Developing Applications for MeeGo*

Horace Li
Technical Account Manager
Software & Services Group

Agenda

- MeeGo* Overview
- MeeGo UI Framework
- Create MeeGo Touch App
- MeeGo SDK

MeeGo* Strategy – Spans Multiple Segments

Netbooks | Connected TV | Handsets | IVI | Media Phone

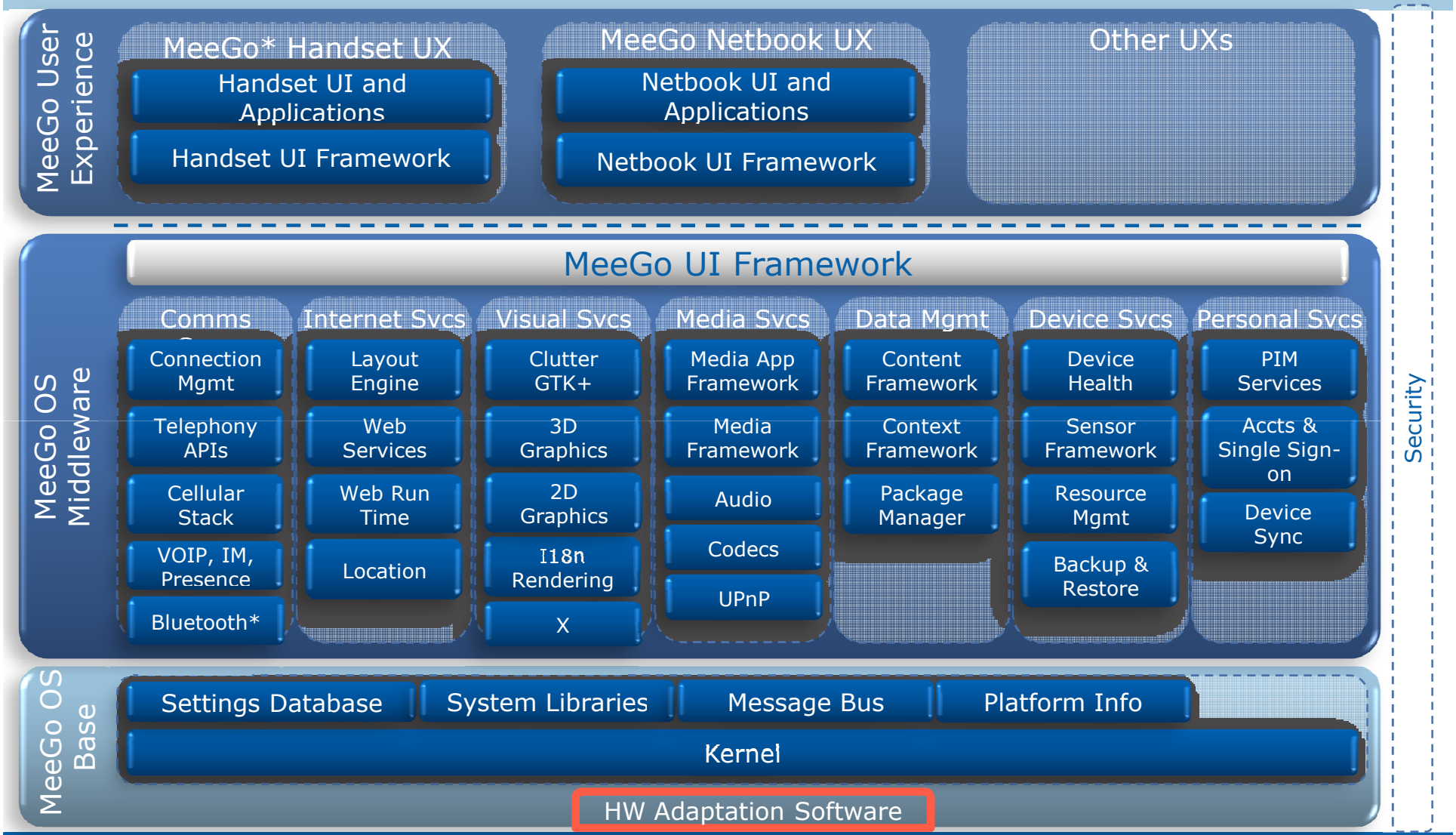
Intel® Atom™ Developer Program & AppUp(SM) Center

