SvnUpdate

Introduction

This program was made to update all `SVN` repositories contained in a directory and its sub-directories. It's implemented in c++ and use cmake for the build-system. A command-line option: `--skip` can be used to set a list of `SVN` subdirectories to skip (relative path, separated by `;`).

The process can be described as:

- list all `SVN` subdirectories
- launch `N` threads to start the update process (where `N` = number of threads in the machine)
- update all the `SVN` repositories and display progress using a progress-bar (using `indicators` c++ header-only library)
- display the list of updated `SVN` repositories (using `tabulate` c++ header-only library)
- log in a file if the `--log` command-line option has been given

Usage

```
# update all subdirectories and log output
svn-update.exe --path "c:\svn-repos" \
               --skip "projectA;projectB" \
               --log "output.log"
```

Build program

Introduction

The build system has moved from the original Visual-Studio solution/project to cmake (faster compile time, easier to maintain, full control of build). A python script: `build.py` has been implemented in order to **download/compile/install** automatically all the third-party libraries.

Build Requirements

In order to build this program, several open-source programs needs to be installed:

- install **visual studio** with Windows SDK and English language pack
  - https://visualstudio.microsoft.com/fr/thank-you-downloading-visual-studio/?sku=Community&rel=16

    - Visual Studio Installer => Modifier => Modules linguistiques => Anglais
- install **git** for windows
    - https://git-scm.com/download/win
- install **cmake** for windows
    - https://cmake.org/download/
- install **python** for windows
    - https://www.python.org/ftp/python/3.9.6/python-3.9.6-amd64.exe
- install **vcpkg** for windows
    - https://vcpkg.io/en/getting-started.html

`vcpkg` is used to compile the third-party librairies.
All `vcpkg` libraries that are required for the program are defined in the standard json file: `vcpkg.json`.

There are two ways of building the program/libraries:

- using **python** `build.py` script which will:
    - install `vcpkg`
    - compile third-party libraries
    - compile program
    - install everything into `x64-windows` directory
- using **visual studio**:
    - compile third-party libraries
    - compile program
    - allow **debug**

> Important: For **visual studio**, an environment variable: `VCPKG_ROOT` with the path of the `vcpkg` installation direction needs to be defined.

Build with python script

Several `python` libraries needs to be installed in order to execute the `build.py` script:

```
# update python pip
python.exe -m pip install --upgrade pip

# install build_tools package
cd build_tools
python.exe -m pip install .
```

Use the `build.py` python script:

```
python.exe build.py --vcpkg-dir="path_to_vcpkg" --build
```

Build with Visual Studio

The following steps needs to be executed in order to build/debug the program:

```
File => Open => CMake...
  Choose CMakelists.txt
Solution Explorer => Switch between solutions and available views => CMake Targets View
Select x64-release or x64-debug
Select the src\program.exe (not bin\program.exe)
```

To add command-line input arguments for debugging the program:

```
Solution Explorer => Project => (executable) => Debug and Launch Settings => src\program.ex
```

```
  "args": [
    "--path \"xxx\"",
    "--skip \"yyy;zzz\"",
    "--log output.log",
  ]
```