rtorrent-ps Documentation

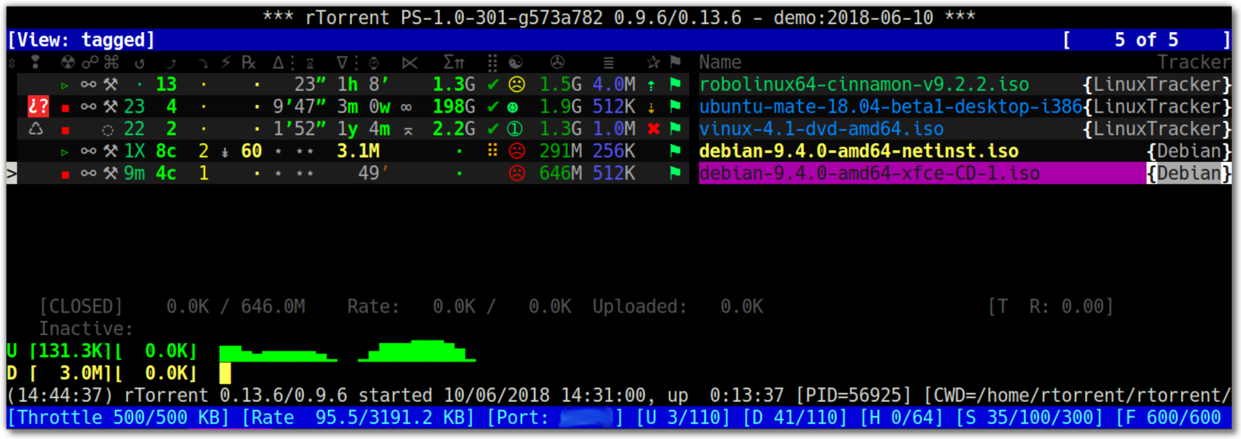
Release 1.1-2018-07

PyroScope Project

Jul 01, 2018

Contents

rTorrent-PS is an extended rTorrent *distribution* with UI enhancements, colorization, some added features, and a comprehensive standard configuration.



```
                    *** rTorrent PS-1.0-301-g573a782 0.9.6/0.13.6 - demo:2018-06-10 ***
[View: tagged]                                                                      [    5 of 5     ]
⇳ ❢  ☢ ☍ ⌘ ↺  ⤴  ⤵ ⚡ ℞ Δ⋮ ⌛  ∇⋮ ⌚  ⋉  Σ⇈  ⣿ ☯  ✇   ≣    ☆ ⚑  Name                                Tracker
      ▹ ⚯ ⚒  · 13  ·    ·   23” 1h 8’    1.3G ✓ ☹  1.5G 4.0M ⇡ ⚑  robolinux64-cinnamon-v9.2.2.iso     {LinuxTracker}
 ¿? ■ ⚯ ⚒ 23  4  ·    · 9’47” 3m 0w ∞  198G ✓ ⊛  1.9G 512K ⇣ ⚑  ubuntu-mate-18.04-beta1-desktop-i386{LinuxTracker}
 ♺  ■    ◌ 22  2  ·    · 1’52” 1y 4m ⌅  2.2G ✓ ①  1.3G 1.0M ✖ ⚑  vinux-4.1-dvd-amd64.iso             {LinuxTracker}
      ▹ ⚯ ⚒ 1X 8c  2 ↡ 60 ⋆ ⋆⋆  3.1M       · ⣿ ☹  291M 256K   ⚑  debian-9.4.0-amd64-netinst.iso            {Debian}
>     ■ ⚯ ⚒ 9m 4c  1    · ⋆ ⋆⋆    49’      ·   ☹  646M 512K   ⚑  debian-9.4.0-amd64-xfce-CD-1.iso          {Debian}




  [CLOSED]    0.0K / 646.0M    Rate:   0.0K /   0.0K  Uploaded:   0.0K                [T  R: 0.00]
  Inactive:
U [131.3K][  0.0K]
D [  3.0M][  0.0K]
(14:44:37) rTorrent 0.13.6/0.9.6 started 10/06/2018 14:31:00, up  0:13:37 [PID=56925] [CWD=/home/rtorrent/rtorrent/
[Throttle 500/500 KB] [Rate  95.5/3191.2 KB] [Port: [redacted]] [U 3/110] [D 41/110] [H 0/64] [S 35/100/300] [F 600/600
```

See *Overview* for a more detailed description and a feature overview, and the rTorrent Handbook for general information regarding *rTorrent*, its configuration, and XMLRPC commands. The latter includes custom command extensions from *rTorrent-PS*.

Hint: Community Contacts and Solving Problems

To get in contact and share your experiences with other users of PyroScope, join the pyroscope-users mailing list or the inofficial `##rtorrent` channel on `irc.freenode.net`.

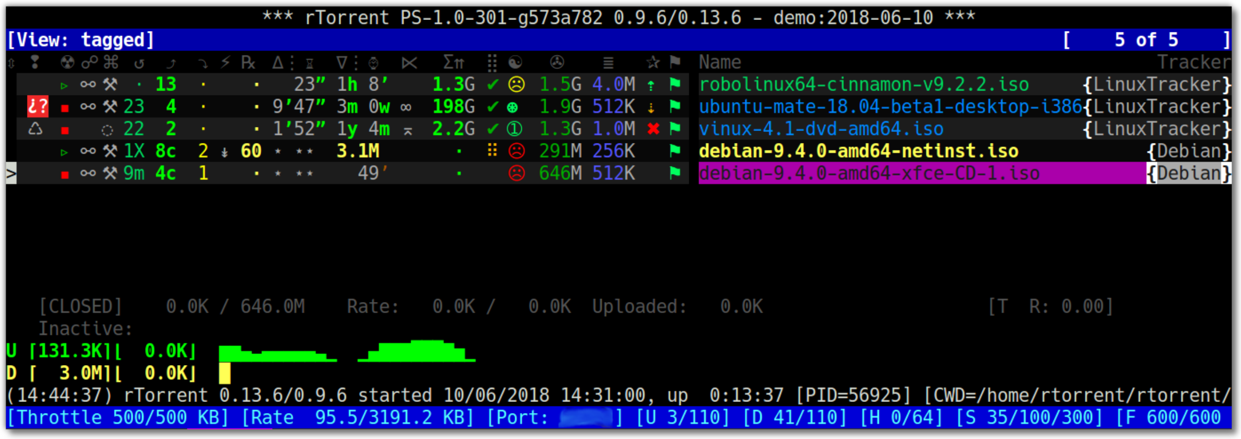
This is also the way to resolve any problems with or questions about your configuration and software installation. *Always* look into the *Trouble-Shooting Guide* as a first measure, which is often the fastest way to get back to a working system. That guide also explains how to efficiently report your problem when you cannot fix it yourself.

CHAPTER 1

Full Contents

Overview

rTorrent-PS is a rTorrent *distribution* (*not* a fork of it), in form of a set of patches that **improve the user experience and stability** of official `rTorrent` releases, and **releases new features more frequently**. The *Feature Overview* below lists the major improvements and extensions. Also see the changelog for a timeline of applied changes, especially those since the last official release.



```
                   *** rTorrent PS-1.0-301-g573a782 0.9.6/0.13.6 - demo:2018-06-10 ***
[View: tagged]                                                              [    5 of 5    ]
⇳ ❢  ☢ ☍ ⌘  ↺  ⤴  ⤵ ⚡ ℞  ∆⋮ ⌛  ∇⋮ ⌚  ⋉  Σ⇈  ⣿ ☯   ✇    ≣   ☆ ⚑  Name                                Tracker
     ▹ ⚯ ⚒  · 13  ·      ·   23″ 1h 8′    1.3G ✔ ☹  1.5G 4.0M ⇡ ⚑  robolinux64-cinnamon-v9.2.2.iso     {LinuxTracker}
 ¿?  ■ ⚯ ⚒ 23  4  ·      · 9′47″ 3m 0w ∞  198G ✔ ⊛  1.9G 512K ⇣ ⚑  ubuntu-mate-18.04-beta1-desktop-i386{LinuxTracker}
 ♺   ■   ◌ 22  2  ·      · 1′52″ 1y 4m ⌅  2.2G ✔ ①  1.3G 1.0M ✖ ⚑  vinux-4.1-dvd-amd64.iso             {LinuxTracker}
     ▹ ⚯ ⚒ 1X 8c  2 ⇣ 60 ⋆ ⋆⋆  3.1M        ·  ⣿ ☹ 291M 256K   ⚑  debian-9.4.0-amd64-netinst.iso            {Debian}
>    ■ ⚯ ⚒ 9m 4c  1      · ⋆ ⋆⋆    49′     ·    ☹ 646M 512K   ⚑  debian-9.4.0-amd64-xfce-CD-1.iso          {Debian}



   [CLOSED]     0.0K / 646.0M     Rate:    0.0K /    0.0K  Uploaded:    0.0K               [T  R: 0.00]
   Inactive:
U [131.3K][   0.0K]
D [   3.0M][   0.0K]
(14:44:37) rTorrent 0.13.6/0.9.6 started 10/06/2018 14:31:00, up  0:13:37 [PID=56925] [CWD=/home/rtorrent/rtorrent/
[Throttle 500/500 KB] [Rate  95.5/3191.2 KB] [Port: [redacted]] [U 3/110] [D 41/110] [H 0/64] [S 35/100/300] [F 600/600
```

Note that *rTorrent-PS* is *not* the same as the *PyroScope* command line utilities (*pyrocore*), and doesn't depend on them; the same is true the other way 'round. It's just that both unsurprisingly have synergies if used together, and some features *do* only work when both are present.

