBuilder
```

Inheritance

[object](#)  ← OptionBuilder

Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

Methods

AsLongOption()

Sets the option as a long option. Throws a BadName exception if the name is already taken. The option will also not be accepted without this, or the AsShortOption method, an AmbiguousOption exception will be thrown.

```
public OptionBuilder AsLongOption()
```

Returns

[OptionBuilder](#)

AsRequired()

Sets the option as required. If it is not set when the program runs, an UnsetOption exception will be thrown.

```
public OptionBuilder AsRequired()
```

Returns

[OptionBuilder](#)

AsShortOption()

Sets the option as a short option. Throws a `BadName` exception if the name is not 1 character long, or if the name is already taken. The option will also not be accepted without this, or the `AsLongOption` method, an `AmbiguousOption` exception will be thrown.

```
public OptionBuilder AsShortOption()
```

Returns

[OptionBuilder](#)

SetOption()

```
public void SetOption()
```

WithAlias(params string[])

Allows the user to set an alias for the option. A `BadName` exception is thrown if the name is already taken. It can take numerous aliases.

```
public OptionBuilder WithAlias(params string[] alias)
```

Parameters

`alias` [string](#)[↗][]

The alias for the option.

Returns

[OptionBuilder](#)

WithGroup(params int[])

Allows the user to set a group for the option. The group is defined by its number (integer). Options within different groups can not appear together on the command line. If an option doesn't have a group set, it can be used with everything. Options can have multiple groups. This will be displayed in the help and a ForbiddenCombination exception will be thrown if different groups appear together on the command line.

```
public OptionBuilder WithGroup(params int[] groupNum)
```

Parameters

groupNum [int\[\]](#)

The group numbers for the option.

Returns

[OptionBuilder](#)

WithHelp(string)

Allows the user to define the help text for the option.

```
public OptionBuilder WithHelp(string help)
```

Parameters

help [string](#)

The help text for the option.

Returns

[OptionBuilder](#)

WithPair(string)

Allows the user to set a paired name for the option. If it is a long option, the pair must be a short option, and vice versa. A `BadName` exception will be thrown if the name is already taken, if the name does not conform to the type of option it should be, or if the method is used before the `IsShort` or `IsLong` methods once the option is set.

```
public OptionBuilder WithPair(string name)
```

Parameters

name [string](#) 

The paired name for the option.

Returns

[OptionBuilder](#)

WithParameter<T>(Parameter<T>)

Allows the user to set only one parameter for the option in the form of a built `Parameter` object.

```
public OptionBuilder WithParameter<T>(Parameter<T> parameter)
```

Parameters

parameter [Parameter](#)<T>

The parameter for the option.

Returns

[OptionBuilder](#)

Type Parameters

T

Class ParameterBuilder<T>

Namespace: [CMD Parser](#)

Assembly: CMD_Parser.dll


Builder pattern class used to provide a fluid parameter defining experience to the user, while not exposing them to the Parameter objects.

```
public class ParameterBuilder<T>
```








Type Parameters

T

Inheritance

[object](#)  ← ParameterBuilder<T>

Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

Constructors

ParameterBuilder(string)

Constructor for cases when the ParameterBuilder needs to be accessed directly and when it is being set up as an option parameter.

```
public ParameterBuilder(string name)
```

Parameters

name [string](#) 

The name of the paramter.

Methods

AsRequired()

Specifies whether the parameter is required. If it is not set on the command line (either as a parameter or an option parameter), an UnsetParameter exception will be thrown.

```
public ParameterBuilder<T> AsRequired()
```

Returns

[ParameterBuilder](#)<T>

GetParameter()

Allows the user to get the fully built Parameter object to pass to the OptionBuilder class. Only available if the instance of the builder class was created directly, otherwise a ForbiddenMethod exception will be thrown.

```
public Parameter<T> GetParameter()
```

Returns

[Parameter](#)<T>

SetParameter()

Allows the user to set the parameter (plain option). Only available if the builder class was accessed through the AddParameter method in ParserControl, otherwise a ForbiddenMethod exception will be thrown. These parameters will capture plain arguments in the order they were set.

```
public void SetParameter()
```

WithDefaultValue(T)

Allows the user to set a default value for the parameter.

```
public ParameterBuilder<T> WithDefaultValue(T value)
```

Parameters

value T

The default value.

Returns

[ParameterBuilder](#)<T>

WithHelp(string)

Allows the user to define the help text for the parameter.

