KIRIGAMI

Qt day

Riccardo Iaconelli

Who am I?

Open Source Project Leader

Qt day

Who am I?

Mozilla Tech Speaker

Founder @ WikiToLearn

KDE developer since....

wikitolearn

moz://a

Qt day

Who am I?

Open Source Developer since 2004

KDE Translations, PIM, Amarok, Plasma...

KDE 4.0

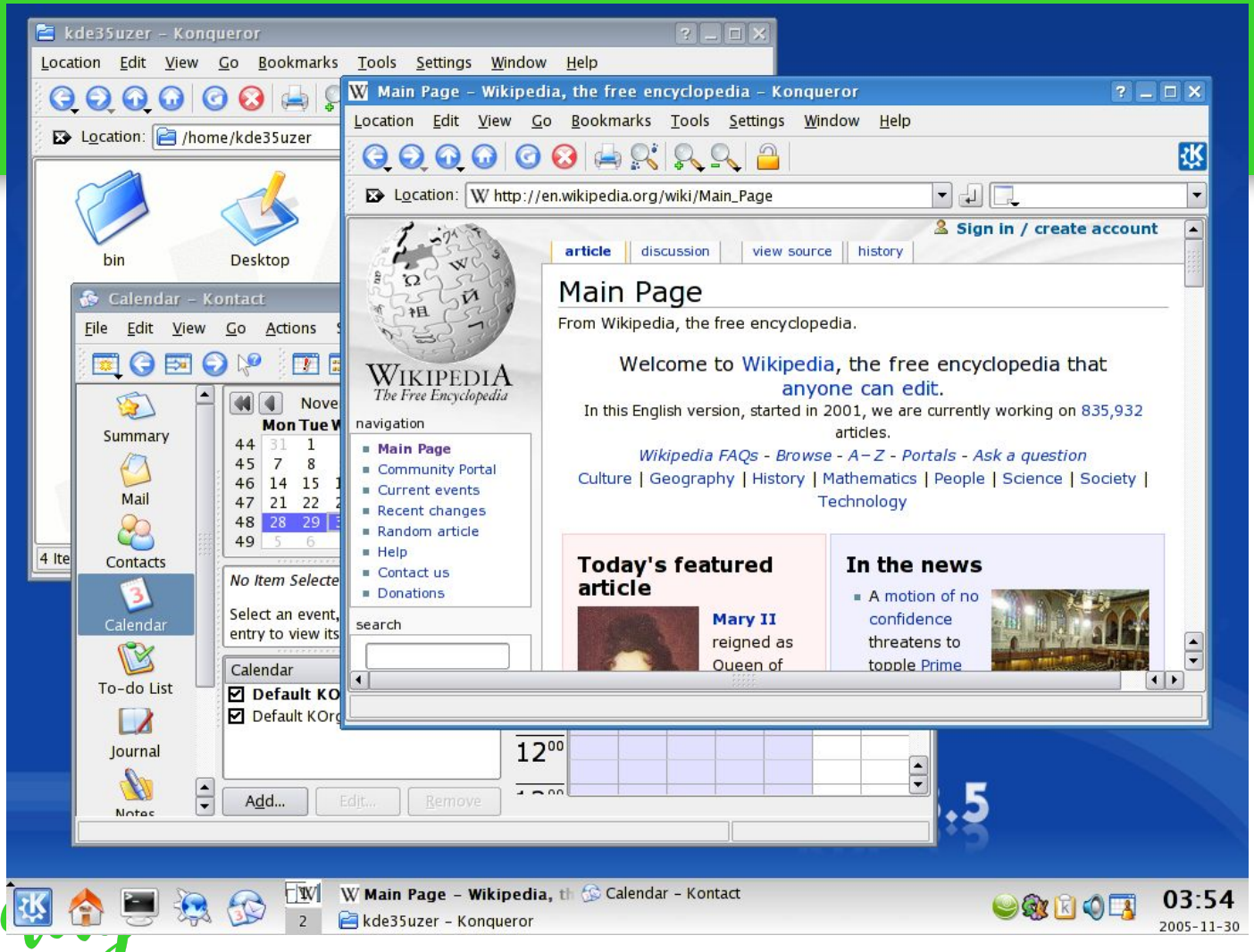
Qt day

in 2005

Qt day

in 2005

Qt day


kde35uzer – Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: /home/kde35uzer
bin
Desktop
Calendar – Kontact
File Edit View Go Actions
Summary
Mail
Contacts
Calendar
To-do List
Journal
Add...
Edit...
Remove
Main Page – Wikipedia, the free encyclopedia – Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: http://en.wikipedia.org/wiki/Main_Page
Sign in / create account
article discussion view source history
Main Page
From Wikipedia, the free encyclopedia.
Welcome to Wikipedia, the free encyclopedia that anyone can edit.
In this English version, started in 2001, we are currently working on 835,932 articles.
Wikipedia FAQs - Browse - A–Z - Portals - Ask a question
Culture | Geography | History | Mathematics | People | Science | Society | Technology
WIKIPEDIA
The Free Encyclopedia
navigation
Main Page
Community Portal
Current events
Recent changes
Random article
Help
Contact us
Donations
search
Today's featured article
Mary II reigned as Queen of
In the news
A motion of no confidence threatens to topple Prime
Main Page – Wikipedia, th
Calendar – Kontact
kde35uzer – Konqueror
03:54
2005-11-30

2005: Plasma Manifesto

A New Beginning

When Apple unleashed the Macintosh in 1984 the desktop as we know it became mainstream. Two decades later and we still have essentially the same desktop concepts: file and applications icons on the desktop, a separate bar for menus and launchers. Incremental improvements have been made to these metaphors, such as the addition of the taskbar in the 1990s. But very little has actually changed.

During this same period of time the Internet has become an integral part of business and even many people's personal lives. Computing is network-centric and multimedia is no longer a buzzword, it's the norm. Movies, music, instant messaging, workgroup collaboration and dazzling graphics all make up the modern computing experience.

And yet our desktop icons still sit there much as they did back in 1984. Our desktop panels remain essentially unchanged since the 1990s. More and more users are abandoning the desktop as a place to put icons as our needs outstrip what they can provide.

It is time that the desktop caught up with modern computing practices and once again made our lives easier and more interesting when sitting in front of the computer. Just like those icons did for people back in 1984.

Development of KDE4 has just begun, and it is during these major release cycles that we have the opportunity to retool and rethink our applications and environment at the fundamental level. The fact that the current desktop concepts have lasted this long is a testament to their effectivity, and we should not simply abandon all sense of the familiar and the useful. Yet we can not stay where we are either.

This, then, is the goal and mandate of Plasma: to take the desktop as we know it and make it relevant again. Breathtaking beauty, workflow driven design and fresh ideas are key ingredients and this web site is your portal onto its birth.