To get in contact and share your experiences with other users of PyroScope, join the pyroscope-users mailing list or the inofficial `##rtorrent` channel on `irc.freenode.net`.

This is also the way to resolve any problems with or questions about your configuration and software installation. *Always* look into the *Trouble-Shooting Guide* as a first measure, which is often the fastest way to get back to a working system. That guide also explains how to efficiently report your problem when you cannot fix it yourself.

Feature Overview

The main changes compared to vanilla rTorrent are these:

- **self-contained installation** into any location of your choosing, including your home directory, offering the ability to **run several versions at once** (in different client instances).
- **rpath-linked** to the major dependencies, so you can upgrade those independently from your OS distribution's versions.
- **extended command set**:
  - sort views by more than one value, and set the sort direction for each of these.
  - (re-)bind keys in the downloads display to any command, e.g. change the built-in views.
  - record and display network traffic.
  - … and many more, see *Command Extensions* for details.
- interface additions:
  - easily **customizable colors**.
  - **collapsed 1-line item display** with a condensed information presentation.
  - almost **fully customizable downloads display** (selection of columns & adding new ones).
  - **network bandwidth graph**.
  - **displaying the tracker domain** for each item.
  - some more minor modifications to the download list view.

To get those features, you just need to either follow the build instructions, or download and install a package from Bintray — assuming one is available for your platform. See the *Installation Guide* on how to get your machine set up.

Supported Platforms

The tested and officially supported platforms are *Debian* and any Debian-related Linux distro, specifically *Ubuntu*. *Arch Linux* is supported via contributors, and *Fedore 26* has some limited support – anything else vanilla rTorrent runs on will likely work too, but in the end *you* have to fix any problems that might occur. If you want better support for your platform, provide a *working* derivative of `Dockerfile.Debian` and necessary modifications to `build.sh` via a PR.

Manual Turn-Key System Setup provides instructions to set up a working *rTorrent-PS* instance in combination with `pyrocore`, on Debian and most Debian-derived distros — i.e. a manual way to do parts of what pimp-my-box does automatically for you. Another option for automatic setup of a similar system is the one by chros73.

To help out other users, you can add your own content with configuration tricks and the like to the project's wiki, and show to the world you're awesome.

Launching a Demo in Docker

To try out *rTorrent-PS* without any headaches, and *if you have Docker 17.06+ installed*, you can use the launch-on-stretch.sh script to run an *ephemeral* instance in a *Debian Stretch* container, like this:

```
git clone "https://github.com/pyroscope/rtorrent-ps.git" ~/src/rtorrent-ps
~/src/rtorrent-ps/docker/launch-on-stretch.sh
```

Detach from the process using `Ctrl-P Ctrl-Q`, and call `reset` to reset your terminal.

Reattach with `docker attach rtps-on-stretch`, then enter `Ctrl-A r` to refresh the `tmux` screen.

If you want to do that directly via SSH from a remote machine, call `ssh -t YOU@HOST "docker attach rtps-on-stretch"` and then refresh with `Ctrl-A r`.

Another use for such a container instance, besides taking a first look, is trying out experimental configurations, without any impact on your main installation.

Incidentally, that script is also a condensed version of the *Manual Turn-Key System Setup* instructions, and can be used as a blue-print for your own `Dockerfile`.

Installation Guide

Note that this cannot be a Linux shell 101, so if the terminology and commands that follow are new for you, refer to the usual sources like The Debian Administrator's Handbook, The Linux Command Line, and The Art of Command Line to get yourself acquainted.

General Installation Options

See *Build from Source* on using the provided `build.sh` script, which will install *rTorrent-PS* into `~/.local/rtorrent/‹version›`.

The stable rTorrent version **0.9.6** is built by default, but 0.9.4 is also supported (but not tested anymore). And not all patches are applied equally, depending on whether they're needed, or applicable at all.

After installation, make sure to read through the *Setup & Configuration* chapter in order to get the display-related changes set up correctly, since on many machines this requires some special configuration of your terminal.

Also take note of the pimp-my-box project that does it all (almost) automatically for Debian-type systems (and is the preferred way to install on those systems). The automation is done using Ansible, which implies you can easily admin several systems with it, and also maintain them – so it's not a one-shot installation bash script creating a setup that can never be changed again.

Note: If you also install the PyroScope command line utilities, do not forget to activate the extended features available together with *rTorrent-PS*, as mentioned in the Configuration Guide. Starting with *version 1.1*, that activation is automatic.

OS-Specific Installation Options

Installation Using Debian Packages

For a limited set of Debian-derived platforms, there are packages available that contain pre-compiled binaries (and only those, no configuration or init scripts). You can download and install such a package from Bintray — assuming one is available for your platform. The packages install the *rTorrent-PS* binary including some libraries into `/opt/rtorrent`.

Example on *Debian Stretch*:

```
version="0.9.6-PS-1.0-272-g388cfab~stretch_amd64"
bintray="https://bintray.com/artifact/download/pyroscope"
cd /tmp
```

```
curl -Lko rt-ps.deb "$bintray/rtorrent-ps/rtorrent-ps_$version.deb"
dpkg -i rt-ps.deb
```

After installation, you must provide a configuration file (`~/.rtorrent.rc`), and either use the absolute path to the binary to start it, or link it into `/usr/local` like this:

```
ln -s /opt/rtorrent/bin/rtorrent /usr/local/bin
```

Note: You can safely install the package and test it out in parallel to an existing installation, just use the absolute path `/opt/rtorrent/bin/rtorrent` to start *rTorrent*. Your (session) data is in no way affected, as long as you normally run a *0.9.x* version.

Installation on Arch Linux

There are now two options contributed by xsmile for installing on *Arch* via `pacman`.

1. The `pkg2pacman` command of *The Build Script* creates a package similar to the *Debian* one, embedding a tested version combination of dependencies. See *Building the Debian Package* for general instructions on building that variant, and use `pkg2pacman` instead of `pkg2deb`.
2. The *"Arch User Repository"* (AUR) PKGBUILDs maintained by @xsmile. These use a standard *Arch* build process, but include the usual *rTorrent-PS* patches.

   There is one package for `libtorrent-ps`, and one for `rtorrent-ps`, and both *take their dependencies from the normal OS packages*:

   - https://aur.archlinux.org/packages/libtorrent-ps/
   - https://aur.archlinux.org/packages/rtorrent-ps/

Before building binaries or packages yourself, install these packages on top of the `base` and `base-devel` groups (**list is user-provided, report any problems**):

```
pacman -S \
    lsb-release subversion git time lsof tmux wget \
    python2-setuptools python2-virtualenv python2 python2-cffi \
    cppunit libxml2 libxslt
```

There is also the rtorrent-pyro-git AUR package. It is *not* the same as you get from using `build.sh`, and not recommended anymore by *this* project, given the new options above.

If you have problems with building or installing any of these packages, contact *their maintainer*.

Homebrew Tap for Mac OSX

See the homebrew-rtorrent-ps repository for instructions to build *rTorrent-PS* and related dependencies on Mac OSX.

Right now, it is not maintained by anyone.

Manual Turn-Key System Setup

Introduction

The following shows installation instructions for a working *rTorrent* instance in combination with *PyroScope* **from scratch**, on *Debian* and most Debian-derived distros. Note that the pimp-my-box project does all this automatically for you, and is the tested and maintained way of installation — this page is just a reference of the core installation steps (if you run into problems, join the `freenode` IRC channel for help).

While the package names and the use of `apt-get` are somewhat dependent on *Debian*, the *Preparatory Steps* commands which are executed under `root` are similar for other distributions, and the compilation instructions should work as-is on practically any Linux and (F)BSD. These instructions are explicitly known to work on *Debian Jessie + Stretch*, and *Ubuntu Xenial + Bionic*.

The whole procedure takes 15 – 20 minutes, including full compilation from source. Subtract about 5 minutes if you install *rTorrent* via a package. This on a quad-core 3.3 GHz Xeon CPU with 32 GiB RAM, and assuming you are familiar with the procedure, or just blindly paste the command blocks that follow. Add plenty of reading time when doing your first setup, and it's still under an hour.

Note: If you don't understand a word of what follows, hit The Debian Administrator's Handbook so then you do.

