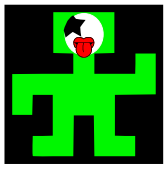
Class CommandLine



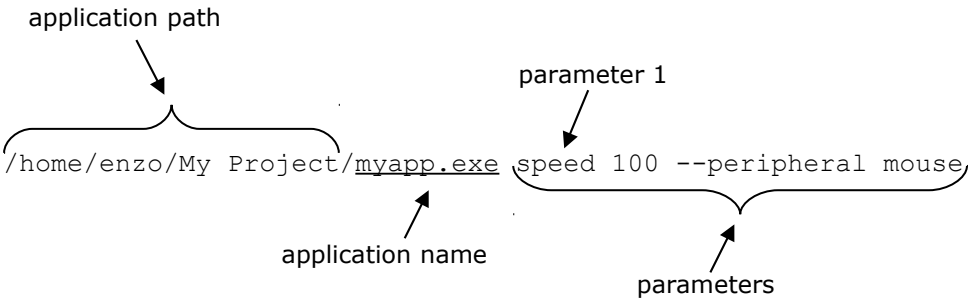
Propose: C++ class to handle data from the command line.

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

The standard structure of a command line:



Members:

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

int getParametersNumber()
string getParameter( int parameterPosition )

setOptionPrefix( string optionPrefix )
setOptionPostfix( string optionPostfix )

bool hasOption( string option )

string getOptionValue( string option )
string getOptionLongValue( string option )

optionCaseSensitive()
optionCaseInsensitive()
```

`string getApplicationName()`
Description: Retrieves the name of the application for the current process.

Example:

```
#include <CommandLine.h>

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

int main( int argc, char *argv[] ) {

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

    try {
        environs::CommandLine commandLine;
        cout << commandLine.getApplicationName() << endl;
        return EXIT_SUCCESS;

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

```

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:

```
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        cout << commandLine.getApplicationPath() << endl;
        return EXIT_SUCCESS;

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

```
}

```

Output:

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

`string getCurrentWorkingDirectory()` ⬆
Description: Retrieves the current working directory for the current process.

Example:

```
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        cout << commandLine.getApplicationPath() << endl;
        cout << commandLine.getCurrentWorkingDirectory() << endl;
        return EXIT_SUCCESS;

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

```

Output:

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

`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:

```
#include <CommandLine.h>

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

int main() {

    using std::cout;
```

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

    try {
        environs::CommandLine commandLine;
        cout << commandLine.getParametersNumber() << endl;
        return EXIT_SUCCESS;

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

```

Output:

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

`string getParameter( int parameterPosition )` ⬆
Description: Retrieves the parameter of the specified parameter position on the command line for the current process.

Example:

```
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        cout << commandLine.getParameter( 2 ) << endl;
        return EXIT_SUCCESS;

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

```

Output:

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

`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:

```cpp
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        commandLine.setOptionPrefix( "--" );
        cout << commandLine.getOptionValue( "price" ) << endl;
        return EXIT_SUCCESS;

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

```

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:

```cpp
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        commandLine.setOptionPostfix( "=" );
        cout << commandLine.getOptionValue( "price" ) << endl;
        return EXIT_SUCCESS;

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

```
        return EXIT_FAILURE;
    }
}

```

Output:

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

---

bool hasOption(string option) ⬆
Description: Checks if the specified option exists.

Example:

```
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        commandLine.setOptionPostfix( ":" );
        if( commandLine.hasOption( "ISBN-10" ) ) {
            cout << "yes" << endl;
        } else {
            cout << "no" << endl;
        }
        return EXIT_SUCCESS;

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

```

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:

```
#include <CommandLine.h>

#include <iostream>
```

```cpp
#include <cstdlib>
#include <stdexcept>

int main() {

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

    try {
        environs::CommandLine commandLine;
        commandLine.setOptionPostfix( ":" );
        cout << commandLine.getOptionValue( "Language" ) << endl;
        return EXIT_SUCCESS;

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

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
```

`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:

```cpp
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        commandLine.setOptionPrefix( "--" );
        cout << commandLine.getOptionLongValue( "peripheral" );
        cout << endl;
        return EXIT_SUCCESS;

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

```
26
```

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:

```
#include <CommandLine.h>

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

int main() {

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

    try {
        environs::CommandLine commandLine;
        commandLine.optionCaseSensitive();
        commandLine.setOptionPrefix( "--" );
        cout << commandLine.getOptionLongValue( "PERIPHERAL" );
        cout << endl;
        return EXIT_SUCCESS;

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

```

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:

```
#include <CommandLine.h>

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

```

```cpp
int main() {

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

    try {
        environs::CommandLine commandLine;
        commandLine.optionCaseInsensitive();
        commandLine.setOptionPrefix( "--" );
        cout << commandLine.getOptionLongValue( "PERIPHERAL" );
        cout << endl;
        return EXIT_SUCCESS;

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

```

Output:

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