Qt day

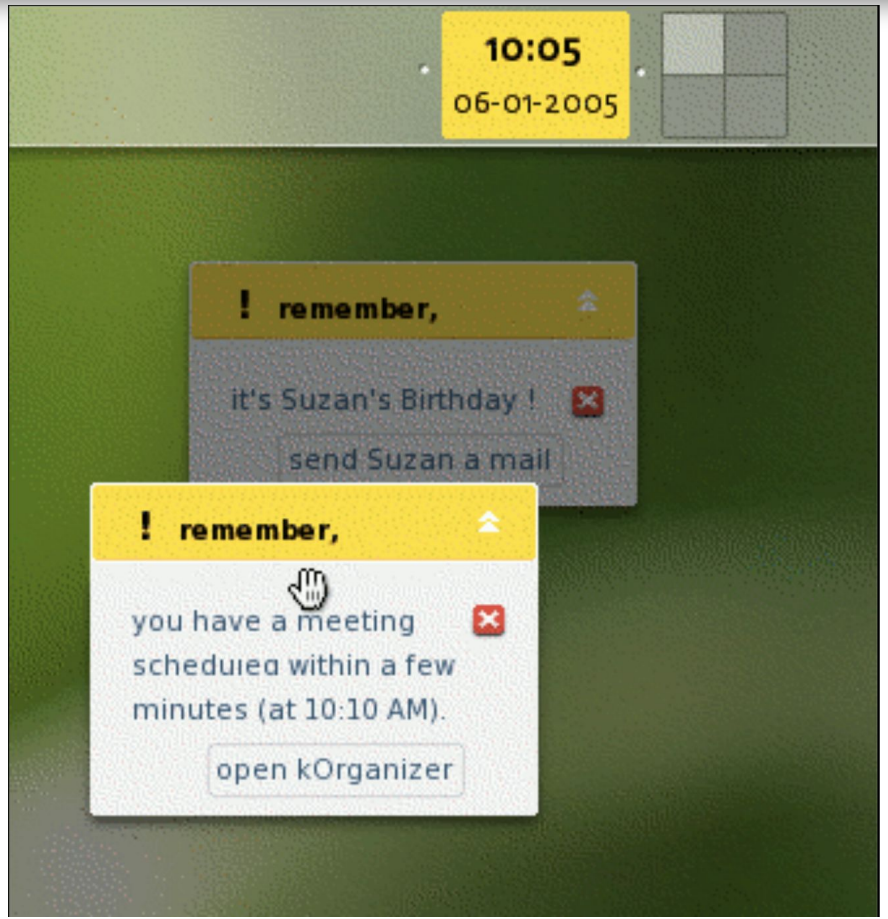
Re-invent your desktop

Applets

Extenders

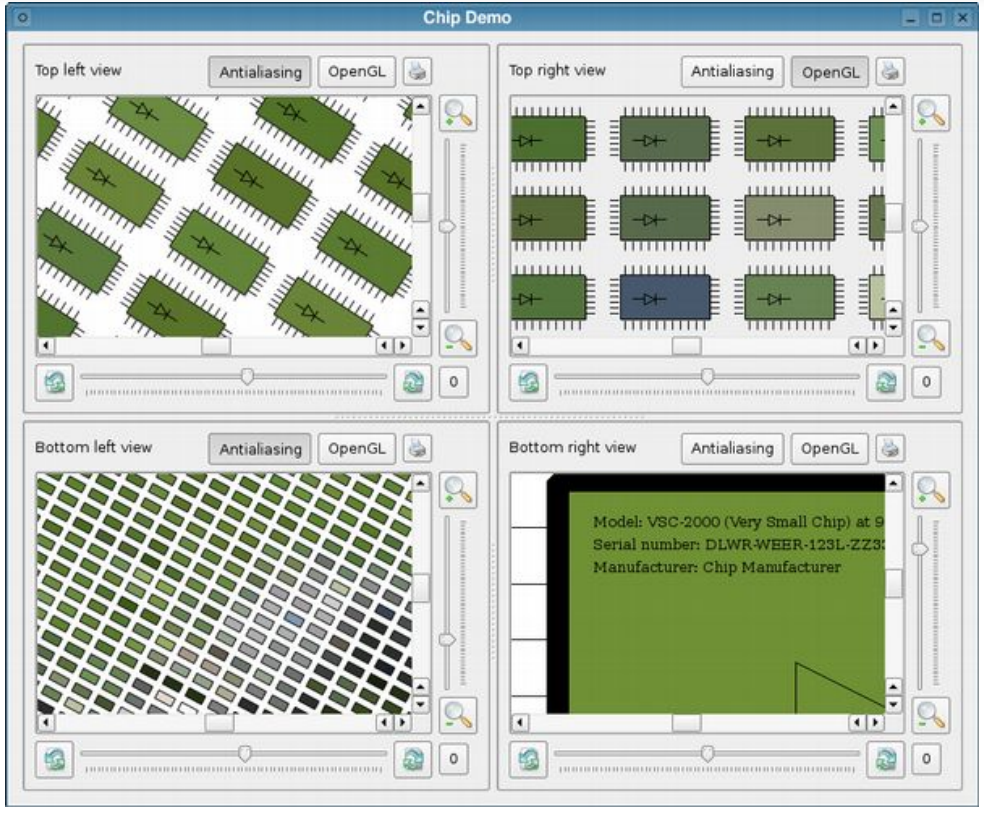
Make use of your desktop space!

...back to 2007



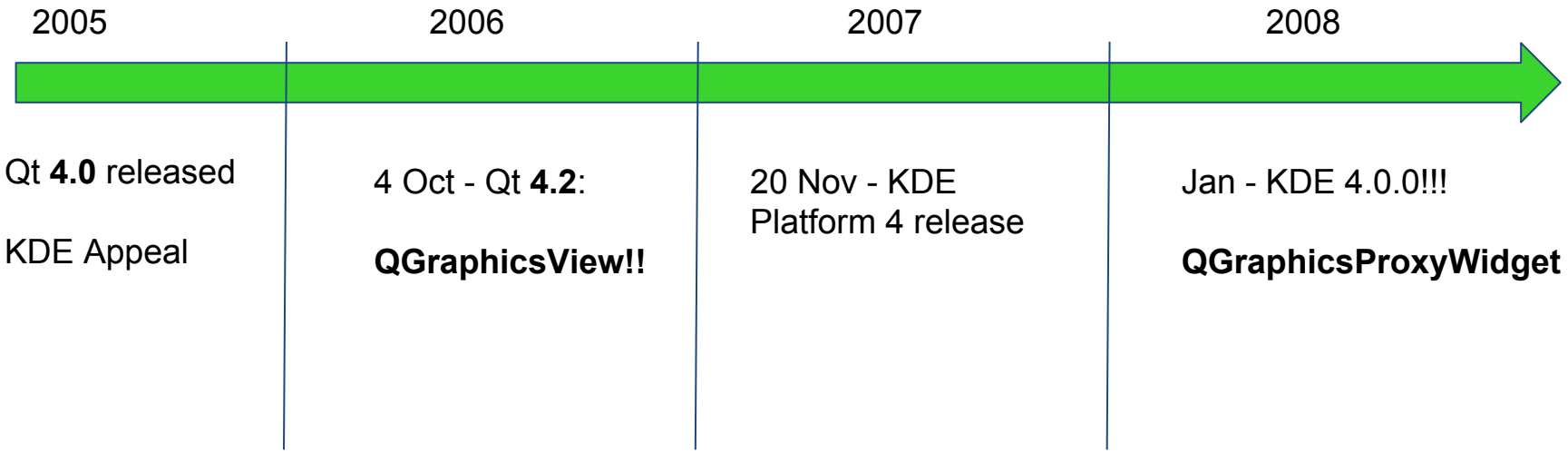
Qt day

Can you remember?



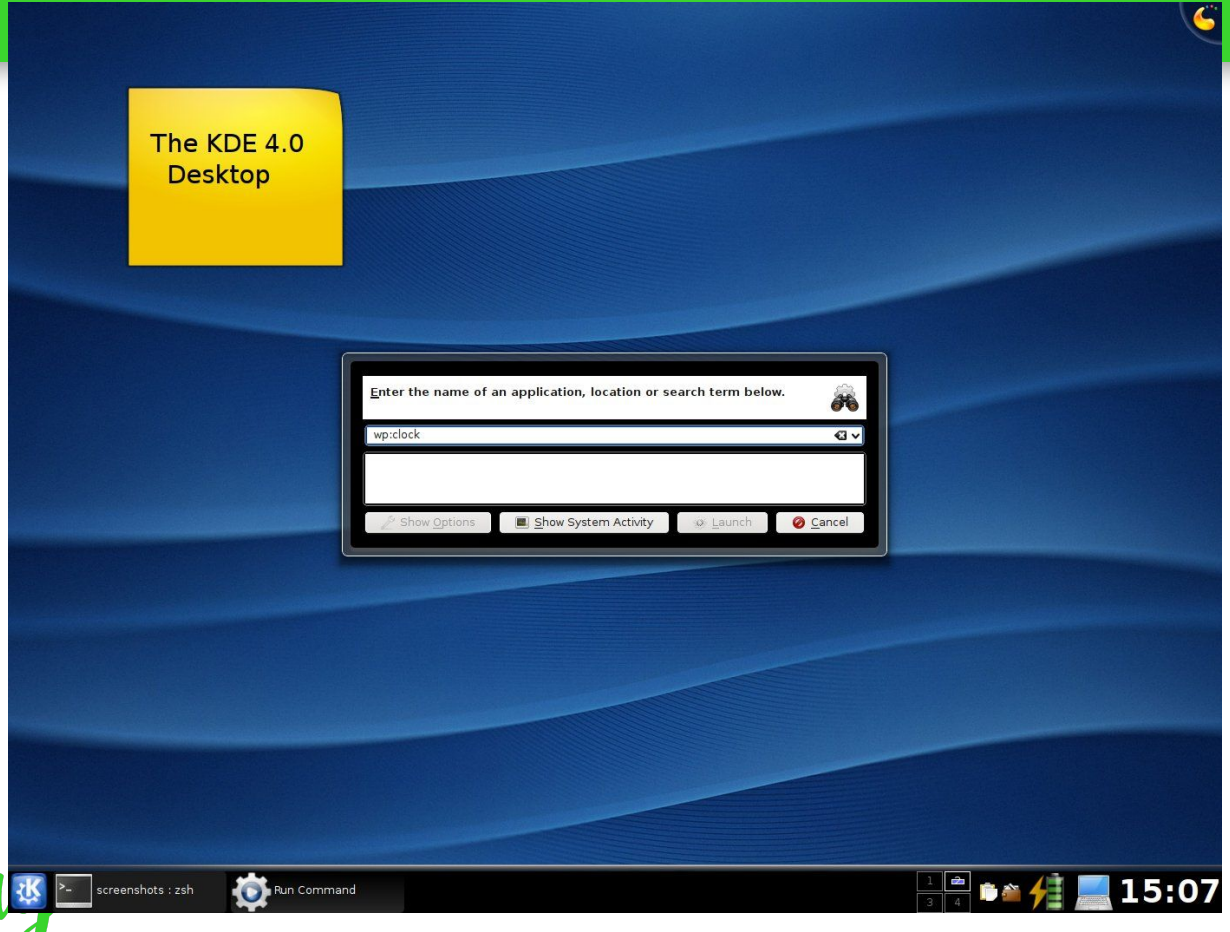
Qt day

Technologies



Qt day

KDE 4.0?


The KDE 4.0 Desktop
Enter the name of an application, location or search term below.
wp:clock
Show Options
Show System Activity
Launch
Cancel
screenshots : zsh
Run Command
15:07

Qt day

KDE 4.0?