Non-packaged software is installed exclusively into your normal user account (home directory), i.e. this description works OK for non-root users as long as the required packages are installed before-hand. The default install location is `~/.local/rtorrent/«version»`, which means you can easily delete any installed software, and also run several versions concurrently.

For shared multi-user setups, this works fine also — compile and install to `/opt/rtorrent` using `./build.sh install`, then provide access to all users by calling `chmod -R go+rX /opt/rtorrent`. Perform the steps from *PyroScope Installation* onwards for each user repeatedly, so they get their own instance.

Important: Most of the command blocks further below can be copied & pasted wholesale into a terminal. Note that `bash` *here documents* (`... <<'EOF'`) **MUST** be pasted at once, up to and including the line having a single `EOF` on it.

Warning: If you have an existing `/usr/local` installation of *rTorrent* / *libtorrent*, it is *very* prudent to `make uninstall` that before compiling another version. Those *might* prevent successful compilation if your lookup paths somehow bring those versions to the front.

In the same vein, remove any packages of `libtorrent` and `rtorrent` you have on your machine. The build instructions on this page then ensure that it is *no* problem to have several versions concurrently on your machine. If anything goes wrong, you can easily reinstall the packages provided by your OS.

Preparatory Steps

Setting Up Locales

Commonly locales are already set up for you, but bare-bones installs often come without locale support, which *rTorrent-PS* absolutely requires due to its use of *Unicode* characters.

This ensures at least the common `en_US.UTF-8` one is available:

```
apt-get install locales
test "$LANG" = "en_US.UTF-8" \
    || { echo "en_US.UTF-8 UTF-8" >>/etc/locale.gen ; locale-gen --lang en_US.UTF-8; }
```

Installing Build Dependencies

You need to install a few **required** packages — **and no, this is not optional in any way**. These are the only steps that must be performed by the `root` user (i.e. in a root shell, or by writing `sudo` before the actual command):

```
apt-get install sudo lsb-release build-essential pkg-config \
    subversion git time lsof binutils tmux curl wget \
    python-setuptools python-virtualenv python-dev \
    libssl-dev zlib1g-dev libncurses-dev libncursesw5-dev \
    libcppunit-dev autoconf automake libtool \
    libffi-dev libxml2-dev libxslt1-dev
```

Note that you can always show Debian's current build dependencies for rTorrent using this command:

```
echo $(apt-cache showsrc rtorrent libtorrent-dev | \
    grep Build-Depends: | cut -f2 -d: | tr ",)" " \\n" | cut -f1 -d"(")
```

On *Fedora* (26), use this (**list is user-provided, report any problems**):

```
dnf install -y \
    redhat-lsb-core make autoconf automake libtool gcc gcc-c++ pkgconf-pkg-config \
    subversion git time lsof binutils tmux curl wget which \
    python-setuptools python-virtualenv python-devel python2-cffi \
    openssl-devel zlib-devel ncurses-devel cppunit-devel libxml2-devel libxslt-devel
```

For *Arch*, see the `pacman` command in *Installation on Arch Linux*.

Optional `root` Setup Steps

If you're security-conscious, you can create a `rtorrent` user and do all the following setup steps under that new account. Doing that ensures that there is *no way*, on a properly maintained nix system, for the build and setup scripts to break either your machine or your normal user account.

```
groupadd rtorrent
useradd -g rtorrent -G rtorrent,users -c "rTorrent client" \
        -s /bin/bash --create-home rtorrent
chmod 750 ~rtorrent
su - rtorrent -c "mkdir -p ~/bin"
```

rTorrent Installation

Install via Debian Packages

See *Installation Using Debian Packages* above for details. After adding the right package for your platform, skip the next section and continue with *PyroScope Installation*.

Note: During *rTorrent* instance setup, do not forget to change the value of `pyro.extended` to 1 so the extended features are actually accessible! Starting with *version 1.1*, that activation is automatic.

Build from Source

Get the build script via direct download or a `git clone`, and call it with the `all` parameter as shown below; the script will then download, build, and install all necessary components, storing temporary files in the current directory. You can pass the `clean_all` parameter to remove those temporary files later on, after everything works. Make sure you followed the *Preparatory Steps* in the section further up on this page.

Note: Be sure to select the version of rTorrent you want to compile, as determined by the settings at the start of the script. If you have no preference otherwise, stick to the default set in the script. Note that such a choice is sticky once you performed the `download` step, until you call `clean_all` again.

All installations go to `~/.local/rtorrent/«version»/`, and disturb neither any host setup nor another version of rTorrent you've installed the same way.

```
# Run this in your NORMAL user account, or as 'rtorrent'!
mkdir -p ~/src/; cd $_
git clone https://github.com/pyroscope/rtorrent-ps.git
cd rtorrent-ps

# Use this if you have the resources, adapt for the number of cores
# and the amount of free memory you have available.
export MAKE_OPTS="-j4"

# Check the VERSION SELECTION at the top of the script, and edit as needed
nice time ./build.sh all  # build 'deps', 'vanilla', and then 'extended'
```

Note that the unpatched version is still available as `rtorrent-vanilla`, and you can simply switch by changing the symlink in `~/bin`, or by calling either version with its full path. See the *User's Manual* for more details on the changes applied.

The Build Script describes more use-cases like building in *Docker*, or an incremental update after a `git fetch` with new *rTorrent-PS* changes.

Note: If you use the configuration as outlined below, do not forget to change the value of `pyro.extended` to 1 in case you want to unlock the additional features of the extended version! Starting with *version 1.1*, that activation is automatic.

PyroScope Installation

The installation of `pyrocore` is done from source, see its manual for more details.

```
# Run this in your NORMAL user account, or as 'rtorrent'!
mkdir -p ~/bin ~/.local
git clone "https://github.com/pyroscope/pyrocore.git" ~/.local/pyroscope

# Pass "/usr/bin/python2", or whatever else fits, to the script as its
# 1st argument, if the default of "/usr/bin/python" is not a suitable
# version.
~/.local/pyroscope/update-to-head.sh

# Check success
exec $SHELL -l
pyroadmin --version
```

The last call's output should look similar to this:

```
$ pyroadmin --version
pyroadmin 0.6.1.dev20180601 on Python 2.7.13
```

rTorrent Instance Setup

To be able to use several different instances of *rTorrent* (e.g. a second one for experimental configuration changes), this setup doesn't use `~/.rtorrent.rc` at all, but keeps everything in one place under the `~/rtorrent` directory. If you change the assignment to `RT_HOME`, you can place it anywhere you like, or create alternate instances with ease.

rTorrent Startup Script

First, create the instance's directories and a start script:

```
# Run this in your NORMAL user account, or as 'rtorrent'!
export RT_HOME="${RT_HOME:-$HOME/rtorrent}"
mkdir -p $RT_HOME; cd $_
mkdir -p .session log work done watch/{start,load,hdtv,cleaned}
cp ~/.local/pyroscope/docs/examples/start.sh ./start
chmod a+x ./start
```

Note that this script is needed on modern systems, else the special installation layout allowing concurrent use of several versions will not work as expected. So always call that script, and not `rtorrent` directly.

Tip: Safely storing downloads on a mounted device

In case your data resides on a mounted device (e.g. an external USB disk), **add a check to the start script** that it is actually present. To do that, create a `.mounted` file in the root of your device, and `exit` the start script if not found. For your convenience, the code for that is already there at the top of `start`, but commented out.

If you don't check, that might lead to rehashing several terabytes of data, because *rTorrent* will mark the downloads stored on an absent device as broken (which they are without their data).

rTorrent Configuration

Next, a not-so-simple rtorrent.rc is created. It already provides everything needed to use all features of the *PyroScope* tools.

Note that built-in `pyrocore` settings are read from a provided include file, that in turn loads snippets from the `~/.pyroscope/rtorrent.d` directory. The same mechanism is used in the main `rtorrent.rc` file, so you can easily add your own customizations in new `rtorrent.d/*.rc` files.

To get all this set up for you, call this provided script:

```
# Run this in your NORMAL user account, or as 'rtorrent'!
~/.local/pyroscope/src/scripts/make-rtorrent-config.sh
```

After this, you should check at least the `rtorrent.d/20-host-var-settings.rc` file and adapt the values to your environment and preferences. Consider copying the commands for the settings you want to adapt to the `_rtlocal.rc` file – read on as to why.

The `_rtlocal.rc` file is the place for some simple custom settings, like additional resource limits or changing default values. The `make-rtorrent-config.sh` script does not copy that optional file. So create it yourself, and pick what you like from the example _rtlocal.rc, e.g. the logging configuration.

The script can be called again to get updates from *GitHub*, **but will overwrite all standard configuration files** with their new version. To safely customize configuration, provide your own version of standard files and list the replaced files in the `.rcignore` file, add your own *additional* files, or as mentioned use the `_rtlocal.rc` file.

