

(新) JTSDK2.0 インストールおよびコンパイル手順書 @4

JTSDKv2.0をインストールする方法とwsjtx 1.9.1 のソースコードをコンパイルしてアプリケーションソフトをビルドする方法を説明します。 海外のOM, ki7mtによりWebで公開されたものに合わせ日本語で追加説明しています。 インストールのエラーまたはコンパイル実行中のエラーに関しては当方では対応できませんので、あらかじめご承知頂き各自の責任に於いてご対応の程お願いいたします。 当方Windows10 1709 32bitOS, 64bitOSの環境にてインストール、ビルドを確認しております。 以下、インストール、アップデートの説明はWindow10をもとに書いてあります。

JTSDKのインストールに関しては日本語環境でも可能ですがソースコードをコンパイルしてアプリをビルドする時は必ず**英語環境が必要**となります。 WindowsOSへの言語の追加は **en_USの選択**をします。 コンパイル時にエラーが起きるがどうしてなのかとの問い合わせの多くはOSが英語(米語)環境になっていない事です。 JTSDKをインストール開始する前に先に英語(米語)環境の構築をお勧めします。 方法については広くインターネットのウェブサイトにて公開されていますのでここでは割愛します。

前置きが長くなりましたがJTSDKを実行するには幾つかの手順を踏む必要があります。 要点を纏めますと次の様になります。

1. JTSDKインストールに必要なプログラムをダウンロードします。
2. インストールしたプログラムをアップデートします。
3. 2項が完了した時点でコンパイル環境をQT5.2からQT5.5へとアップグレードします。
4. Hamlib3をビルドします。
5. wsjtx-1.9.1, jtdx 18.1.xx をビルドします。

• DOWNLOADS

<https://sourceforge.net/projects/jtsdk/files/win32/2.0.0/>

から以下の 8 個のファイルをダウンロードします。

sourceforge のページでは各ファイル名にリンクが貼られていますのでクリックするだけで OK です。

Click the following links to download.

- [MS-VCredist \(2010\)](#)
- [MS-VCredist \(2013\)](#)
- [OmniRig](#)
- [JTSDK Main Installer](#)
- [JTSDK Update-1](#)
- [JTSDK Update-2](#)
- [JTSDK Update-3](#)
- [JTSDK Update-4](#)

OmniRig 古いバージョンの物がすでにインストールされていた場合、リンク先の物 (zip file: 7/25/2015 9:12am, CRC-32 FBFE7073) に入れ替えましょう。Afreet Software, Inc のダウンロードサイドにある OmniRig と同じ物の様です。

• INSTALLATION

ダウンロード出来ましたら下記のパッケージファイルを記載されている順番に正しくインストールします。 15分以内でインストール出来るようなことが書かれていましたが 32bitOS と 64bitOS ではかなりの違いを経験しました。私の 32bitOS core i5 では何だかんだで 60分以上を要してます。

インストール中に出た指示に従いインストールを続けて全て完了させてください。尚、JTSDK がインストールされる場所は固定されていて (C:\¥JTSDK) 変更することは出来ません。

- Install MS-VCredist (2010), follow the prompts
- Install MS-VCredist (2013), follow the prompts
- Unzip, then run the OmniRig Installer, follow the prompts
- JTSDK Main Installer, follow the prompts
- JTSDK Update-1, follow the prompts
- JTSDK Update-2, follow the prompts, provides QT 5.5 and GCC 4.9
- JTSDK Update-3, follow the prompts, adds Ruby and AsciiDoctor
- JTSDK Update-4, Move Build scripts to a stable branch (Important Upgrade!)

• 開発環境 2.0.3 を 2.0.6 へと UPDATE, UPGRADE

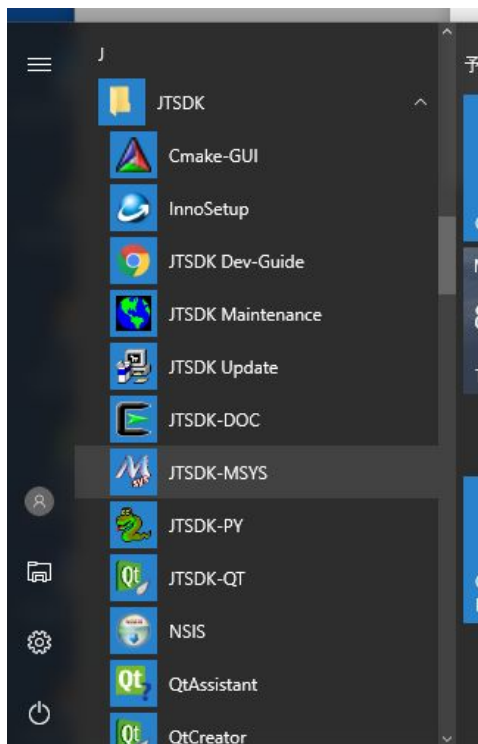
JTSDK のインストールが完了したことを前提に進めます。

デスクトップ上のアイコン **JTSDK-MSYS** をマウスで起動させます。

仮にアイコンが作成されていない場合は：

スタートアップから起動 (画像とパスは Windows10 のものです)

C:\¥ProgramData¥Microsoft¥Windows¥Start Menu¥Programs¥JTSDK



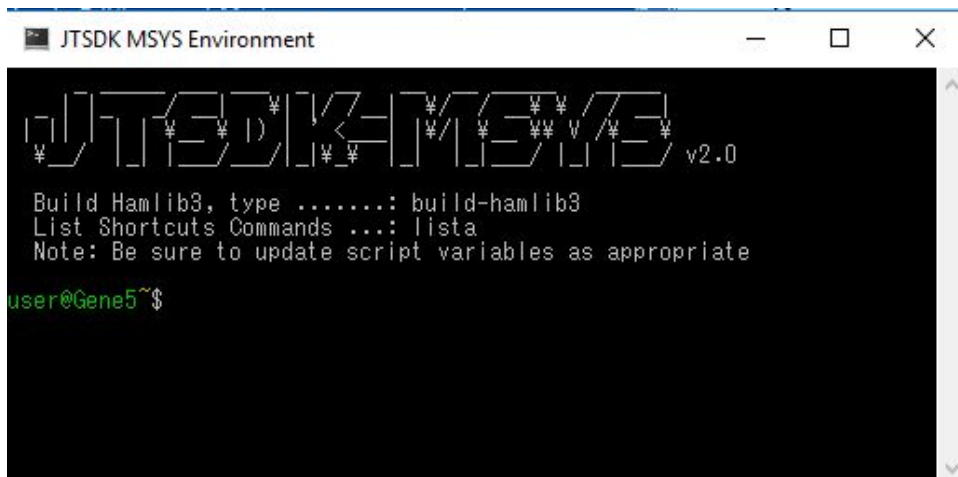
JTSDK-MYSY を起動後の画像

A screenshot of a terminal window titled "JTSDK MSYS Environment". The terminal displays the output of the JTSDK-MSYS setup process. The text is as follows:

```
JTSDK-MSYS - Setting Up User Files
-----
..installing: JTSDK bashrc to /home/user/.bashrc
..installing: JTSDK bash_aliases to /home/user/.bash_aliases
..installing: JTSDK bash_bashrc to /home/user/.bash_profile
..installing: JTSDK inputrc to /home/user/.inputrc
..installing: JTSDK minttyrc to /home/user/.minttyrc

JTSDK-MSYS - Setup Complete: -->> RESTART REQUIRED <<--
At the prompt, type: exit, then re-launch JTSDK-MSYS
user@Gene5~$ |
```