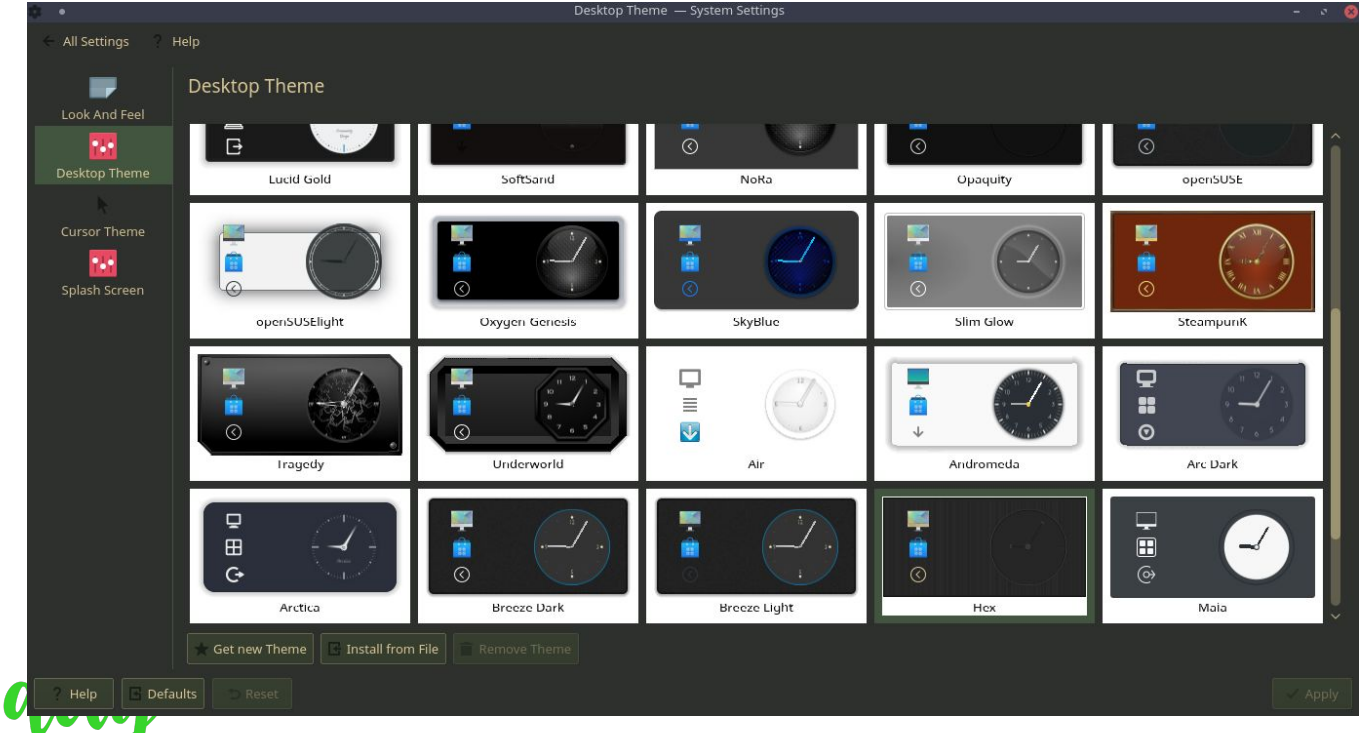
We broke almost all of the stack

Graphics drivers were a pain

X11 compositing anyone?

KDE 4.0?

Consistent theming throughout the desktop



What problems did we find with QGV?

Only really basic items

QGraphicsItems are not QWidgets!

QGraphicsProxyWidget is.... UGH!

...but still

C++ applets were OK

Support for scripting (big <3 KDE Libs)

De-coupling the visualizazations: **DataEngines** (remote!)

...most importantly

REALLY REALLY SLOW!

(depends what you do)

< 10-15 fps

...most importantly

muenalan

October 13, 2010 at 11:42

Will there be a point where I will be laughed at.. because all my QGraphicsView/-Item code became legacy ?

Qt day

Qt day

What is QML

A new way to create user interfaces

Just declare... we will take care of the rest

FAST! > 60 fps!

What is QML

```
Rectangle {
 opacity: 0.5
 color: "red"
 width: 100; height: 100
 Rectangle {
 color: "blue"
 x: 50; y: 50; width: 100; height: 100
 }
}
```

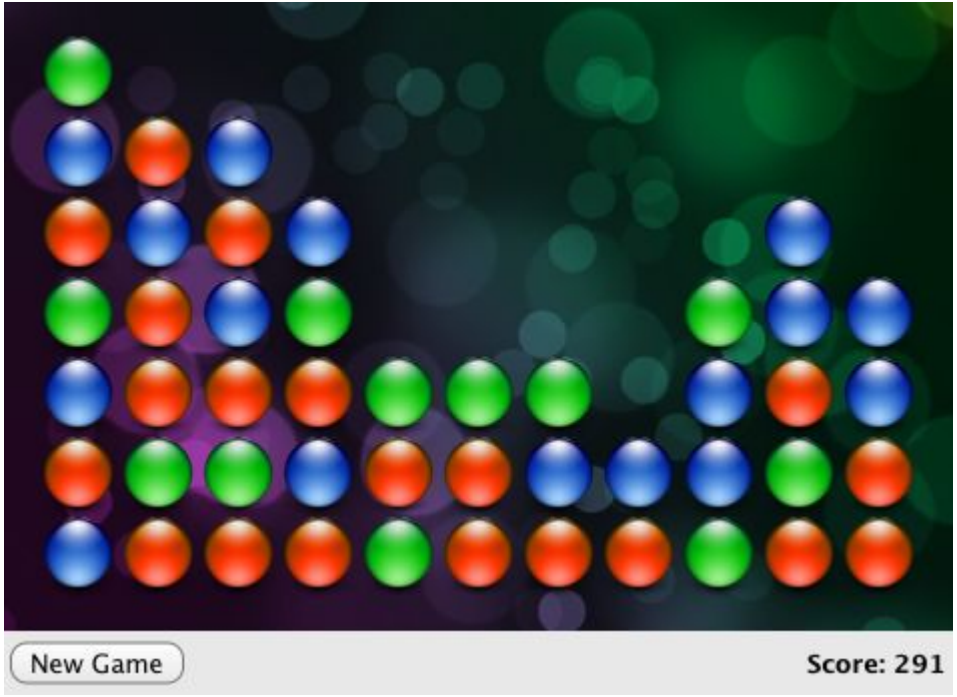
Qt day

What is QML

Rectangle
Text
Image
MouseArea, ...

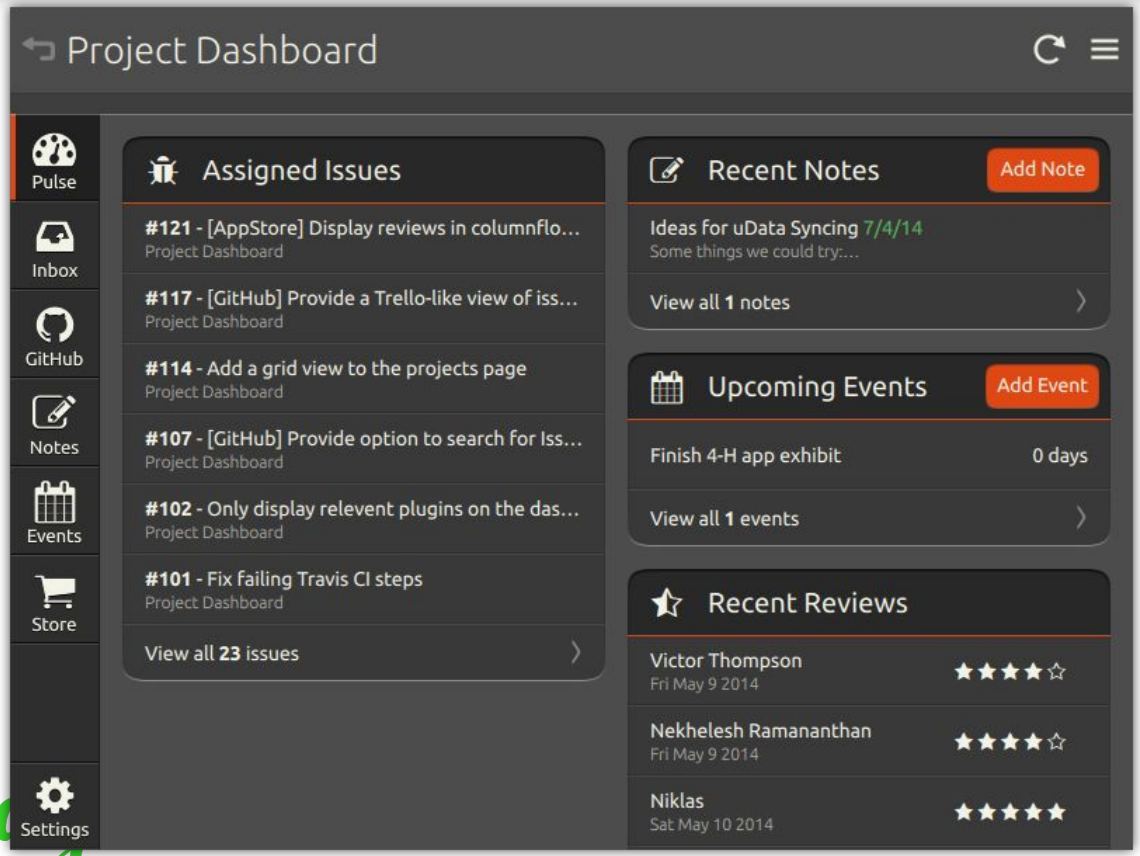
There are no “components”

