Class CommandLine



Propose: C++ class to handle data from the command line.

Version 2.0.0

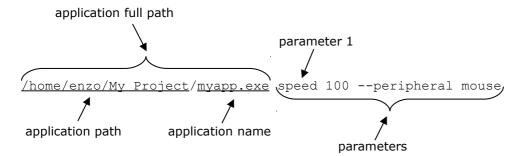
Enzo Roberto Verlato - enzover@ig.com.br https://github.com/FreeSource

Supported and tested platforms:

0.S.	Compiler	Make
WindowsXP SP2	MinGW gcc 4.6.1	gmake 3.82
Linux openSUSE 11.4 / 12.2	gcc 4.5.1 / 4.7.1	gmake 3.82
OpenIndiana 151a	gcc 3.4.3	gmake 3.81
FreeBSD 9.0	gcc 4.2.1	gmake 3.82
Solaris 9 / 10	gcc 3.3.2 / 3.4.6	gmake 3.80 / 3.81
Mac OS X 10.8.2	gcc 4.2.1	gmake 3.81

Copyright (c) 2012 Enzo Roberto Verlato. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

The standard structure of a command line:



Members:

```
string getApplicationName()
string getApplicationPath()
string getApplicationFullPath()
string getCurrentWorkingDirectory()

bool hasParameters()
bool hasParameter( int parameterPosition )

int getParametersNumber()
string getAllParameters()
string getParameter( int parameterPosition )
int getParameterAsInteger( int parameterPosition )
float getParameterAsFloat( int parameterPosition )
gotoFirstParameter()
bool gotoNextParameter()
int getCurrentPosition()
```

```
string getCurrentParameter()
int getCurrentParameterAsInteger()
float getCurrentParameterAsFloat()
string getFirstParameter()
int getFirstParameterAsInteger()
float getFirstParameterAsFloat()
string getLastParameter()
int getLastParameterAsInteger()
float getLastParameterAsFloat()
setOptionPrefix( string optionPrefix )
setOptionPostfix( string optionPostfix )
string getOptionPrefix()
string getOptionPostfix()
bool hasOption( string option )
string getOptionValue( string option )
int getOptionValueAsInteger( string option )
float getOptionValueAsFloat( string option )
string getOptionLongValue( string option )
optionCaseSensitive()
optionCaseInsensitive()
bool isOptionCaseSensitive()
```

string getCommandLine() 🏠

Description: Retrieves the command line string for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <windows.h>
 4 #include <iostream>
 5 #include <cstdlib>
 6 #include <stdexcept>
 8 using std::cout;
9 using std::endl;
10 using std::string;
11 using std::runtime error;
12
13 int WINAPI WinMain ( HINSTANCE hInstance, HINSTANCE hPrevInstance,
14 PSTR szCmdLine, int iCmdShow ) {
15
       try {
16
           util::CommandLine commandLine;
17
           cout << commandLine.getCommandLine() << endl;</pre>
18
           return EXIT SUCCESS;
19
       catch ( runtime error &error ) {
           cout << "Exception occurred: " << error.what() << endl;</pre>
           return EXIT FAILURE;
23
       }
24 }
25
```

```
linux:/home/enzo # ./myapp My first example
/home/enzo/myapp My first example
```

string getApplicationName() $\hat{\mathbf{q}}$

Description: Retrieves the name of the application for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main( int argc, char *argv[] ) {
       try {
13
14
           util::CommandLine commandLine;
15
           cout << commandLine.getApplicationName() << endl;</pre>
           return EXIT SUCCESS;
16
17
       }
18
       catch ( runtime error &error ) {
19
           cout << "Exception occurred: " << error.what() << endl;</pre>
20
           return EXIT_FAILURE;
       }
21
22 }
23
```

Output:

```
linux:/home/enzo # ./myapp
myapp
```

string getApplicationPath() $\ensuremath{ \upphi}$

Description: Retrieving the application path of the current process, not including the name of the program itself.