MeeGo* = Moblin™ + Best of Maemo

Platforms based on Intel processors

MeeGo is a continuation of the Intel Atom brand software strategy execution

MeeGo* Architecture

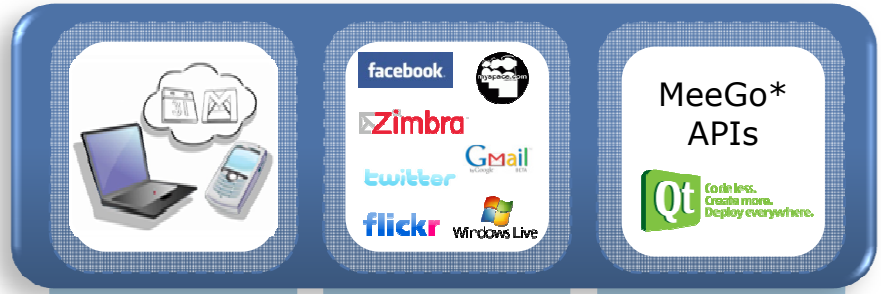


MeeGo* on Intel® Atom™ Processor Features Overview

Segment Specific User Experiences

Connection Manager for data connectivity

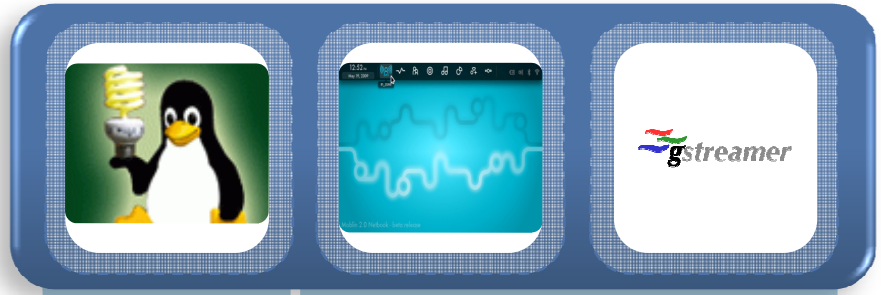
Telephony Framework



Cloud-device sync of PIM Data

Integrated Social Networking

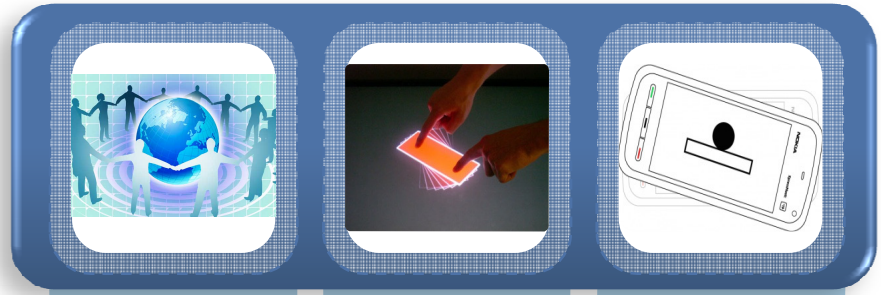
Application Development Environment



Improved Power Management

Fastboot & Shutdown Optimization

Support for Multiple Multimedia Framework



International-ization with UI guidelines

Gesture & MultiTouch Framework

Sensor Framework

Note: Some features listed may be segment-specific

Agenda

- MeeGo* Overview
- MeeGo UI Framework
- Create MeeGo Touch App
- MeeGo SDK

MeeGo* Application Development Environment

MeeGo offers Qt* and Web runtime for app development:

- Qt for native C++ and Web runtime for Web applications (HTML, JS, CSS, etc.)
- Qt and Web runtime bring cross platform development so apps can span multiple platforms
- Native development tool: Qt Creator
- Web development tools: plug-ins for standard web development tools