What is QML



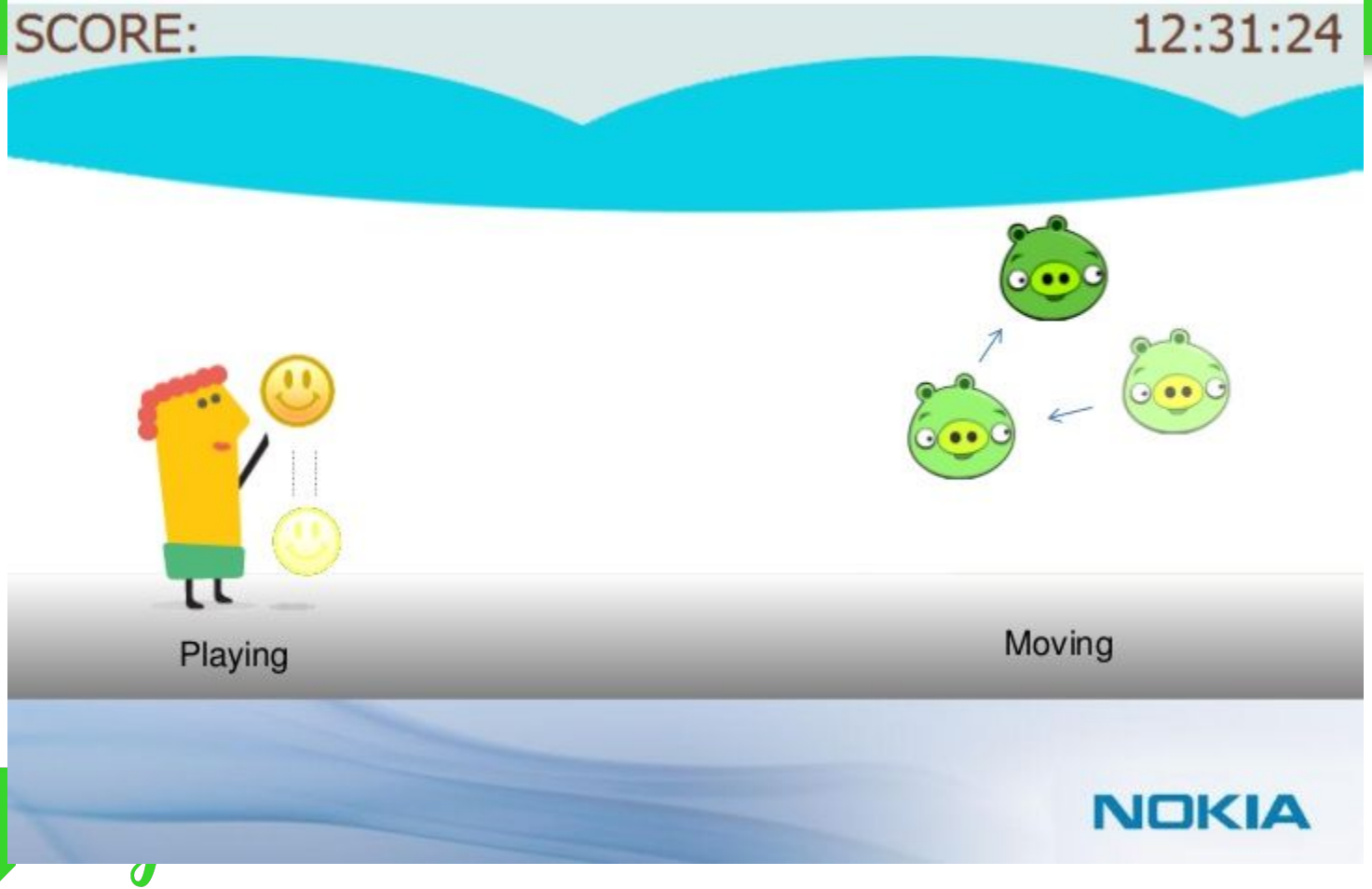
Qt day

What is QML


Project Dashboard
Pulse
Inbox
GitHub
Notes
Events
Store
Settings
Assigned Issues
#121 - [AppStore] Display reviews in columnflo...
Project Dashboard
#117 - [GitHub] Provide a Trello-like view of iss...
Project Dashboard
#114 - Add a grid view to the projects page
Project Dashboard
#107 - [GitHub] Provide option to search for Iss...
Project Dashboard
#102 - Only display relevent plugins on the das...
Project Dashboard
#101 - Fix failing Travis CI steps
Project Dashboard
View all 23 issues
Recent Notes
Add Note
Ideas for uData Syncing 7/4/14
Some things we could try:...
View all 1 notes
Upcoming Events
Add Event
Finish 4-H app exhibit
0 days
View all 1 events
Recent Reviews
Victor Thompson
Fri May 9 2014
Nekhelesh Ramananthan
Fri May 9 2014
Niklas
Sat May 10 2014

Beautiful **Developer** Design

Game: Angry Developer

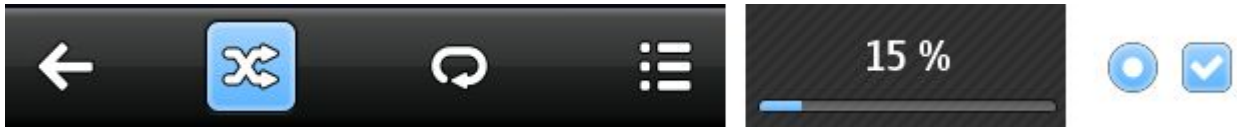


NOKIA

Qt

QtQuickControls! (v1)

Nokia created Symbian/Meego Components



Qt day

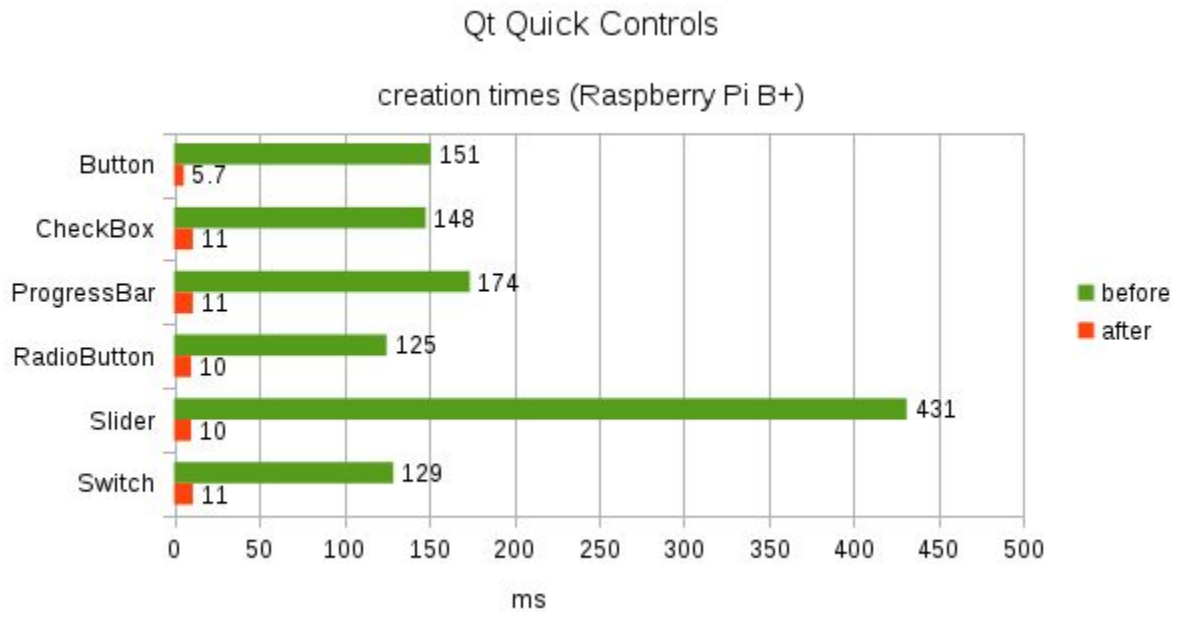
QtQuickControls 1

Everything implemented with QML + Javascript

... everything ...

