Cross-Platform Native App & Game Development with Qt 5.3 & V-Play

Mobile Developer After-Work
01.07.2014

by Christian Feldbacher, MSc
CEO & Co-Founder V-Play GmbH

About Me

- Experience
  - iOS, Android, Symbian App Development
  - Studied Mobile Computing FH Hagenberg
  - C++ 10+ years
  - Qt 5+ years
- Co-Founder of V-Play GmbH
  - Started in 2009, full-time 2011
  - Founded V-Play GmbH December 2012
  - 5 employees
  - Based in Vienna (Wien Mitte)

Agenda

- Qt Introduction
- QML Features with Live Example
- How to create multi screen apps with Qt
- Native Controls
- Deployment
- V-Play Qt 5 Plugins
- V-Play Game Engine

Qt Introduction

What is so „cute“ about Qt?

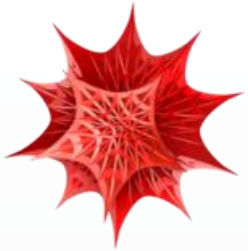
- Cross-Platform C++ Framework
- Develop once, deploy anywhere:
  - PC, iOS*, Android*, WP8**, BB10, …
- Write applications in C++ and/or **QML & JavaScript**

* Fully supported since Qt 5.2 (Dec 13), already since 2012 in V-Play
** Beta support since Qt 5.3 (Apr 14), full support in Oct 14

Some popular Qt applications

BlackBerry

jolla



skype™

Qt

Autodesk® Maya

Qt History

- Qt 1.0, „Trolltech“ 1991
- Qt 4.0 2005
- 2008, Nokia acquires Trolltech
- 2011 Digia acquires Qt from Nokia
- Dec 2012: Qt 5.0
  - Focus on QML (=**Q**t **M**eta **L**anguage, part of Qt Quick)
  - New Scene Graph Renderer
- Dec 2013: Qt 5.2
  - Official iOS & Android Support
- Apr 2014: Qt 5.3
  - Windows Phone & Windows Runtime Beta Support

QML Features by Example

QML FEATURES

Overview

- Declarative
- UI Elements & Positioners
- Property Propagation, Signals & Slots
- Animations
- State Machines
- Performance

QML FEATURES

Declarative

- Much faster to develop: less code ➡ less errors

Real Game Example:

- 60% less code than C++
- 10-20% less code than other scripting languages like Lua

V-Play Code

```
import VPlay 1.0
import QtQuick 1.1

Scene {
  Rectangle {
    color: "red"
    anchors.fill: parent
  }

  Item {
    anchors.right: parent.right
    y: 15
    width: 30
    height: 20

    Rectangle {
      anchors.fill: parent
      anchors.rightMargin: 5
      color: Qt.rgba(0.5, 0.5, 0, 1)
    }
  }
}
```

Cocos2d-x Code

```
#include "cocos2d.h"

// MyScene is another class, where we put the contents into
CCScene *scene = MyScene::scene();

CCLayerColor *layerColor = CCLayerColor::layerWithColorWidthHeig
      ccc4(255, 0, 0, 255),
      scene->getContentSize().width,
      scene->getContentSize().height);
scene->addChild(layerColor);

CCNode *node = new CCNode::node();
node->setContentSize(CCSize(30, 20));
node->setPositionX( node->getParent()->getContentSize().width -
node->setPositionY(15);
layerColor->addChild(node);

CCLayerColor *layerColor2 = CCLayerColor::layerWithColorWidthHei
      ccc4(255*0.5, 255*0.5, 0, 255),
      node->getContentSize().width - 5,
      node->getContentSize().height);
node->addChild(layerColor2);
```

QML FEATURES

QML Live Demo

- Demonstrates all QML Features
- Source code at:
  http://doc.v-play.net/examples-qmldemonstration.html

QML FEATURES

Tricky in the beginning - Example

```
import VPlay 1.0
import QtQuick 1.1

GameWindow {

  Scene {
    id: myScene

    property int counter

    // changed-handler is auto-generated
    onCounterChanged: {
      // imperative
      myTextImperative.text = counter
    }

    signal sceneClicked()
    onSceneClicked: { /* any logic here */ }

    MouseArea {
      anchors.fill: parent

      onClicked: {
        myScene.counter++
        myScene.sceneClicked()
      }
    }

    // these two text elements have the same value
    Text { id: myTextImperative }
    Text {
      id: myTextDeclarative
      text: myScene.counter
    }
  }// Scene
}// GameWindow
```

Id – intra-object, restricted to the same document and parent objects.

Property – inter-object, accessible from outside.

Signal – inter-object, accessible from outside. Read-only.

