Die Brummbeere Documentation

Release 0.0.1

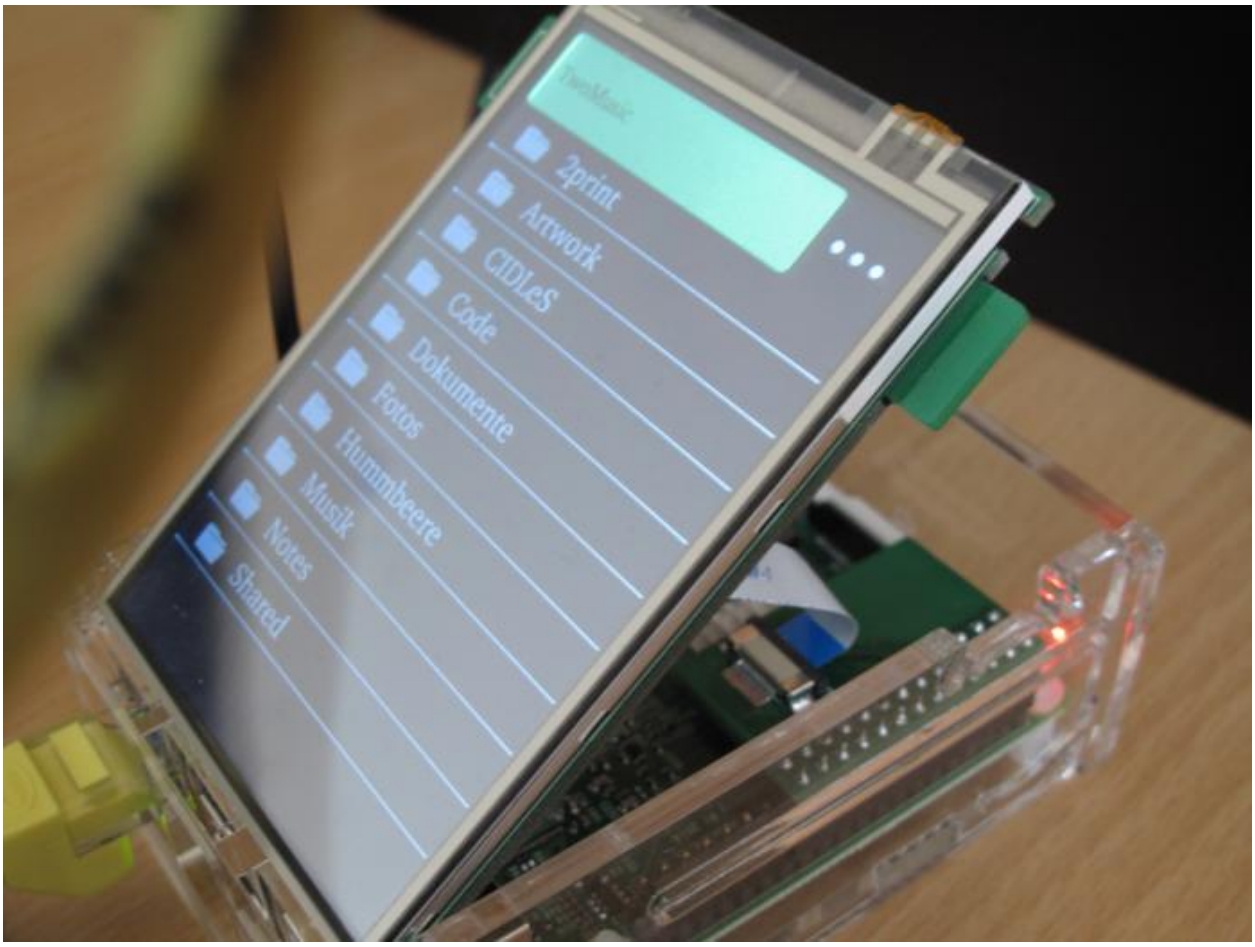
Peter Bouda

February 21, 2016

Contents

CHAPTER 1

ownCloud Music Player



Watch Die Brummbeere in action

Die Brummbeere is an ownCloud audio player based on the Qt framework. The application was written for embedded systems in mind, so it features a very simplistic user interface suitable for low resolutions and optimized for low energy consumption. It still runs on Linux and Mac.

View on GitHub || Download .zip

CHAPTER 2

Contents

2.1 Compile for desktop

2.1.1 Prerequisites

The only requirement to compile Die Brummbeere is Qt, which you can download from here:

http://www.qt.io/download/

This will in contain all the Qt libraries and the Qt Creator, which we will use to compile Die Brummbeere on desktop computers.