QtQuickControls 2

Everything re-implemented in C++

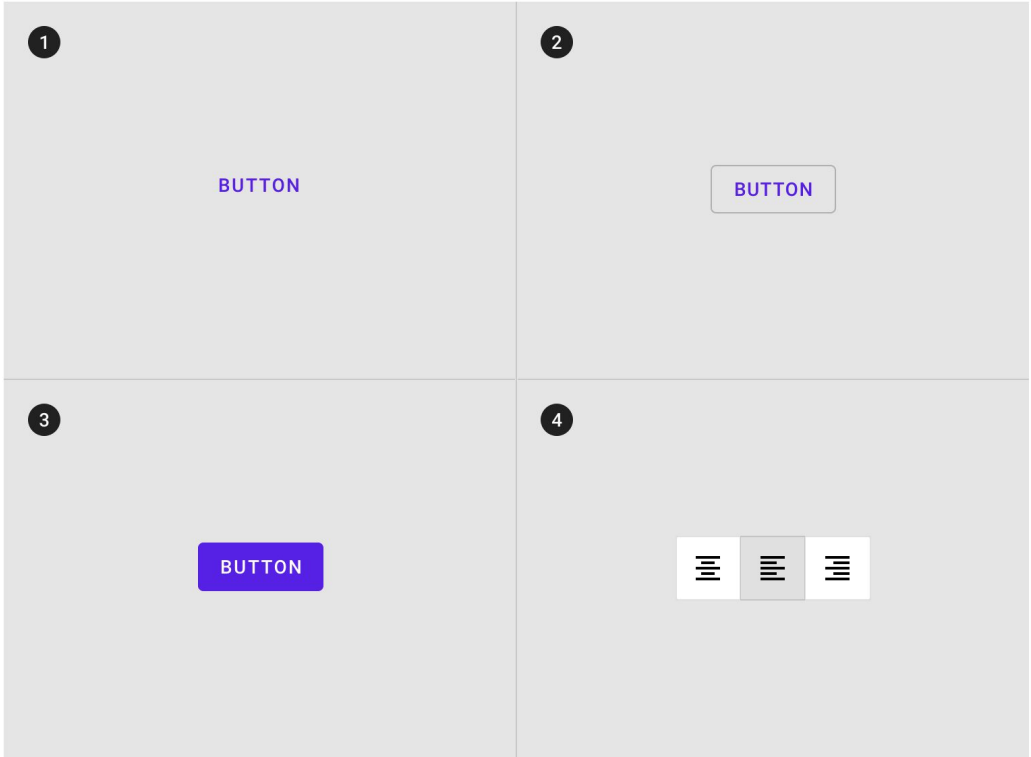


Qt day

Material Design

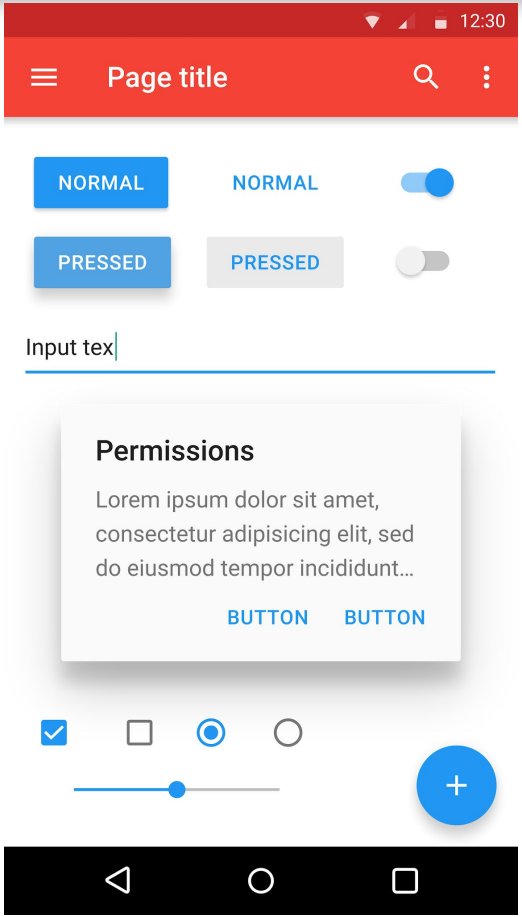
Qt day

Material Design



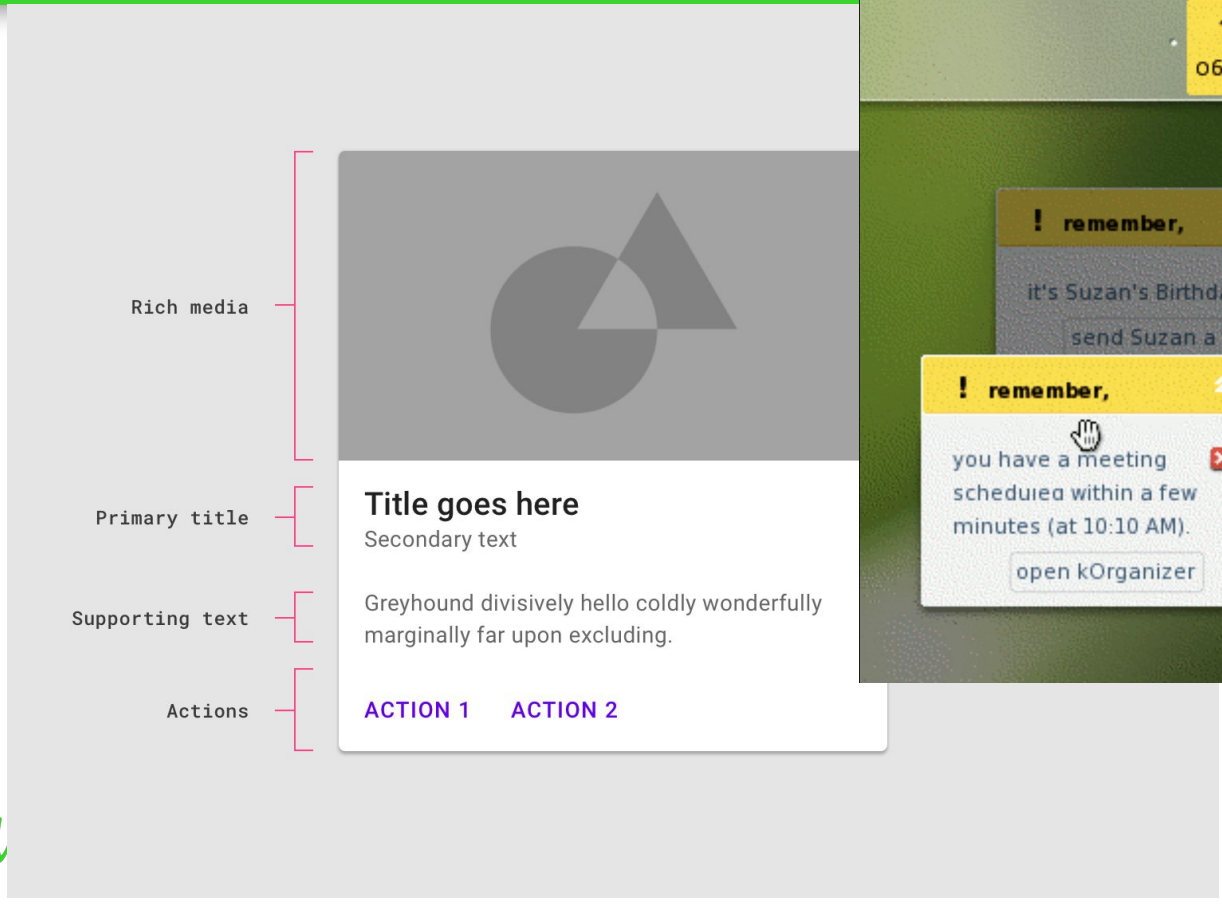
Qt day

Material Design

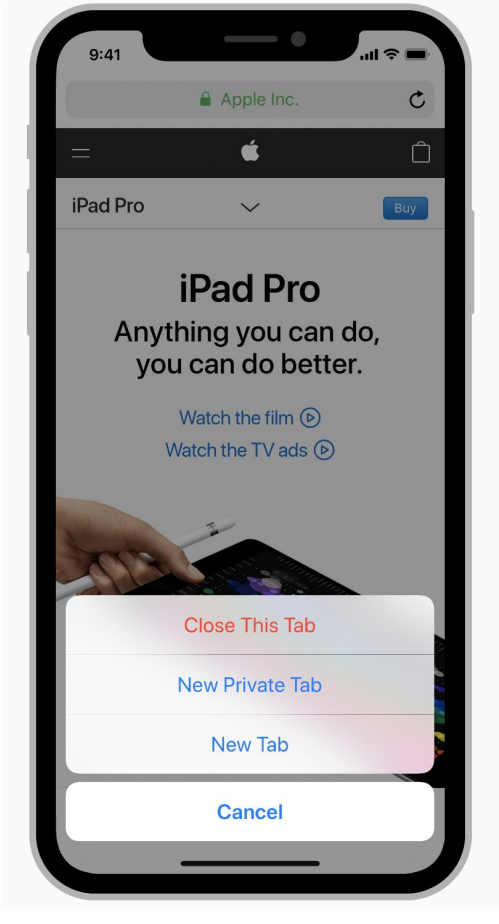


Qt day

Material Design: cards



Apple Design Guidelines



Qt day

KIRIGAMI

Qt day

Kirigami UI

Bring the same abstraction and complexity on top of components to Qt and the “free world”

Inqlude

inqlude.org

#inqlude

The Qt library archive

Qt day

KDE Frameworks

Every Framework has a **Tier** and a Type

Tier 1: No other dependency other than Qt

(Kirigami is Tier 1)

Design phase, HIG work in progress

- Based upon Design guidelines of visual Design Group
  https://hig.kde.org

- The fastest way to have consistent apps with the HIG is to have pre-made GUI components/controls
- You can find the controls in kirigami.git
- Plan: make it a tier-1 framework
- Multiplatform: has to work on Desktop Linux, Plasma Mobile, Android, Ubuntu Phone, Windows, iOS...