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
12 int main() {
13
     try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getApplicationPath() << endl;</pre>
           return EXIT SUCCESS;
16
17
       }
```

```
catch ( runtime_error &error ) {
    cout << "Exception occurred: " << error.what() << endl;
    return EXIT_FAILURE;
}
}</pre>
```

```
linux:/home/enzo # ./myapp
/home/enzo
```

string getApplicationFullPath() ♠

Description: Retrieving the application path of the current process, including the name of the program itself.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getApplicationFullPath() << endl;</pre>
16
           return EXIT SUCCESS;
17
       }
       catch ( runtime error &error ) {
18
           cout << "Exception occurred: " << error.what() << endl;</pre>
19
           return EXIT FAILURE;
       }
21
22 }
23
```

Output:

```
linux:/home/enzo # ./myapp
/home/enzo/myapp
```

string getCurrentWorkingDirectory() ♠

Description: Retrieves the current working directory for the current process.

```
#include <CommandLine.h>

#include <iostream>
#include <cstdlib>
#include <stdexcept>

using std::cout;
```

```
8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getApplicationPath() << endl;</pre>
16
           cout << commandLine.getCurrentWorkingDirectory() << endl;</pre>
17
           return EXIT_SUCCESS;
       }
18
       catch ( runtime error &error ) {
19
           cout << "Exception occurred: " << error.what() << endl;</pre>
           return EXIT FAILURE;
21
       }
23 }
24
```

```
linux-hevv:/home/enzo/CommandLine/main # /home/enzo/myapp
/home/enzo
/home/enzo/CommandLine/main
```

bool hasParameters() ♠

Description: Checks if the command line has parameters.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 6
 7 using std::cout;
 8 using std::endl;
9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
           if( commandLine.hasParameters() ) {
15
                cout << "yes" << endl;</pre>
16
           }
17
           else {
18
               cout << "no" << endl;</pre>
19
           }
20
           return EXIT SUCCESS;
21
       }
22
       catch ( runtime error &error ) {
23
           cout << "Exception occurred: " << error.what() << endl;</pre>
24
           return EXIT FAILURE;
25
       }
26
27 }
28
```

```
linux:/home/enzo # ./myapp parameter1 parameter2 parameter3
```

yes

bool hasParameter(int parameterPosition) ♠
Description: Checks if a specified parameter exists.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime_error;
12 int main() {
13
       try {
14
            util::CommandLine commandLine;
15
            if( commandLine.hasParameter( 2 ) ) {
                cout << "yes" << endl;</pre>
16
17
           }
           else {
18
                cout << "no" << endl;</pre>
19
           }
           return EXIT SUCCESS;
21
       }
22
       catch ( runtime_error &error ) {
23
           cout << "Exception occurred: " << error.what() << endl;</pre>
24
           return EXIT FAILURE;
25
       }
26
27 }
28
```

Output:

```
linux:/home/enzo # ./myapp How To Pass Parameters To Main() And Use Them
yes
```

int getParametersNumber() ♠

Description: Returns the total number of parameters on the command line for the current process, not including the name of the program itself.

```
#include <CommandLine.h>

#include <iostream>
#include <cstdlib>
#include <stdexcept>

using std::cout;
using std::endl;
using std::string;
using std::string;
int main() {
 try {
```

```
14
            util::CommandLine commandLine;
15
            cout << commandLine.getParametersNumber() << endl;</pre>
16
            return EXIT SUCCESS;
17
       }
       catch ( runtime error &error ) {
18
           cout << "Exception occurred: " << error.what() << endl;</pre>
19
            return EXIT FAILURE;
20
       }
21
22 }
23
```

```
linux:/home/enzo # ./myapp The C++ Programming Language
4
```

string getAllParameters() 🏠

Description: Retrieves all the parameters on the command line for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
12 int main() {
13
       try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getAllParameters() << endl;</pre>
16
           return EXIT SUCCESS;
17
18
       catch ( runtime error &error ) {
           cout << "Exception occurred: " << error.what() << endl;</pre>
19
           return EXIT FAILURE;
20
21
       }
22 }
23
```

Output:

```
linux:/home/enzo # ./myapp High thoughts must have high language
High thoughts must have high language
```

string getParameter(int parameterPosition) $\hat{\gamma}$

Description: Retrieves the parameter of the specified parameter position on the command line for the current process.

