fedora

KDE 4 & developers

Lukáš Tinkl <ltinkl@redhat.com>
Jaroslav Řezník <jreznik@redhat.com>

revision 20090909-1

Overview

Qt 4

- Foundation library, different API
- **Arthur** - rendering engine (OpenGL, SVG, X Render, Postscript, PDF)
- **D-BUS** support
- Integrated **ECMA scripting** (QtScript classes)
- New model/view framework
- **Phonon** (since KDE4.0/Qt 4.4) - multimedia
- **QtWebKit** (since Qt 4.4) – HTML engine
- **LGPL** license

CMake

- CMake is now default build system
- Cross-platform
  - Linux, BSD, Mac OS, Windows, etc...
- “Cross-make”
  - Support for autotools, Visual Studio projects, etc.
- Easy to write CMakefiles
- Modules
- Out-of-source builds

Phonon - overview

- New KDE multimedia Framework
- Different backends (GStreamer, Xine, DirectX, Quicktime)
- Unlike aRts not a sound server, rather a unifying API for KDE apps
- First in KDE 4.0, included in Qt 4.4, now standalone
- Graphs based
  - MediaObjects
  - Sinks
  - Paths

Phonon – playing sound

- When you want to play sound as easy as possible
  - Phonon::createPlayer
- C++ example

```
MediaObject *music =
    createPlayer(MusicCategory, MediaSource("sound.wav"));

music->play();
```

- Don't forget to link with Phonon!
  - Find package not needed anymore
  - to target-link-libraries add ${KDE4_PHONON_LIBS}

Phonon – simple video player

- When you want to play video
  - You can use Phonon::VideoPlayer
  - Or you can build your own graph!
- You will need
  - MediaObject, VideoWidget, AudioOutput and connect them together

```
MediaObject *mediaObject = new MediaObject(this);

VideoWidget *videoWidget = new VideoWidget(this);
createPath(mediaObject, videoWidget);

AudioOutput *audioOutput = new AudioOutput(VideoCategory, this);
createPath(mediaObject, audioOutput);
```

Phonon – playing streams

- You can play even network stream
  - Phonon::AbstractMediaStream
  - Push/Pull approaches

```
PullStream::PullStream(QObject *parent)
    : AbstractMediaStream(parent)
{
    setStreamSize(getMediaStreamSize());
}
void PullStream::needData()
{
    const QByteArray data = getMediaData();
    if (data.isEmpty()) {
        endOfData();
    } else {
        writeData(data);
    }
}
```

Solid - overview

- Similarly to Phonon, Solid is a hardware abstraction layer
- Built on top of HAL (Linux) or native API (Windows, OS X)
  - DeviceKit WIP
- Removable media, network discovery, etc.

Solid – network status

- When you need current network connection status
  - Solid::Networking::status()
    - Unknown — the networking system is not active or unable to report its status - proceed with caution
    - Unconnected — the system is not connected to any network
    - Disconnecting — the system is breaking the connection
    - Connecting — the system is not connected to any network
    - Connected — the system is currently connected to a network
- Simple C++ example

```
#include <solid/networking.cpp>

if ( Solid::Networking::status() == Solid::Networking::Connected )
    kDebug() << "Nice! You are online!";
else
    kDebug() << "What's wrong with you? You are offline!!!";
```

- Notifications
  - Solid::Networking::Notifier

Solid – devices listing

- On Linux built on top of HAL
- Command line usage
  - solid-[bluetooth | hardware | network | powermanagement]
- Simple example

```
[jreznik@dhcp-lab-147 ~]$ solid-hardware list
udi  = '/org/freedesktop/Hal/devices/platform_bluetooth'
udi  = '/org/freedesktop/Hal/devices/acpi_CPU0'
udi  = '/org/freedesktop/Hal/devices/acpi_CPU1'
...
[jreznik@dhcp-lab-147 ~]$ solid-hardware details
'/org/freedesktop/Hal/devices/acpi_CPU0'
udi  = '/org/freedesktop/Hal/devices/acpi_CPU0'
  parent = '/org/freedesktop/Hal/devices/computer'  (string)
  vendor = ''  (string)
  product = 'Intel(R) Xeon(R) CPU            5110  @ 1.60GHz'
...
```

Solid - storage

- Predicates for filtering device list results
- For storage access devices

```
[[StorageVolume.usage == 'FileSystem' OR StorageVolume.usage == 'Encrypted']
OR [ IS StorageAccess AND StorageDrive.driveType == 'Floppy' ]]
```

- And another C++ example

```
const QList<Device> &deviceList = Device::listFromQuery(predicate);

foreach(const Device &device, deviceList)
{
    const StorageAccess *access = device.as<StorageAccess>();
    const StorageVolume *volume = device.as<StorageVolume>();
    const Block *block = device.as<Block>();

    kDebug() << volume->label();
    kDebug() << access->filePath();
    kDebug() << block->device();
}
```

Plasma - overview

- Brand new concept of a desktop (panel, applets, etc.)
- **plasmoids** – desktop widgets
- SVG rendering, premade data engines
- Replaces the old *kdesktop* and *kicker*
- **KickOff** – the new menu system
- Uses data engine (plasmaengineexplorer) as model and plasmoids as view

Plasma - plasmoids

- Plasma panel

- Clock (analog, digital, binary), notes, battery status, trash, calculator, dictionary, comic strip etc...
- Alternative launchers
  - Lancelot
- 3rd party plasmoids

Plasma – simple plasmoid

- Solid + Plasma = simple network status plasmoid
- Two parts
  - data engine reads current status from Solid
  - plasmoid shows it
- Example...