Qt day

Basic assumptions behind the design

- Convergence with optimized UIs
- Focus on content consumption instead of creation
- Give as much space to content as possible
- Optimize for one-handed use
- Optimize for hierarchically organized content

What Kirigami is NOT about

What Kirigami is NOT about:

- Bottons, checkboxes etc:
- Not in topic, use QQuickControls2

ApplicationWindow

- Base class for applications
- Derives from QQC ApplicationWindow
- Implements some of the central UI elements
- Properties for Drawers
- Passive notifications
- The page navigation looking like a scrollable row (our main way to go back between pages is trough gestures, very effective except corner cases)

Page

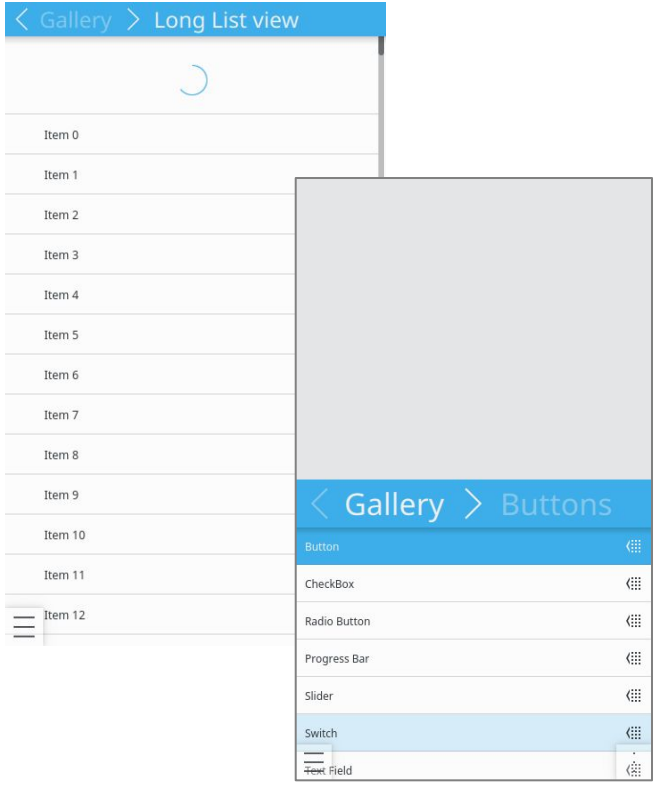
- The application is divided in “pages”
- The class sets a standard behavior
- Exposes “actions” properties
  - main: main Primary Action Button action
  - left: optional smaller button at left of Primary Action Button
  - Right: optional smaller button at right of Primary Action Button
  - contextualActions: list that will go in the contextual drawer

Page

```
Page {
    id: page
    title: "Page Title"

    actions {
        main: Action {
            iconName: : "document-edit"
            text: "Main Action Text"
            onTriggered:...
        }
        left: Action {...}
        right: Action {... }
        contextualActions: [
            Action {
                text:"Action for buttons"
                iconName: "bookmarks"
                onTriggered: ...
            },
            Action {
                text:"Disabled Action"
                iconName: "folder"
                enabled: false
            }
        ]
    }
//Page Contents...
```

ScrollingPage

- Many Pages on mobile devices are scrollable controls, either lists of items or just complicated layouts that don't fit in the screen
- ScrollingPage manages that without the need of explicitly including Flickables
- Supports the popular “pull down to refresh” gesture present in many mobile apps
- By overpulling, the whole UI gets dragged down making it reachable by thumb



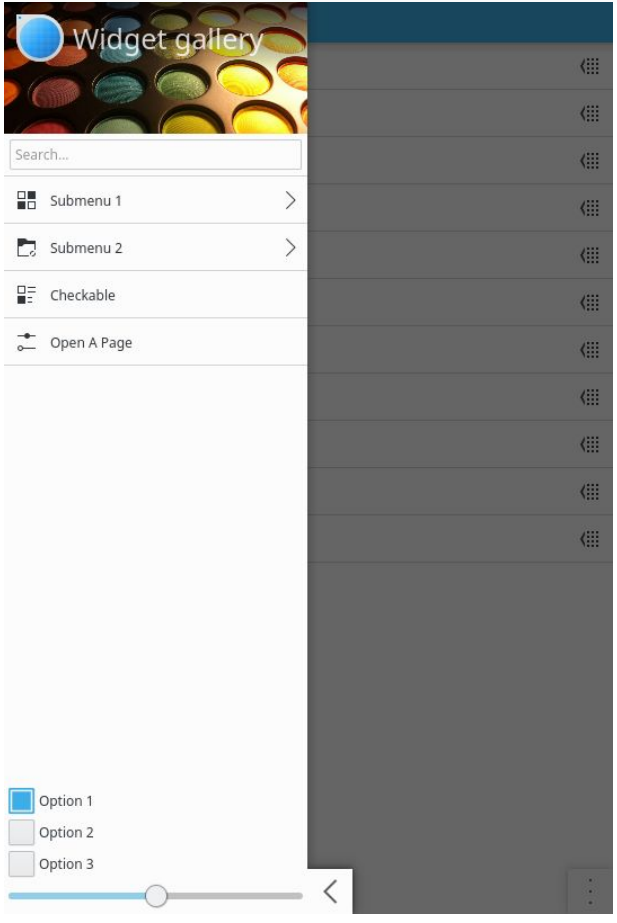
Qt day

Drawers

- ApplicationWindow supports two drawers that will be overlaid on top of the application contents
- Base class: OverlayDrawer
- It's empty, can be used everywhere
- It can have 4 orientations: left, right, top, bottom
- API compatible with QQuickControls2 Drawer
- Edge slide from left and right, and from bottom on platforms that have sides reserved for the system (Ubuntu, Windows 8-10)

GlobalDrawers

- Edge slide from left (screenedge or Primary Action button)
- Big pretty title
- Navigable menu, like a menubar
- Bottom and top areas to put arbitrary controls



Qt day

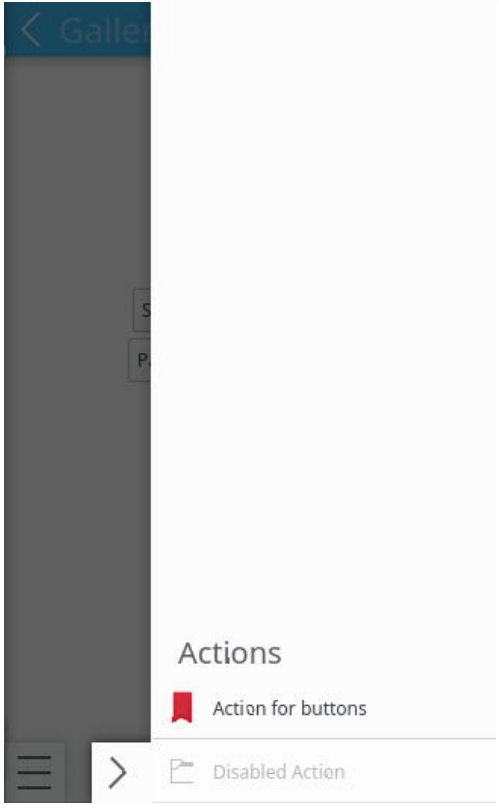
GlobalDrawer

```
Kirigami.ApplicationWindow {
    globalDrawer: Kirigami.GlobalDrawer {
        title: "Widget gallery"
        titleIcon: "graphics"
        bannerImageSource: "banner.jpg"

        actions: [
            Kirigami.Action {
                text: "Submenu 1"
                iconName: "view-list"
                Kirigami.Action {
                        text: "Action 1"
                        OnTriggered: ...

                }
                Kirigami.Action {...}
                Kirigami.Action {...}
            },
            Kirigami.Action {
                text: "Checkable"
                checkable: true
                onChackedChanged: ...
            }
        ]

        //Arbitrary content
        Item {...}
    }

    //Application content
}
```

Qt day

ContextDrawer

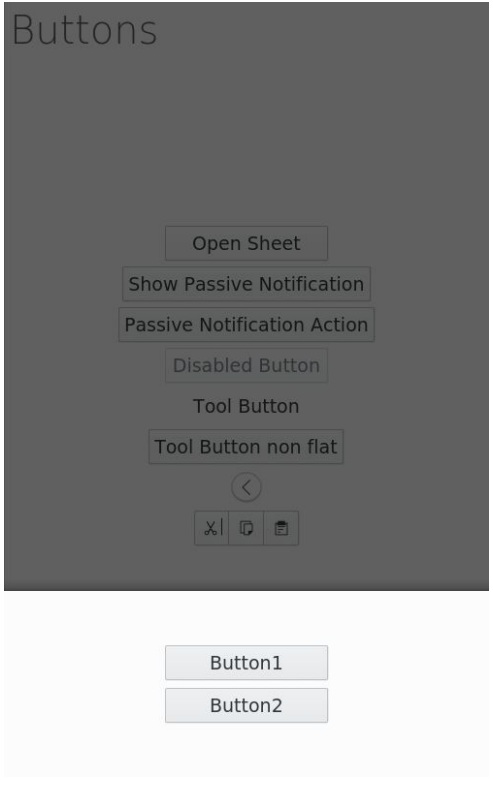
- Edge slide from right (screen edge, Action Button or bottom-right gesture)
- Actions that depend from the current app page: context-dependent
- Bottom-aligned to be thumb-friendly