プロンプトが表示されたら `exit` を入力して一旦終了します。
再度 JTSDK-MSYS を起動させます。



```
JTSDK MSYS Environment

JTSDK Maintenance v2.0

Build Hamlib3, type .....: build-hamlib3
List Shortcuts Commands ...: lista
Note: Be sure to update script variables as appropriate

user@Gene5~$
```

exit を入力して JTSDK-MSYS を終了させます。

次に **JTSDK Maintenance** を起動します。

アップデートで使用するスクリプト **JTSDK Maintenance** の起動方法が Windows のバージョンにより異なります。

Vista / Win7

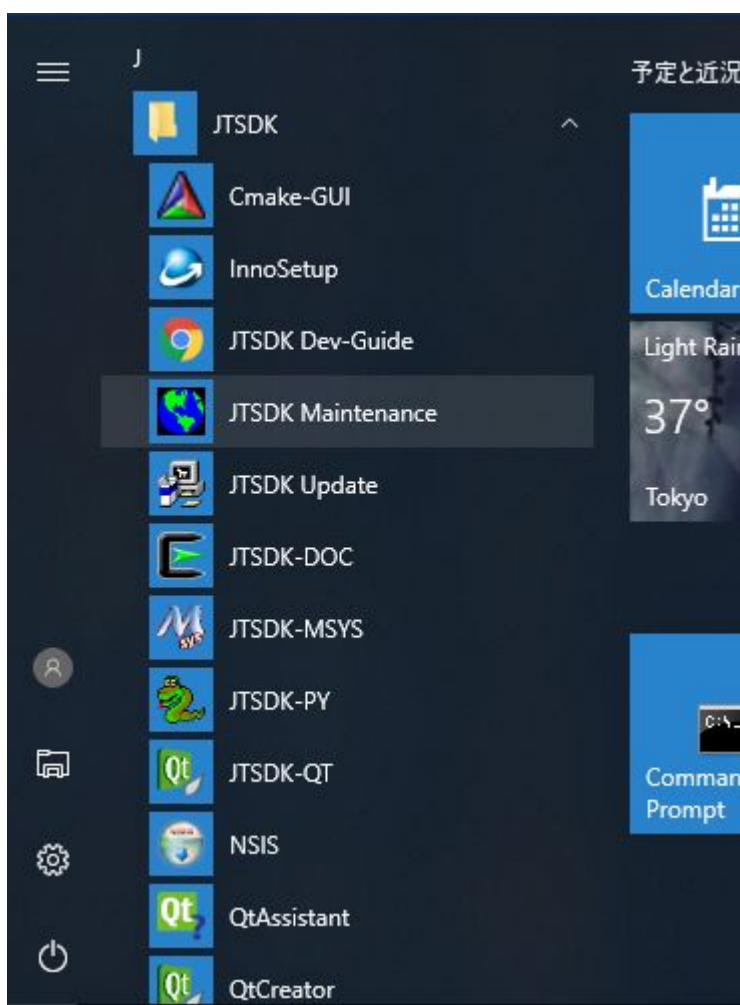
- Start >> Programs >> JTSDK >> Tools >> **JTSDK Maintenance** (JTSDK Maintenance を起動)
- Then type: update (update を入力そして Enter)
- Then type: upgrade (upgrade を入力そして Enter) 少々時間を要する
- Close JTSDK-Maint (プロンプトが出たら JTSDK-Maintenance を exit を入力して終了)
- Re-Open JTSDK Maint, then type: version (再度 JTSDK-Maintenance を起動、version を入力そして Enter)
- Ensure you are on v2.0.6 (v2.0.6 の表示を確認)

Win8 / Win10

- Launchers should be listed under a location similar to:
- All Apps >> J >> JTSDK >> **JTSDK Maintenance** (JTSDK Maintenance を起動する)
- Then type: update (update を入力そして Enter)
- Then type: upgrade (upgrade を入力そして Enter) 少々時間を要する

- Close JTSDK-Maint (プロンプトが出たら JTSDK-Maintenance を exit を入力して終了)
- Re-Open JTSDK Maint, then type: version (再度 JTSDK-Maintenance を起動、version を入力そして Enter)
- Ensure you are on v2.0.6 (v2.0.6 の表示を確認)

スタートアップから JTSDK-Maintenance を起動します。Windows10 の場合



```
JTSDK General Maintenance Environment - 2.0.3-
-----
* Provides Access To: Subversion, Gnu Tools and AsciiDoctor
* Upgrades JTSDK Main Scripts and Packages when needed

TO UPDATE and UPGRADE
Type .....: update
Then Type ..: upgrade

TO DISPLAY CURRENT VERSION INFORMATION
Type .....: version

GENERAL: MAINTENANCE
With this env, you have access to all the Gnu
Tools, Subversion and AsciiDoctor. It can be used
to perform most any task needed by the SDK.
There are no Tool-Chains or Frameworks ( Qt / Python )
in #PATH#

C:;%JTSDK>update_
```