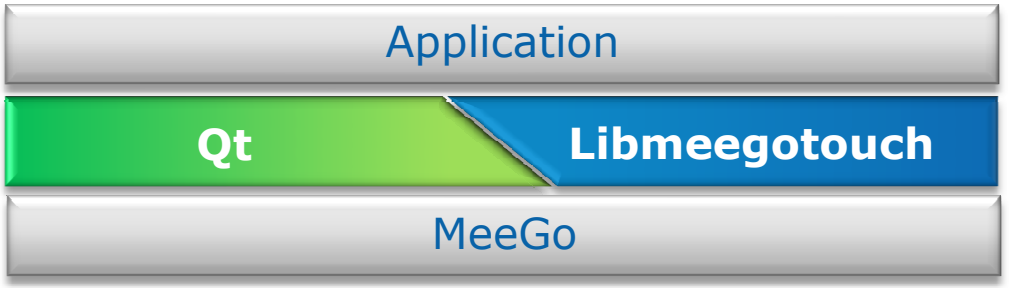
MeeGo offers a complete set of tools for developers to easily and rapidly create a variety of innovative applications

MeeGo Touch UI Framework

- **Libmeegotouch, a touch friendly visual widget library.**
- **Highlighted features,**
  - A complete set of widgets that provides a specific UI style primarily targeting touch screen devices.
  - Various graphics layouts support, including linear layout, grid layout, etc.
  - Native transformations support, including rotated, scaled or perspective adjusted.
  - Seamlessly integration with Qt modules like I/O, SQL & XML programming.
  - Internationalization enabled, 33 languages support.
  - Inter-process communication support, allows applications to simply either use or serve an interface
- **MeeGo touch UI framework is LGPL licensed.**
- **Source repository,**

```
$ git clone git://gitorious.org/meegotouch/libmeegotouch.git
```

Relationship with Qt



- Libmeegotouch is built on top of Qt library, having native gesture/multi-touch event recognition and handling.
- Both Qt and libmeegotouch can be used directly.
  - libmeegotouch is primarily used to create the UI layouts, animations, and touch response.
  - Qt is used for lower layer logic.
- While Qt creates traditional desktop UI, libmeegotouch make scene based applications easier.

Agenda

- MeeGo* Overview
- MeeGo UI Framework
- Create MeeGo Touch App
- MeeGo SDK

Sample MeeGo Touch App

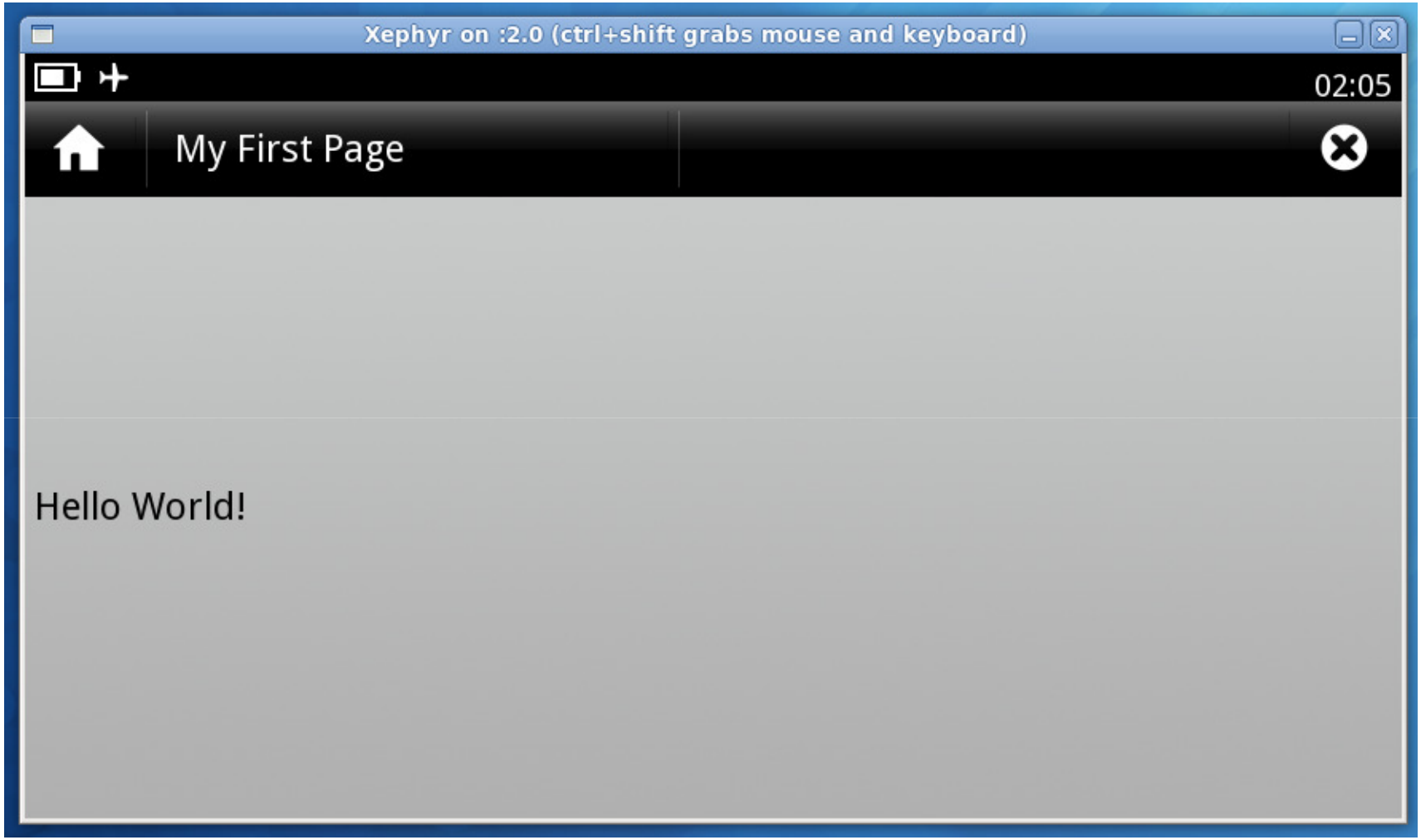
- A simple MeeGo touch 'Hello World' application, saved as *main.cpp*:

```
#include <MApplication>
#include <MApplicationWindow>
#include <MApplicationPage>
#include <MLabel>

int main(int argc, char **argv)
{
        MApplication app(argc, argv);
        MApplicationWindow window;
        MApplicationPage page;

        page.setTitle("My First Page");
        page.setCentralWidget(new MLabel("Hello World!"));
        page.appear(&window);
        window.show();
        return app.exec();
}
```

