Build System & Scripts

Quick Notes

For some of the commands the tools that are required need to be installed on your dev/build box first. The mono build chain has not been finished yet.

The reason we're using msbuild scripts over using something like powershell or ablacore is that we can leverage msbuild & xbuild without having to write everything twice.

build.cmd arguments

here some sample usage of the build.cmd file

$ build [targets=simple] [buildarea=all] [configuration=release]
$ build
$ build commit all release
$ build simple core
$ build commit all debug

Build Area

the configuration property is called Area. /p:Area=all

- all - Everything that is to be released.
- analyzers - The Contrib.Analyzers project.
- contrib - The Contrib projects, minus the Lucene.Net project.
- contrib-core - The Contrib.Core project
- core - The core Lucene.Net project.
- fastvectorhighlighter - The Contrib.FastVectorHighlighter project.
- highlighter - The Contrib.Highlighter project.
- queries - The Contrib.Queries project.
- regex - The Contrib.Regex project.
- similarity - The Contrib.Similarity project.
- simplefacetedsearch - The Contrib.SimpleFacetedSearch project.
- snowball - The Contrib.Snowball project.
- spatial - The Contrib.Spatial project.

Configuration

- release - configures to build the projects in "Release" mode
- debug - cogfigures to build the projects in "Debug" mode

Common Build Targets On Windows

- **clean** - cleans specified files & directories.
- **build** - builds the specified projects.
- **test** - uses gallio & nunit to run the tests.
- **test-report-html** - uses gallio & nunit to build html test reports.
- **test-report-xml** - uses gallio & nunit to build xml test reports.
- **coverage** - will use ncover3, if installed, to build code coverage reports.
- **document** - will build documentation, if sandcastle is installed.
- **rules** - will use analyze rules against code, if fxcop is installed.
- **simple** - calls the targets: build;copy-release
- **commit** - calls the targets: clean;build;test-report-html;rules;coverage;document;copy-release
- **nightly** - same as commit right now

Build Properties of Note

- Configuration - Release or Debug
- Area - see the "Build Area" section above.
- NETFRAMEWORK - .net or mono, defaults to .net

Build Flow

The **build.cmd** sets values and invokes **build.targets**. build.targets acts almost like a controller, it takes the arguments and imports files & variables based on the incoming property values.

Depending on the specified **Area**, which defaults to 'all', the correct **project.targets** & **documentation.targets** are loaded into the build.targets. These files are under the subfolders in /build/scripts. i.e. build/scripts/Analyzers.

If the build property **NETFRAMEWORK** is defaulted, the **dot-net-tools.targets** are loaded. If mono is specified, the **mono-tools.targets** will be loaded.

The target will only be invoked on the assemblies that are listed by the project.targets & documentation.targets files. So if you run the command **build document analyzers**, the build scripts will only create documentation for the Lucene.Net.Contrib.Analyzers.dll assembly and put it into the artifacts folder for that area, i.e. /build/artifacts/Analyzers/docs.

Adding a new project

If you are adding a new project, then most likely a contrib project. If your new project is Contrib.AwesomeFacetedSearch, your simple name will be "AwesomeFacetedSearch".

Make sure that the assembly is signed using the Lucene.Net.snk in ~/lib/Lucene.Net.snk. If the Project is on a version that is less than Lucene.Net 3.0, then suppress warning 618.

The output path needs to be set to the right bin folder. In release mode, it should be "..\..\..\build\bin\contrib\AwesomeFacetedSearch\Release\" with xml comments enabled. In debug mode, it should be "..\..\..\build\bin\contrib\AwesomeFacetedSearch\Debug\"

Create a new subfolder in "build/scripts" with the simple name of your project. For example,

Copy the document.targets & project.targets from the build/scripts/Queries folder. Then do a search and replace on "Queries" with a case match. In the project.targets file, replace the lowercased "queries" with "awesomefacetedsearch".

Then add the following line to build.targets

build.targets import

```
<Import Project="AwesomeFacetedSearch/project.targets" Condition="'$(Area)' == 'awesomefacetedsearch'" />
```

And add this line to /build/scripts/All/project.targets & build/scripts/Contrib/project.targets

all & contrib project.targets

```
<Import Project="../AwesomeFacetedSearch/project.targets"  />
```

Add this line to the docs.shfbproj file

docs.shfbproj

```
  <Import Project="AwesomeFacetedSearch/document.targets" Condition="'$(area)' == 'awesomefacetedsearch'" />
```

You should now be able to run build commands against 'awesomefacetedsearch' or have it build with contrib or all.

try running. build document awesomefacetedsearch

Common Build Targets On Mono

- TODO: get working