Building Native Apps on BlackBerry 10

Shadab Rashid
Application Development Consultant
Developer Relations
Jan 14th, 2013
Agenda

- BlackBerry 10 Overview
- Native/Cascades Development
- Resources / Getting started
- Lots of Sample apps
- Q&A
Integrated  Social  Beautiful
Beautiful

Flow
Social Connected Apps Share Framework
Integrated Unified Inbox
IT GLANCES

Lets you keep an eye on what’s happening.

Peek from wherever you are

Content always on the foreground

Active thumbnails showing app context
IT THINKS
An intelligent, predictive keyboard that learns from and writes for you.

- Lets you type fast with just one finger
- Predicts the next letter, with the entire word
- Just swipe up on the letter to finish the word
IT REMEMBERS

Time Warp photo lets you capture a moment even if you just missed it.

No more odd expressions or awkward retakes

No more eyes closed!
Roads to BlackBerry 10 Development

C++/Qt
Cascades

C/C++
Native SDK

HTML5 BlackBerry®
WebWorks™
Open standards

- Very easy to port C/C++
- Platform libraries built on open standards
- POSIX and STL friendly (*stdc++*, *socket*, *pthread*, etc.)
Open Standards (2)

- POSIX and STL
- C++, Sockets (libstdc++, libsockets)
- OpenGL ES 1.1 / 2.0 (libEGL, libGLESv1_CM, libGLESv2)
- OpenAL 1.1 (libOpenAL, libalut)
- Imaging (libpng, libjpeg)
- Compression (libz, libbz2)
- Fonts (libfreetype)
- Math (libm)
- XML (libxml2, libxslt)
- Database (libsqlite)
Development Tools

- Supports Windows, Mac OS X, Linux
- GCC 4.6.3 based compiler tool chain
- GDB 7.3 debugging
- Device Deployment Tools
- VMware Simulator Images
- Eclipse CDT 8 based IDE (QNX Momentics)
BlackBerry 10 Overview

Native C++ Application Developers

UI Framework
BlackBerry Cascades

Platform, device and cloud APIs
Qt, QtCore

Web Application Developers
HTML5, CSS, JavaScript

WebWorks

Community and Content Developers

BlackBerry 10 OS
QNX

Multi-threading, Memory management, Security

Adobe AIR™
Android Runtime
Gaming Platforms

Marimade, unity
Zooming in on the NDK…

**QNX**

- Posix compatible
- C-APIs
- BPS event model
- Low Level Platform APIs
- Core open source libraries and standards
Open Source libraries
• Many already ported to QNX
• http://blackberry.github.com/ndk
• …and more getting contributed
Game platforms already ported to QNX
• ScoreLoop
• Unity
• Marmalade
• GameLoft
This part of the NDK is already available on PlayBook!

• http://developer.blackberry.com/native
Native/Cascades Development

C++/Qt Cascades

C/C++ Native SDK
Cascades in the native SDK

Cascades
The Frosting
- Beautiful UI out of the box
- C++ App Framework
- Based on Qt and QML
- BlackBerry 10 UX built-in
- BlackBerry Integration APIs

Core Native
The Sponge Cake
- Porting heaven
- Game developer’s delight
- POSIX compliant
- Raw OS access
- Open source friendly

Figure 1: BlackBerry 10 Native SDK
The NDK – Two layers

High Level – Cascades
- High level APIs / C++
- Qt, QML, JavaScript
- Built-in Elegance, beautiful UI
- Provides the BlackBerry 10 UX
- Visual tooling

Low Level – C API
- Posix/Low level C; great for porting
- Raw OS access, windowing, etc.
- OpenGL ES, OpenAL, ...Excellent for games
- Great for Open Source Integration
Why Use Cascades?

Elegant UI Framework
• Great looking core UI components
• Easy to build custom UI components
• Dedicated UI tooling including design preview (w/ Photoshop Plugin)

Increase Productivity
• Higher level APIs
• QT flavoured C++ and declarative UI approach
Cascades & Qt

- Mature C++ application framework
- Great APIs
- Signals and Slots
- Many helper classes
- QML
Cascades and QT
What’s the deal?

