Designing for Testability

By Bo Thorsen

Todays Menu

- Me
- Testing?
- The fundamental model
- Project layout
- Testability
- QMake
- CMake
- Closing thoughts

Fionia Software – Bo Thorsen

- Qt Experts
- Solving hard Qt issues
- Implement full Qt projects
- Project management
- Any project size
- Part of the team
- Focus on Qt Embedded and Windows/Linux Desktop

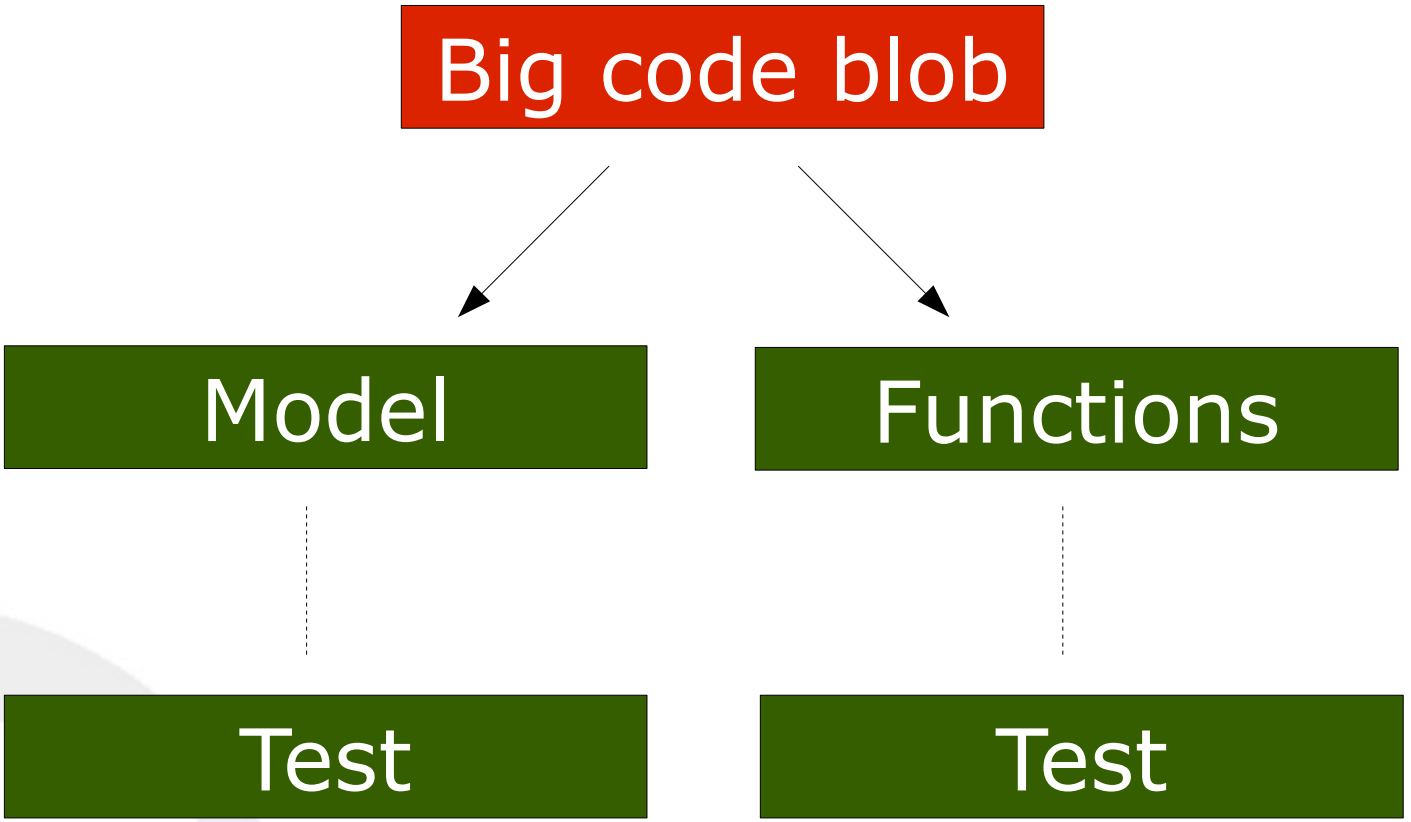
Testing

- Unit testing
- Regression testing
- User testing
- Integration testing
- System testing
- UI testing

Unit Testing

- Tests small code parts:
  Call a method and check the situation
- Regression test suite
- Necessary for agile processes
- Can't replace user testing or other types of tests
- Not the fundamental development instrument

The Fundamental Model



Project Layout 1 – FS Editor

- LibTest
- LibModel
- LibModelCommands
- LibUi
- Application
- Unittest

Download on http://fioniasoftware.dk/fseditor.html

FS Editor – QMake

- One big test (doesn't have to be so)
- Each dir has a unittest subdir with tests
- unittest.pro includes all the **/unittest/unittest.pri
- Runs the unit test at the end of the build

FS Editor – QMake

```
QT += script xml testlib
SOURCES += fstest.cpp
HEADERS += fstest.h

include (../model/model.pri)
include (../modelcommands/modelcommands.pri)
include (../ui/ui.pri)
include (../model/unittest/model-unittest.pri)

win32 {
    CONFIG(debug, debug|release) { TDIR=debug/ }
      else { TDIR=release/ }
} else {
    TDIR=
}
QMAKE_POST_LINK = $${OUT_PWD}/$${TDIR}
     $${TARGET}$${TARGET_EXT}
```

Project Layout 2

- libs/tools
  libs/test
  libs/lgpltools
- codegenerators/libs
  cg/db
  cg/rest
- src/model
  src/functionality1
  src/ui
  src/application

Customer Project – CMake

- Huge project => Huge test => one test per subproject
- Each dir has a unittest subdir with tests
- CMake code to make the test builds easy
- Doesn't run the unit tests at the end of each build
- Regression test server runs the tests every night

CMake code example 1

```
SET(headers ...)
SET(moc_headers ...)
SET(sources ...)

QT4_WRAP_CPP(moc_generated ${moc_headers})

INCLUDE(${QT_USE_FILE})

ADD_LIBRARY(model STATIC ${headers} ${moc_headers}
   ${sources} ${moc_generated})
TARGET_LINK_LIBRARIES(model ${QT_LIBRARIES})

ADD_SUBDIRECTORY(unittest)
```

CMake code example 2

```
INCLUDE(${PROJECT_SOURCE_DIR}/cmake/FSTest.cmake)
SET(test_model TournamentTest)
SET(test_model_libs model )
FSTEST(test_model Model)
```

- FSTest.cmake macro FSTEST creates the test application, adds the proper libraries and adds the test files
- In the example above, it will add unittest/tournamenttest.h and .cpp

Other test/build types

- Squish
- Valgrind
- QBS
- ...

Closing Toughts

- Write the tests when it makes sense, not before (well, doh!)
- Use them for debugging
- Choose the parts that absolutely must be completely test covered

Questions?