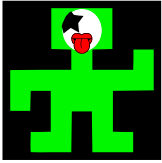


Class CommandLine



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>

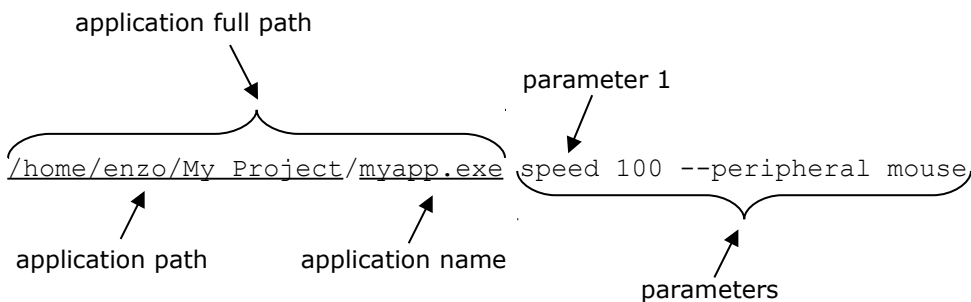
Supported and tested platforms:

O.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 getCommandLine()

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>
2
3  #include <windows.h>
4  #include <iostream>
5  #include <cstdlib>
6  #include <stdexcept>
7
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;
18         return EXIT_SUCCESS;
19     }
20     catch ( runtime_error &error ) {
21         cout << "Exception occurred: " << error.what() << endl;
22         return EXIT_FAILURE;
23     }
24 }
25

```

Output:

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

string `getApplicationName()` ↗

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

Example:

```
1 #include <CommandLine.h>
2
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( int argc, char *argv[] ) {
13     try {
14         util::CommandLine commandLine;
15         cout << commandLine.getApplicationName() << endl;
16         return EXIT_SUCCESS;
17     }
18     catch ( runtime_error &error ) {
19         cout << "Exception occurred: " << error.what() << endl;
20         return EXIT_FAILURE;
21     }
22 }
23
```

Output:

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

string `getApplicationPath()` ↗

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

Example:

```
1 #include <CommandLine.h>
2
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;
15         cout << commandLine.getApplicationPath() << endl;
16         return EXIT_SUCCESS;
17     }
18 }
```

```

18     catch ( runtime_error &error ) {
19         cout << "Exception occurred: " << error.what() << endl;
20         return EXIT_FAILURE;
21     }
22 }
23

```

Output:

```

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>
2
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;
15         cout << commandLine.getApplicationFullPath() << endl;
16         return EXIT_SUCCESS;
17     }
18     catch ( runtime_error &error ) {
19         cout << "Exception occurred: " << error.what() << endl;
20         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.

Example:

```

1  #include <CommandLine.h>
2
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;
15         cout << commandLine.getApplicationPath() << endl;
16         cout << commandLine.getCurrentWorkingDirectory() << endl;
17         return EXIT_SUCCESS;
18     }
19     catch ( runtime_error &error ) {
20         cout << "Exception occurred: " << error.what() << endl;
21         return EXIT_FAILURE;
22     }
23 }
24

```

Output:

```

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

```

bool hasParameters() ↗

Description: Checks if the command line has parameters.

Example:

```

1 #include <CommandLine.h>
2
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;
15         if( commandLine.hasParameters() ) {
16             cout << "yes" << endl;
17         }
18         else {
19             cout << "no" << endl;
20         }
21         return EXIT_SUCCESS;
22     }
23     catch ( runtime_error &error ) {
24         cout << "Exception occurred: " << error.what() << endl;
25         return EXIT_FAILURE;
26     }
27 }
28

```

Output:

```

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

```

```
yes
```

bool hasParameter(**int** parameterPosition) ↗

Description: Checks if a specified parameter exists.

Example:

```
1 #include <CommandLine.h>
2
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;
15         if( commandLine.hasParameter( 2 ) ) {
16             cout << "yes" << endl;
17         }
18         else {
19             cout << "no" << endl;
20         }
21         return EXIT_SUCCESS;
22     }
23     catch ( runtime_error &error ) {
24         cout << "Exception occurred: " << error.what() << endl;
25         return EXIT_FAILURE;
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.

Example:

```
1 #include <CommandLine.h>
2
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;
15     cout << commandLine.getParametersNumber() << endl;
16     return EXIT_SUCCESS;
17 }
18 catch ( runtime_error &error ) {
19     cout << "Exception occurred: " << error.what() << endl;
20     return EXIT_FAILURE;
21 }
22 }
23

```

Output:

```

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>
2
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;
15         cout << commandLine.getAllParameters() << endl;
16         return EXIT_SUCCESS;
17     }
18     catch ( runtime_error &error ) {
19         cout << "Exception occurred: " << error.what() << endl;
20         return EXIT_FAILURE;
21     }
22 }
23

```

Output:

```

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

```

string getParameter(int parameterPosition) ↗

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

Example:

