Qt World Summit 2019

The Future is Written with Qt

Berlin, 4-6 November / Tokyo, 29 November

www.qtworldsummit.com

Introducing a new OS for Qt: Lindows… or is it Winux?

Romain Pokrzywka

Wall UX Architect

BLUESCAPE

In memory of
Guillermo Amaral (@gamaral)

1981 - 2019

BLUESCAPE

Overview

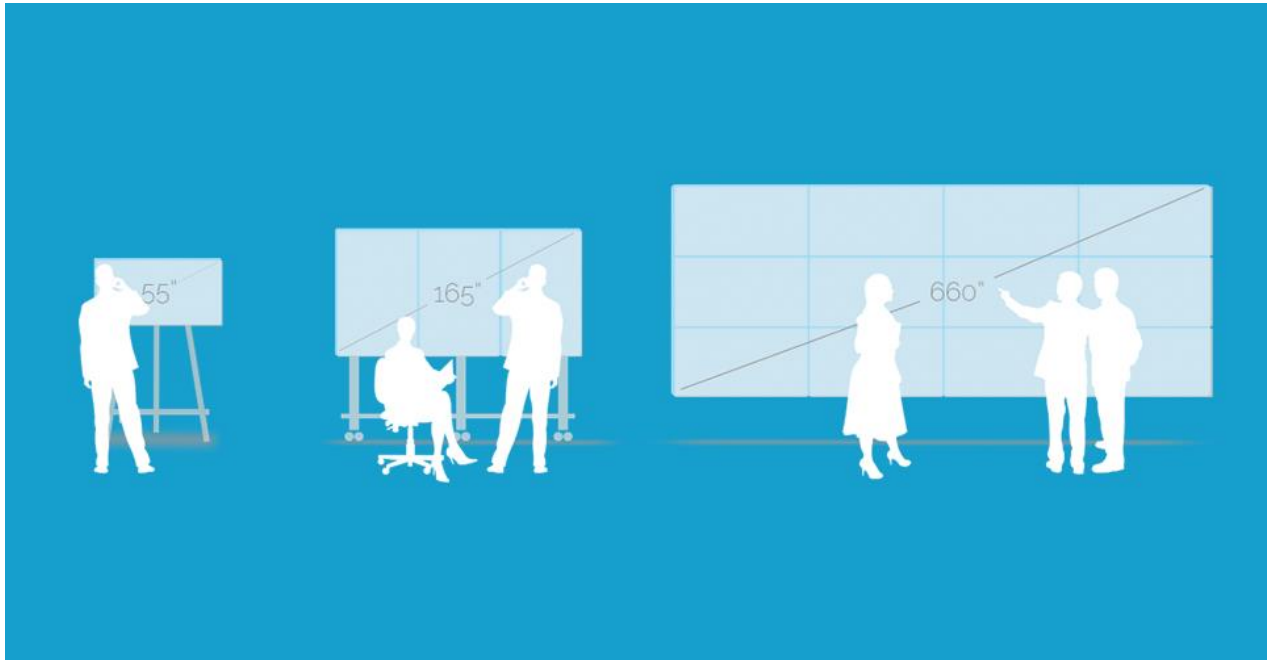
- Bluescape: a story about OS migration
- WSL : Windows 10’s cleverest trick for developers
- Tips & tricks for cross-platform scripting

BLUESCAPE

Bluescape: A visual collaborative platform

Qt + Bluescape =

- QML/QtQuick
  - most UX elements
- QtOpenGL
  - fullscreen rendering
- QtWebEngine
  - Web Browser object
  - WebRTC
  - Object container (documents, videos)
- QtDBus
  - Cross-process IPC



Migrating from Linux to Windows

Qt makes it (mostly) easy:

- Platform abstractions
- qmake/cmake

…but what about:

- 3rd party dependencies
- Deployment/packaging shell scripts
- Developer environment, tools, etc.

Migrating from Linux to Windows

Our options:

- Installing Cygwin/MSYS:

- Porting your scripts to Batch/PS:

Enter WSL - the Windows Subsystem for Linux

- Essentially “Native Linux on Windows”
  - Native code, no VM. Binaries are 100% identical to Linux
  - Native filesystem access, Windows drives show up under `/mnt`
- Several distributions installable: Ubuntu, Fedora, OpenSUSE
- Officially limited to console mode

  …but there’s unofficial X11 support via a Windows XServer:

  (Xming, VcXsrv…)

Integrating your scripts with WSL

- Windows OS detection
  - uname gives info about the system and the kernel:

```
romain@Romain-asset1946:~$ uname -srvo
Linux 4.4.0-17763-Microsoft #379-Microsoft Wed Mar 06 19:16:00 PST 2019 GNU/Linux
```

  - to use inside bash scripts:

```
if [[ $(uname -r) =~ Microsoft ]]; then
    echo "Running inside WSL"
fi
```