- Cascades uses QT to provide plumbing and high level services
  - Object and event models
  - Markup parser (QtDeclarative)
  - Signals/Slots (i.e. callbacks)
  - Core services (e.g: XML-parsing, SQL libs.. Many more)

- Cascades does not use QtGui or QtQuick
  - BlackBerry 10 has it’s own UI framework and roadmap
  - QtQuick QML != Cascades QML
Define the UI in QML

import bb.cascades 1.0

Page {
    content:
    Label {
        text: "Hello World!"
    }
}

It's like HTML and JSON had a baby.
import bb.cascades 1.0

Page {
    content:
    Label {
        text: "Hello World!"
        horizontalAlignment: HorizontalAlignment.center
    }
}

This looks a bit like CSS. Only not.
Add some JavaScript!

```javascript
import bb.cascades 1.0

Page {
    Container {
        Label {
            id: hello
            text: "Hello World!"
            horizontalAlignment: HorizontalAlignment.center
        }
        Button {
            text: "German-ize"
            onClicked: {
                hello.text = "Hallo Welt!";
            }
        }
    }
}

Finally!
```
Declarative QML

QML = Qt Modeling Language

QML can:

1. Used by UI designers
2. Rapid prototype UI
3. Do white label/branding/promotions of applications (with different QML files)

import bb.cascades 1.0

Page {
    content:
    Label {
        text: “Hello word!”
    }
}

(Simple QML designed to create a Hello World page)
Embeddable JavaScript!

- No need to reinvent the wheel
- Do more than just markup in QML
- Use separate JS files, create functions, handle signals

```qml
Button {
    text: "Expand"
    maxWidth: 200
    onClicked: {
        if (text == "Expand") {
            text = "Contract"
            maxWidth = 400
        } else {
            text = "Expand"
            maxWidth = 200
        } // else
    } // onClicked
} // Button
```
import bb.cascades 1.0

TabbedPane {
    showTabsOnActionBar: true
    Tab {
        title: qsTr("Tab 1")
        Page {
            id: tab1
            actions: [
                // define the actions for first tab here
                ActionItem {
                    title: qsTr("Rotate")
                    onTriggered: {
                        imgTab1.rotationZ = imgTab1.rotationZ + 90;
                    }
                },
                ActionItem {
                    title: qsTr("Break")
                    onTriggered: {
                        imgTab1.imageSource = "asset://images/picture1br.png";
                    }
                }
            ]
        }
    }
}
QML Visual Tooling

QML editor with syntax highlighting/code completion

QML design view

Component library (Drag and drop to QML)

QML tree view, QML properties view

```qml
import bb.cascades 1.0

Page {
    content: Container {
        layout: DockLayout {

            // Setting the background color to a color created from the string #262626
            background: Color.create("#262626")

            // The content (images, slider ...) is placed in a container
            content: Container {
                layout: StackLayout {

                    // The content should be centered
                    layoutProperties: DockLayoutProperty
                        horizontalAlignment: HorizontalAlignment.Center,
                        verticalAlignment: VerticalAlignment.Center

                    // The two images are put on top of each other
                    // A nice looking image frame is placed around the first

                    // A control that indicates that a process is being completed.
                    ActivityIndicator

                    // A double button with fixed height and optional text and image.
                    Button

                    // A check box is a control with two states, either 'checked' or 'not checked'.
                    Checkbox

                    // A container that can be used for grouping controls and other containers.
                    Container

                    // A control for selecting a date or time.
                    DateTimePicker

                }
            }
        }
    }
}
```
C++ and QML – A two way street.

C++ code can load QML files and access QML objects.

QML files can reference C++ values and objects.

Everyone is happy 😊
Invoking C++ from QML

```cpp
/**
 * Invokable function used for triggering sounds from QML.
 *
 * @param msg a string describing which sound should be played.
 */
Q_INVOKABLE
void playSound(const QString &msg);

// Background ImageView.
ImageView
imageSource: "asset://images/background.png"

// If we touch the background we
onTouch: {
  if (event.isDown ()) {
    cowbellApp.playSound("moc"
  }
}
```
Which layer for what?

- For UI – QML
- C++ for application logic (QtCore based)
- C and BPS to access system features
- Our github repo for popular frameworks and libs:
  github.com/blackberry
C++/Qt with C API – an example

Invoke:
- Play
- Pause
- Stop
- Seek

Update:
- Status
- Position
- Errors

UI (QML)

Qt Signals

Wrapper class (C++/Qt)

Actions - C API

Events – C API

Cascades
Cascades Cookbook

In this cookbook, I’ve put a list of ingredients for making user interfaces. Just like sushi, the end result will depend on your skill of aligning and balancing the components until they look just right.

Good luck!
/The Chef
UI Framework

Custom QML components
Create your own custom QML components to supplement the core controls, and learn how to reuse your components.

Images
Learn the strategies and best practices for creating and displaying images in your application.

Layouts
Explore the different layout patterns that are available in Cascades and learn when to use them to arrange your UI controls.
**Updated: Navigation**

Provide navigation in your apps by creating new screens, arranging screens in a stack, or using tabs to display your content.

