Construindo um Minigame com Qt e Raspberry Pi

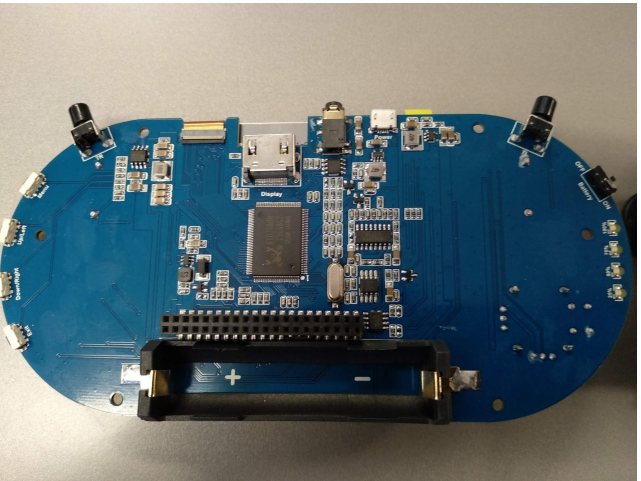
Luis Gustavo S. Barreto
<gustavo@ossystems.com.br>

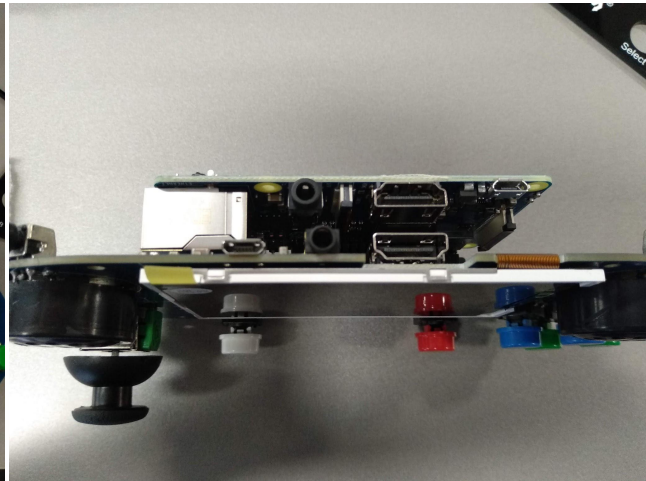

QT
HD
HISTORY.COM

Introdução

O projeto consiste em um Gamepad (mini-game) desenvolvido inteiramente com Qt/Qml utilizando o seguinte hardware:

- Raspberry Pi 3 Model B+
- Waveshare Game HAT





Prototipagem inicial

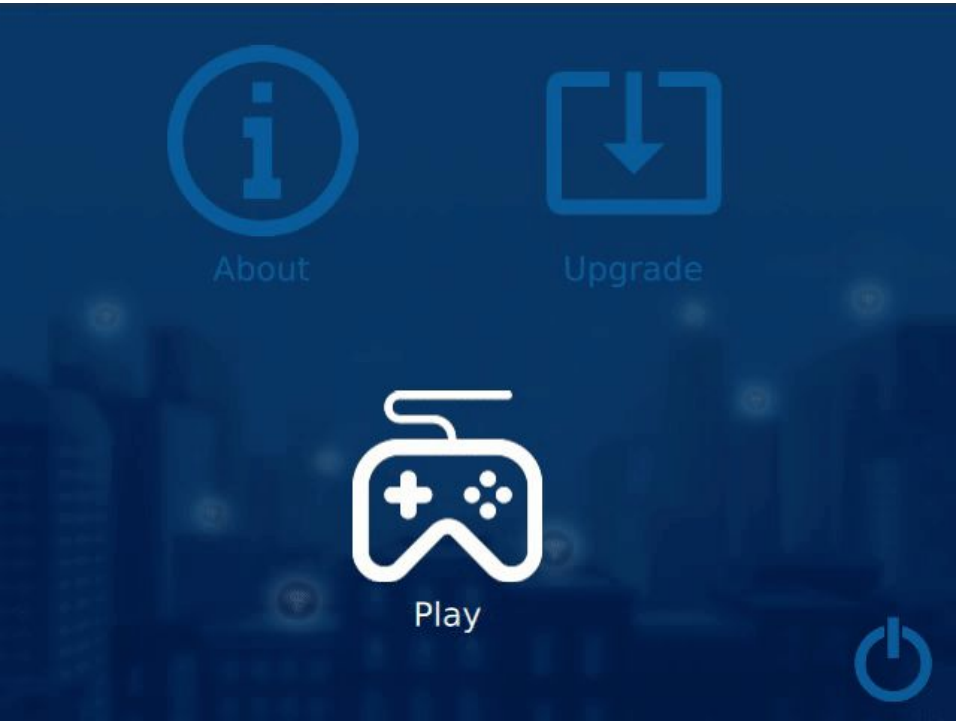
A prototipagem inicial se deu em cima de uma instalação padrão do Raspbian.