```
#include <CommandLine.h>

#include <iostream>
```

```
4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
       try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getParameter( 2 ) << endl;</pre>
           return EXIT SUCCESS;
16
17
       }
       catch ( runtime_error &error ) {
18
          cout << "Exception occurred: " << error.what() << endl;</pre>
19
           return EXIT FAILURE;
20
21
       }
22 }
23
```

```
linux:/home/enzo # ./myapp C++ evolved from C
evolved
```

int getParameterAsInteger(int parameterPosition) $\widehat{\phi}$

Description: Retrieves the parameter of the specified parameter position on the command line as integer for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
   try {
           util::CommandLine commandLine;
14
           cout << commandLine.getParameterAsInteger( 2 ) << endl;</pre>
15
           return EXIT SUCCESS;
16
17
       } catch ( runtime error &error ) {
          cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
20
21 }
22
```

```
linux:/home/enzo # ./myapp --price 1.99
1
```

float getParameterAsFloat(int parameterPosition)

Description: Retrieves the parameter of the specified parameter position on the command line as float for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
           util::CommandLine commandLine;
14
15
           cout << commandLine.getParameterAsFloat( 2 ) << endl;</pre>
           return EXIT SUCCESS;
16
       } catch ( runtime error &error ) {
17
           cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
21 }
```

Output:

```
linux:/home/enzo # ./myapp --price 1.99
1.99
```

gotoFirstParameter() ♠

Description: Points to the first parameter on the command line for the current process.

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
8 using std::endl;
9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
     try {
           util::CommandLine commandLine;
14
15
           commandLine.gotoNextParameter();
16
           cout << commandLine.getCurrentParameter() << endl;</pre>
17
           commandLine.gotoFirstParameter();
18
           cout << commandLine.getCurrentParameter() << endl;</pre>
           return EXIT SUCCESS;
19
20
       catch ( runtime error &error ) {
21
           cout << "Exception occurred: " << error.what() << endl;</pre>
```

```
23          return EXIT_FAILURE;
24     }
25 }
26
```

```
linux:/home/enzo # ./myapp Principles and Practice using C++
and
Principles
```

bool gotoNextParameter() ♠

Description: Points to the next parameter on the command line for the current process. Returns false if is at the last parameter.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
12 int main() {
13
       try {
14
           util::CommandLine commandLine;
15
16
                cout << commandLine.getCurrentParameter() << endl;</pre>
17
           while( commandLine.gotoNextParameter() );
18
           return EXIT SUCCESS;
19
       }
20
       catch ( runtime error &error ) {
           cout << "Exception occurred: " << error.what() << endl;</pre>
21
           return EXIT FAILURE;
22
       }
23
24 }
25
```

Output:

```
linux:/home/enzo # ./myapp GCC the GNU Compiler Collection
GCC
the
GNU
Compiler
Collection
```

int getCurrentPosition()

Description: Retrieves the current position parameter on the command line for the current process.

```
1 #include <CommandLine.h>
2
```

```
3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
8 using std::endl;
 9 using std::string;
10 using std::runtime error;
12 int main() {
13
      try {
           util::CommandLine commandLine;
14
15
               cout << commandLine.getCurrentPosition() << endl;</pre>
16
17
           while( commandLine.gotoNextParameter() );
           return EXIT_SUCCESS;
18
19
20
       catch ( runtime error &error ) {
           cout << "Exception occurred: " << error.what() << endl;</pre>
21
           return EXIT FAILURE;
       }
23
24 }
25
```

```
linux:/home/enzo # ./myapp Principles and Practice using C++
1
2
3
4
5
```

string getCurrentParameter() 🏟

Description: Retrieves the current parameter on the command line for the current process.

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
       try {
14
           util::CommandLine commandLine;
15
16
                cout << commandLine.getCurrentParameter() << endl;</pre>
17
           while( commandLine.gotoNextParameter() );
           return EXIT SUCCESS;
18
19
20
       catch ( runtime error &error ) {
           cout << "Exception occurred: " << error.what() << endl;</pre>
           return EXIT FAILURE;
       }
```

```
24 }
25
```

```
linux:/home/enzo # ./myapp Principles and Practice using C++
Principles
and
Practice
using
C++
```

int getCurrentParameterAsInteger()

Description: Retrieves the current parameter on the command line as integer for the current process. **Example:**

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
8 using std::endl;
9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
     try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getCurrentParameterAsInteger() << endl;</pre>
           return EXIT SUCCESS;
16
       } catch ( runtime error &error ) {
17
          cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
20
21 }
```

Output:

```
linux:/home/enzo # ./myapp 1.99 cents
1
```

float getCurrentParameterAsFloat() ☆

Description: Retrieves the current parameter on the command line as float for the current process.