Example for a `~/rtorrent/_rtlocal.rc` file:

```
# Reduce retention period of uncompressed logs
pyro.log_archival.days.set = 1
```

Note: In `rtorrent.rc`, change the value of `pyro.extended` to 1 so the extended *rTorrent-PS* features are actually accessible! Starting with *version 1.1*, that activation is automatic.

CLI Tools Configuration

This adds a minimal configuration, so that the defaults are taken from the installed software, which makes later updates a lot easier.

```
# Run this in your NORMAL user account, or as 'rtorrent'!
pyroadmin --create-config

cat >~/.pyroscope/config.ini <<EOF
# PyroScope configuration file
#
# For details, see https://pyrocore.readthedocs.org/en/latest/setup.html
#

[GLOBAL]
# Location of your rTorrent configuration
rtorrent_rc = ~/rtorrent/rtorrent.rc

# XMLRPC connection to rTorrent
scgi_url = scgi://$HOME/rtorrent/.scgi_local

[FORMATS]
filelist = {{py:from pyrobase.osutil import shell_escape as quote}}{{#
    }}{{for i, x in looper(d.files)}}{{d.realpath | quote}}/{{x.path | quote}}{{#
        }}{{if i.next is not None}}{{chr(10)}}{{endif}}{{#
    }}{{endfor}}

movehere = {{py:from pyrobase.osutil import shell_escape as quote}}{{#
    }}mv {{d.realpath | quote}} .

# Formats for UI commands feedback
tag_show = {{#}}Tags: {{ chr(32).join(d.tagged) }} [{{ d.name[:33] }}...]

[SWEEP]
# Settings for the "rtsweep" tool

# Use the rules from the named [SWEEP_RULES_<name>] sections
default_rules = builtin, custom
```

```
# Minimum amount of space that must be kept free (adds to the space request)
space_min_free = 10g

[SWEEP_RULES_CUSTOM]
# Rules to manage disk space
#
# Rules are ordered by the given priority. You can disable built-in rules
# found in the [SWEEP_RULES_BUILTIN] section by changing "default_rules"
# in the [SWEEP] section. Use "rtsweep show" to list active rules.
#
# Default sort order for each rule is by "loaded" date (oldest first).
# Note that active, prio 3, and ignored items are protected!
#
# If the active rules fail to provide enough space, as much of the oldest
# items as needed are removed.

# Seeded and bigger than 500M after 7 days, inactive and big items first
seeded7d.prio   = 910
seeded7d.sort   = active,-size
seeded7d.filter = ratio=+1.2 size=+500m loaded=+5d

[ANNOUNCE]
# Add alias names for announce URLs to this section; those aliases are used
# at many places, e.g. by the "mktor" tool and to shorten URLs to these aliases

# Public / open trackers
PBT     = http://tracker.publicbt.com:80/announce
          udp://tracker.publicbt.com:80/announce
PDT     = http://files2.publicdomaintorrents.com/bt/announce.php
ArchOrg = http://bt1.archive.org:6969/announce
          http://bt2.archive.org:6969/announce
OBT     = http://tracker.openbittorrent.com:80/announce
          udp://tracker.openbittorrent.com:80/announce
Debian  = http://bttracker.debian.org:6969/announce
Linux   = http://linuxtracker.org:2710/
EOF
```

Read the pyrocore Configuration Guide for more information regarding this file. You can come back to customizing it later, the system will work fine with the above default.

First Start and Testing

tmux Configuration

We spruce up `tmux` a bit using a custom configuration, before we start it the first time. This also makes it more homey for long-time `screen` users:

```
# Run this in your NORMAL user account, or as 'rtorrent'!
cp --no-clobber ~/.local/pyroscope/docs/examples/tmux.conf ~/.tmux.conf
```

Starting a tmux Session

You're now ready to start your shiny new *rTorrent-PS*, so just do it:

```
# Run this in your NORMAL user account, or as 'rtorrent'!
tmux -2u new -n rT-PS -s rtorrent "~/rtorrent/start; exec bash"
```

The `exec bash` keeps your `tmux` window open if `rtorrent` exits, which allows you to actually read any error messages in case it ends *unexpectedly*. If such an error occurs (e.g. about your terminal not providing enough colors), check out *Setup & Configuration* and the *Trouble-Shooting Guide* for a fix.

After that, test the XMLRPC connection by using this command in a new `tmux` window:

```
# Open a new tmux terminal window by pressing "Ctrl-a" followed by "c", and then...
rtxmlrpc system.time_usec
```

You can of course add more elaborate start scripts, like a cron watchdog, init.d scripts, or a systemd unit. Put the above `tmux` call into `ExecStart`, and use `...  new -d ...` to run a detached session – see the rTorrent wiki for detailed examples.

Continue with reading the 'pyrocore' manual to get acquainted with that, and *Setup & Configuration* for providing the necessary configuration regarding your terminal.

Setup & Configuration

The main part of configuration regarding *rTorrent-PS* itself is already done, if you followed *Manual Turn-Key System Setup* or used pimp-my-box for it. If you used neither, look into what make-rtorrent-config.sh does (see also *rTorrent Configuration*), in order to get all the features described in the *User's Manual*, which in large part rely on those standard configuration snippets.

This chapter provides some background on the standard configuration and how you can tweak it, and contains hints on what you might need to do regarding the runtime environment and your system setup.

You can skip to the *next chapter* to learn about the special *rTorrent-PS* features and come back to this later, provided everything looks OK to you when you first started *rTorrent-PS* (especially that all special characters render correctly).

Setting up Your Terminal Emulator

General Concerns

There are two major obstacles for a proper display of the extended canvas, and that is selecting the right font(s) and providing a terminal setup that supports 256 or more colors.

Read the following sections on how to solve any problems you might encounter within your environment.

Font Selection & Encoding Issues

Whatever font you use in your terminal profile, it has to support the characters used in the column headers and some of the displayed values, else you're getting a degraded user experience. Also, your terminal **must** be set to use UTF-8, which nowadays usually is the default anyway.

On *Linux*, that means `LANG` should be something like `en_US.UTF-8`, and `LC_ALL` and `LC_CTYPE` should **not** be set at all! If you use a terminal multiplexer like most people do, and the display doesn't look right, try `tmux -u` or `screen -U` to force UTF-8 mode.

Also make sure you have the `locales` package installed on Debian-type systems, and the `en_US.UTF-8` locale actually created. See *Setting Up Locales* for that.

The following command lets you easily check whether your font supports the most important characters and your terminal is configured correctly:

```
PYTHONIOENCODING=utf-8 python -c 'print(u"\u22c5 \u22c5\u22c5 \u201d \u2019 \u266f␣
↪\u2622 " \
    u"\u260d \u2318 \u2730 \u28ff \u26a1 \u262f \u2691 \u21ba \u2934 \u2935 \u2206␣
↪\u231a " \
    u"\u2240\u2207 \u2707 \u26a0\xa0\u25d4 \u26a1\xa0\u21af \xbf \u2a02 \u2716 \u21e3␣
↪\u21e1 " \
    u"\u2801 \u2809 \u280b \u281b \u281f \u283f \u287f \u28ff \u2639 \u2780 \u2781␣
↪\u2782 " \
    u"\u2783 \u2784 \u2785 \u2786 \u2787 \u2788 \u2789 \u25b9\xa0\u254d \u25aa \u26af␣
↪\u2692 " \
    u"\u25cc \u21c5 \u21a1 \u219f \u229b \u267a ")'
```

In case you have unsolvable problems with only a few specific glyphs, see *Defining Your Own Columns* below on how to change them to ones working for you, or even switch them to plain ASCII.

Terminal Setup on Windows

To get full coverage of all Unicode glyphs used in the *extended canvas*, the steps below show you how to use font linking to make `Everson Mono` complement `DejaVu Sans Mono` when used in `PuTTY` version 0.70 or higher.

1. Download and install the DejaVu Sans Mono and Everson Mono fonts.
2. Next, add or edit a multi-string value for your preferred font under this Windows registry key:

   ```
   HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows␣
   ↪NT\CurrentVersion\FontLink\SystemLink
   ```

   To do that start `regedit` and go to that folder. Any key there names a font for which fallback fonts are registered. So add/edit the `DejaVu Sans Mono` key, and add fallback font references as its value.

   - Right click `SystemLink` and select `New`, then `Multi String Value`.
   - Use `DejaVu Sans Mono` as the key's name.
   - Double-click the new key, and enter this as the value:

     ```
     Everson Mono.ttf,EversonMono
     ```

3. After closing `regedit`, logout from Windows and back in again to activate the font link, but a full reboot is safer (hey, it's Windows, you should be used to it).
4. Start `PuTTY` and select `Change settings` from the menu.
   - In `Window › Appearance` select `DejaVu Sans Mono`.
   - Set `UTF-8` in `Window / Translation`
   - Under `Terminal` check `Use background colour to erase screen`.
   - In `SSH › Data`, make sure to use `putty-256color` for the `terminal` setting.
5. Connect, and check the display.