update を入力

```
JTSDK General Maintenance Environment - 2.0.3-
-----
* Provides Access To: Subversion, Gnu Tools and AsciiDoctor
* Upgrades JTSDK Main Scripts and Packages when needed

TO UPDATE and UPGRADE
Type .....: update
Then Type ..: upgrade

TO DISPLAY CURRENT VERSION INFORMATION
Type .....: version

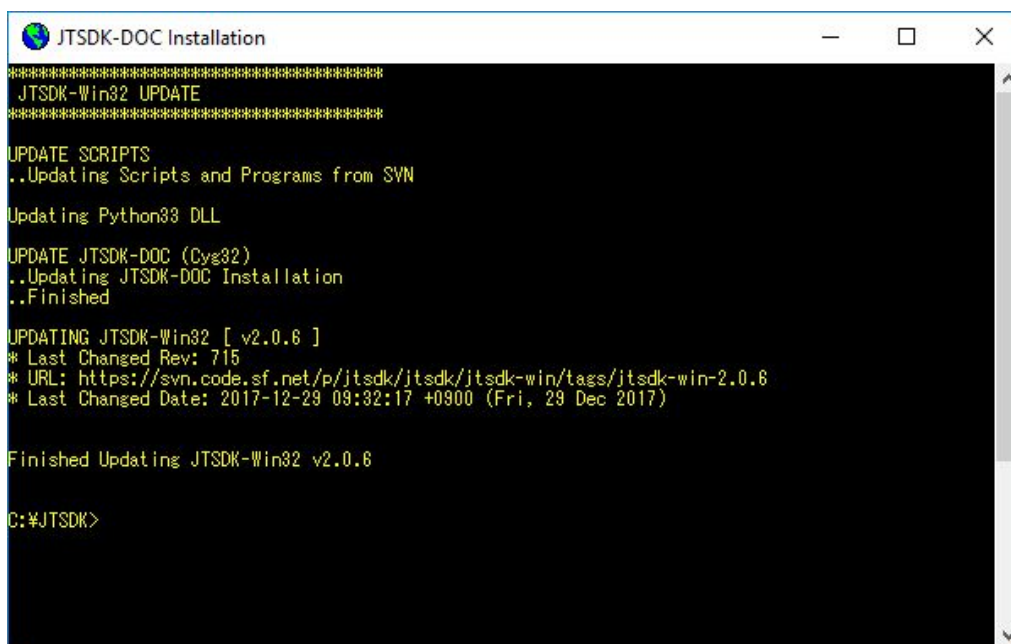
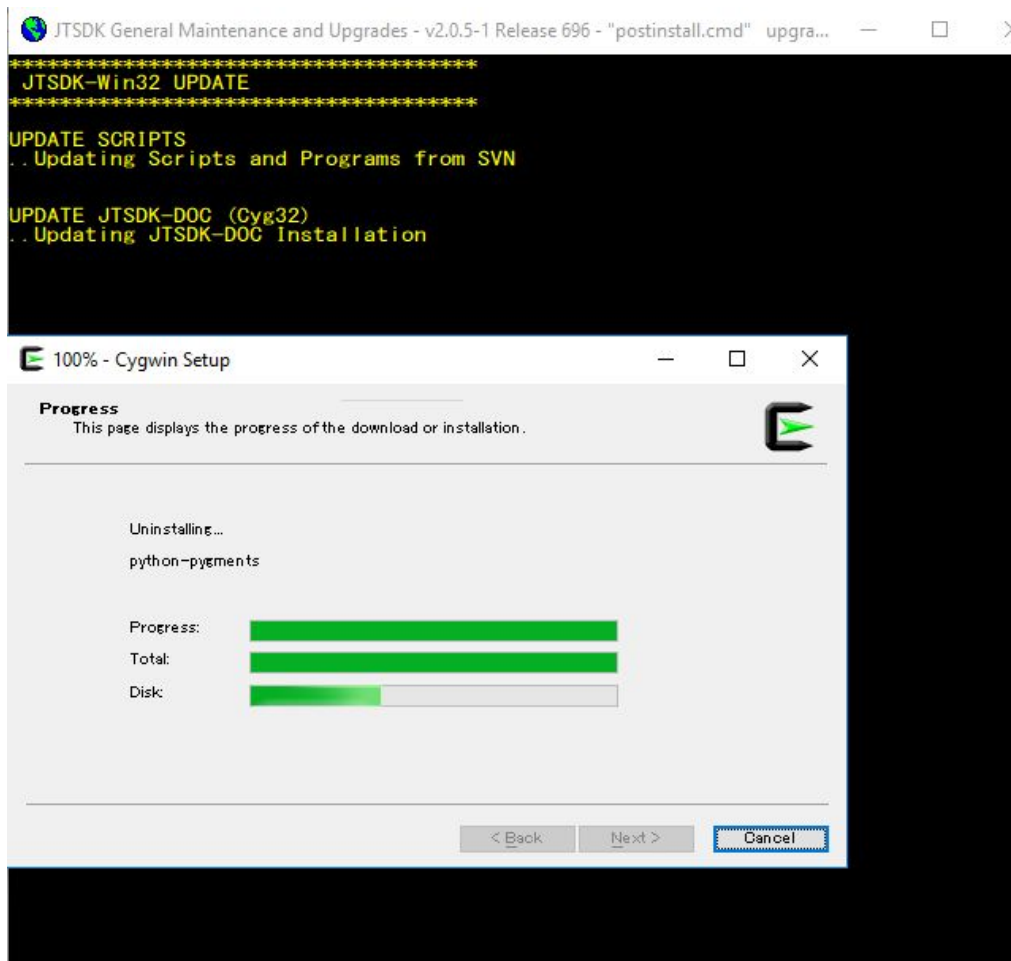
GENERAL: MAINTENANCE
With this env, you have access to all the Gnu
Tools, Subversion and AsciiDoctor. It can be used
to perform most any task needed by the SDK.
There are no Tool-Chains or Frameworks ( Qt / Python )
in #PATH#

C:;%JTSDK>update
A postinstall.cmd
Export complete.

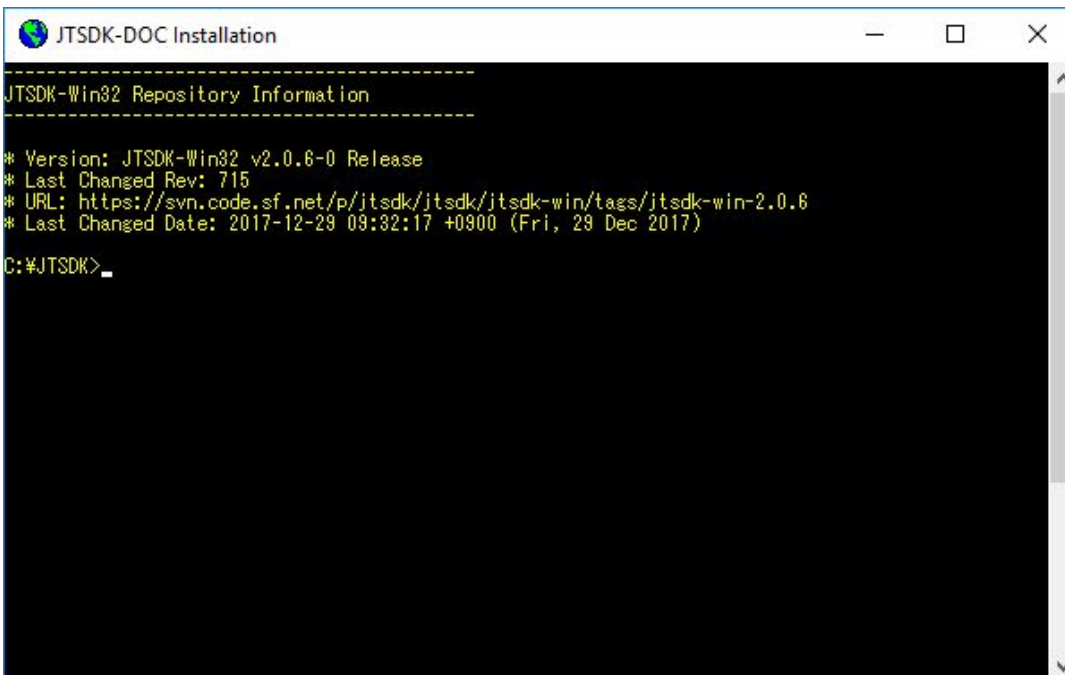
C:;%JTSDK>upgrade_
```