QML vs C++

Choose your Weapon

QML Strenghts	C++ Strengths
Development Speed, Interpreted	Maximum Performance, Compiled
Less Code -> Less Errors	Best Portability
Custom GUI Styles (**Games**!)	Existing Business Logic?
Animated GUIs, Mobile GUIs	
Mixing with C++ easy, C++ API for JS	
Focus of Qt in future is mobile	
Easy to learn, Designer-Friendly	

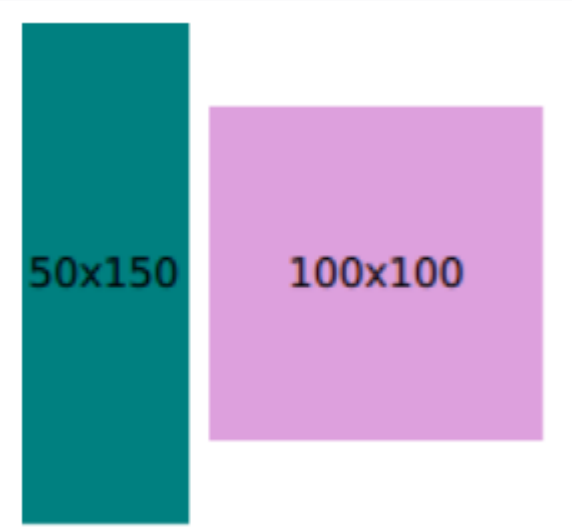
Multi Screen with Qt

Multi Screen with Qt

Layouts

- Positioners with fixed size:
  - Row, Column, Grid
- Flexible Layout elements:

```
RowLayout {
    id: layout
    anchors.fill: parent
    spacing: 6
    Rectangle {
        color: 'teal'
        Layout.fillWidth: true
        Layout.minimumWidth: 50
        Layout.preferredWidth: 100
        Layout.maximumWidth: 300
        Layout.minimumHeight: 150
        Text {
            anchors.centerIn: parent
            text: parent.width + 'x' + parent.height
        }
    }
    Rectangle {
        color: 'plum'
        Layout.fillWidth: true
        Layout.minimumWidth: 100
        Layout.preferredWidth: 200
        Layout.preferredHeight: 100
        Text {
            anchors.centerIn: parent
            text: parent.width + 'x' + parent.height
        }
    }
}
```

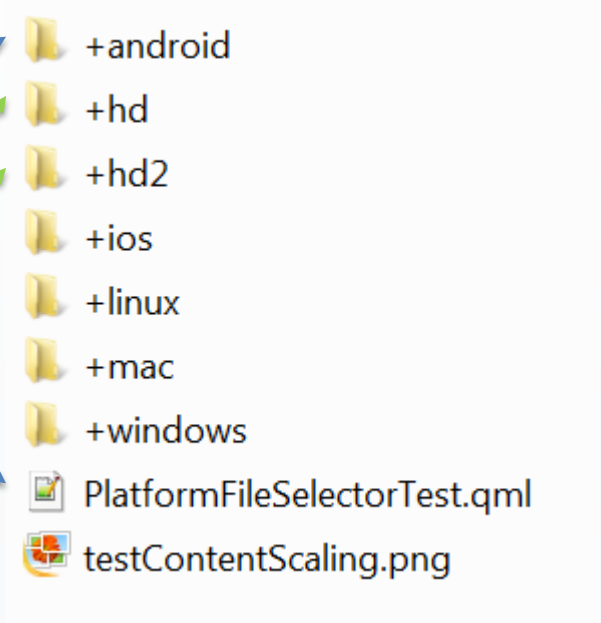


Multi Screen with Qt

File Selectors

- File access generalized across platforms

```
import QtQuick 2.0

Item {

  PlatformFileSelectorTest {}

  Image {
    source: "testContentScaling.png"
  }
}
```



- Works behind the scenes

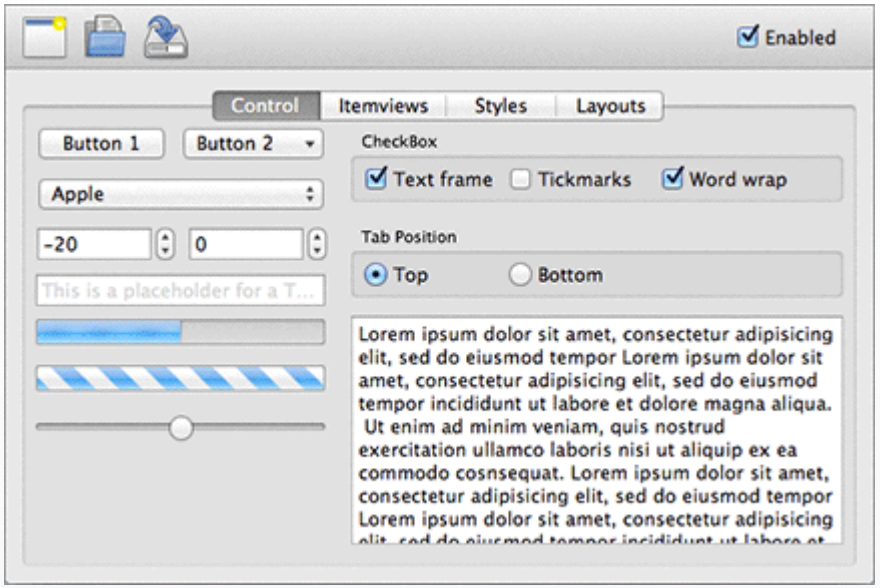
→ Less code & easier to read (no ifdefs)