Other fonts that were suggested are `Andale Mono`, and `GNU Unifont`. You have to try out yourself what looks good to you and works with your specific system and terminal emulator. Read more about fallback fonts on *superuser.com*.

– based on feedback by @NoSubstitute, with help from superuser.com and MSDN

See also:

Font linking on Windows and Using KiTTY instead of PuTTY

Terminal Setup on Linux

When you use `gnome-terminal`, everything should work out of the box, given you use the `start` script, which sets `TERM` and `LANG` correctly. Also always call `tmux` with the `-2u` options.

If you use `urxvt`, you have to provide fallback fonts as on *Windows*. Add the following to your `~/.Xresources`:

```
URxvt*font: xft:DejaVu Sans Mono:style=regular:pixelsize=15,xft:Noto Sans Mono CJK␣
↪JP:pixelsize=15,xft:FreeSerif
```

Note that *15pt* is a threshold for the font size, below it `urxvt` thinks there's not enough space to render the glyphs.

Generally, to cope with problems like this or find other fonts that suit you better, the `ttfdump` tool can help to check out fonts on the technical level. Another helper is the `gucharmap` GUI tool, that allows you to explore your installed fonts visually.

– based on feedback by @ymvunjq

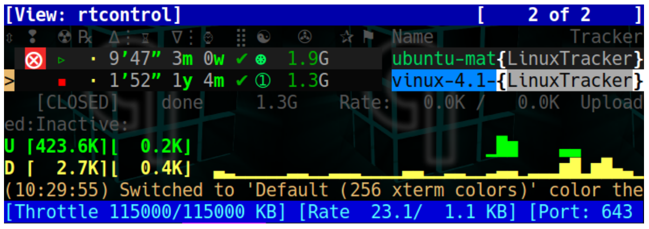
Color Scheme Configuration

This section describes how you can change all the colors in *rTorrent-PS* to your liking. Note that there are existing color schemes provided by the *pyrocore* configuration, and you can rotate through them using the ~ key thanks to rtorrent.d/theming.rc.

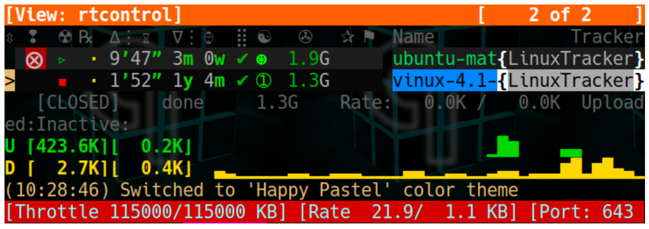
You can copy one of the provided color scheme `*.rc.default` files in `~/.pyroscope/color-schemes/` to a new `‹mytheme›.rc` file. Details about how to specify colors and so on are in the sub-sections that follow.

Color Configuration Details

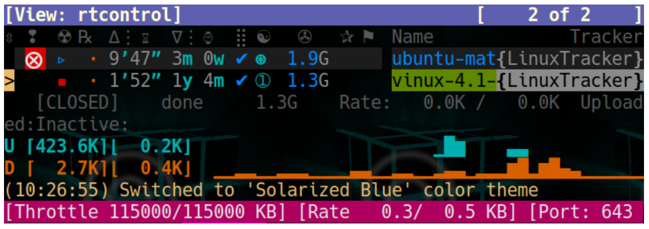
And here are some visuals of the default schemes…



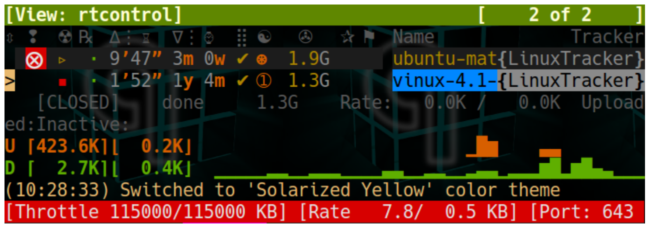
```
[View: rtcontrol]                            [     2 of 2     ]
                                      Name              Tracker
  ⊗ ▷  · 9’47” 3m 0w ✓ ⊛  1.9G        ubuntu-mat{LinuxTracker}
>    ■ · 1’52” 1y 4m ✓ ①  1.3G        vinux-4.1-{LinuxTracker}
   [CLOSED]   done     1.3G    Rate:  0.0K /   0.0K  Upload
ed:Inactive:
U [423.6K][  0.2K]
D [  2.7K][  0.4K]
(10:29:55) Switched to 'Default (256 xterm colors)' color the
[Throttle 115000/115000 KB] [Rate  23.1/  1.1 KB] [Port: 643
```



```
[View: rtcontrol]                            [     2 of 2     ]
                                      Name              Tracker
  ⊗ ▷  · 9’47” 3m 0w ✓ ⊛  1.9G        ubuntu-mat{LinuxTracker}
>    ■ · 1’52” 1y 4m ✓ ①  1.3G        vinux-4.1-{LinuxTracker}
   [CLOSED]   done     1.3G    Rate:  0.0K /   0.0K  Upload
ed:Inactive:
U [423.6K][  0.2K]
D [  2.7K][  0.4K]
(10:28:46) Switched to 'Happy Pastel' color theme
[Throttle 115000/115000 KB] [Rate  21.9/  1.1 KB] [Port: 643
```



```
[View: rtcontrol]                         [     2 of 2    ]
⇕ ❢  ☢ ℞ Δ ⋮ ⌛  ∇ ⋮ ⌚  ⣿ ☯   ✇   ☆ ⚑  Name              Tracker
  ⊗  ▹   · 9’47” 3m 0w ✔ ⊛  1.9G       ubuntu-mat{LinuxTracker}
>    ▪   · 1’52” 1y 4m ✔ ①  1.3G       vinux-4.1-{LinuxTracker}
    [CLOSED]    done     1.3G    Rate:   0.0K /   0.0K  Upload
ed:Inactive:
U [423.6K][  0.2K]
D [  2.7K][  0.4K]
(10:26:55) Switched to 'Solarized Blue' color theme
[Throttle 115000/115000 KB] [Rate   0.3/  0.5 KB] [Port: 643
```



```
[View: rtcontrol]                         [     2 of 2    ]
⇕ ❢  ☢ ℞ Δ ⋮ ⌛  ∇ ⋮ ⌚  ⣿ ☯   ✇   ☆ ⚑  Name              Tracker
  ⊗  ▹   · 9’47” 3m 0w ✔ ⊛  1.9G       ubuntu-mat{LinuxTracker}
>    ▪   · 1’52” 1y 4m ✔ ①  1.3G       vinux-4.1-{LinuxTracker}
    [CLOSED]    done     1.3G    Rate:   0.0K /   0.0K  Upload
ed:Inactive:
U [423.6K][  0.2K]
D [  2.7K][  0.4K]
(10:28:33) Switched to 'Solarized Yellow' color theme
[Throttle 115000/115000 KB] [Rate   7.8/  0.5 KB] [Port: 643
```

Supporting 256 or More Colors

Having 256 colors available means you can select very dark shades of grey, and that can be used for subtle even / odd line backgrounds in the collapsed canvas of *rTorrent-PS*.

To enable 256 colors, your terminal must obviously be able to support them at all (i.e. have a `xterm-256color` terminfo entry, or similar). But even if that is the case, you often need to give a little nudge to the terminal multiplexers; namely start `tmux` with the `-2` switch (that forces 256 color mode), or for `screen` start it with the terminal already set to 256 color mode so it can sense the underlying terminal supports them, i.e. use this in your startup script:

```
if [ "$TERM" = "${TERM%-256color}" ]; then
    export TERM="$TERM-256color"
fi
tmux ...
```

Then, within the terminal multiplexer's environment, you must **again** ensure the terminal is set to a 256 color terminfo entry. See the .tmux.conf by @chros73 for possible solutions for any tmux-related problems.

The reward for jumping through all those hoops is that you can then use color gradients for ratio coloring, and much more appropriate pallid color shades for backgrounds.

Showing a Terminal's Palette

The term-256color.py script can help you with showing the colors your terminal supports, an example output using Gnome's terminal looks like the following...

Working With Color Schemes

If your terminal works as intended, you now might want to find you own color theme. The easiest way is to use a second shell and `rtxmlrpc`. Try out some colors, and add the combinations you like to your `~/.rtorrent.rc`.