```
#include <CommandLine.h>

#include <iostream>
#include <cstdlib>
#include <stdexcept>

using std::cout;
using std::endl;
using std::string;
using std::runtime_error;
```

```
11
12 int main() {
13
       try {
14
           util::CommandLine commandLine;
           cout << commandLine.getCurrentParameterAsFloat() << endl;</pre>
15
           return EXIT SUCCESS;
16
       } catch ( runtime error &error ) {
17
          cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
20
21 }
```

```
linux:/home/enzo # ./myapp 1.99 cents
1.99
```

string getFirstParameter() 🏠

Description: Retrieves the first parameter on the command line for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
12 int main() {
13
       try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getFirstParameter() << endl;</pre>
16
           return EXIT SUCCESS;
17
       }
18
       catch ( runtime error &error ) {
           cout << "Exception occurred: " << error.what() << endl;</pre>
19
           return EXIT FAILURE;
20
21
       }
22 }
23
```

Output:

```
linux:/home/enzo # ./myapp C++ Development Environment
C++
```

string getFirstParameterAsInteger() ↔

Description: Retrieves the first parameter on the command line as integer for the current process.

```
#include <CommandLine.h>
```

```
3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
12 int main() {
13
      try {
           util::CommandLine commandLine;
14
           cout << commandLine.getFirstParameterAsInteger() << endl;</pre>
15
           return EXIT SUCCESS;
16
       } catch ( runtime error &error ) {
17
          cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
20
21 }
```

```
linux:/home/enzo # ./myapp 1.99 cents
1
```

string getFirstParameterAsFloat() ♠

Description: Retrieves the first parameter on the command line as float for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime_error;
12 int main() {
1.3
    try {
14
           util::CommandLine commandLine;
15
           cout << commandLine.getFirstParameterAsFloat() << endl;</pre>
           return EXIT SUCCESS;
16
17
       } catch ( runtime error &error ) {
          cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
20
21 }
22
```

Output:

```
linux:/home/enzo # ./myapp 1.99 cents
1.99
```

string getLastParameter() n



Description: Retrieves the last parameter on the command line for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
   try {
14
           util::CommandLine commandLine;
           cout << commandLine.getLastParameter() << endl;</pre>
15
           return EXIT SUCCESS;
16
     }
17
      catch ( runtime error &error ) {
18
19
          cout << "Exception occurred: " << error.what() << endl;</pre>
           return EXIT FAILURE;
       }
22 }
23
```

Output:

```
linux:/home/enzo # ./myapp The C++ Standard Library
Library
```

string getLastParameterAsInteger() ♠

Description: Retrieves the last parameter on the command line as integer for the current process.

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
     try {
           util::CommandLine commandLine;
14
15
           cout << commandLine.getLastParameterAsInteger() << endl;</pre>
           return EXIT SUCCESS;
16
       } catch ( runtime error &error ) {
17
          cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
21 }
```

```
linux:/home/enzo # ./myapp Book price = 49.99
49
```

string getLastParameterAsFloat() ↔

Description: Retrieves the last parameter on the command line as float for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
      try {
13
           util::CommandLine commandLine;
14
15
           cout << commandLine.getLastParameterAsFloat() << endl;</pre>
16
           return EXIT SUCCESS;
17
       } catch ( runtime_error &error ) {
           cout << "Exception occurred: " << error.what() << endl;</pre>
18
           return EXIT FAILURE;
19
       }
21 }
```

Output:

```
linux:/home/enzo # ./myapp Book price = 49.99
49.99
```

setOptionPrefix(string optionPrefix) ♠

Description: Define the prefix (string added in front of the option name) used to recognize an option on the command line.