upgrade を入力

upgrade を入力してアップグレードが実行中の画面です。



下は version を入力後に表示される画面です。



```
JTSDK-DOC Installation
-----
JTSDK-Win32 Repository Information
-----
* Version: JTSDK-Win32 v2.0.6-0 Release
* Last Changed Rev: 715
* URL: https://svn.code.sf.net/p/jtsdk/jtsdk/jtsdk-win/tags/jtsdk-win-2.0.6
* Last Changed Date: 2017-12-29 09:32:17 +0900 (Fri, 29 Dec 2017)
C:\#JTSDK>
```

アップデート、アップグレードの終了です。Exit で JTSDK Maintenance を終了します。
2018 年 10 月 14 日現在 Last Change Rev は **725** になります。

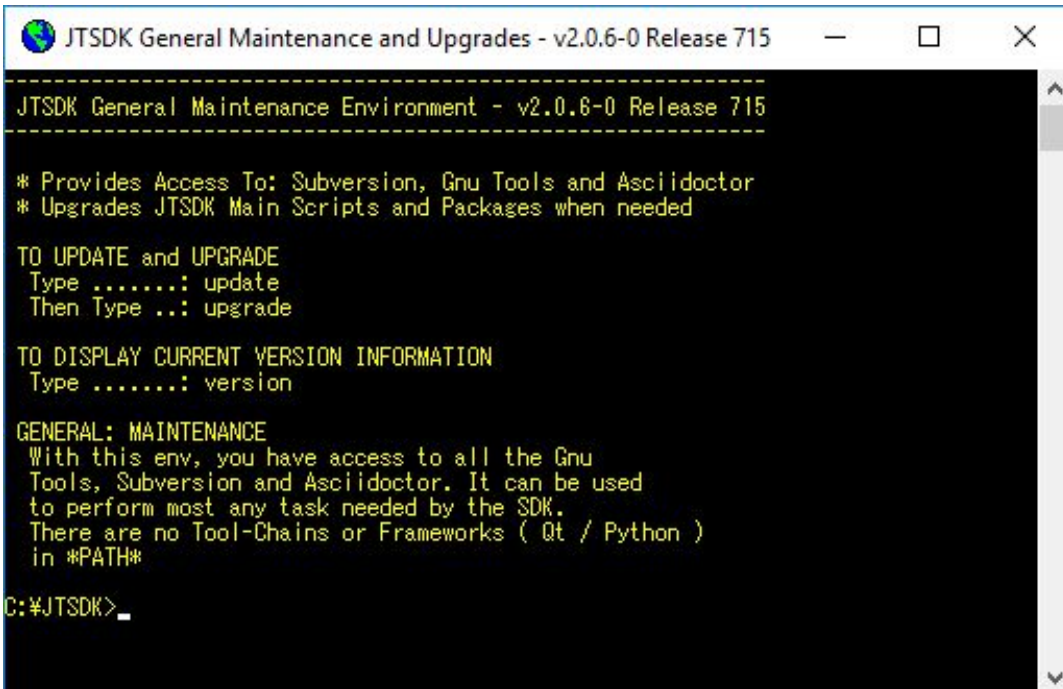
すでに JTSDK がインストール済みであり、あなたの JTSDK の開発環境が v2.0.5-1 Release 696 でしたら以下コマンドを実行して v2.0.6-0 Release 725 へ変更します。

TO UPGRADE:

- * Open JTSDK Maintenance (JTSDK Maintenance の起動)
- * Type: update
- * Type: upgrade
- * Close, then reopen JTSDK Maintenance (JTSDK Maintenance を一旦終了。再度起動)
- * Type: version

注意) 今まで v2.0.5-1 Release 696 の開発環境、QT-55 で Hamlib3 をビルドされていた方も改めて v2.0.6-0 Release 725 の下で Hamlib3 をビルドし直すことをお勧めします。そして定期的にビルドする事もお勧めします。初めての方は次のページ以降を参照の上 Hamlib3 をビルドします。

通常状態で JTSDK-Maintenance を起動した画像



```
JTSDK General Maintenance and Upgrades - v2.0.6-0 Release 715
-----
JTSDK General Maintenance Environment - v2.0.6-0 Release 715
-----
* Provides Access To: Subversion, Gnu Tools and AsciiDoctor
* Upgrades JTSDK Main Scripts and Packages when needed

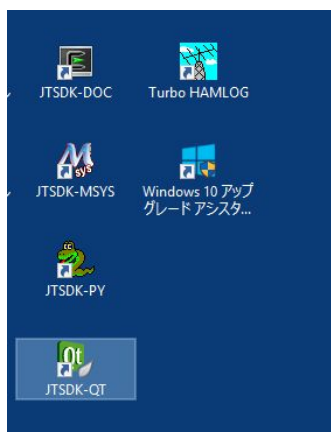
TO UPDATE and UPGRADE
Type .....: update
Then Type ..: upgrade

TO DISPLAY CURRENT VERSION INFORMATION
Type .....: version

GENERAL: MAINTENANCE
With this env, you have access to all the Gnu
Tools, Subversion and AsciiDoctor. It can be used
to perform most any task needed by the SDK.
There are no Tool-Chains or Frameworks ( Qt / Python )
in *PATH*

C:¥JTSDK>_
```