- Desafios
  - Integrar os botões do Gamepad no X11 (joy2key)
- Vantagens
  - Universo debian: apt, utilitários, etc..
  - Ambiente gráfico semelhante ao desktop de desenvolvimento
  - Fácil e rápido de enviar a aplicação com SSH/SCP
- Desvantagens
  - Difícil de reproduzir e replicar o que foi feito
  - Empacotamento complicado
  - Portabilidade
  - Tamanho da imagem (4GB)
  - Como manter atualizado?


TL
TR
Q.C. Pass
NO 86
Get Ready!
TAP
TAP
X
A
Y
B
Game HAT
Start
Select


TL
TR
Play
X
A
Y
B
Game HAT
Start
Select


TL
TR
updatehub
Checking for updates...
X
A
Y
B
Game HAT
Start
Select


About
Upgrade
Play


Get Ready!
TAP
TAP

Embarcando

- Framework Linux Embarcado - Yocto Project
- Over-The-Air Update - UpdateHub
- Framework Gráfico - Qt Quick

1º

Framework Linux Embarcado

Yocto Project

- Framework para criação e desenvolvimento de distribuições Linux embarcado
- Código aberto (mas pode ser usado para compilação de código proprietário)
- Utilizado por grandes fabricantes da indústria de dispositivo embarcados
- O que ele faz?
    - Download do código fonte
    - Aplicação de patches
    - Compilação cruzada
    - Gerenciamento de pacotes
- O que ele gera?
    - Pacotes binários
    - Imagens de sistema Linux
    - Toolchains
    - SDKs

2°

Over-The-Air Update

UpdateHub

É uma solução para atualização de dispositivos IoT e Linux embarcado.

- Suporta Linux e RTOS (Zephyr Project)
- Integração com o Yocto Project
- SDK para integração com aplicações: Python, Go e Qt/Qml
- 100% código aberto (cliente e servidor)
- Opção de servidor SaaS (enterprise)
- Suporte a atualizar *rootfs* e *bootloader* (*U-Boot/Grub*)
- Suporte i.MX, Rockchip, Texas Instruments, Broadcom, ...

3º

Framework Gráfico

Qt Quick

O Qt Quick é uma coleção de tecnologias para ajudar no desenvolvimento de interfaces intuitivas, modernas e fluidas.

- **QML**, a linguagem
- **Qt Creator,** o editor/IDE

QML

A sintaxe do QML é parecida com JSON e suporta expressões JavaScript combinado a um sistema dinâmico de propriedades. Além integrar com C++ por meio de plugins.