Qt day

Custom Drawers

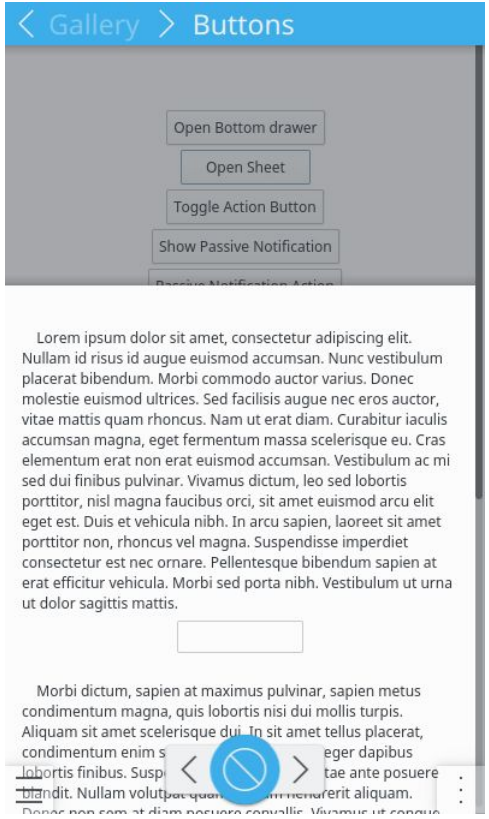
- The developer can use an arbitrary number of custom drawers
- Drawers from bottom useful to be used as a kind of “dialog”
- If you know your content will always just take a tiny portion of screen



Qt day

OverlaySheet

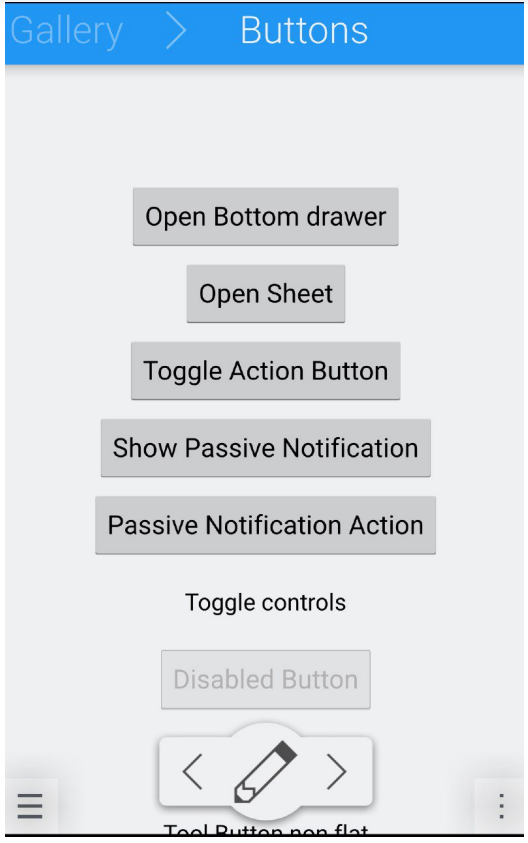
Better than drawers - if it's needed big, scrolling dialog-like pages that can be dismissed by gesture, use OverlaySheet



Qt day

Primary Action Button

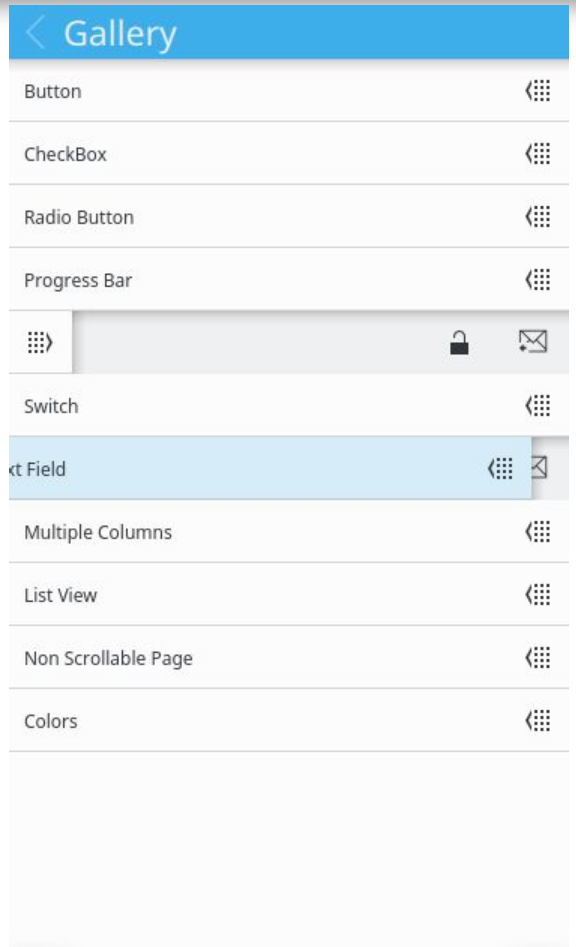
- Not a class that can be directly instantiated
- Triggers actions and acts as an handle for the side drawers
- The Page provides the actions as a model, Kirigami decides how to visually represent them



Qt day

SwipeListItem

- A list item with a standard look, plus a draggable handle to reveal actions
- Recommended when there is no contextdrawer
- The swipe gesture is mobile-specific, on desktop the actions will appear on mouse over, with no code changes

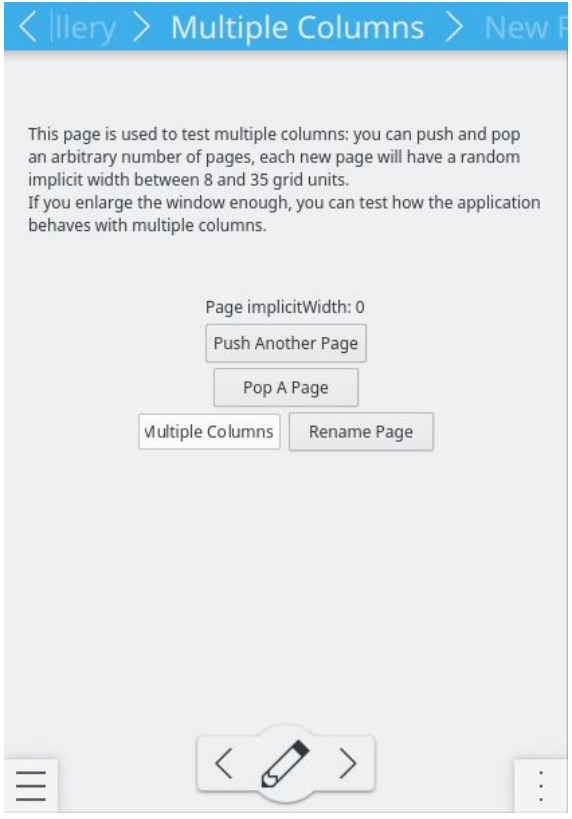


Qt day

```
ListView {
    id: mainListView

    model: //QAIM or simpler QML models
    delegate: Kirigami.SwipeListItem {
        id: listItem

        Kirigami.Label {
            text: model.text
            color: listItem.checked ? listItem.activeTextColor :
listItem.textColor
        }

        onClicked: ...
        actions: [
            Kirigami.Action {
                iconName: "document-decrypt"
                onTriggered: ...
            },
            Kirigami.Action {
                iconName: "mail-reply-sender"
                onTriggered: ...
            }
        ]
    }
}
```

ApplicationHeader

- The top of the application is reserved to a “title” area that acts as a breadcrumb in the currently open page hierarchy
- Content can be customized using AbstractapplicationHeader instead



Qt day

Other classes

- Action: “model” representation of an action
- ListItem/BasicListItem
- Heading
- Label
- SplitDrawer

Getting it built

- More than one way: what works on some platforms doesn't on others (plugins don't work on iOS)
- CMAKE
  - Build as a plugin (default)
  - Static
- QMAKE
  - Pro file, build as a plugin
  - Pri file, statically include
  - Global pri file, as a plugin and like any other KF5

Getting it built

- Cmake:
  - Plugin: find_package(KF5Kirigami CONFIG REQUIRED)
  - Static: cmake -DSTATIC_LIBRARY=ON
- Qmake:
  - Plugin: QT += Kirigami
  - Static: include (kirigami/kirigami.pri)

Quick and dirty qmake on iOS

- Clone kirigami.git in your project
- Clone breeze-icons.git in the kirigami folder

```
TEMPLATE = app

QT += qml quick
CONFIG += c++11

SOURCES += main.cpp
RESOURCES += qml.qrc

include(deployment.pri)

iphoneos {
    include (kirigami/kirigami.pri)
}
```

Qt day

Quick and dirty qmake on iOS

```
#include <QGuiApplication>
#include <QQmlApplicationEngine>
#include "kirigami/src/kirigamiplugin.h"

int main(int argc, char *argv[])
{
    QGuiApplication app(argc, argv);
#ifdef Q_OS_IOS
    KirigamiPlugin::registerTypes();
#endif
    QQmlApplicationEngine engine;
    engine.load(QUrl(QStringLiteral("qrc:/main.qml")));

    return app.exec();
}
```