Native Controls

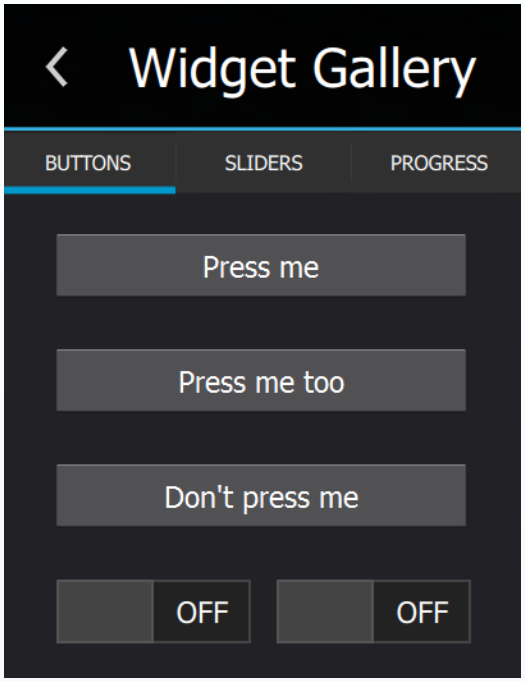
Native Controls

Different Styles

Native Style



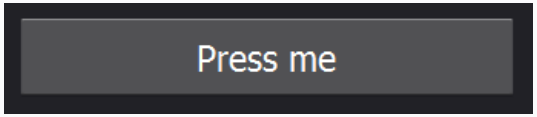
Custom Style



Native Controls Styling

Custom Styling

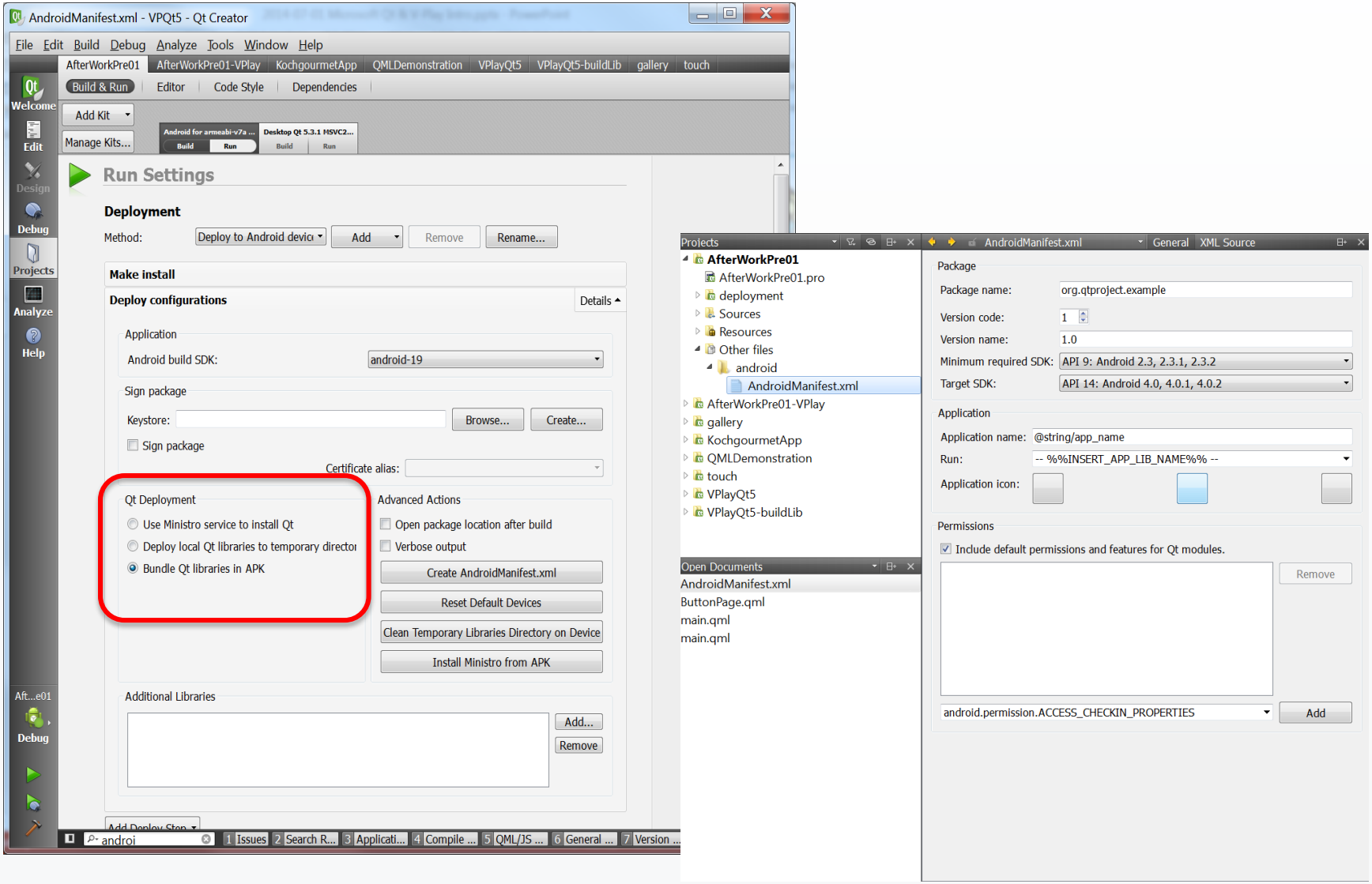
```
import QtQuick 2.2
import QtQuick.Controls 1.2
import QtQuick.Controls.Styles 1.1

Button {
  text: "Press me"
  style: touchStyle
}
```




```
Component {
  id: touchStyle
  ButtonStyle {
    panel: Item {
      implicitHeight: 50
      implicitWidth: 320
      BorderImage {
        anchors.fill: parent
        antialiasing: true
        border.bottom: 8
        border.top: 8
        border.left: 8
        border.right: 8
        anchors.margins: control.pressed ? -4 : 0
        source: control.pressed ? "../images/button_pressed.png"
                                : "../images/button_default.png"
        Text {
          text: control.text
          anchors.centerIn: parent
          color: "white"
          font.pixelSize: 23
          renderType: Text.NativeRendering
        }
      }
    }
  }
}
```