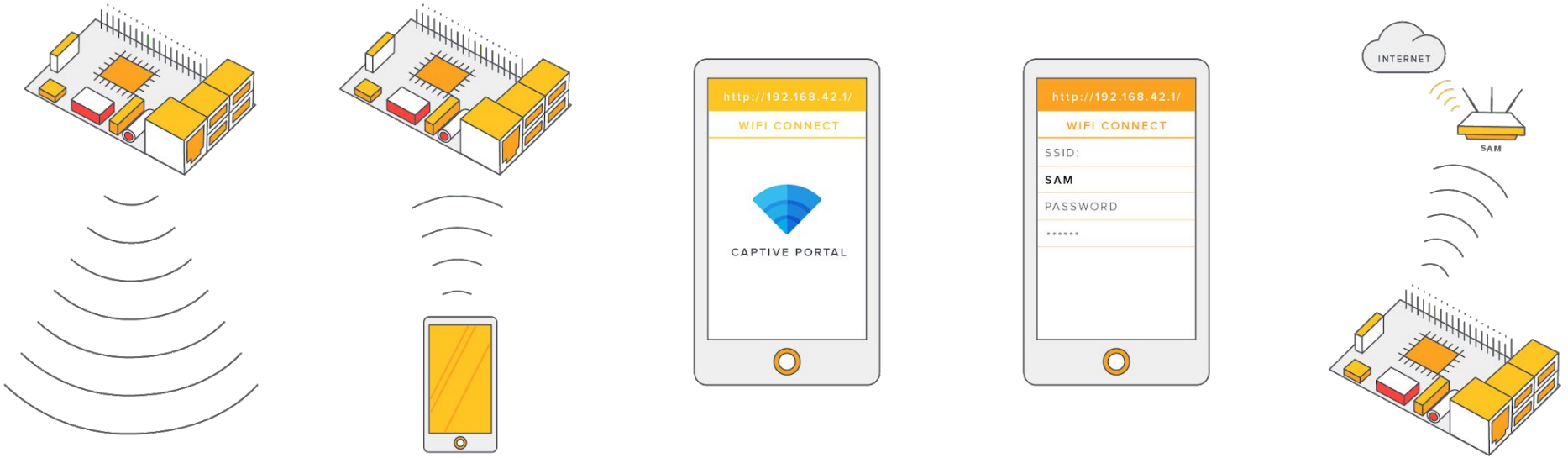
Logo, QML =

QML no desenvolvimento de jogos

- V-Play Engine (proprietária)
- QML Box2D plugin (open source)

Desafios

Resolvendo o problema de conectividade



1. **ADVERTISE:** Device Creates Access Point
2. **CONNECT:** User Connects Phone To Device Access Point
3. **PORTAL:** Phone Shows Captive Portal To Use
4. **CREDENTIALS:** User Enters Local WiFi Network Credentials On Phone
5. **CONNECTED!:** Device Connects To Local WiFi Network

Traduzindo os botões para o evdev

```
dtoverlay=gpio-key,gpio=5,keycode=103,label="KEY_UP"
dtoverlay=gpio-key,gpio=13,keycode=105,label="KEY_LEFT"
dtoverlay=gpio-key,gpio=6,keycode=108,label="KEY_DOWN"
dtoverlay=gpio-key,gpio=19,keycode=106,label="KEY_RIGHT"
....
```

Desenvolvimento

Arquitetura do sistema

- Sistema operacional
  - Linux (Yocto)
- Serviços do sistema
  - UpdateHub
  - Resin Wifi Connect
  - NetworkManager
- Bibliotecas e Plugins
  - Qt...
  - UpdateHub SDK
  - QML Utils
- UI
  - Homescreen
  - Game

Yocto e os layers necessários

```
$ git clone -b thud git://git.yoctoproject.org/poky
$ git clone -b thud git://git.openembedded.org/meta-openembedded
$ git clone -b thud git://git.yoctoproject.org/meta-raspberrypi
$ git clone -b thud https://github.com/meta-qt5/meta-qt5.git
$ git clone -b thud https://github.com/UpdateHub/meta-updatehub
```

Receita da UI principal

```
SRCREV = "dc9a18955cb8cfbd2f517ef7bb959e152b2b6484"
SRC_URI = "git://github.com/gustavosbarreto/gamepad;protocol=https"

do_install() {
  cp -r ${S}/* ${D}${datadir}/${PN}
}

RDEPENDS_${PN} += " \
  qtbase-plugins \
  qtdeclarative-qmlplugins \
  qtdeclarative-tools \
  qtgraphicaleffects-qmlplugins \
  updatehub-sdk-qt-qmlplugin \
"
```

Receita do jogo

```
SRCREV = "c450ade7d5b56690d265c7a2a0abe797e1ac5421"
SRC_URI = "git://github.com/gustavosbarreto/qml-flappy-bird;protocol=http;branch=rasp"

do_install() {
  cp -r ${S}/* ${D}${datadir}/${PN}
}

RDEPENDS_${PN} += " \
  qtbase-plugins \
  qtdeclarative-qmlplugins \
  qtdeclarative-tools \
"
```

