openPablo manual v0.1

Inhaltsverzeichnis

1 Introduction

1.1 What is openPablo?

As an open source project openPablo can harrness the inherent power of community power.

if you need premium support (and you will likely do so in a commercial and productive environment) currently it is better to use commercial products like Elipcal's Claro, Softcolors Color or OneVisions Amendo. This might change though, depending on how much support the project gets.

Still, some of the software out there is quite buggy, and so its quite legitimate to ask: Why pay for bananaware, if you can have similar functions for free?

Rumors are that curent image enhancement technology is not better than ImageMagick, a free tool. Unluckily ImageMagick does not come with a ready-to-use interface, contrary, using it can be quite delicate. There are even very sophisticated scripts, but using them is hard, modifying them can be harder.

The tool aims for the Newspaper Market, where it is not crucial to have 100 percent superb images, and where cost is a major problem. Therefore collaborating on such issues like color management in an open manner can be beneficial for the whole branch.

· No more dongles, licences, encrypted, propriary ticket formats or price-discussions

· Strange Behaviour can be identified by your local hacker.

· Latest technology library updates will result in speed. (we do not fear breaking something, if so, we can fix and release it on a daily basis).

· Forum to talk about common problems, place for experts to discuss and exchange.

· Technology is crossplattform as possible, allows it to be run on mobile phones. (at least some controller?)

As an open source project, we can use the all the power of the open source world:

- OpenCV
- Lensfun
- Exiv 2
- Little CMS 2
- Magick++
- Qt
- PoDoFo
- Boost
- OpenMP

1.2 Shell Features

openPablo supports the following fileformats (thanks to Magick++):

- JPG
- JPG2000
- TIFF
- PNG
- PSD
- PDF (via podofo)
- RAW (via dcraw++)

openPablo can handle the following metadata (thanks to Exiv2):

- EXIF
- IPTC
- XMP

- PSD Section

Tickets can be written either in JSON or in XML format. Any tool that can edit these fileformats can be used to create or modify tickets. The ticket API itself is open and documented.

You can run several instances next to each other. For example each one could monitor some hotfolder.

1.3 Library Features

- personal preferences
- scene recognition/gist
- recognition for special branches
- vanishing points
- autorotate
- lens correction
- white balance
- sky recoloring
- timestampremoval
- chromatic aberration
- noise reduction
- sharpening
- split toning
- tone curve
- saturation
- vibrance
- clarity
- black/white point

- shadows/highlights
- contrast
- exposure
- tint
- temperature

1.4 ToDos and Ideas

- Plugin System We will try to implement a plugin-system, so you can develop commercial plugins without being forced to give away your code.
- exclude image conditions
- better layer support/alpha channels
- GPU
- UI
- filerouting
- web2.0/crowd
- scene and object recognition
- create your own local server, or cloud-service
- test system/test set images
- JDF
- use smile or bson format internaly for the blobs.

1.5 The other side of the coin

Though we will take some care not to include patented algorithms, we cannot take any guarantees that by using this software, you are making use of third-party patents. If you see anything that infringes any patent or copyright, please tell us, we will remove the offending parts immediately and will try to find another workaround.

1.6 Remarks

Why openPablo has been initiated? My main goal is to implement several image enhancement routines. But there is none open source library where I could add my code to. Just producing these routines would be some kind of time, as without some application around it, these ideas would simply be left to die. But with some nice application around it, these might get used by a broader community. Also there hopefully be more exchange on these topics, between developers and users of these software.

2 Using openPablo

2.1 Simple example

You can also specify a whole folder with settings via the '–settingsfolder' option. openPablo will then read all valid tickets inside this folder and process them. This way you can easily save more complex setups.

3 Tickets

```
{
   "Settings": {
      "id": "Offset1",
      "Color": {
         "ICC": {
            ”Path”: ”data/iccprofiles”,
            ”Input”: ”Autodetect”,
            ”Output”: ”ISOcoated_v2_300_bas.ICC”
        }
    },
    ”FileHandling”: {
        ”OutputFormat”: ”JPEG”,
        ”Compression”: 87
    }
},

”Input”: {
    ”InputFile”: ”cmyk.jpg”,
    ”InputPath”: ”data”
},

”Output”: [
    {
        ”id”: ”Web”,
        ”Settings”: ”Offset1”,
        ”Width”: 320,
        ”Height”: 240,
        ”OutputPath”: ”tmp/tmpweb/”,
”RenamePattern”: ”%f_%d.%e”
    },

    {
        ”id”: ”Print”,
        ”Settings”: ”Offset1”,
        ”Width”: 1024,
        ”Height”: 1024,
        ”OutputPath”: ”tmp/tmpprint/”
    }
]
}
```

3.1 Input

Either an explicit input file is given, or a wildcard. In this case, openPablo will go into a hotfolder-state, i.e. it will monitor the given input-path and process every incoming file.

3.2 Output

Several output can be defined. Each output must have a distinct id which identifies it. A Settings have to be specified by a settings id, where the specific settings will be used for processing. Other options are: Width, Height. The outputpath will be the path where the processed files will be put into.

3.3 Logs

format of log (=HTML, XML, JSON?, TXT), place for log, filename for log.

4 Test cases

TestCases are skripted via Python. Logs can be read and interpreted. only blackbox testing.

5 Technical side

... Uses ImageMagick (i.e. Magick++) to read images. This format is then converted to BSON, decorated with some minor extra-infos (e.g. was the image JPEG-compressed?) and forwarded to the engine. After the engine finished its processing, returning another image in BSON format, the data is then pumped back to Magick++ which then will write the image as specified in the ticket.