Sample Hello World Screenshot

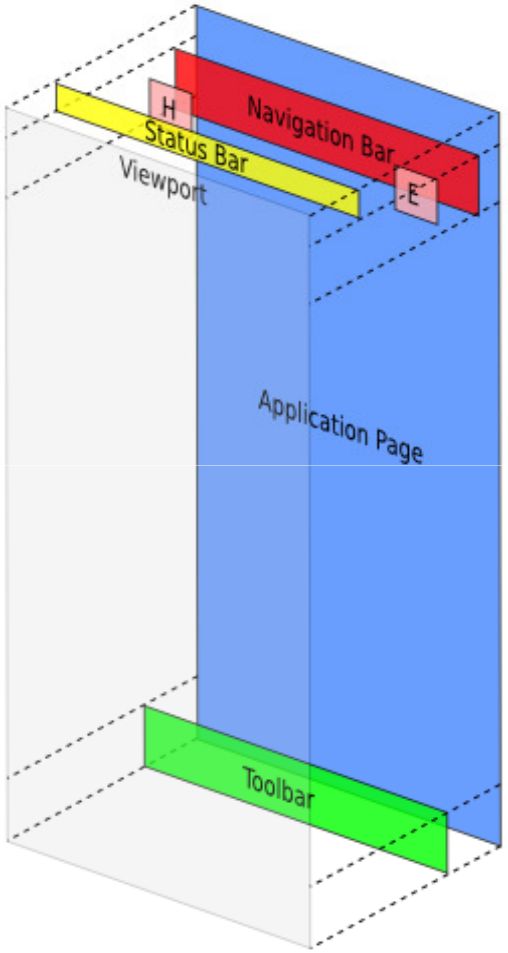


MApplication

- The first object to be created in MeeGo* touch application
- Derived from QApplication
- Mainly created to:
  - Handle the main event loop
  - Manage the GUI application's control flow and main settings
  - Construct an instance of QDbus service, MApplicationService
- By default, only a single instance of any application is allowed
- To allow multiple instances of an application…
  - Derive a class from default MApplicationService
  - Override its launch() function

MApplicationWindow

- Derived from MWindow
- Create a top-level application window
  - Status bar
  - Navigation bar
  - Application Page
  - Toolbar
- To insert content into the window, a MApplicationPage has to be created and shown
- Orientation changes and in-scene windows are automatically managed
- Set MApplicationWindow full screen will hide status bar; set back normal mode will show status bar again