・コンパイル環境を QT5.2 から QT5.5 へと変更



インストール直後の JTSDK-QT の環境は QT5.2 です。これを QT5.5 へと変更します。

JTSDK-QT を起動します。 プロンプト C:¥JTSDK が表示されましたら **enable-qt55** を入力、そして Enter します。 一旦 exit で終了して再度 JTSDK-QT を起動します。プロンプトが(JTSDK-QT 5.5) になっているのを確認します。

```
JTSDK-QT v2.0.6-0 Release 715
BUILD APPLICATIONS: ( WSJT-X WSPR-X MAP65 )
-----
USAGE: build-(app_name) (type)
App Names .....: wsjtx wsprx map65
Release Types .....: rconfig rinstall package
Debug Types .....: dconfig dinstall
HELP SCREENS
-----
JTSDK-QT Help, Type ..: help-qt5env
Checkout Help, Type ..: help-checkout
Build Help, Type .....: help-(app_name)
List Options, Type ....: list-options
COMPILER INFO ( Mingw 49_32 )
-----
C++ .....: 4.9.2
GFortran ...: 4.9.2
GNU Make ...: 4.1
CRITICAL APP INFO
-----
Asciidoctor..: 1.5.3
Cmake .....: 3.0.2
Cpack .....: 3.0.2
QT5 .....: 5.5.0
QMake .....: 3.0
NSIS .....: v3.0b1
InnoSetup ...: 5.5.5a
Pkg-Cfg .....: 0.28
(JTSDK-QT 5.5 ) C:¥JTSDK)_
```

その他、推奨される環境です。

同様に JTSDK-QT 起動時に 以下のコマンドを入力します。

すべて enable に設定とオリジナルでは推奨されていますが、JTDX のビルド若しくはコードの修正をされる方もおられると思い、フル自動でのコード取得、自動でのコンパイル開始、進行状況無表示にはしていません。

- enable-separate
- disable-autosvn
- disable-autorun
- enable-clean
- enable-rcfg
- disable-quiet

enable-qt55 の場合は有効にするために JTSDK-QT の再起動が必要でしたが上記オプションの場合は再起動の必要はありません。

- Hamlib3 のビルド



JTSDK-MSYS を起動します。



プロンプトが表示されたら **build-hamlib3** を入力して、Enter を押します。ビルド終了まで時間を要しますが終了したら JTSDK-MSYS を exit します。 QT5.5 の環境で使用できる Hamlib3 の作成は終了です。

・ ビルドの種類

ビルドにはデバッグモード、リリースモード、パッケージモードが用意されています。

デバッグモード :ソースコードの一部を変更した時に簡易的に WSJT-X の実行ファイルのチェックが出来るモードです。

リリースモード : パッケージ版をインストールせずに JTSDK がインストールされた同じ PC のデスクトップに作成するアイコンから WSJT-X の実行ファイルを起動出来るモードです。その際、ユーザーフォルダー以下に作成されるフォルダー名を任意に変更もできます。

パッケージモード :JTSDK がインストールされてない別の PC へアプリをインストールするためのインストーラー付きのパッケージソフトを作成するためのモードです。

デフォルトスクリプトコマンドはWSJT-X 開発版の最新状態のソースコードを用いて自動的にビルドします。コマンドを実行させた時に出来るホルダーと実行ファイルのパスは list-optionsのseparateオプション (enable-separate, disable-separate) により異なります。 リビジョン番号がホルダー名として作成されるか否かの違いです。

disable-separate (Separate : No) の場合 :

デバッグモード : C:\JTSDK\wsjtx\devel\qt55\1.9.0\Debug\install\bin\wsjtx.cmd

リリースモード : C:\JTSDK\wsjtx\devel\qt55\1.9.0\Release\install\bin\wsjtx.exe

パッケージモード :

C:\JTSDK\wsjtx\devel\qt55\1.9.0\Release\package\wsjtx-1.9.0-rc4-wins32.exe

enable-separate (Separate : Yes) リビジョン番号 8633 の場合 :

デバッグモード : C:\JTSDK\wsjtx\devel\qt55\1.9.0\8633\Debug\install\bin\wsjtx.cmd


リリースモード : C:\JTSDK\wsjtx\devel\qt55\1.9.0\8633\Release\install\bin\wsjtx.exe

パッケージモード :

C:\JTSDK\wsjtx\devel\qt55\1.9.0\8633\Release\package\wsjtx-1.9.0-rc4-wins32.exe

初めてのビルド

WSJT-X のソースコードはネット上のファイルサーバーから自動的にダウンロードされます。PC にソースコードが無い場合はすべてのファイルがダウンロードされますが、それ以外の場合は新しいファイルのみのダウンロードとなります。まずは、デバッグモードにてビルドした実行ファイルが動作するのか確認しましょう。JTSDK-QT を起動後、コマンドラインに **build-wsjtx dinstall** と入力してコンパイラーを動作させます。エラーが無ければビルドサマリーが表示されて終了します。



```
JTSDK QT 5.5 Development Environment - v2.0.6-0 Release 715

JTSDK-QT v2.0.6-0 Release 715

BUILD APPLICATIONS: ( WSJT-X WSPR-X MAP65 )
-----
USAGE: build-(app_name) (type)

App Names .....: wsjtx wsprx map65
Release Types .....: rconfig rinstall package
Debug Types .....: dconfig dinstall

HELP SCREENS
-----
JTSDK-QT Help, Type ..: help-qtenv
Checkout Help, Type ..: help-checkout
Build Help, Type .....: help-(app_name)
List Options, Type ...: list-options

COMPILER INFO ( Mingw 49_32 )
-----
C++ .....: 4.9.2
GFortran ...: 4.9.2
GNU Make ...: 4.1

CRITICAL APP INFO
-----
Asciidoctor...: 1.5.3
Cmake .....: 3.0.2
Cpack .....: 3.0.2
QT5 .....: 5.5.0
QMake .....: 3.0
NSIS .....: v3.0b1
InnoSetup ...: 5.5.5a
Pkg-Cfg .....: 0.28

(JTSDK-QT 5.5 ) C:%JTSDK)build-wsjtx dinstall_
```

```
C:\JTSDK\scripts\qtenv-build-wsjtx.cmd dinstall

-----
SVN Check
-----
Update from SVN Before Building? ( y/n )
Type Response: _
```