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
           commandLine.setOptionPrefix( "--" );
15
           cout << commandLine.getOptionValue( "price" ) << endl;</pre>
```

```
return EXIT_SUCCESS;

return EXIT_SUCCESS;

catch ( runtime_error &error ) {
    cout << "Exception occurred: " << error.what() << endl;
    return EXIT_FAILURE;
}

}
</pre>
```

```
linux:/home/enzo # ./myapp --price 0.99
0.99
```

setOptionPostfix(string optionPostfix) ♠

Description: Define the postfix (string added to the end of the option name) used to recognize an option on the command line.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
     try {
14
           util::CommandLine commandLine;
           commandLine.setOptionPostfix( "=" );
15
           cout << commandLine.getOptionValue( "price" ) << endl;</pre>
16
           return EXIT SUCCESS;
17
18
19
       catch ( runtime error &error ) {
          cout << "Exception occurred: " << error.what() << endl;</pre>
20
           return EXIT FAILURE;
21
       }
22
23 }
24
```

Output:

```
linux:/home/enzo # ./myapp price=0.99
0.99
```

string getOptionPrefix() ♠

Description: Returns the prefix (string added in front of the option name) used to recognize an option on the command line.

```
#include <CommandLine.h>
#include <iostream>
```

```
4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
15
           commandLine.setOptionPrefix( "--" );
           cout << commandLine.getOptionPrefix() << endl;</pre>
16
17
           return EXIT SUCCESS;
18
       catch ( runtime error &error ) {
19
           cout << "Exception occurred: " << error.what() << endl;</pre>
20
           return EXIT FAILURE;
21
       }
23 }
24
```

```
linux:/home/enzo # ./myapp
--
```

string getOptionPostfix() ♠

Description: Returns the postfix (string added to the end of the option name) used to recognize an option on the command line.

Example:

```
1 #include <CommandLine.h>
 2
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
          util::CommandLine commandLine;
           commandLine.setOptionPostfix( "=" );
15
16
           cout << commandLine.getOptionPostfix() << endl;</pre>
17
           return EXIT SUCCESS;
18
       catch ( runtime error &error ) {
19
          cout << "Exception occurred: " << error.what() << endl;</pre>
20
           return EXIT FAILURE;
21
       }
22
23 }
24
```

```
linux:/home/enzo # ./myapp
```

=

bool hasOption (string option) $\hat{\phi}$

Description: Checks if the specified option exists.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
12 int main() {
13
       try {
14
            util::CommandLine commandLine;
            commandLine.setOptionPostfix( ":" );
15
            if( commandLine.hasOption( "ISBN-10" ) ) {
16
                cout << "yes" << endl;</pre>
17
18
           }
19
           else {
                cout << "no" << endl;</pre>
20
21
           }
           return EXIT SUCCESS;
       }
23
       catch ( runtime_error &error ) {
24
           cout << "Exception occurred: " << error.what() << endl;</pre>
25
           return EXIT FAILURE;
26
       }
27
28 }
29
```

Output:

```
linux:/home/enzo # ./myapp Paperback: 208 pages Publisher: O'Reilly
Media; 1 edition (August 19, 2011) Language: English ISBN-10: 1449397670
Weight: 14.4 ounces
yes
```

string getOptionValue(string option) $\hat{\gamma}$

Description: Retrieves the value of the specified option on the command line for the current process.

```
#include <CommandLine.h>

#include <iostream>
#include <cstdlib>
#include <stdexcept>

using std::cout;
using std::endl;
using std::string;
using std::runtime_error;
```

```
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
15
           commandLine.setOptionPostfix( ":" );
           cout << commandLine.getOptionValue( "Language" ) << endl;</pre>
16
           return EXIT SUCCESS;
17
18
       catch ( runtime error &error ) {
19
           cout << "Exception occurred: " << error.what() << endl;</pre>
20
           return EXIT FAILURE;
21
       }
23 }
24
```