Static with CMake: iOS

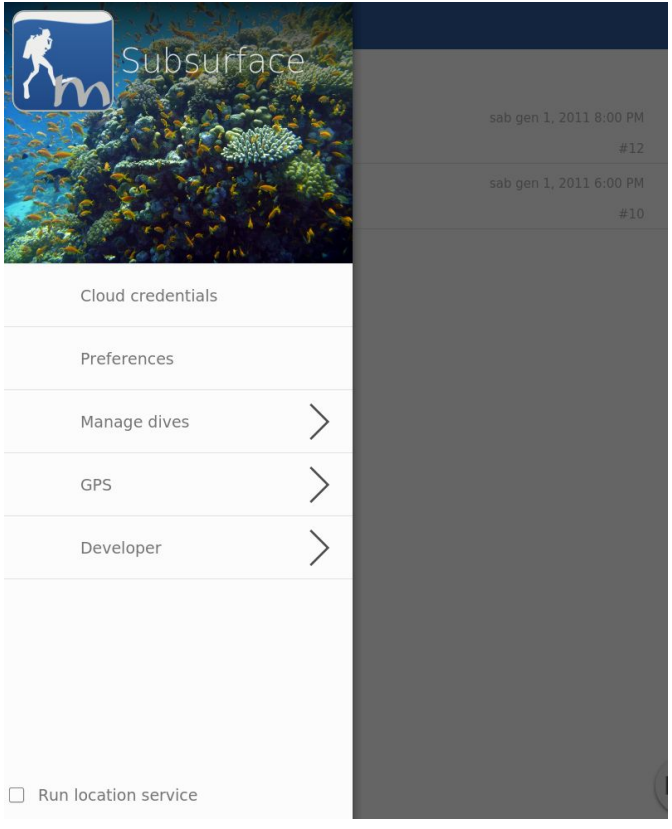
```
add_executable(main main.cpp)
IF(CMAKE_SYSTEM_NAME STREQUAL iOS)
target_link_libraries(main
            ${CMAKE_SOURCE_DIR}/libkirigamiplugin.a)
ELSE()

#normal linking...

ENDIF()
```

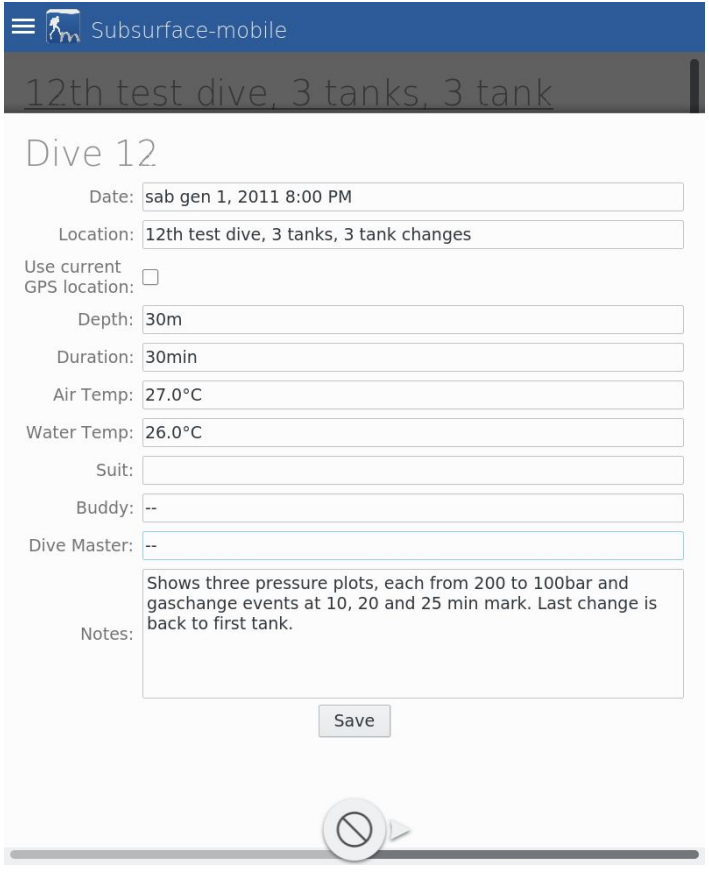
Qt day

Success Story

- SubSurface
- Dive log app written by Linus Torvalds, Dirk Hohndel and others
- Desktop app
- Used to be GTK+, migrated to Qt
- Released mobile version for Android (iOS version in beta),
- Mobile version shares most backend code with desktop version, Android and iOS code almost identical
- Early adopter for Kirigami, very good collaboration including patches from Dirk

Subsurface-mobile

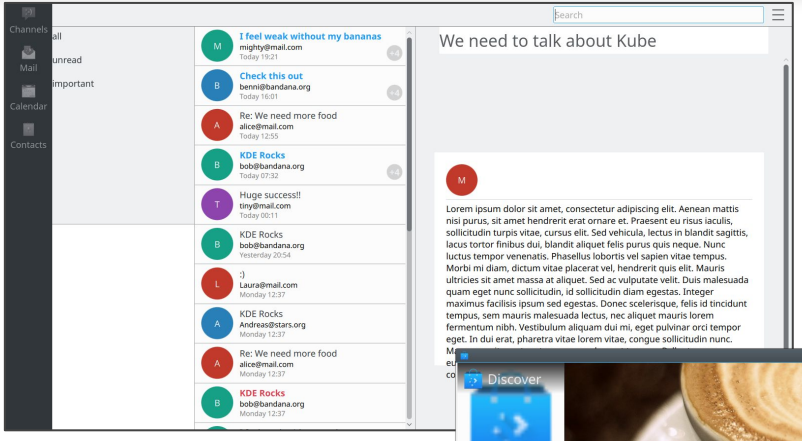


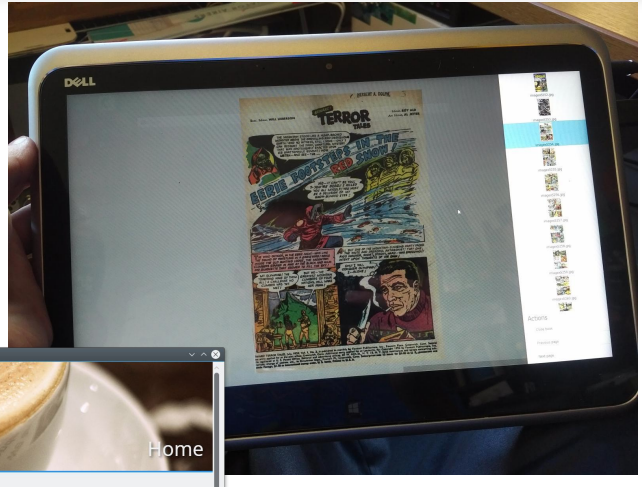


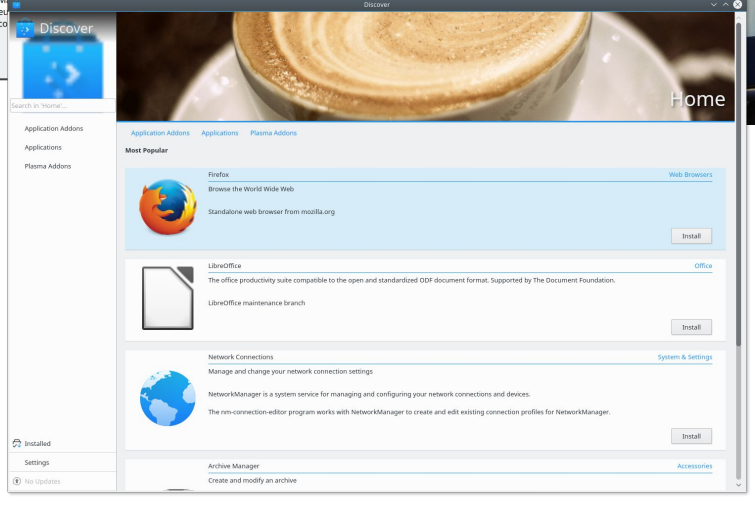
More Kirigami-Based Applications

- Peruse (released for desktop Linux and Windows, in development for mobile)
- Kube (in development for mobile and desktop Linux, Windows, OS X, Android)
- Discover (in development for mobile and desktop Linux)

More Kirigami-Based Applications







Qt day

Feedback

“The Android demo "Kirigami gallery" is running surprisingly fluidly on my Nexus 7 tablet from 2012. Top.”

“I'm excited by the slick looking Kirigami and the possibility of building applications to deploy on Linux, Android, and Windows with C++.”

“It didn't take long to get used to swiping the central button, and it is easy and comfortable to use. It's also very efficient: one button has three actions (swipe left, swipe right, click). The user just has to know what to do with it.”

“Superb function, beautiful design” (feedback on subsurface-mobile)

Qt day

Key Takeaways

- Kirigami is a **Tier 1** KDE Framework
- Ready for experimentation and wider usage
- Test the libary, contribute feedback and patches!

Additional Resources...

- HIG: https://hig.kde.org
- API: https://api.kde.org/playground-api/libs-apidocs/kirigami/html/index.html
- Gallery on Android: https://play.google.com/store/apps/details?id=org.kde.kirigamigallery
- IRC: #plasma or #kde-devel on freenode
- Mailing list: plasma-devel@kde.org or kde-devel@kde.org

Qt day

Differenze con MD/Altro

https://hig.kde.org/patterns/navigation/column.html

https://hig.kde.org/components/actionbutton.html
(now added in MD!)
https://material.io/design/components/app-bars-bottom.html

Qt day

Questions?

KIRIGAMI

Riccardo Iaconelli

riccardo@kde.org

@ruphy