Integrating your scripts with WSL

- Filesystem paths conversion
  - Manual conversion is doable via `/mnt`, but WSL provides the **`wslpath`** tool to help:

```
romain@Romain-asset1946:~$ wslpath -u "C:\Program Files\Bluescape"
/mnt/c/Program Files/Bluescape
romain@Romain-asset1946:~$ wslpath -w /mnt/c/Users/Romain.Pokrzywka
```

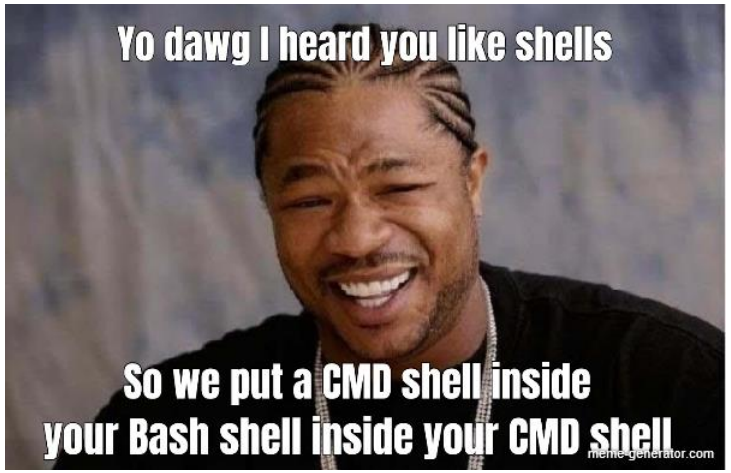
  - Also works through symlinks:

```
romain@Romain-asset1946:~$ ls -l dev
lrwxrwxrwx 1 romain romain 10 Apr 10  2019 dev -> /mnt/c/dev
romain@Romain-asset1946:~$ wslpath -w ~/dev/tsx_wall
C:\dev\tsx_wall
```

… but no access to Linux fs from Windows!

```
romain@Romain-asset1946:~$ wslpath -w $HOME
wslpath: /home/romain: Invalid argument
```

Integrating your scripts with WSL



- Environment propagation
  - Only **PATH** is automatically propagated to WSL
  - **WSLENV** controls additional envvars to propagate

```
C:\Users\RomainPokrzywka>set BR
BREAKPAD_BIN_DIR=C:\dev\bin
BREAKPAD_SYMBOLS_DIR=C:\dev\symbols\linux
C:\Users\RomainPokrzywka>set WSLENV=BREAKPAD_BIN_DIR/p:BREAKPAD_SYMBOLS_DIR/p
C:\Users\RomainPokrzywka>wsl.exe

romain@Romain-asset1946:/mnt/c/Users/RomainPokrzywka$ set | grep BREAKPAD
BREAKPAD_BIN_DIR=/mnt/c/dev/bin
BREAKPAD_SYMBOLS_DIR=/mnt/c/dev/symbols/linux
WSLENV=BREAKPAD_BIN_DIR/p:BREAKPAD_SYMBOLS_DIR/p
```

Integrating your Windows toolchain with WSL

- The GCC/Clang packages provided with WSL are the *Linux* toolchains!

  They will give you Linux executables, not Windows

- We still want to build native Windows executables, using Visual Studio or mingw
- Visual Studio has a build env script: **`vcvars*.bat`**

  ☺ which we can call from WSL

  ☹ but which doesn't accept commands

Integrating your Windows toolchain with WSL

- The trick: `vcvars_env_run.sh`
  - Runs commands inside a VS build env
  - Commands pre-parsed by Bash
  - Supports forwarding of `&& ||` operators and envvars

- Script and examples shared on GitHub:
  https://github.com/kromain/wsl-utils

vcvars_env_run.sh examples

- Simple command: building 3rd party deps with vcpkg:

```
local vcpkg_cmd="vcpkg.exe --triplet %VCPKG_DEFAULT_TRIPLET%"
local vcpkg_packages="aws-sdk-cpp[s3] breakpad expat libconfig tbb"

cd $VCPKG_DIR

vcvars_env_run.sh bootstrap-vcpkg.bat
vcvars_env_run.sh $vcpkg_cmd install $vcpkg_packages
```

vcvars_env_run.sh examples

- Complex command: qmake out-of-source build

```
cd $BUILD_DIR

local build_cmd=( \
    $(wslpath -w ${DEPS_DIR}/bin/qmake.exe) \
        -r \
        CONFIG+=${QMAKE_CONFIG} \
        $(wslpath -w ${SOURCE_DIR}/tsx_root.pro) \
    "&&" ${MAKE_TOOL} \
)

${SCRIPT_DIR}/vcvars_env_run.sh "${build_cmd[@]}"
```

Thank you!

Questions?

memeguy.com

PS: WSL2 is coming

- Now a “lightweight VM”
- Faster I/O
- Access to Linux FS from Windows

ICYMI: https://github.com/kromain/wsl-utils