```

1  #include <CommandLine.h>
2
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;
15         cout << commandLine.getParameter( 2 ) << endl;
16         return EXIT_SUCCESS;
17     }
18     catch ( runtime_error &error ) {
19         cout << "Exception occurred: " << error.what() << endl;
20         return EXIT_FAILURE;
21     }
22 }
23

```

Output:

```

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

```

int getParameterAsInteger(**int** parameterPosition) ↗

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

Example:

```

1 #include <CommandLine.h>
2
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;
15         cout << commandLine.getParameterAsInteger( 2 ) << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22

```

Output:

```

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>
2
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;
15         cout << commandLine.getParameterAsFloat( 2 ) << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22
```

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.

Example:

```
1 #include <CommandLine.h>
2
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;
15         commandLine.gotoNextParameter();
16         cout << commandLine.getCurrentParameter() << endl;
17         commandLine.gotoFirstParameter();
18         cout << commandLine.getCurrentParameter() << endl;
19         return EXIT_SUCCESS;
20     }
21     catch ( runtime_error &error ) {
22         cout << "Exception occurred: " << error.what() << endl;
23     }
24 }
```

```

23     return EXIT_FAILURE;
24 }
25 }
26

```

Output:

```

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>
2
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;
15         do
16             cout << commandLine.getCurrentParameter() << endl;
17         while( commandLine.gotoNextParameter() );
18         return EXIT_SUCCESS;
19     }
20     catch ( runtime_error &error ) {
21         cout << "Exception occurred: " << error.what() << endl;
22         return EXIT_FAILURE;
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.

Example:

```

1  #include <CommandLine.h>
2

```

```

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;
15         do
16             cout << commandLine.getCurrentPosition() << endl;
17         while( commandLine.gotoNextParameter() );
18         return EXIT_SUCCESS;
19     }
20     catch ( runtime_error &error ) {
21         cout << "Exception occurred: " << error.what() << endl;
22         return EXIT_FAILURE;
23     }
24 }
25

```

Output:

```

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.

Example:

```

1 #include <CommandLine.h>
2
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;
15         do
16             cout << commandLine.getCurrentParameter() << endl;
17         while( commandLine.gotoNextParameter() );
18         return EXIT_SUCCESS;
19     }
20     catch ( runtime_error &error ) {
21         cout << "Exception occurred: " << error.what() << endl;
22         return EXIT_FAILURE;
23     }
24 }
25

```

```
24 }
25
```

Output:

```
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>
2
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;
15         cout << commandLine.getCurrentParameterAsInteger() << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22
```

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.

Example:

```
1 #include <CommandLine.h>
2
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;
15         cout << commandLine.getCurrentParameterAsFloat() << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22

```

Output:

```

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>
2
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;
15         cout << commandLine.getFirstParameter() << endl;
16         return EXIT_SUCCESS;
17     }
18     catch ( runtime_error &error ) {
19         cout << "Exception occurred: " << error.what() << endl;
20         return EXIT_FAILURE;
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.

Example:

```

1 #include <CommandLine.h>
2

```

```

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;
15         cout << commandLine.getFirstParameterAsInteger() << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22

```

Output:

```

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>
2
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;
15         cout << commandLine.getFirstParameterAsFloat() << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22

```

Output:

```

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

```

string getLastParameter() ↗

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

Example:

```
1 #include <CommandLine.h>
2
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;
15         cout << commandLine.getLastParameter() << endl;
16         return EXIT_SUCCESS;
17     }
18     catch ( runtime_error &error ) {
19         cout << "Exception occurred: " << error.what() << endl;
20         return EXIT_FAILURE;
21     }
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.

Example:

```
1 #include <CommandLine.h>
2
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;
15         cout << commandLine.getLastParameterAsInteger() << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22
```

Output:

```
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>
2
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;
15         cout << commandLine.getLastParameterAsFloat() << endl;
16         return EXIT_SUCCESS;
17     } catch ( runtime_error &error ) {
18         cout << "Exception occurred: " << error.what() << endl;
19         return EXIT_FAILURE;
20     }
21 }
22
```

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.

Example:

```
1 #include <CommandLine.h>
2
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;
15         commandLine.setOptionPrefix( "--" );
16         cout << commandLine.getOptionValue( "price" ) << endl;
17     }
18 }
```



```

17     return EXIT_SUCCESS;
18 }
19 catch ( runtime_error &error ) {
20     cout << "Exception occurred: " << error.what() << endl;
21     return EXIT_FAILURE;
22 }
23 }
24


```

Output:

```

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>
2
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;
15         commandLine.setOptionPostfix( "=" );
16         cout << commandLine.getOptionValue( "price" ) << endl;
17         return EXIT_SUCCESS;
18     }
19     catch ( runtime_error &error ) {
20         cout << "Exception occurred: " << error.what() << endl;
21         return EXIT_FAILURE;
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.