Deployment

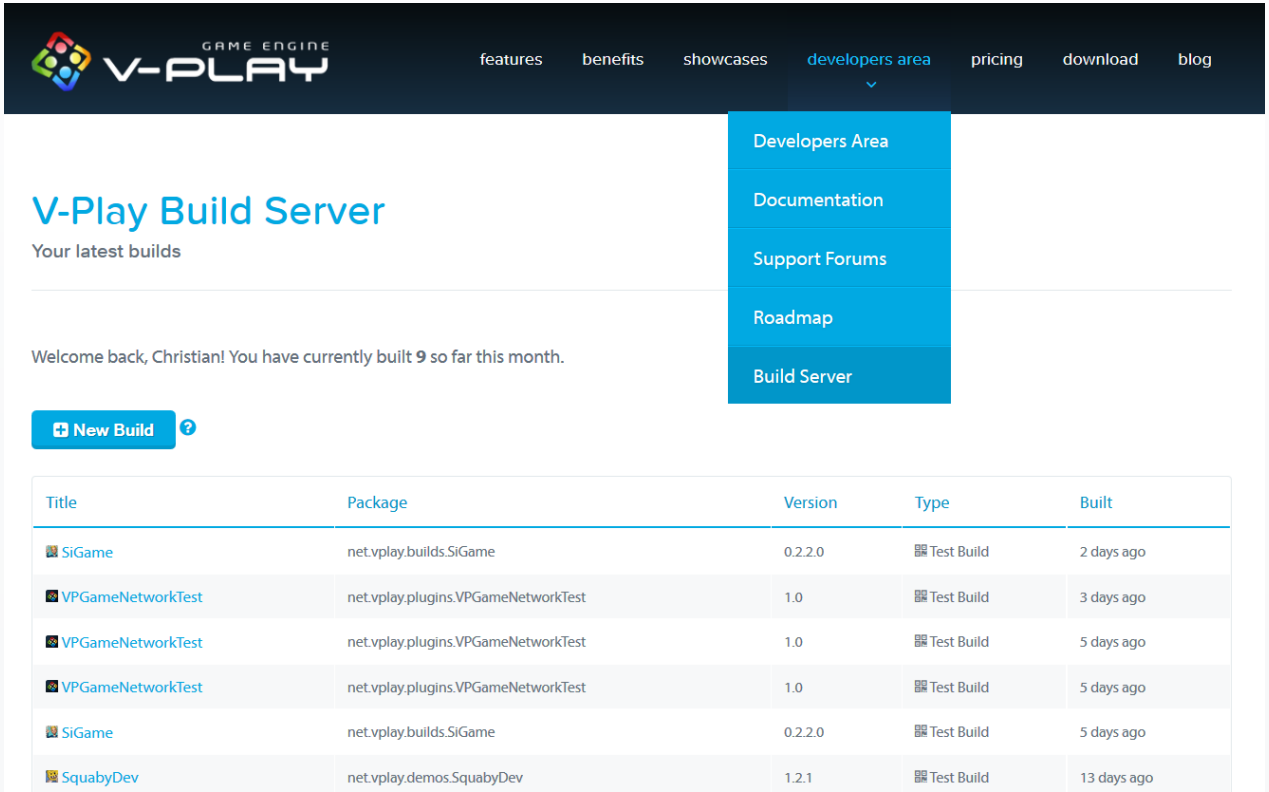
Deployment via Qt Creator



Deployment V-Play Build Server

Build Cockpit

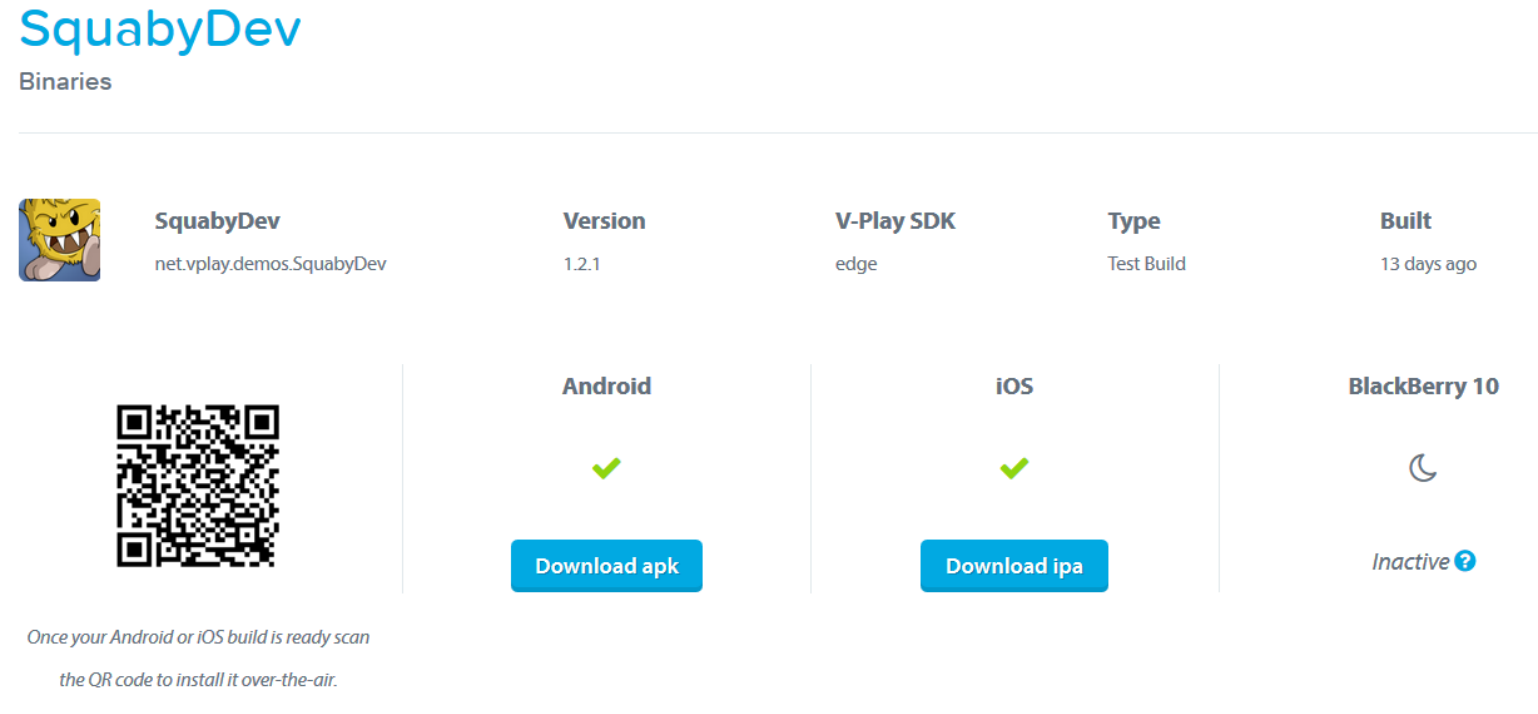
- https://builds.v-play.net
- Register with V-Play username & password
- Upload contents of qml directory with „New Build“



Deployment V-Play Build Server

Project Download

- Download with QR code directly to phone



For iOS devices, send UDID to support@v-play.net

Deployment Summary

- Android:
  a.) Ministro (200 kB app + Ministro app 1 MB)
  b.) Bundle Qt libs into app (~10 MB)
- iOS: statically linked only (~7 MB)
- Win RT: Beta in Qt 5.3, full support with Qt 5.4

1.) Build for Windows Phone 8 in Qt Creator
2.) `C:\Qt\5.3\winphone_arm\bin\qmake.exe -tp vc CONFIG+=windeployqt C:\MyProject.pro`
3.) Open the generated `.vcxproj` file in VS
4.) Start `qtd3dservice.exe` to generate shaders at deployment

The deployment process will improve with Qt 5.4 in October 2014!