上記画像のような問い合わせがありますが、全くソースコードフォルダー C:\JTSDK\src\wsjtx が空の場合には自動的にダウンロードが開始されます。すでにソースコードがダウンロード済みであれば n を入力します。 y であれば差分のみをダウンロードが開始されます。

```
JTSDK QT 5.5 Development Environment - v2.0.6-0 Release 715

-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/bin/fmtave.exe
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/bin/fcal.exe
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/bin/fmeasure.exe
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/bin/rigctl-wsjtx.exe
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/bin/rigctl-wsjtx.exe
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/README
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/COPYING
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/AUTHORS
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/THANKS
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/NEWS
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/INSTALL
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/BUGS
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/wsmtx/JPLEPH
-- Installing: C:/JTSDK/wsmtx/garc/qt55/1.9.0/8631/Debug/install/share/doc/WSJT-X/wsmtx-main-1.9.0-rc4.html
-- Generating Debug Batch File for ( wsjtx-1.9.0-rc4 )

-----
Build Summary
-----

Name .....: wsjtx-1.9.0-rc4 Release Candidate
Version .....: 1.9.0
SVN .....: r8631
Type .....: Debug
Target .....: install
Tool Chain ....: qt55
SRC .....: C:\JTSDK\src\wsjtx-1.9.0-rc4
Build .....: C:\JTSDK\wsmtx\garc\qt55\1.9.0\8631\Debug\build
Install .....: C:\JTSDK\wsmtx\garc\qt55\1.9.0\8631\Debug\install
SVN URL .....: http://svn.code.sf.net/p/wsjt/wsjt/tags/wsjtx-1.9.0-rc4

(JTSDK-QT 5.5 ) C:\JTSDK_
```

デバッグモードでビルドが終了しビルドサマリーが表示された画像です。ファイルパスにある `wsjtx.cmd` を起動してみましょう。動作しているようであれば重欠点の不良は無いと思われます。仮にコンパイルエラーで途中で止まってしまった場合には

```
C:\JTSDK\wsjtx\devel\qt55\1.9.0\Debug\build
```

```
C:\JTSDK\wsjtx\devel\qt55\1.9.0\Debug\install\bin
```

のホルダーを削除して再コンパイルしてみます。 ファイルがビルド出来る場合もあります。 再びエラーが生じる場合はソースコード自体に問題があるのかもしれませんが。

デバッグモード上でアプリの動作が確認できれば、使用する環境に合わせて次のコマンドでビルドします。

リリースモードは **`build-wsjtx rinstall`**

パッケージモードは **`build-wsjtx package`**

以上

ヘルプ

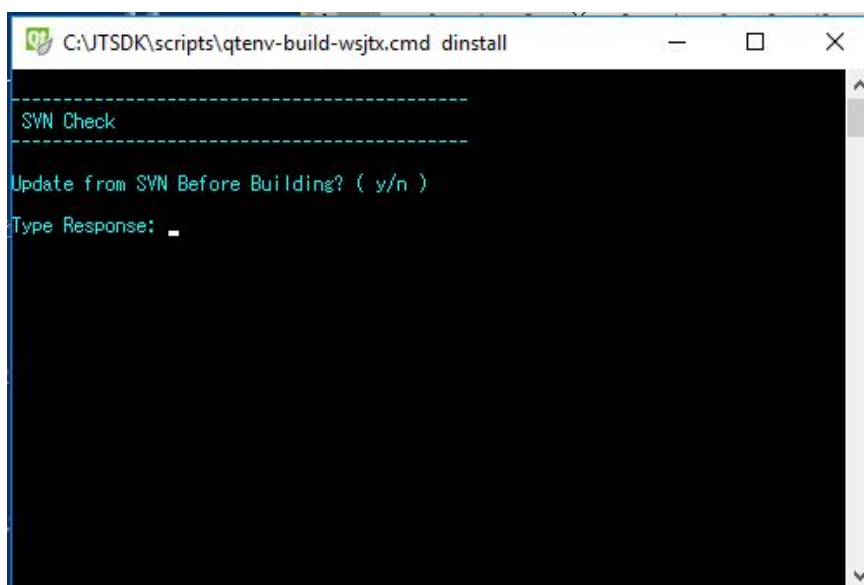
1. JTDX 18.1.xxx のビルドはどうするのですか？

JTSDK-QTのコマンドラインでのコマンド入力はwsjtx 1.9.1の作成と全く同一です。ソースコードは JTDX Website (<https://www.release.jtdx.tech/Windows/Source%20code/>) からZipファイルで11/23/2018現在、提供されています。

Zipファイルを展開して出来たwsjtxフォルダーをtrunkにリネームします。

C:\JTSDK\src\wsjtx へコピーしますがその前にC:\JTSDK\src\wsjtx\trunkがある場合にはtrunkを削除します。ビルドコマンドは **build-wsjtx dinstall** , **build-wsjtx rinstall** , **build-wsjtx package**のいずれかです。

サーバー (Repository) からソースコードのダウンロードの必要はありませんのでビルドコマンドを入力して次に現れる問い合わせには必ず **n** を入力します。yを入力して開始しますとビルド中にエラーになります。



ビルド終了時点で出来上がるホルダーは以下のパスになります。今後バージョンが変わった場合やビルドシステムが変更された場合は、パスが変更される可能性があります。

C:\JTSDK\wsjtx\trunk\qt55\18.1.95\Debug、C:\JTSDK\wsjtx\trunk\qt55\18.1.95\Release

デバッグモードで起動確認される場合は、作成されるwsjtx.cmdのスク립トコマンド中にあるCALLで飛ぶ先をjtdx.exeに修正する必要があります。(CALL jtdx.exe)

2. JS8Call のソースコードの入手先はどこですか？

JS8Call : <https://bitbucket.org/widefido/wsjtx/downloads/?tab=tags>

JTDX ビルドの方法を参考にして JTSDK により **JS8Call** がビルド可能です。

3. WSJTX 2.0 のビルドは可能ですか？

はい、可能です。 但し、JTSDK ファイルの一部を修正する必要は有ります。また、ソースコードは以下の url から入手可能となります。 wsjtx-2.0.0-xxx.tgz の階層には tar の圧縮ファイルがあります。 最終的に wsjtx.tar を解凍しますとフォルダー wsjtx のソースコードが得られます。

<https://physics.princeton.edu/pulsar/k1jt/wsjtx.html>

次のファイル操作を理解できない方にはビルドをお勧めいたしませんので読み飛ばしてください。

3-1. C:\JTSDK\qtenv.cmd をテキストエディターで開き下記の図の背景色が黄色であるところに

DOSKEY build-wsjtx2=pushd %cd% ^& %scr%\qtenv-build-wsjtx2.cmd \$* を一行追加します。

```
REM -----
REM  BUILD COMMANDS
REM  -----
DOSKEY build-wsprx=pushd %cd% ^& %scr%\qtenv-build-wsprx.cmd $*
DOSKEY build-map65=pushd %cd% ^& %scr%\qtenv-build-map65.cmd $*
DOSKEY build-wsjtx=pushd %cd% ^& %scr%\qtenv-build-wsjtx.cmd $*
DOSKEY build-wsjtx2=pushd %cd% ^& %scr%\qtenv-build-wsjtx2.cmd $*
DOSKEY wsjtx-list=pushd %cd% ^& %based%\scripts\qtenv-build-list.cmd $*
DOSKEY build-hamlib3="%based%\scripts\build-hamlib3.cmd"
```