```
linux:/home/enzo # ./myapp Paperback: 208 pages Publisher: O'Reilly
Media; 1 edition (August 19, 2011) Language: English ISBN-10: 1449397670
Weight: 14.4 ounces
English
```

int getOptionValueAsInteger(string option) $\hat{\phi}$

Description: Retrieves the value of the specified option on the command line as integer for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime_error;
11
12 int main() {
13
       try {
           util::CommandLine commandLine;
14
           commandLine.setOptionPostfix( ":" );
15
16
           cout << commandLine.getOptionValueAsInteger( "Weight" );</pre>
           cout << endl;</pre>
17
           return EXIT SUCCESS;
18
19
       } catch ( runtime error &error ) {
          cout << "Exception occurred: " << error.what() << endl;</pre>
20
           return EXIT FAILURE;
21
22
       }
23 }
24
```

```
linux:/home/enzo # ./myapp Paperback: 208 pages Publisher: O'Reilly
Media; 1 edition (August 19, 2011) Language: English ISBN-10: 1449397670
Weight: 14.4 ounces
14
```

float getOptionValueAsFloat(string option) 命

Description: Retrieves the value of the specified option on the command line as float for the current process.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
       try {
           util::CommandLine commandLine;
14
           commandLine.setOptionPostfix( "=" );
15
           cout << commandLine.getOptionValueAsFloat( "price" ) + 0.01;</pre>
16
           cout << endl;</pre>
17
           return EXIT SUCCESS;
18
       } catch ( runtime error &error ) {
19
           cout << "Exception occurred: " << error.what() << endl;</pre>
           return EXIT FAILURE;
       }
23 }
24
```

Output:

```
linux:/home/enzo # ./myapp price=0.98
0.99
```

string getOptionLongValue(string option) \widehat{w}

Description: Retrieves the long value of the specified option (a range of parameters delimited by the next option if it exists) on the command line for the current process.

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
           commandLine.setOptionPrefix( "--" );
15
16
           cout << commandLine.getOptionLongValue( "peripheral" );</pre>
17
           cout << endl;</pre>
           return EXIT SUCCESS;
```

```
linux:/home/enzo # ./myapp --speed 100 --peripheral mouse display
keyboard --price 1000
mouse display keyboard
```

optionCaseSensitive() 🏟

Description: Differ use of uppercase and lowercase letters on the option parameter for the other functions. Option parameter is case sensitive by default.

Example:

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
15
           commandLine.optionCaseSensitive();
           commandLine.setOptionPrefix( "--" );
16
17
           cout << commandLine.getOptionLongValue( "PERIPHERAL" );</pre>
           cout << endl;</pre>
18
19
           return EXIT SUCCESS;
     }
21
       catch ( runtime_error &error ) {
          cout << "Exception occurred: " << error.what() << endl;</pre>
           return EXIT_FAILURE;
23
       }
24
25 }
26
```

Output:

```
linux:/home/enzo # ./myapp --peripheral mouse
```

optionCaseInsensitive() 🏟

Description: No differ use of uppercase and lowercase letters on the option parameter for the other functions.

```
1 #include <CommandLine.h>
```

```
3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
14
           util::CommandLine commandLine;
15
           commandLine.optionCaseInsensitive();
           commandLine.setOptionPrefix( "--" );
16
17
           cout << commandLine.getOptionLongValue( "PERIPHERAL" );</pre>
           cout << endl;</pre>
18
           return EXIT_SUCCESS;
19
       catch ( runtime_error &error ) {
21
           cout << "Exception occurred: " << error.what() << endl;</pre>
           return EXIT FAILURE;
23
       }
24
25 }
26
```

```
linux:/home/enzo # ./myapp --peripheral mouse
mouse
```

bool isOptionCaseSensitive() $\hat{\mathbf{Q}}$

Description: Checks if the options are case sensitive.

```
1 #include <CommandLine.h>
 3 #include <iostream>
 4 #include <cstdlib>
 5 #include <stdexcept>
 7 using std::cout;
 8 using std::endl;
 9 using std::string;
10 using std::runtime error;
11
12 int main() {
13
      try {
           util::CommandLine commandLine;
15
           if( commandLine.isOptionCaseSensitive() ) {
16
                cout << "yes" << endl;</pre>
17
           }
           else {
18
               cout << "no" << endl;</pre>
19
           }
20
           return EXIT SUCCESS;
21
22
       catch ( runtime error &error ) {
23
           cout << "Exception occurred: " << error.what() << endl;</pre>
24
           return EXIT FAILURE;
25
```

26 **}**27
28

Output:

linux:/home/enzo # ./myapp
yes