Example:

```

1  #include <CommandLine.h>
2
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;
15         commandLine.setOptionPrefix( "--" );
16         cout << commandLine.getOptionPrefix() << endl;
17         return EXIT_SUCCESS;
18     }
19     catch ( runtime_error &error ) {
20         cout << "Exception occurred: " << error.what() << endl;
21         return EXIT_FAILURE;
22     }
23 }
24

```

Output:

```

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>
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;
15         commandLine.setOptionPostfix( "=" );
16         cout << commandLine.getOptionPostfix() << endl;
17         return EXIT_SUCCESS;
18     }
19     catch ( runtime_error &error ) {
20         cout << "Exception occurred: " << error.what() << endl;
21         return EXIT_FAILURE;
22     }
23 }
24

```

Output:

```

linux:/home/enzo # ./myapp
=

```

`bool hasOption(string option)` ↗

Description: Checks if the specified option exists.

Example:

```
1 #include <CommandLine.h>
2
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;
15         commandLine.setOptionPostfix( ":" );
16         if( commandLine.hasOption( "ISBN-10" ) ) {
17             cout << "yes" << endl;
18         }
19         else {
20             cout << "no" << endl;
21         }
22         return EXIT_SUCCESS;
23     }
24     catch ( runtime_error &error ) {
25         cout << "Exception occurred: " << error.what() << endl;
26         return EXIT_FAILURE;
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)` ↗

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

Example:

```
1 #include <CommandLine.h>
2
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;
15         commandLine.setOptionPostfix( ":" );
16         cout << commandLine.getOptionValue( "Language" ) << endl;
17         return EXIT_SUCCESS;
18     }
19     catch ( runtime_error &error ) {
20         cout << "Exception occurred: " << error.what() << endl;
21         return EXIT_FAILURE;
22     }
23 }
24

```

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
English

```

int getOptionValueAsInteger(string option) ↗

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

Example:

```

1  #include <CommandLine.h>
2
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;
15         commandLine.setOptionPostfix( ":" );
16         cout << commandLine.getOptionValueAsInteger( "Weight" );
17         cout << endl;
18         return EXIT_SUCCESS;
19     } catch ( runtime_error &error ) {
20         cout << "Exception occurred: " << error.what() << endl;
21         return EXIT_FAILURE;
22     }
23 }
24

```

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
14

```

`float` `getOptionValueAsFloat(string option)` ↗

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

Example:

```
1 #include <CommandLine.h>
2
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;
15         commandLine.setOptionPostfix( "=" );
16         cout << commandLine.getValueAsFloat( "price" ) + 0.01;
17         cout << endl;
18         return EXIT_SUCCESS;
19     } catch ( runtime_error &error ) {
20         cout << "Exception occurred: " << error.what() << endl;
21         return EXIT_FAILURE;
22     }
23 }
24
```

Output:

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

`string` `getOptionLongValue(string option)` ↗

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.

Example:

```
1 #include <CommandLine.h>
2
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;
15         commandLine.setOptionPrefix( "--" );
16         cout << commandLine.getLongValue( "peripheral" );
17         cout << endl;
18         return EXIT_SUCCESS;
19     }
20 }
```

```

19     }
20     catch ( runtime_error &error ) {
21         cout << "Exception occurred: " << error.what() << endl;
22         return EXIT_FAILURE;
23     }
24 }
25

```

Output:

```

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>
2
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;
15         commandLine.optionCaseSensitive();
16         commandLine.setOptionPrefix( "--" );
17         cout << commandLine.getOptionLongValue( "PERIPHERAL" );
18         cout << endl;
19         return EXIT_SUCCESS;
20     }
21     catch ( runtime_error &error ) {
22         cout << "Exception occurred: " << error.what() << endl;
23         return EXIT_FAILURE;
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.

Example:

```

1  #include <CommandLine.h>

```

```

2
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;
15         commandLine.optionCaseInsensitive();
16         commandLine.setOptionPrefix( "--" );
17         cout << commandLine.getOptionLongValue( "PERIPHERAL" );
18         cout << endl;
19         return EXIT_SUCCESS;
20     }
21     catch ( runtime_error &error ) {
22         cout << "Exception occurred: " << error.what() << endl;
23         return EXIT_FAILURE;
24     }
25 }
26

```

Output:

```

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

```

bool isOptionCaseSensitive() ↗

Description: Checks if the options are case sensitive.

Example:

```

1 #include <CommandLine.h>
2
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;
15         if( commandLine.isOptionCaseSensitive() ) {
16             cout << "yes" << endl;
17         }
18         else {
19             cout << "no" << endl;
20         }
21         return EXIT_SUCCESS;
22     }
23     catch ( runtime_error &error ) {
24         cout << "Exception occurred: " << error.what() << endl;
25         return EXIT_FAILURE;
26     }
27 }

```

```
26 }  
27  
28
```

Output:

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