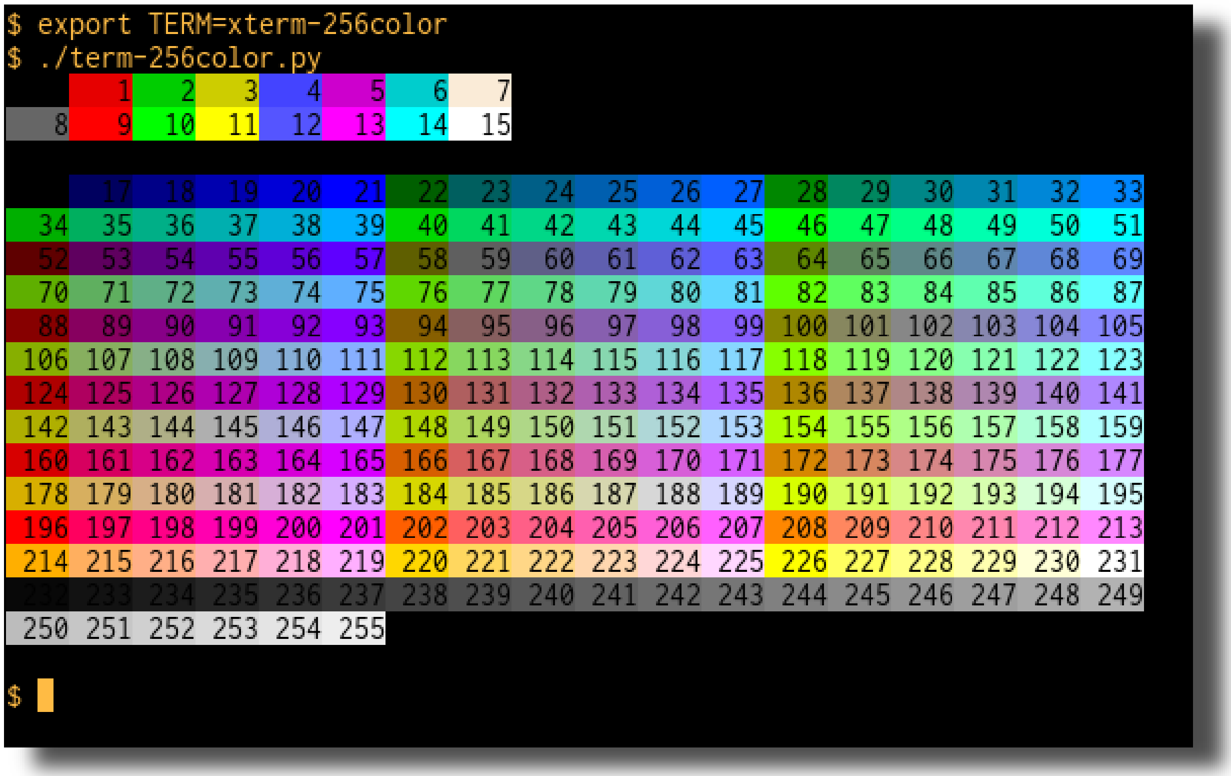
```
# For people liking candy stores...
rtxmlrpc ui.color.title.set "bold magenta on bright cyan"
```

You can use the following code in a terminal to dump a full color scheme from the running client:

```
for i in $(rtxmlrpc system.listMethods | egrep '^ui.color.[^.]+$'); do
    echo $i = $(rtxmlrpc -r $i | tr "'" '"') ;
done
```

Adding Your Own Color Schemes

Color schemes are lists of `ui.color.‹color name›.set` commands in your configuration, or in a `*.rc[.default]` file in the `~/.pyroscope/color-schemes/` directory. If you want to use scheme rotation at all (via the ~ key), put your own schemes into extra `color-schemes/*.rc` files and not the main configuration.



```
$ export TERM=xterm-256color
$ ./term-256color.py
       1   2   3   4   5   6   7
   8   9  10  11  12  13  14  15

      17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33
  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69
  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87
  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105
 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123
 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141
 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177
 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195
 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213
 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231
 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249
 250 251 252 253 254 255

$
```

Fig. 1.1: Output of **term-256-color.py**

The set of color names is predetermined and refers to the *meaning* of the color, i.e. where it is used on the canvas. `footer` and `alarm` are obvious examples.

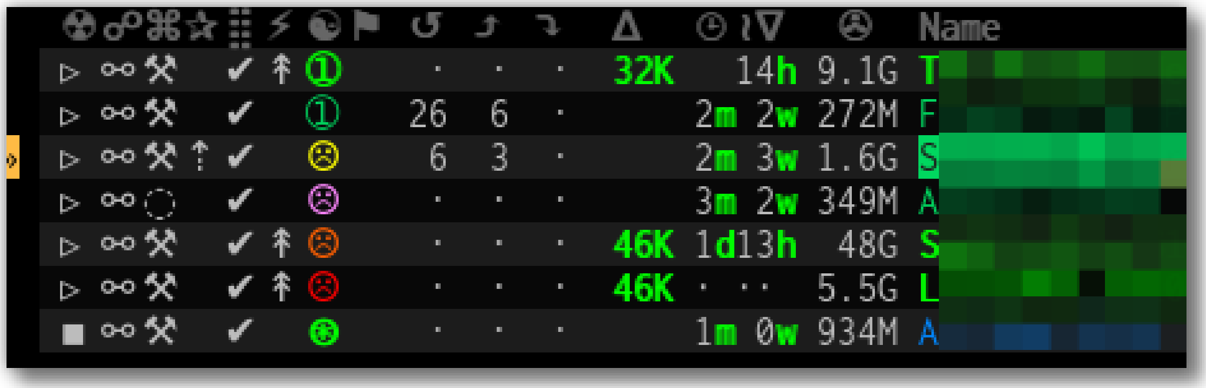
Here's a configuration example showing all the commands and their defaults:

```
# UI/VIEW: Colors
ui.color.alarm.set="bold white on red"
ui.color.complete.set="bright green"
ui.color.even.set=""
ui.color.focus.set="reverse"
ui.color.footer.set="bold bright cyan on blue"
ui.color.incomplete.set="yellow"
ui.color.info.set="white"
ui.color.label.set="gray"
ui.color.leeching.set="bold bright yellow"
ui.color.odd.set=""
ui.color.progress0.set="red"
ui.color.progress20.set="bold bright red"
ui.color.progress40.set="bold bright magenta"
ui.color.progress60.set="yellow"
ui.color.progress80.set="bold bright yellow"
ui.color.progress100.set="green"
ui.color.progress120.set="bold bright green"
ui.color.queued.set="magenta"
ui.color.seeding.set="bold bright green"
ui.color.stopped.set="blue"
ui.color.title.set="bold bright white on blue"
```

See the ui.color.* command reference for details on these and related commands.

The following color settings work better than the default ones in a 256 color terminal (gnome-terminal), for me at least. Your mileage (color table) may vary. Having 256 colors means you have very dark shades of grey, and that is used here to set the even / odd backgrounds.

```
ui.color.complete.set=41
ui.color.stopped.set=33

ui.color.footer.set="bright cyan on 20"
ui.color.even.set="on 234"
ui.color.odd.set="on 232"

ui.color.progress0.set=196
ui.color.progress20.set=202
ui.color.progress40.set=213
ui.color.progress60.set=214
ui.color.progress80.set=226
ui.color.progress100.set=41
ui.color.progress120.set="bold bright green"
```



```
  ☢☍⌘☆⣿⚡☯⚑ ↺  ⤴  ⤵   Δ    ⌚ ≀∇   ☹  Name
 ▷ ⚯ ⚒   ✔ ↟ ①  ·  ·  ·  32K   14h 9.1G T
 ▷ ⚯ ⚒   ✔   ①  26 6  ·      2m 2w 272M F
›▷ ⚯ ⚒ ⇡ ✔   ☹  6  3  ·      2m 3w 1.6G S
 ▷ ⚯ ◌   ✔   ☹  ·  ·  ·      3m 2w 349M A
 ▷ ⚯ ⚒   ✔ ↟ ☹  ·  ·  ·  46K 1d13h  48G S
 ▷ ⚯ ⚒   ✔ ↟ ☹  ·  ·  ·  46K ·  ··  5.5G L
 ■ ⚯ ⚒   ✔   ☺  ·  ·  ·      1m 0w 934M A
```

Note that you might need to enable support for 256 colors in your terminal, see *Supporting 256 or More Colors* for a description. In a nutshell, you need to install the `ncurses-term` package if you don't have it already, and also add these commands to your *rTorrent* start script:

```
if [ "$TERM" = "${TERM%-256color}" ]; then
    export TERM="$TERM-256color"
fi
```

Customizing the Display Layout

Canvas v2 Topics

Canvas v2 Overview

The main display with the downloads list is flexible and can be configured to your will, in *rTorrent-PS 1.1* and up. This is also known as *canvas v2*.

Use the following rtxmlrpc command to check if you have a version that can do this:

```
$ rtxmlrpc "system.has=,canvas_v2"
1
# The '1' means you have canvas v2 on board;
# a '0' or "Method 'system.has' not defined" means you don't.
```

The only fixed parts are the position indicator at the very left of the display, and the combined name / tracker column on the right. The latter takes all the space left by other columns.