Plasma – data engine

- Data engine serves as model for plasma applet(s)
- One or more data sources
- For polling updateSourceEvent

```
bool NetworkstatusEngine::updateSourceEvent(const QString &name)
{
    if (name == I18N_NOOP("NetworkStatus")) {
        if ( status() == Connected )
            setData(I18N_NOOP("NetworkStatus"), I18N_NOOP("status"),
              "connected";
        else
        ...

        return true;
    }

    return true;
}
```

Plasma – applet

- Applet serves as view for data engine(s) ;-)
- Connect to source

```
Plasma::DataEngine* networkstatusEngine =
    dataEngine("networkstatus");
networkstatusEngine->connectSource("NetworkStatus", this, 0);
```

- Read data source value
  - NetworkstatusApplet::dataUpdated(const QString& source, const Data &data) to m_status
- Paint plasmoid (paintInterface)

```
m_theme->paint(p, contentsRect, "globe");
if (m_status == "connected")
    m_theme->paint(p, contentsRect, "connected");
```

KWin 4 - overview

- Default KDE window manager
- New generation with compositing support like Compiz
- Eye-candy/usability effects
  - Exposé
  - Live applications switching applet
  - Wobbly windows
  - Magnifier
  - Even cube ;-)

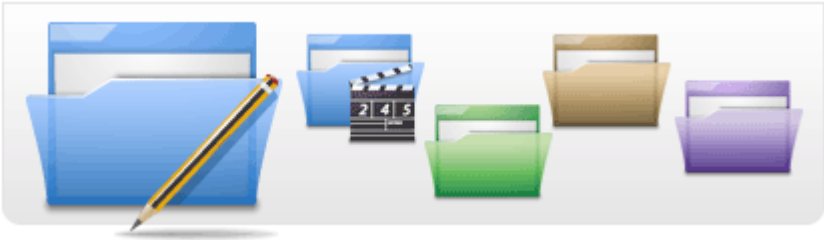
Decibel, Akonadi, KitchenSync, Kross

- Telephony API (XMPP, Jingle and SIP) – it's Decibel.
- Central PIM storage (SQL based) – it's Akonadi.
- Sync framework based on OpenSync.
- Full and transparent scripting with Python, Ruby and KDE JavaScript – it's Kross.