```
public ParameterBuilder<T> WithHelp(string help)
```

Parameters

help [string](#)

The help text for the parameter.

Returns

[ParameterBuilder](#)<T>

WithLambdaValidation(Func<T, bool>)

Allows the user to enter a lambda function for parameter verification.

```
public ParameterBuilder<T> WithLambdaValidation(Func<T, bool> func)
```

Parameters

func [Func](#)<T, [bool](#)>

The lambda function for value verification.

Returns

Class Parameter<T>

Namespace: [CMD Parser](#)

Assembly: CMD_Parser.dll

Generic parameter class which holds the parameter data.

```
public class Parameter<T>
```








Type Parameters

T

Inheritance

[object](#)  ← Parameter<T>

Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

Methods

Value()

Parameter value getter.

```
public T Value()
```

Returns

T

Class ParserControl


Namespace: [CMD.Parser](#)

Assembly: CMD.Parser.dll








The loadbearing class, with which the user is able to communicate when defining options and their parameters, as well as when parsing the values obtained from these options during runtime.

```
public class ParserControl
```

Inheritance

[object](#)  ← ParserControl

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#) 

Methods

AddOption(string)

Provides the user with an instance of `OptionBuilder`, which they can set up, as an option instance within the `ParserControl` object.

```
public OptionBuilder AddOption(string name)
```

Parameters

name [string](#) 

The short or long name of the option.

Returns

[OptionBuilder](#)

An instance of the `OptionBuilder` class.

AddParameter<T>(string)

Provides the user with an instance of the generic ParameterBuilder, which they can set up, as a parameter (plain option) instance within the ParserControl object. Can be parametrized as if it were an option parameter, the default is a string parameter.

```
public ParameterBuilder<T> AddParameter<T>(string name)
```

Parameters

name [string](#)

The name of the parameter.

Returns

[ParameterBuilder](#)<T>

An instance of the ParameterBuilder class.

Type Parameters

T

GetHelp()

Allows the user to get formatted help for all of the options, parameters and option parameters, as well as information on groups and usage.

```
public string GetHelp()
```

Returns

[string](#)

A string which holds the entirety of the help.

GetHelp(string)

Allows the user to get the help text for the chosen option or parameter, to which they can refer by any of their valid names.

```
public string GetHelp(string name)
```

Parameters

name [string](#)

The name of the option or parameter, the help for which we want.

Returns

[string](#)

A string which holds the help text.

GetOptionNames()

Allows the user to access the names of all options and parameters that have been set in the command line, it will always return the name of the long option if it has both a long and short variants, and only return the short name if that is the only variant of that command.

```
public List<string> GetOptionNames()
```

Returns

[List](#) <[string](#)>

A list of strings representing the names of the options.

GetOptionParameterHelp(string)

Allows the user to get the help text for the parameter of the chosen option, to which they can refer by any of its valid names.

```
public string GetOptionParameterHelp(string optionName)
```

Parameters

`optionName` [string](#)

The name of the option, for which we want the parameter help.

Returns

[string](#)

A string which holds the help text.

GetUncaught()

Allows the user to retrieve all command line arguments which hadn't been caught by any specified option, parameter or option parameters, all as strings, identical to how they were on the command line.

```
public List<string> GetUncaught()
```

Returns

[List](#) <[string](#)>

A list of strings which holds all uncaught arguments.

GetValue(string)

Allows the user to retrieve the Parameter object for a specified option. The user is assumed to know what type of parameters they expect from each of the options. If the parameter isn't set and there is no default value set, true will be returned.

```
public dynamic GetValue(string optionName)
```

Parameters

`optionName` [string](#)

The name of the option for which the parameter is requested.

Returns

dynamic

A Parameter object corresponding to the specified option name.

IsSet(string)

Allows the user to check whether a specified option or parameter was set on the command line.

```
public bool IsSet(string name)
```

Parameters

name [string](#)

The name of the option or parameter, the status of which we are checking.

Returns

[bool](#)

A boolean value, based on whether the option was set.

ParseArgs(string[])

Parses all the arguments on the command line, and prepares the ParserControl instance for queries about the options, parameters and option parameters through its methods. If any required parameters are missing a MissingParameter exception will be thrown.

```
public void ParseArgs(string[] args)
```

Parameters

args [string](#)[]

The array of command-line arguments to be parsed.