Receita da imagem

```
inherit core-image updatehub-image

CORE_IMAGE_EXTRA_INSTALL += '\
  gamepad \
  qml-flappy-bird \
  wifi-connect \
'

SYSTEMD_DEFAULT_TARGET = "graphical.target"
```

Construindo a imagem

```
bitbake gamepad-image
```

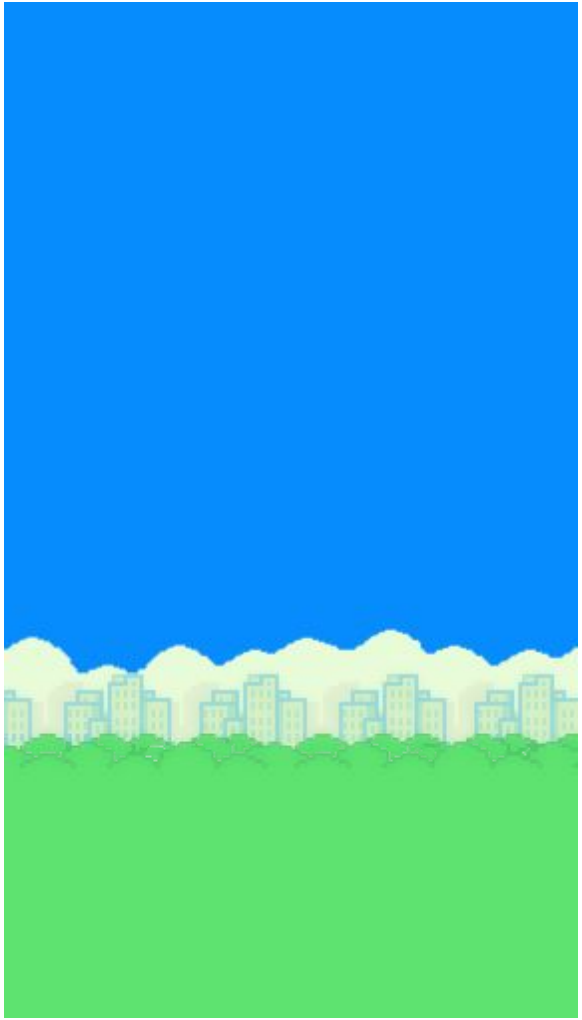
Estrutura da aplicação QML

```
StackView {
  Menu {
  }

  Game {
    Keys.onPressed: bird.jump()

    Bird {
      image: "bird.png"
    }

    Pipe {
    }
  }
}
```

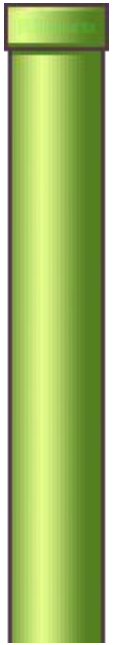
Componentes

Os elementos gráficos da cena (imagens) se transformam em componentes.

- Reaproveitamento em outros momentos da cena
- Reaproveitamento até em projetos diferentes
- Evita duplicação de código
- Facilita a compreensão do código separando cada “pedaço” em um arquivo




MEDAL
SCORE
MOST




Get Ready!


0 1234567890


Game Over

Animando

Timers são responsáveis por darem vida aos elementos da cena. Geralmente incrementando ou decrementando propriedades de outros componentes como *x*, *y*, *largura e altura*.

- Timer de animação do fundo (paisagem)
- Timer de animação do chão
- Timer de “simulação” de gravidade
- Timer da fábrica de canos

Dicas de QML

- QtGraphicalEffects
  - FastBlur - Aplica um blur em elementos
  - ColorOverlay - Altera a cor de um elemento aplicando um overlay
- BorderImage
  - Utilizado para criar bordas em um imagem repetindo porções da imagem original

Algumas curiosidades

- Poucas linhas de código
  - LOC do Homescreen: 373
  - LOC do Game: 578
- *C++-free* (sem uso de C++)

Mais informações

- **Código fonte do home screen do mini game:** http://github.com/gustavosbarreto/gamepad/
- **Código fonte do jogo:** http://github.com/gustavosbarreto/qml-flappy-bird/
- **Código fonte do UpdateHub:** https://github.com/updatehub/
- **Código fonte do Wifi Connect:** https://github.com/balena-io/wifi-connect
- **Sobre o UpdateHub:** http://updatehub.io

Obrigado!