2.1.2 Windows, Mac and Linux

Note: Playing music won't currently work on Windows, due to a missing feature in Qt, i.e. streaming audio from password protected URLs. This is an open issue in the Qt bugtracker and might be solved in future Qt versions.

On desktop computers Die Brummbeere can be compiled just like any other Qt software. You may clone the git repository or just download and unzip the current master branch. When you clone you have to pull in all submodules (currently Die Brummbeere uses the project beere-qml-components). On Mac or Linux just open a shell and type:

```
$ git clone https://github.com/pbouda/brummbeere
$ cd brummbeere
$ git submodule init
$ git submodule update
```

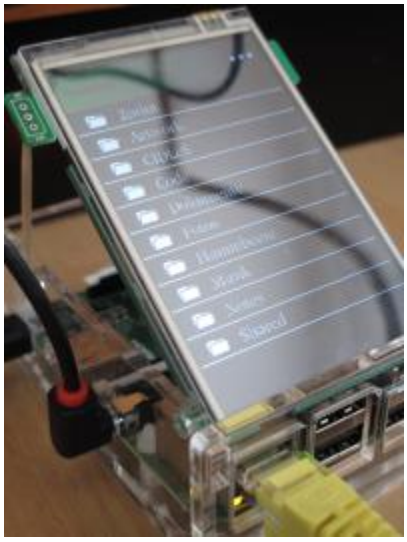
This will create a folder named `brummbeere`, which contains a folder `src` with the code of Die Brummbeere. Just open the project file `Brummbeere.pro` in the Qt Creator and build and run the project.

For manual compilation on the command line you have to call `qmake` followed by `make` in the `src` folder:

```
$ cd src
$ qmake Brummbeere.pro
$ make
```

This will create a binary `brummbeere` in the `mainapp` folder.

2.2 Embedded Brummbeere on the Raspberry



This tutorial shows how to build a custom embedded Linux for the Raspberry that runs Die Brummbeere to play audio files from ownCloud. The system will boot a minimal Linux environemnt with Qt libraries and automatically start Die Brummbeere. Optionally, it will load drivers for touchscreens and/or audio boards. We use the Qt Multimedia module and the Linux ALSA sound environment to play back the files. This setup can be used to run any other Qt apps on the Raspberry, of course. It is a general guide how to boot a minimal system with the latest Qt library, including support for touchscreens, OpenGL and multimedia.

At the moment, this guide is for the Raspberry Pi 2 and Raspberry A/B(+). You need to run Linux to compile your own embedded Linux with buildroot.

As an example for a touchscreen we will use the Tontec 3,5" TFT. It comes with a nice case that you can see on the picture above. The TFT is available on Amazon, for example. It is easily connected to the Raspberry GPIO header and its drivers are part of the Raspberry Linux distribution.

2.2.1 Prepare SD card

The SD card has to co be prepared with a certain partition layout in order to be bootable on the Raspberry. The standard layout is a small FAT partition and a larger ext4 partition in this order. The easiest way to get prepare the card in this way is to install a standard Raspbian on the card. You can find information about the process on the Raspberry download page:

https://www.raspberrypi.org/downloads/

Just follow the instructions given on the page under the `Raspbian` heading.

Optional: Use latest device tree overlays

The latest Raspberry kernels and images support device tree overlays (DTO) for specific hardware like TFT display with touch or audio card. The usage of device trees makes support for different hardware easier, compared to the configuration of device drivers via kernel modules. If you plan to use a TFT touch screen (like the Tontec TFT) or an audio card (like Hifiberry) you can enable the device trees for your hardware. The following example enables the Tontec TFT with touch support and sets the correct default resolution. As we will use the EGLFS backend of Qt, we have to turn on HDMI even if no screen is connected. The embedded system then uses fbcp to copy the current HDMI output to the framebuffer of the TFT screen.

First we will update the DTOs of the Raspbian boot partition that was installed in the previous step. Mount the partitions of your SD card and check that you have a folder `overlays`. We will replace it with the current firmware overlays in the Raspberry GitHub repository. Download and unpack the firmware:

```
$ wget https://github.com/raspberrypi/firmware/archive/master.zip
$ unzip master.zip
```

In the folder `firmware-master` you will find the folder `boot`, which contains the latest firmwares, kernels and overlays. You can just copy the full content of the `boot` folder to the boot partition of your SD card. Choose to overwrite any existing files on the card. The buildroot system will later overwrite the kernel of the boot partition with a self-compiled version, the rest of the boot partition will stay as it is now.