Inspecting Your Display

To list the columns you have in your setup, call rtxmlrpc like so:

```
$ rtxmlrpc method.get=,ui.column.render | sed -re 's/ /␣/g' | sort
100:3C95/2:␣␣
110:2C92/2:␣
120:?2:␣
130:?2:␣
400:?3C23/3:␣␣
410:?3C24/3:␣␣
420:?3C14/3:␣␣
500:?2:␣
510:3C28/3:℞␣␣
520:6C96/6:␣␣␣
530:6C90/6:␣␣␣
800:3:␣
900:?5C24/3C21/2:␣Σ␣␣
910:2C94/2:␣
920:3C93/3:␣␣
930:5C15/3C21/2:␣␣␣␣
970:2C91/2:␣
980:2C16/2:␣
```

The important thing here are the numbers in front, which define the sort order of columns from left to right. They also allow to address a specific column, which becomes important in a moment.

All these are built-in defaults, except the throttle indicator with index 800, which is defined in ~/rtorrent/rtorrent.d/05-rt-ps-columns-v2.rc.include of pimp-my-box.

Important: You **MUST** update your pimp-my-box configuration if you used that to set up your system. Otherwise you'll get duplicate columns.

To show the full column definitions with their code, call pyroadmin:

```
$ pyroadmin --dump-rc | grep -A1 ui.column.render | egrep '^(method.set_key|    )'
method.set_key = ui.column.render, "100:3C95/2:  ", \
    ((array.at, {"  ", " ", " ", " ", " ", " ", "¿?", " "}, ((d.message.alert)) ))
method.set_key = ui.column.render, "110:2C92/2: ", \
    ((string.map, ((cat, ((d.is_open)), ((d.is_active)) )), {00, ""}, ..., {11, ""}))
...
method.set_key = ui.column.render, "980:2C16/2: ", \
    ((array.at, {" ", ""}, ((d.views.has, tagged)) ))
```

Column Layout Definitions

The keys of the `ui.column.render` multi-command must follow a defined format, namely ‹`index`›:⟨?⟩‹`width`›⟨‹`color definition`›⟩:‹`title`›. There are three fields, separated by colons. The parts in ⟨`...`⟩ are optional.

‹`index`› was already mentioned, used for sorting and addressing columns.

The second field can start with a `?` to tag this column as 'sacrificial', i.e. optional in the face of too narrow terminals. ‹`width`› is a column's width in characters. The ‹`color definition`› determines what terminal attributes are used to render these characters, and is a sequence of `C`‹`color index`›/‹`length`› elements.

Finally, ‹ `title` › is used for the column's heading. Make sure to end it with a space to leave room for wide Unicode glyphs, and always make it as long as the column width.

To get a color index table, try this command:

```
rtxmlrpc system.has.private_methods \
    | egrep '^ui.color.*index$' \
    | xargs -I+ rtxmlrpc -i 'print="+ = ",(+)'
```

Since the `ui.color.*index` commands are private, the output must go to the *rTorrent* console. This is what you'll see (timestamps removed):

```
ui.color.alarm.index = 22
ui.color.complete.index = 23
ui.color.custom1.index = 1
ui.color.custom2.index = 2
ui.color.custom3.index = 3
ui.color.custom4.index = 4
ui.color.custom5.index = 5
ui.color.custom6.index = 6
ui.color.custom7.index = 7
ui.color.custom8.index = 8
ui.color.custom9.index = 9
ui.color.even.index = 30
ui.color.focus.index = 19
ui.color.footer.index = 18
ui.color.incomplete.index = 27
ui.color.info.index = 21
ui.color.label.index = 20
ui.color.leeching.index = 28
ui.color.odd.index = 29
ui.color.progress0.index = 10
ui.color.progress20.index = 11
ui.color.progress40.index = 12
ui.color.progress60.index = 13
ui.color.progress80.index = 14
ui.color.progress100.index = 15
ui.color.progress120.index = 16
ui.color.queued.index = 26
ui.color.seeding.index = 24
ui.color.stopped.index = 25
ui.color.title.index = 17
```

There are also columns with *dynamic* color schemes, using a color index 90, which map to a 'normal' color index depending on an item's attributes. An example is `3C95/2` for the alert column, which changes to red (`ui.color.alarm`) if there is an active alert.

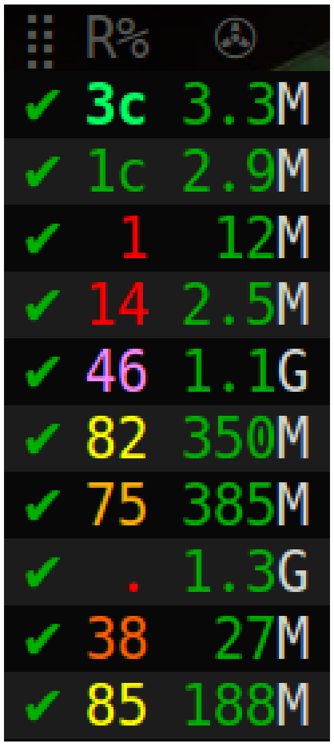
This is a list of the dynamic color schemes:

- 90: `DOWN_TIME` – Download (*leeching*) or time display (*info* + *seeding*/*incomplete*)
- 91: `PRIO` – A color for , depending on `d.priority`: *progress0*, *progress60*, *info*, *progress120*
- 92: `STATE` – A color for , depending on `d.is_open` (*progress0* if not) and `d.is_active` (*progress80* or *progress100*)
- 93: `RATIO` – A *progress* color for from 0 to 120
- 94: `PROGRESS` – A *progress* color from 0 to 100 for the column
- 95: `ALERT` – For , *info* or *alarm* depending on alert state

- 96: `UP_TIME` – Upload (*seeding*) or time display (*info* + *seeding*/*incomplete*)

The mixed `DOWN_TIME` and `UP_TIME` schemes must span the full width of the column, and can only be used with *one* color definition in the column key (anything after them is ignored).

Defining Your Own Columns



```
  R%
✓ 3c 3.3M
✓ 1c 2.9M
✓  1  12M
✓ 14 2.5M
✓ 46 1.1G
✓ 82 350M
✓ 75 385M
✓  . 1.3G
✓ 38  27M
✓ 85 188M
```

This example shows how to replace the ratio column (920) with a pure ASCII version. You can see the result on the right.

Place this code in your custom configuration, e.g. in the `_rtlocal.rc` file (when using pimp-my-box).

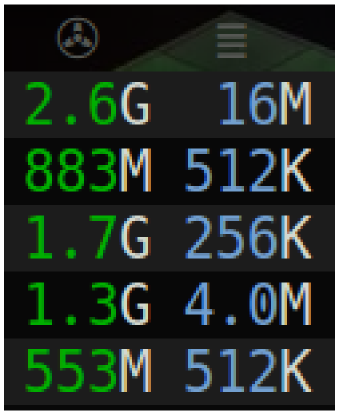
```
# Remove the default column
method.set_key = ui.column.render, (ui.column.spec, 920)

# Add ASCII ratio in percent
# (1..99 for incomplete; 1c = 1.0; 1m = 10.0; ...)
method.set_key = ui.column.render, "922:3C93/3:R% ", \
    ((string.replace, ((convert.magnitude, ((math.div, ((d.ratio)), 10)) )), \
                     {"", "."} ))
```

To construct a column definition like this, you need to understand rTorrent Scripting first – more so than what's sufficient for writing simple configurations.

Looking at the original column definition often helps, e.g. to grab a few snippets for your own version:

```
$ pyroadmin --dump-rc | egrep -A1 '"920:.+"'
method.set_key = ui.column.render, "920:3C93/3:  ", \
    ((string.substr, "           ", \
                    ((math.mul, 2, ((math.div, ((d.ratio)), 1000)) )), 2, " "))
```

Also, try to understand how all the other column definitions work, you can learn a few tricks that are typical for column rendering.



```
2.6G  16M
883M 512K
1.7G 256K
1.3G 4.0M
553M 512K
```

Especially if you want to display additional values in the same format as an existing column, you just have to swap the command accessing the displayed item's data. Here's a chunk size column, all you need to do is replace `d.size_bytes` in the code of column 930 with `d.chunk_size`, and give it a new index and heading.

```
ui.color.custom9.set = "bright blue"
method.set_key = ui.column.render, "935:5C9/3C21/2:    ", \
    ((convert.human_size, ((d.chunk_size)) ))
```

That example also shows how to use a custom color.

Disabling Columns

The `ui.column.show` and `ui.column.hide` commands provide the means to easily change the visibility of columns, without touching their definition. They both take a list of column keys as their arguments, as either strings or values.