ファイル qtenv.cmd を保存します。

3-2. C:\JTSDK\scripts\qtenv-build-wsjtx.cmd をテキストエディターで開き下記の図の背景色が黄色である行を削除します。

```
:SVN-CHECKOUT
ECHO.
ECHO -----
ECHO SVN Check
ECHO -----
ECHO.
CD /D %srcd%
ECHO Checking Branch ^( %nopt% ^)
ECHO SVN URL ...^: %burl%/nopt%
ECHO Local .....^: %branch_srcd%%nopt%
IF NOT EXIST %branch_srcd%%nopt%.svn%NUL (
start /wait svn checkout %burl%/nopt% %branch_srcd%%nopt%
IF ERRORLEVEL 1 ( GOTO SVN-CO-ERROR )
GOTO GET-SVER
) ELSE ( GOTO ASK-SVN-UPDATE )
```

削除した後

```
:SVN-CHECKOUT
ECHO.
ECHO -----
ECHO SVN Check
ECHO -----
ECHO.
CD /D %srcd%
ECHO Checking Branch ^( %nopt% ^)
ECHO SVN URL ...^: %burl%/nopt%
ECHO Local .....^: %branch_srcd%%nopt%
GOTO GET-SVER
```

修正したファイルを **qtenv-build-wsjtx2.cmd** として保存します。 **wsjtx2** にする事を忘れないでください。

3-3. 以後、Web からソースコードをダウンロードして WSJTX 2.0 をビルドする際に使用するコマンドは以下のようになります。

build-wsjtx2 dinstall, build-wsjtx2 rinstall, build-wsjtx2 package.

3-4. build-wsjtx を入力してエラーとなった場合は C:\JTSDK\src\wsjtx\trunk\svn 削除する

4. JTDX 用の修正 Hamlib3 ビルドを使用する場合は？

hamlib3-qt55_backup_v2018xxxx_mod.zip を解凍すると hamlib3-qt55 のフォルダーが出来ます。そのフォルダーを C:\JTSDK\以下に移します。現存する C:\JTSDK\hamlib3-qt55 をリネームするかは個人の責任で行ってください。後は通常通りに JTDX のビルドを行います。

5. Hamlib3 のソースコードを修正するには？

C:\JTSDK\src\hamlib3\src にあるファイルが Hamlib3 のソースコードにあたります。必要とするファイルを修正したのちに Hamlib3 を再ビルドします。

別途提供された Hamlib3 のソースコード(C 言語)を入手された場合は？

C:\JTSDK\src\hamlib3\src\以下に入手されたソースコードを書きこみます。現存するファイルをリネームするかは個人の責任で行ってください。書き込み後、Hamlib3 を再ビルドします。残念ながら提供されているソースコードに svn フォルダーが無い場合にはビルド時にエラーとなります。最近書かれたソースコードでしたら新規にダウンロードした Hamlib3 のソースコードの svn フォルダーをダミーとしてしようすると解決できるかもしれません。

6. JTSDK-QT コマンドラインへのビルドコマンドオプション記入

wsjtx 開発バージョン以外の物をビルドする際に必要となるコマンドです。JTSDK-QT を実行します。GA and RC List内に目的の物がリストされているか **wsjtx-list -a**にて確認をします。無い場合には**wsjtx-list -u**を入力してSVN repository listのアップデートを行います。現在は **wsjtx-1.9.1**が一番新しい物となっています。

例えば公開版v1.9.1 GA (r8747) のコマンドオプションは

build-wsjtx -b gar -n wsjtx-1.9.1 -c debug -t install

or

build-wsjtx -b gar -n wsjtx-1.9.1 -c release -t install

or

build-wsjtx -b gar -n wsjtx-1.9.1 -c release -t package

7. コマンドヘルプ

WSJT-X COMMAND LINE OPTIONS

(JTSDK-QT 5.5) C:\¥JTSDK) build-wsjtx -b gar -n wsjtx-1.9.0-rc3 -c release -t package

Usage ..:: build-wsjtx [-h] [-b] [-n] [-c] [-t]

Example....: build-wsjtx -b dev -n wsjtx -c release -t install

OPTIONS:

- h Displays this message
- b (dev | gar)
dev = development branches ^/branches
gar = GA and RC branches ^/tags
- n Branch Name: wsjtx, wsjtx-1.6, wsjtx-1.6.0-rc1, etc
- c Cmake Build Type: (release | debug)
- t install package docs | user-defined

* To Display this message, type ..:: build-wsjtx -h

* To List available branches, type ..: wsjtx-list -a

* Return to Main Menu, Type ..:: main-menu

WSJT-X DEFAULT BUILD COMMANDS

(JTSDK-QT 5.5) C:\¥JTSDK) build-wsjtx package

Usage ..:: build-wsjtx [OPTION]

Example....: build-wsjtx rinstall

OPTIONS:

- | | |
|----------|-----------------------------------|
| rconfig | WSJTX Devel, Release, Config Only |
| dconfig | WSJTX Devel, Debug, Config Only |
| rinstall | WSJTX Devel, Release, Install |
| dinstall | WSJTX Devel, Debug, Install |
| package | WSJTX Devel, Release, Package |
| docs | WSJTX Devel, Release, User Guide |