**Animations**

Discover the powerful implicit and explicit animation system in Cascades, and learn how to precisely control your animations.

**Updated: Text**

Add some visual appeal to your text controls and learn how to display different versions of the virtual keyboard.
UI Framework (3)

Demo time: Cascades Cookbook, Cross fade, Poem Maker

Updated: Lists
Take advantage of the power and flexibility of lists, the basis of most mobile apps, and customize them to create the look you want.

Updated: Web content
Incorporate a WebView into your application, to browse webpages or use HTML inside your app. Learn about WebView.

New: Dialogs, prompts, and toasts
Learn how to alert users of events that occur on the device, and provide other useful information in your apps.
Invocation framework, BBM

App integration
Launch another application or an external service from within your app.

New: BBM Social Platform
Add social interaction and increase your app's popularity and discoverability by integrating with the BBM Social Platform.
• Invoke other apps from your own!

• Flow seamlessly between applications
  ▶ Provide that awesome experience

• Embed other apps (“viewers”) in your own!
  ▶ Example: Just want to play a media file?
  ▶ Just Invoke the media previewer!
I need to open this .doc file. Anyone?

- I can open .doc files
- I can open .jpeg files
- I am good for noting 😊
I want Dr. Doc to open this .doc file.

I can open .doc files
BBM Framework

• **Increased discoverability:**
  ▶ Reach over **45 million** users worldwide!

• **Increased engagement and app stickiness:**
  ▶ Get the addicting social element!
  ▶ Share / download between users, engage friends

• **Increased monetization:**
  ▶ viral BBM discoverability + increased engagement = *Richer developers* 😊
Storage, App Sandbox, Languages

**Updated: Data storage**
Learn how to access data from external sources, such as an SQL database or a JSON file, and display the data in your apps.

**File system access**
Learn about the application sandbox and how to access shared content on the file system of the device.

**New: Internationalization**
Prepare your application for markets around the world by translating your content and customizing it for a particular locale.
Database Support

Cascades supports both:

- QtSql
- libsql

Easiest to use QtSql in a Cascades app, provided by QtCore
QString dbPath = "data/db.db";
QSqlDatabase database
QSqlDatabase::addDatabase("QSQLITE");
database.setDatabaseName(dbPath);
database.open();

QSqlQuery query(database);
query.prepare("SELECT * FROM table");
query.exec();
Screen and Windowing

The Screen and Windowing library provides a framework that makes it easier to create a user-interface. Learn how to use the API to create screens and windows so that you can display graphics, images and more.

New: Audio / Video

Learn how to create visually astonishing UIs for your mobile apps with fully functioning multimedia support for both audio and video, as well as streams.

some multimedia samples!
Image, Camera, Notifications

**Image**
Learn how to use the API to render images for your apps.

**New: Camera**
Find out how to use the camera to create apps that allow your users to take photos and record video.

**New: Notifications**
Create notifications that highlight events that occur in your app, and let users customize how they want to receive these notifications.
Location
Learn about how Cascades has reimplemented the Qt Mobility Location API to support positioning.

New: Sensors
Gather data from the various sensors on the device, including the accelerometer, magnetometer, and rotation sensor.

Networking
Learn about how Cascades uses the QtNetworking module and how to send and receive data using HTTP communication.
Sensor Subsystems Available

- GPS
- Accelerometer
- Gyroscope
- Magnetometer
- NFC
- Bluetooth
- USB
Where do you Start?

https://developer.blackberry.com/
Getting Started
4 Easy Steps!

Download the Native SDK
developer.blackberry.com/native

Register for Signing Keys

Run the getting started wizard
- Window -> Preferences -> BlackBerry
  Sets up your environment, keys, debug token

Create a new Hello World app and deploy it to your device
Getting Started

Resources

- BlackBerry Native Development
  - http://developer.blackberry.com/native

- BlackBerry Developer Blog
  - http://devblog.blackberry.com

- BlackBerry Jam Sessions (LOTS of slides & videos)
  - http://www.blackberryjamconference.com/americas/content/sessions

- Forums:

- Twitter
  - @BlackBerryDev
Lots of sample apps!
UX Trumps all!

- UX Guidelines For BlackBerry 10

- Key Principles
  - A Cinematic Experience
  - Multitasking as 2nd nature
  - Content is king
  - Fluid Workflow
  - Moments of Charm

One last app: Quotes
Thanks

Shadab Rashid

Application Development Consultant
Developer Relations

@surashid
sh rashid@rim.com