More infos: http://blog.qt.digia.com/blog/2014/05/20/qt-weekly-11-getting-started-with-winrt-2/

More Qt Modules

Not covered in detail

- WebKit support, Chromium in Qt 5.4
- Local SQLite Database, persistent storage
- ListModel, remote data parsing JSON or XML

```
import QtQuick 2.0
import QtQuick.XmlListModel 2.0

XmlListModel {
    id: xmlModel
    source: "http://www.mysite.com/feed.xml"
    query: "/rss/channel/item"

    XmlRole { name: "title"; query: "title/string()" }
    XmlRole { name: "pubDate"; query: "pubDate/string()" }
}
```

```
ListView {
    width: 180; height: 300
    model: xmlModel
    delegate: Text { text: title + ": " + pubDate }
}
```

More Qt Modules

Not covered in detail

- WebKit support, Chromium in Qt 5.4
- Local SQLite Database, persistent storage
- ListModel, remote data parsing JSON or XML
- Qt Cloud Services
  - Backend-as-a-Service
  - Backend code in Qt, Node.js or Ruby
- Multimedia (Audio, Video)
- Particles, Shader effects, 3D models

Qt for Apps Recap

Key Benefits:

+ Rapid Development (QML+JS, same look guarantee)
+ High Performance (C++ internally, scene graph)
+ Great for custom UIs & animation-heavy apps
+ Native connection with C++/Obj-C easy
+ Qt is free & open-source (LGPL)! *

Gotchas:

- Size overhead 5-10 MB (Ministro not user friendly)
- No predefined native control style for iOS, Android, WP8 yet (coming in October 2014 with Qt 5.4)

* commercial Qt license costs 109€/month per developer - includes all mobile platforms (esp. useful for iOS to avoid publishing object code)

V-Play Qt 5 Plugins

Why QML Plugins?

The V-Play history

2010
QML / Qt Quick
Wholy shit
That's cool!!!

2011
Game prototype
For iOS & Android
QGraphicsView

2011
Custom cocos2d-x
Renderer beginning

2012
V-Play Launch
Ready to publish high performance
iOS & Android games

2013
QML Plugins
For apps & games

Facebook Plugin

Example: Posting to the user wall

Use-Cases:

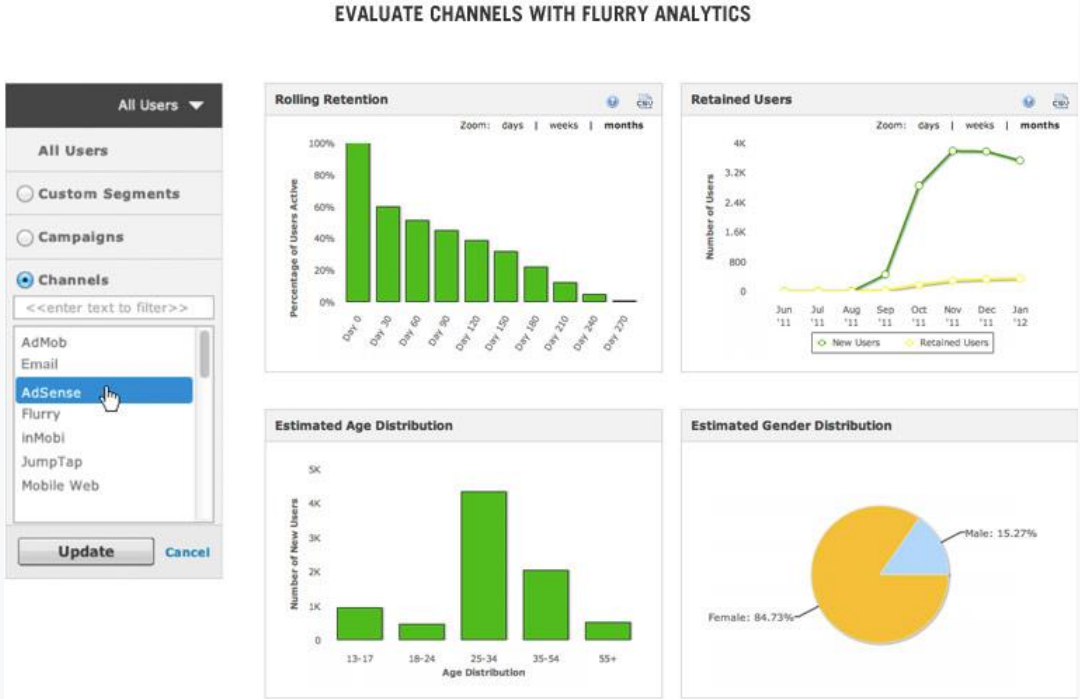
Like App, Post to User Timeline, Invite friends to app, Highscores & achievements, graph requests, get email

```
import QtQuick 1.1
import VPlay 1.0
import VPlay.plugins.Facebook 1.0

GameWindow {
  Scene {

    Facebook {
      id: facebook
      appId: "YOUR_APP_ID"

      readPermissions: ["email", "read_friendlists"]
      publishPermissions: ["publish_actions"]

      onSessionStateChanged: {
        if (sessionState === Facebook.Opened) {

          facebook.openWallDialog( {
            "link" : "www.v-play.net",
            "name" : "V-Play name!",
            "caption" : "V-Play caption ...",
            "description" : "This is the description" })
        }
      }

      Component.onComplete: {
        facebook.openSession();
      }
```