MApplicationPage

- Derived from MSceneWindow
- Provide a framework for building an application's user interface
- Create an pannable viewport
- Only one page can be displayed at any given time
- Navigation between pages are also technically achievable
  - Call child page appear() will make current page disappear and show child page instead
- Page navigation history is managed by MSceneManager
- Manage navigation bar display mode

Build Up MeeGo* Touch App

- MeeGo* touch is created on top of Qt*, sharing Qt building system
- Firstly generate project (.pro) file, by using command

```
$ qmake -project
```

- A typical generated project file content

```
#####################################################
# Automatically generated by qmake (2.01a) Tue Apr 20 14:00:01 2010
#####################################################
TEMPLATE = app
TARGET =.
DEPENDPATH += .
INCLUDEPATH += .
# Input
SOURCES += main.cpp
```

- Add libmeegotouch into building system, edit project file

```
CONFIG += meegotouch
```

- Following standard qmake steps

```
$ qmake
$ make
```

Agenda

- MeeGo* Overview
- MeeGo UI Framework
- Create MeeGo Touch App
- MeeGo SDK

MeeGo* SDK Introduction

- The MeeGo* SDK consists of:
  - A MeeGo handset or netbook image, which contains a Simulator for MeeGo applications (Linux* only currently), based on QEMU (a generic and open source machine emulator and virtualizer), and Qt* Creator which can be configured to deploy to remote MeeGo devices
  - A new development tool, MADDE, integrated with Qt Creator to provide common application development and debug environment.

- MeeGo SDK Portal URL: http://meego.com/developers/getting-started

- Pre-requisites
  - Intel® Architecture hardware platform, both 32bit Intel® Atom™ Processor and Intel® Core™2 Duo Processor are acceptable
  - A reasonably modern Linux distribution (e.g. Fedora* 12, Ubuntu* 10.04)
  - Access authority as root user,

```
$ su -i
```

Prepare System for QEMU

- Using QEMU for MeeGo development requires:
  - capable of VT (Virtualization Technology) support in your system.
  - hardware accelerated graphics installed.

- To check for VT support, run the following from a terminal.

```
$ egrep '^flags.*(vmx|svm)' /proc/cpuinfo
```

- Any output is success. Here's an example:

```
flags : fpu vme de pse tsc msr pae mce cx8 apic mtrr pge mca cmov pat pse36
clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe lm constant_tsc arch_perfmon
pebs bts pni dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm
sse4_1 lahf_lm tpr_shadow vnmi flexpriority
```

Install KVM Module

- Check that the kvm_intel or kvm_amd modules are loaded:

```
$ lsmod | grep kvm
kvm_intel 43816 0
kvm 164576 1 kvm_intel
```

- If needed, load the module.
- For an Intel graphics chipset:

```
$ sudo modprobe kvm_intel
```

- For an AMD graphics chipset:

```
$ sudo modprobe kvm_amd
```

- If the kvm modules are not available or fail to load, VT may not be enabled in the BIOS. Reboot your system, go into the BIOS, and enable VT.

Configure Distributions Package Manager

- Take Ubuntu distribution as example.
- Add the following line to /etc/apt/sources.list file or create a /etc/apt/sources.list.d/meego-sdk.list file containing the following line:

```
# deb http://repo.meego.com/MeeGo/sdk/host/repos/ubuntu/10.04/ /
```

- Add the repository public key:

```
# gpg --keyserver subkeys.pgp.net --recv 0BC7BEC479FC1F8A
# gpg --export --armor 0BC7BEC479FC1F8A | sudo apt-key add -
```

- Update packages database:

```
# aptitude update
```

- Check that the MeeGo repository has been added to the file with,

```
# apt-cache policy madde
```

Install MeeGo SDK

- To install the whole MeeGo SDK:

```
# aptitude install meego-sdk
```

- For ARM only:

```
# aptitude install meego-sdk-armv7l
```

- For IA32 only:
- To install MeeGo SDK components individually:
  - QtCreator
  - MADDE
  - QEMU
  - Toolchains for ARM and IA32

```
# aptitude install meego-sdk-qtcreator
# aptitude install madde qt-tools
# aptitude install arm-2009q1
# aptitude install meego-sdk-i586-toolchain
# aptitude install qemu-arm qemu-gl
```

Simulator Screenshot

Create and test a target with MADDE

- Create MeeGo target in MADDE:

```
# mad-admin create -f <target>
```

  - <target> is **meego-core-armv7l-1.1** , **meego-handset-ia32-1.1** or **meego-netbook-ia32-1.1**.
  - The **-f** flag instructs MADDE to download and install the appropriate sysroot tarball first.

