

# Cross-compiling using MXE

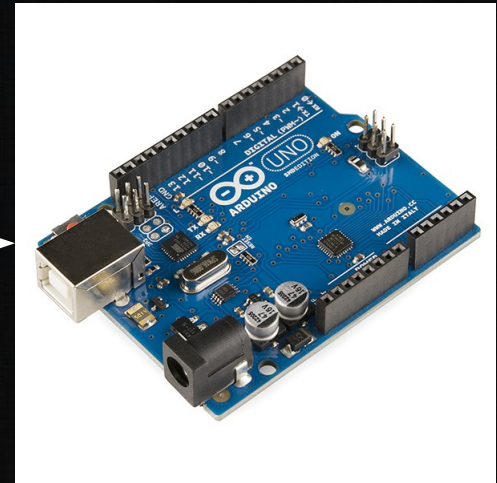
© 2017 Richel Bilderbeek

[www.github.com/richelbilderbeek/CppPresentations](http://www.github.com/richelbilderbeek/CppPresentations)



# Cross compiling

- Compile code for a different operating system



# Linux → Windows

- Use MXE ('M Cross Environment')
- Homepage at <http://mxe.cc/>
- GitHub at <https://github.com/mxe/mxe>
- Active community





# MXE setup

- Clone the repository



```
git clone  
https://github.com/mxe/mxe.git
```

# MXE setup

- Build the executables
- Can take many hours!

```
cd mxe
```

```
make gcc boost qt qt5 qtbase wt qwt  
sfml
```

# MXE setup

- Add MXE to path
- Restart terminal before cross-compiling

```
echo "export  
PATH=/home/riche1/GitHubs/mxe/usr/b  
in:$PATH" >> ~/.bashrc
```



# MXE setup

- Create a Makefile for your project
- Static build: no DLLs needed, just one big .EXE

```
i686-w64-mingw32.static-qmake-qt5  
my_project.pro
```

# MXE setup

- Build your project
- Takes as long as any regular build
- Uses g++ version 4.9.4

```
make
```



# Convenience

- Building MXE can easily be scripted
  - <https://github.com/richelbilderbeek/Ri biLibraries/blob/master/mxe.sh>
- Cross-compiling can easily be scripted
  - <https://github.com/richelbilderbeek/Brainweaver/blob/master/BrainweaverCross compile.sh>

# Problems

- MXE takes a long time to build
  - Overnight
  - Only build what you need



# Problems

- You favorite package may be absent
  - List of all +400 packages at <http://mxe.cc/#packages>
  - Documentation of new packages at <http://mxe.cc/#creating-packages>



# Problems

- Builds may fail, as libraries change
  - Will be fixed within days
  - Do not build the day before delivering the final executable

# Conclusion

- MXE is easy to use
- MXE is harder to build
- Think ahead when using MXE



# Questions?