Flurry Analytics

Measure & improve your app

```
import VPlay 1.0
import VPlay.plugins.flurry 1.0

GameWindow {
  Scene {

    Flurry {
      id: flurry
      applicationKey: "MyQtApp123"
    }

    SimpleButton {
      onClicked: flurry.logEvent("Button1.Clicked")
    }

  }
}
```



Ad Plugins

For Chartboost & AdMob

```
import QtQuick 1.1
import VPlay 1.0
import VPlay.plugins.admob 1.0

GameWindow {
  Scene {

    AdMob {
      publisherId: "<your-publisher-id>"
      anchors.bottom: parent.bottom
      banner: AdMob.Smart
      testDeviceId: "<your testdevice id>"
    }

    Text {
      text: "Hello Smartbanner!"
      color: "green"
      anchors.centerIn: parent
    }

    SimpleButton {
      text: "Back"
      anchors.right: parent.right
    }
```

In-App Purchases

The futurue of app monetization

```
import VPlay 1.0
import VPlay.plugins.store 1.0

GameWindow {

 Store {
   nonConsumables: [

     NonConsumableGood {
       id: noadsGood
       itemId: "noads_id"
       purchaseType: StorePurchase {
          id: noAdPurchase
          productId: noadsGood.itemId
       }
     }

   ]
 }
}
```

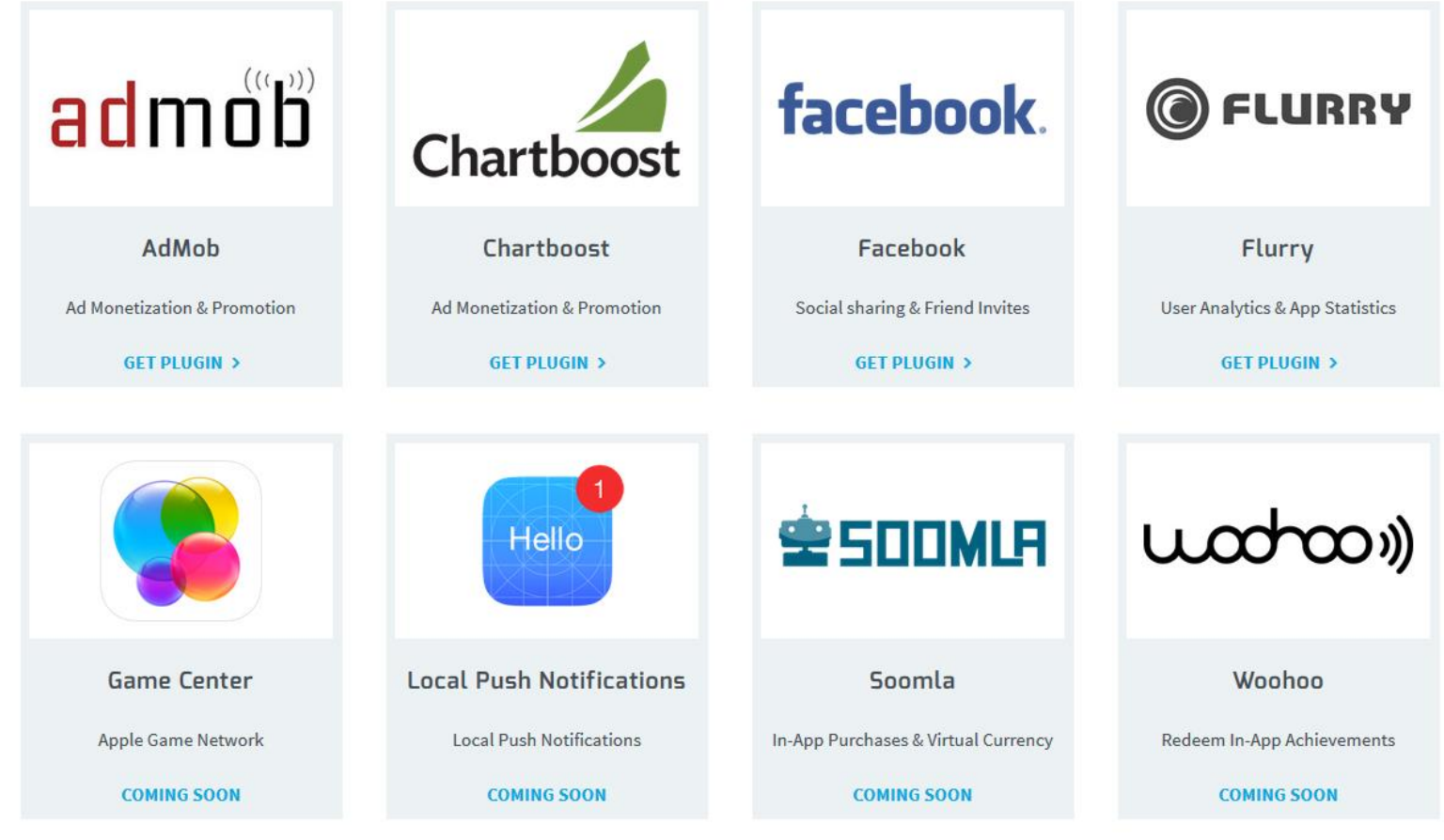
```
AdMob {
  visible: !noadsGood.purchased

  publisherId: "<your-publisher-id>"
  anchors.bottom: parent.bottom
  banner: AdMob.Smart
}
```