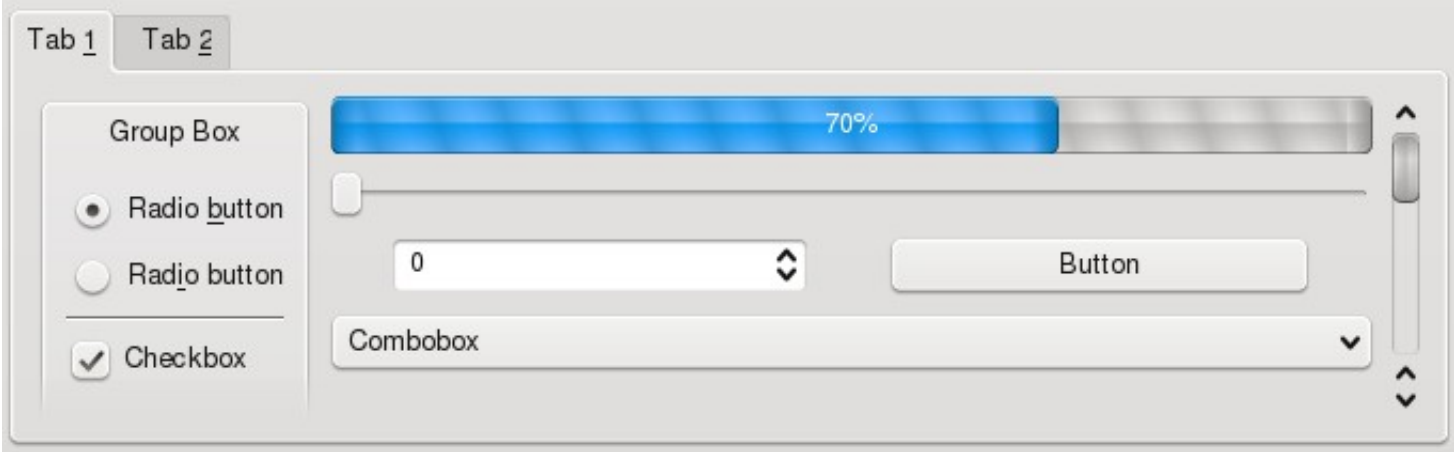
Kross

Oxygen

- New fresh look
- Oxygen icon theme



- Oxygen widgets



New applications

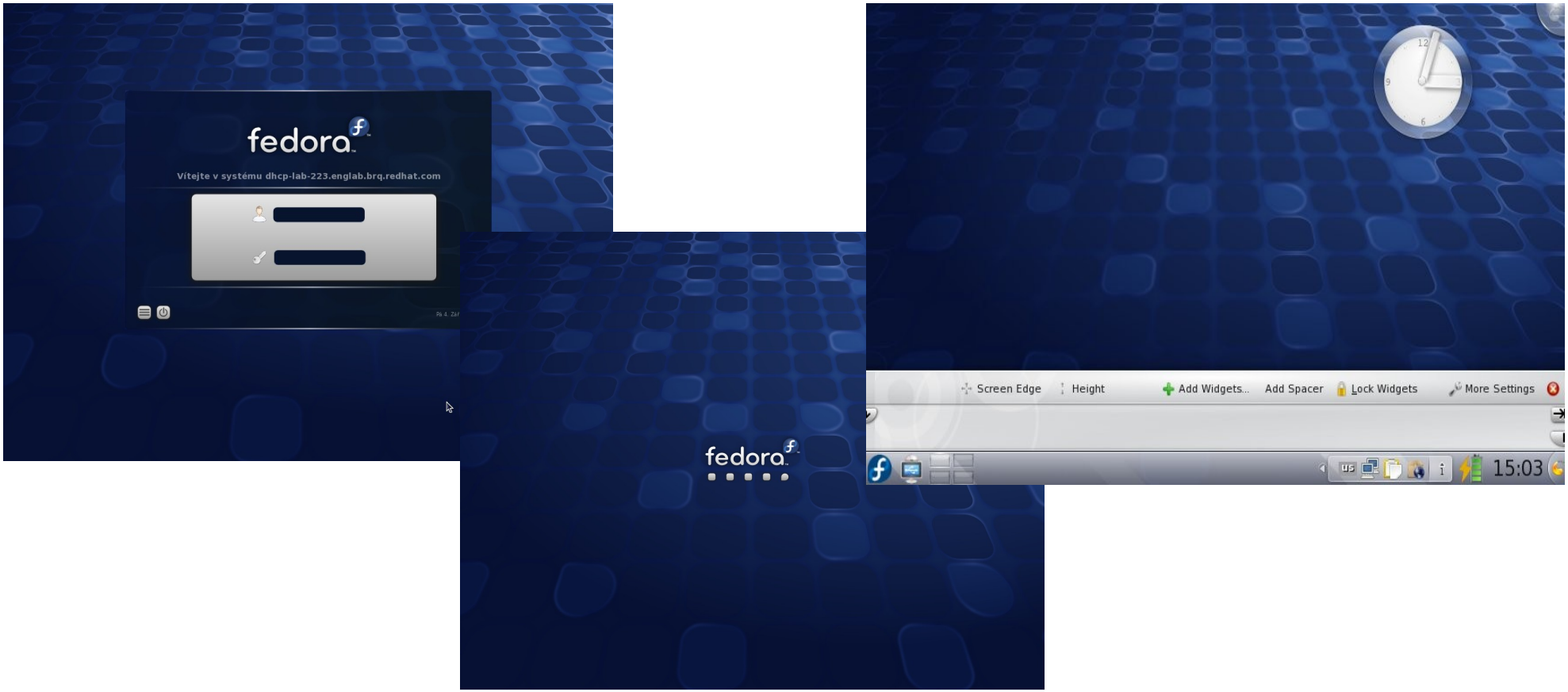
- Dolphin – file manager
- Okular – universal viewers (PS, PDF, DVI, etc.)
- Amarok 2
- New games and edu applications
  - Marble, KStars
- Many improvements in all KDE applications!

KDE 4.3.0

- New default theme – Air
- Improved job and notification management
- PolicyKit integration
  - PolicyKit 1 support WIP
- Ark supports LZMA/XZ
- New Bug Report Tool
- New effects like "Sheet" and "Slide Back" and better performance in KWin

Fedora 12 KDE Desktop Edition

- KDE 4.3.1
- GStreamer Phonon backend as default
- KPackageKit



KDE 4 on Fedora

- 4.3.1 for Fedora 10, 11 & 12
- Can I help?
  - Bug reporting, testing...
  - There are so many missing KDE packages in Fedora repository...
  - Communicate
    - IRC: #kde, #fedora-kde @ FreeNode.org
    - http://www.kde.org/, http://fedoraproject.org/wiki/KDE, http://fedoraproject.org/wiki/SIGs/KDE

Questions...

- Questions...
  - Otázky...
    - Preguntas...
      - Fragen...