WSJT-X List Help Options

(JTSDK-QT 5.5) C:¥JTSDK) wsjtx-list -h

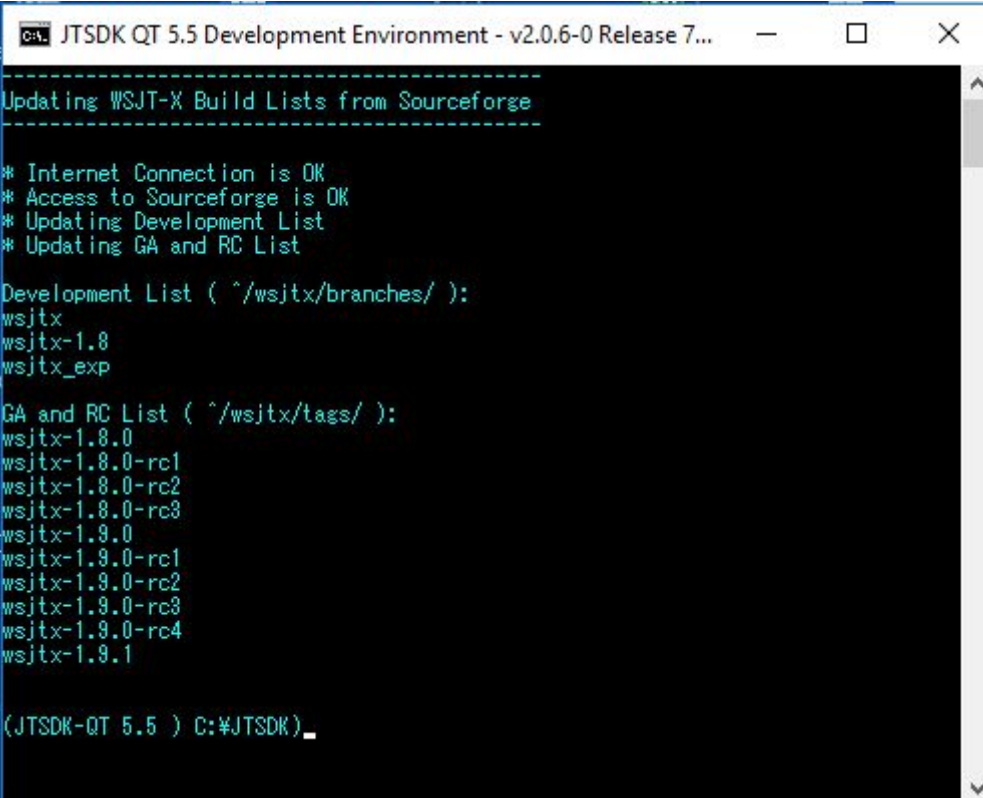
Usage ...: wsjtx-list [option]

Example...: wsjtx-list -h

OPTIONS:

- h Display this message
- a Display all lists
- d Display the development branch list
- g Display the GA and RC tags list
- u Update lists from SVN repository

wsjtx-list -u を実行 利用可能な SVN をアップデートした状態



```
-----  
Updating WSJT-X Build Lists from Sourceforge  
-----  
* Internet Connection is OK  
* Access to Sourceforge is OK  
* Updating Development List  
* Updating GA and RC List  
  
Development List ( ~/wsjtx/branches/ ):  
wsjtx  
wsjtx-1.8  
wsjtx_exp  
  
GA and RC List ( ~/wsjtx/tags/ ):  
wsjtx-1.8.0  
wsjtx-1.8.0-rc1  
wsjtx-1.8.0-rc2  
wsjtx-1.8.0-rc3  
wsjtx-1.9.0  
wsjtx-1.9.0-rc1  
wsjtx-1.9.0-rc2  
wsjtx-1.9.0-rc3  
wsjtx-1.9.0-rc4  
wsjtx-1.9.1  
  
(JTSDK-QT 5.5 ) C:¥JTSDK)_
```

JTSDK OPTION STATUS

(JTSDK-QT 5.5) C:\JTSDK) list-options

USAGE ...: enable-[NAME] or disable-[NAME]

NAME ...: separate qt55 quiet skipsvn autosvn clean rcfg autorun

CURRENT STATUS (オプション値は私の環境です。)

Separate: Yes

Quiet Mode ...: No

Skip SVN: No

Auto SVN: No

Clean First ...: Yes

Auto run: No

Reconfigure ...: Yes

Use QT5.5: Yes

DESCRIPTION

separate: Separate by App Version + SVN Version

quiet: Enable or Disable additional on screen messages

skipsvn: If Enabled, dont ask and dont update from SVN

autosvn: If Enabled, perform the SVN update without asking

clean: Clean the build tree before cmake --build .

autorun: Run the build without asking

rcfg: Re-run cmake configure

qt55: Enable or Disable using QT5.5 Tool Chain

When QT55 is enabled or disabled, you **Must** restart JTSDK-QT before the change can take affect.

* To Display this message, type ...: list-options

* Return to Main Menu, Type: main-menu

8. SVNによるダウンロードエラー回復方法

ビルド開始時点から直ぐにSVN Update エラーが出てビルドに失敗する場合は次のようなケースがあります。サーバーがメンテナンス等により接続制限がありソースコードのダウンロードができない。若しくは何らかの不都合によりダウンロードされているファイルとのリビジョン確認が取れない場合があります。前者は後日再度ビルドを試みます。後者は次のような回復処置でエラーの発生を回避できることもあります。

-
- 1) JTSDK Maintenance を立ち上げます。
 - 2) `cd C:\JTSDK\src\wsjtx\trunk` を実行、若しくはソースコードがあるフォルダーへ移動します。
 - 3) `svn up` を実行します。
 - 4) エラー表示を確認します。
もし、エラーが出ない場合はこのケースには該当していないと思われます。
 - 5) 次に `svn cleanup` を実行します。

注) `svn cleanup` 実行後に `svn up` を行うと `development (devel)`, `general available release (gar)` に関わらず最新のリビジョン (開発版) のソースコードがダウンロードされるようです。公開版 (`gar`) のソースコードを得るには JTSDK-QT のコマンドラインよりオプションコマンドを入力する事で対応します。

例 :

```
C:\JTSDK\src\wsjtx>svn up
svn: E155004: Run 'svn cleanup' to remove locks (type 'svn help cleanup' for details)
svn: E155004: Working copy 'C:\JTSDK\src\wsjtx' locked.
svn: E155004: 'C:\JTSDK\src\wsjtx' is already locked.
```

```
C:\JTSDK\src\wsjtx>svn cleanup
```

```
C:\JTSDK\src\wsjtx>svn up
Updating '!':
U   mainwindow.ui
U   Configuration.cpp
Updated to revision 7558.
```

```
C:\JTSDK\src\wsjtx>
```

追記 :

JTSDK-QTにて`list-options`の `enable-clean` を実行していない場合は 実行する事をお勧めします。 `clean` コマンドが有効になります。

9. wsjtxのリビジョン

Repository (サーバーにおけるソースコードの管理場所) は大きく Development List (^/wsjt/baranches) と GA and RC List (^/wsjt/tags) に分かれています。開発中のソースコードは Development List に属し随時アップデートされ、その都度リビジョンの番号が進みます。Release Candidate となった場合にも GA and RC List でソースコードは新しいリビジョン番号にて管理されます。

GA 若しくは RC がリリースされましたら 新たな Repository にアクセスできるように SVN List をアップデートする必要があります。
その時に使用するコマンドは `wsjtx-list -u` です。

残念ながら SVN List は 1.9.1 以降止まっています。また、今の JTSDK2.0 では `wsjtx-2.0.0` 用の SVN が利用可能ではないようです。11/23/2018 現在、ソースコードがあれば WSJTX2.0, JTDX, JS8Call はビルド可能です。

November 30, 2018

JG1APX

@ 4