- To check that target and toolchain are found in MADDE:

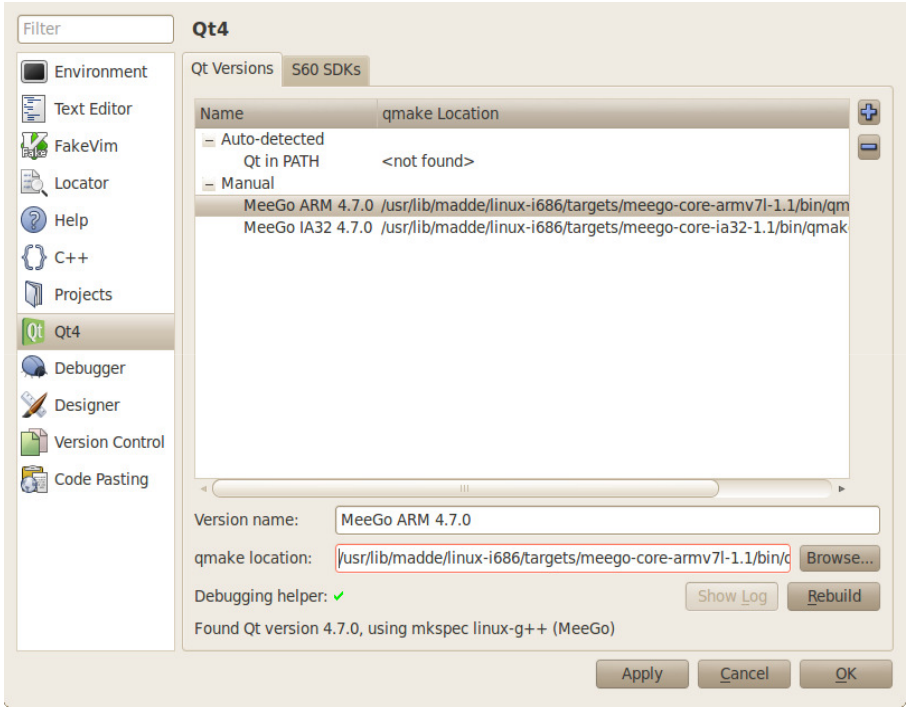
```
$ mad -t <target> pscreate -t qt-simple qthello
$ cd qthello
$ mad -t <target> qmake
$ mad -t <target> make
```

- To check that the qt-simple application is created for the correct target, run:

```
file build/qthello
```

Configure Qt Creator to use MeeGo toolchain(s)

- Start Qt Creator by selecting **Applications > Programming > Qt Creator**.
  1. Go to **Tools** > **Options** > **Qt4** > **Qt Versions**
  2. On the right side of the **Qt Versions** view, click on the plus sign button to add a new version.
  3. On the **Version Name** line, specify a name for the new version
  4. On the **qmake location** line, specify a qmake location for the new version. The toolchain installs by default to usr/lib/madde/linux-i686/targets/<target>/bin/qmake
  5. Click **Rebuild**, **Apply** and **OK**.



Summary

- MeeGo = Best of Moblin + Best of Maemo
- MeeGo is a fully open source software platform
- MeeGo offers a complete set of tools for developers
- Both Qt and libmeegotouch can be used directly
- MeeGo SDK simulates MeeGo environment on your computer

Q&A

Intel - Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life-saving, life-sustaining, critical control or safety systems, or in nuclear facility applications.

Intel products may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel may make changes to dates, specifications, product descriptions, and plans referenced in this document at any time, without notice.

This document may contain information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Intel Corporation may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Wireless connectivity and some features may require you to purchase additional software, services or external hardware.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

Intel, the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

MeeGo is a registered trademark of the Linux Foundation.

*Other names and brands may be claimed as the property of others.


intel®
Leap ahead™