Optional: Modify config.txt for hardware support

Next, we will enable the Tontec TFT touchscreen and set the correct resolution. Open the file `config.txt` on the boot partition of you SD card and paste the following content:

```
dtoverlay=mz61581,rotate=0

hdmi_cvt=320 480 50 2
hdmi_group=2
hdmi_mode=87
hdmi_force_hotplug=1
```

The first line enables the device tree overlay for the TFT, including touch support. The output is not rotated, so we get portrait mode.

The next 4 lines set a custom resolution (we need 320x480 pixels for the Tontec screen) and force HDMI hotplug. We need the latter as we want to run without any external display on HDMI, but still need HDMI enabled to get OpenGL support in Qt and a framebuffer to copy the graphics output to the TFT's framebuffer.

2.2.2 Build Embedded Linux with Die Brummbeere

In this step we will build a complete Linux system including the kernel, drivers and the Qt libraries. The build process depends on buildroot.

Buildroot configuration

First you need to download Die Brummbeere and buildroot. The Raspberry 2 is only supported in the current buildroot git repository, so we clone the current buildroot master. The Brummbeere repository contains a skeleton folder `raspi` that we use to clone into. This folder contains scripts and files that buildroot will use to build the filesystem for the embedded system:

```
$ git clone https://github.com/pbouda/brummbeere.git
$ cd brummbeere
$ git submodule init
$ git submodule update
$ cd raspi
$ git clone git://git.buildroot.net/buildroot
```

In the next step we configure buildroot for the Raspberry Pi 2 and a complete Qt framework with dependencies like ALSA. The folder `raspi/buidroot-config` contains buildroot configuration files to set all options that we need. Enter the `buildroot` folder and load the configuration:

```
$ cd buildroot
$ make defconfig BR2_DEFCONFIG=../buildroot-config/brummbeere-raspi2.config
```

If you want to build Die Brummbeere for Raspberry A/B(+) then choose the "raspi" configuration file during this step:

```
$ cd buildroot
$ make defconfig BR2_DEFCONFIG=../buildroot-config/brummbeer-raspi.config
```

Adding NTP daemon

As the Raspberry does not have a realtime clock, our embedded system start an NTP daemon to set the current date and time. Qt will use the date to validate the SSL certificate of your ownCloud server, if the connection is encrypted. As the embedded system uses buildroot's busybox, we will just add the `ntpd` option to the configuration. Start the menu configuration of busybox:

```
$ make busybox-menuconfig
```

In the menu choose the option `Networking Utilities -> ntpd`. Exit and save.

Download Raspberry tools

To be able to add support for device tree overlays in a later step we need to download the Raspberry tools. The tools contain a script `mkknlimg` that adds a trailer to the self-compiled kernel. It also includes a script `knlinfo` that output whether a given kernel contains the trailer for DTO support. You can just clone the tools from GitHub. The script that installs the root filesystem later expects the script to be located in `brummbeere/raspi/tools/mkimage`, so make sure that you clone into the folder `brummbeere/raspi`:

```
$ cd ..
$ git clone https://github.com/raspberrypi/tools.git
```

Modify installrootfs.sh script

The script that install the root filesystems needs to know the device of your SD card. Please check carefully which device your SD card uses and adapt the script in `raspi/scripts/installrootfs.sh`. Currently the device for the SD card is `/dev/sdc`. Change those device names to your setup **in all locations**.

If your SD card is still mounted from step *Prepare SD card* you might just call `mount` to see a list of all filesystems. Find your SD card in this list and use the device names that are listed (like `/dev/sdc1` and `/dev/sdc2`).

Careful: Your SD card has to prepared with the two Raspberry partitions and should be mounted for the following steps. If you do not edit the script ''installrootfs.sh'' with the correct device names your hard disk might be formatted!

Add config file for ownCloud

As the current version of Die Brummbeere does not contain an onscreen keyboard, you might not be able to edit the URL, user name and password on the Raspberry. To set an initial configuration you can create a file `Brummbeere.conf` in the folder `raspi/userland/target`. The file has the following content:

```
url=https://yourowncloud.domain.com
user=yourusername
password=yourpassword
```

The file will be copied to the correct location on the root filesystem automatically and will be used to access your ownCloud.

Start the build process

You can now start the build process. This will create Linux, all libraries and copy everything to the SD card. If you do not run with root privileges the build process will ask for a root password at some later point (when the filesystem is copied to the SD card). The whole procedure might take a while, up to a few hours. Just run:

```
$ make
```

Good luck and have fun with Die Brummbeere!

2.3 Feedback

I would like to hear from you! If you have any comments, suggestions, bug reports, or just want to learn more about Die Brummbeere, please contact me via one of the following options.

2.3.1 Mail

My mail address is pbouda at outlook dot com.

2.3.2 IRC

My IRC name is *pbouda*, you can normally find me on #qt and #buildroot on Freenode.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search