V-Play Game Network

Make your users return to your app

- *GameCenter on steroids* made for/with Qt
- True cross-platform:
  - Leaderboards
  - Achievements
  - Challenges
  - Cloud storage & syncing
- Deep Facebook connection
- Full offline support
- Skinnable
- Add Gamification also for „normal apps“

V-Play Plugins Recap



Available on http://plugins.v-play.net
100€ per plugin per app, Samples on GitHub

Qt Technology Partner

GAME ENGINE

V-PLAY

cross-platform 2D game development
in days

V-Play Game Engine Highlights

http://www.youtube.com/watch?v=zfvjxcaU-3Q

V-Play Level Editor & Level Sharing

Highly engage users with user generated content

- Save time in internal development
- Let users create levels
- Social Sharing & Rating of Levels → Community
- Monetize content packs with IAP
- Customizable Look

Add **app store for user generated content** to your own apps/games easily.

Examples (in App Stores):

Squaby

Stack And Friends

Particle Editor

Squaby Tower Defense

Create your own Mincecraft

http://www.youtube.com/watch?v=F6xjxQz_ehI

V-Play Game Engine Recap

- Rapid development (QML, sample games)
- Balance game & create levels at runtime
- User Generated Content in your games
- True Cross-Platform social connection (Facebook, leaderboards, achievements)

Starter License from 20€ per month

Includes all platforms, unlimited games

Contact

V-Play Management Team

Christian Feldbacher
Co-Founder
chris@v-play.net

Alex Leutgöb
Co-Founder
alex@v-play.net

V-Play GmbH
Kolonitzgasse 9/11-14
A – 1030 Wien

http://v-play.net
@vplayengine
http://www.facebook.com/vplayengine

References

Awesome Qt Videos & Most Useful Links

Videos:

- V-Play Engine Highlights video: http://www.youtube.com/watch?v=zfvjxcaU-3Q
- Squaby Trailer: http://www.youtube.com/watch?v=F6xjxQz_ehl
- Qt 5 Release Highlights video: http://www.youtube.com/watch?v=vhWS_bN-T3k
- Cool live-effects with shaders & Video: http://www.youtube.com/watch?v=SMRln8FJvKc
- Livecoding video shader effects: http://www.youtube.com/watch?v=P4kv-AoAJ-Q
- Qt Quick Controls video: http://www.youtube.com/watch?v=_6_F6Kpjd-Q
- Qt Quick Designer Intro: http://www.youtube.com/watch?v=cOViDcYWNCI

Documentation:

- List of all QML components: http://qt-project.org/doc/qt-5/qtquick-qmltypereference.html
- Overview of all Qt modules: http://qt-project.org/doc/qt-5/qtmodules.html
- Qt Quick Controls: http://qt-project.org/doc/qt-5/qtquickcontrols-index.html
- V-Play Documentation, Sample Games & Tutorials: http://v-play.net/doc/