The following example shows column 42 only on the *active* and *leeching* views,

```
method.set_key = event.view.show, ~column_42_toggle, \
    "branch = \"string.contains=$ui.current_view=, active, leeching\", \
        ui.column.show=42, ui.column.hide=42"
ui.column.hide = 42
```

The `ui.column.is_hidden` and `ui.column.hidden.list` commands can be used to query the visibility of columns, the first one takes a single column key as its argument.

```
$ rtxmlrpc --repr ui.column.is_hidden '' 42
1
$ rtxmlrpc --repr ui.column.hidden.list
[42]
```

A practical use of `ui.column.is_hidden` is to toggle a column. This code does so for 935, and binds the toggle to the _ key.

```
method.insert = pmb._toggle_chunk_size, simple|private, \
    "branch = ui.column.is_hidden=935, ui.column.show=935, ui.column.hide=935 ; \
     ui.current_view.set = (ui.current_view)"
pyro.bind_key = toggle_chunk_size, _, "pmb._toggle_chunk_size="
```

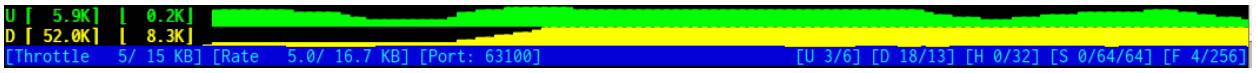
The `ui.current_view.set = (ui.current_view)` part forces a redraw of the canvas, giving you instant feedback.

Adding Traffic Graphs

Add these lines to your configuration:

```
# Show traffic of the last hour
network.history.depth.set = 112
schedule = network_history_sampling,1,32, network.history.sample=
method.insert = network.history.auto_scale.toggle, simple|private, \
    "branch=network.history.auto_scale=, \
        \"network.history.auto_scale.set=0\", \
        \"network.history.auto_scale.set=1\""
method.insert = network.history.auto_scale.ui_toggle, simple|private, \
    "network.history.auto_scale.toggle= ;network.history.refresh="
branch=pyro.extended=,"schedule = bind_auto_scale,0,0, \
    \"ui.bind_key=download_list,=,network.history.auto_scale.ui_toggle=\""
```

And you'll get this in your terminal:



```
U [  5.9K]  [  0.2K]
D [ 52.0K]  [  8.3K]
[Throttle   5/ 15 KB] [Rate   5.0/ 16.7 KB] [Port: 63100]          [U 3/6] [D 18/13] [H 0/32] [S 0/64/64] [F 4/256]
```

Fig. 1.2: rTorrent-PS Network History

As you can see, you get the upper and lower bounds of traffic within your configured time window, and each bar of the graph represents an interval determined by the sampling schedule. Pressing = toggles between a graph display with a baseline of 0, and a zoomed view that scales it to the current bounds.

User's Manual

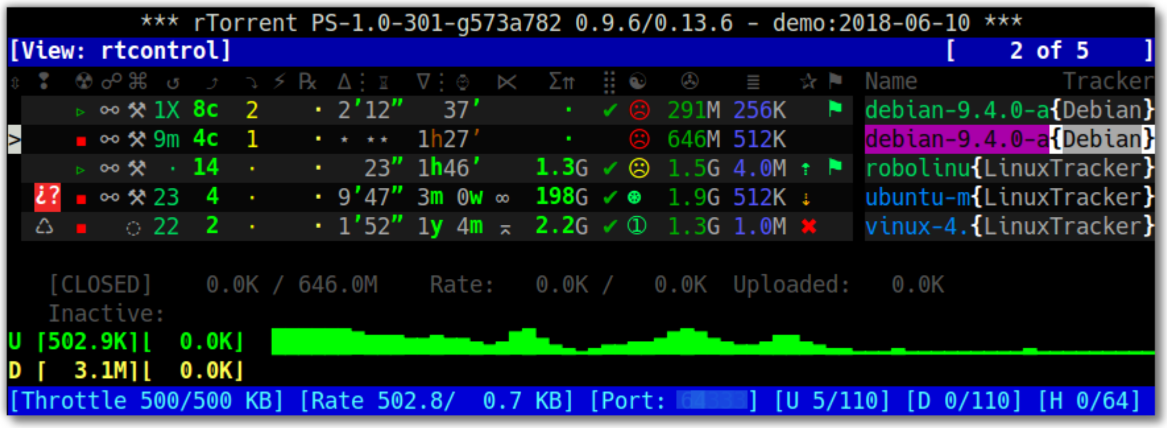
This chapter describes the additional features in *rTorrent-PS*, and other differences to a vanilla *rTorrent* build.

Additional Features

Using the right default configuration (more on that below), you will get the following additional features in your *rTorrent-PS* installation:

1. The `t` key is bound to a `trackers` view that shows all items sorted by tracker and then by name.
2. The `!` key is bound to a `messages` view, listing all items that currently have a non-empty message, sorted in order of the message text.
3. The `^` key is bound to the `rtcontrol` search result view, so you can easily return to your last search.
4. The `?` key is bound to the `indemand` view, which sorts all open items by their activity, with the most recently active on top.
5. The `%` key is bound to the `ratio` view, which sorts all open items by their ratio (descending).
6. The `°` key is bound to the `uploaded` view, which sorts all open items by their total upload amount (descending).
7. The `"` key is bound to the `datasize` view, which sorts all open items by the size of their content data (descending).

8. Add even more views, see Additional Views for details.
9. `Page ↑` and `Page ↓` scroll by 50 items at a time (or whatever other value `ui.focus.page_size` has).
10. `Home` / `End` jump to the first / last item in the current view.
11. The `~` key rotates through all available color themes, or a user-selected subset. See Color Themes for details.
12. The `<` and `>` keys rotate through all added category views (`pyro.category.add=‹ name ›`), with filtering based on the ruTorrent label (`custom_1=‹ name ›`). See Category Views for details.
13. `|` reapplies the category filter and thus updates the current category view.
14. The `u` key shows the uptime and some other essential data of your rTorrent instance.
15. `F2` shows some important help resources (web links) in the console log.
16. `*` toggles between the collapsed (as described on *Extended Canvas Explained*) and the expanded display of the



```
         *** rTorrent PS-1.0-301-g573a782 0.9.6/0.13.6 - demo:2018-06-10 ***
[View: rtcontrol]                                                  [    2 of 5    ]
⇕ ❢  ☢ ☍ ⌘  ↺  ⤴  ⤵ ⚡ ℞  Δ⋮ ⌛  ∇⋮ ⌚  ⋉  Σ⇈  ⣿ ☯  ✇  ≣  ☆ ⚑  Name         Tracker
     ▹ ∞ ⚒ 1X 8c  2    · 2'12"   37'      ·  ✓ ☹ 291M 256K   ⚑ debian-9.4.0-a{Debian}
>    ▪ ∞ ⚒ 9m 4c  1    · ⋆ ⋆⋆  1h27'      ·    ☹ 646M 512K     debian-9.4.0-a{Debian}
     ▹ ∞ ⚒  · 14  ·    ·    23" 1h46'  1.3G ✓ ☹ 1.5G 4.0M ↑ ⚑ robolinu{LinuxTracker}
  ¿? ▪ ∞ ⚒ 23  4  ·    · 9'47" 3m 0w ∞ 198G ✓ ⊛ 1.9G 512K ↓   ubuntu-m{LinuxTracker}
  ♺  ▪   ◌ 22  2  ·    · 1'52" 1y 4m ⌅ 2.2G ✓ ① 1.3G 1.0M ✖   vinux-4.{LinuxTracker}

  [CLOSED]    0.0K / 646.0M    Rate:   0.0K /    0.0K  Uploaded:    0.0K
  Inactive:
U [502.9K][   0.0K]
D [   3.1M][   0.0K]
[Throttle 500/500 KB] [Rate 502.8/  0.7 KB] [Port: [redacted]] [U 5/110] [D 0/110] [H 0/64]
```

current view.

17. `/` toggles the visibility of 'sacrificial' columns – normally, they're only hidden when the terminal width gets too small.
18. The `active` view is changed to include all incomplete items regardless of whether they have any traffic, and then groups the list into complete, incomplete, and queued items, in that order. Within each group, they're sorted by download and then upload speed.
19. Some *canvas v2* columns are added in the *pimp-my-box* configuration – the selected throttle (), a download's chunk size (), and the expected time of arrival () on the *active* and *leeching* displays only. The visibility of the chunk size column can be toggled using the _ key.
20. The commands `s=«keyword»`, `t=«tracker_alias»`, and `f=«filter_condition»` are pre-defined for searching using a Ctrl-X prompt.
21. The `.` key toggles the membership in the `tagged` view for the item in focus, `:` shows the `tagged` view, and `T` clears that view (i.e. removes the tagged state on all items). This can be very useful to manually select a few items and then run `rtcontrol` on them, or alternatively use `--to-view tagged` to populate the `tagged` view, then deselect some items interactively with the `.` key, and finally mass-control the rest. See Additional Views for details.
22. You can use the `purge=` and `cull=` commands (on a Ctrl-X prompt) for deleting the current item and its (incomplete) data.
23. `Ctrl-g` shows the tags of an item (as managed by `rtcontrol`); `tag.add=‹ tag ›` and `tag.rm=‹ tag ›` can be used to change the set of tags, both also show the new set of tags after changing them.
24. Time-stamped log files with rotation, archival (compression), and pruning – with a setting for the number of days to keep logs in uncompressed form, or at all.

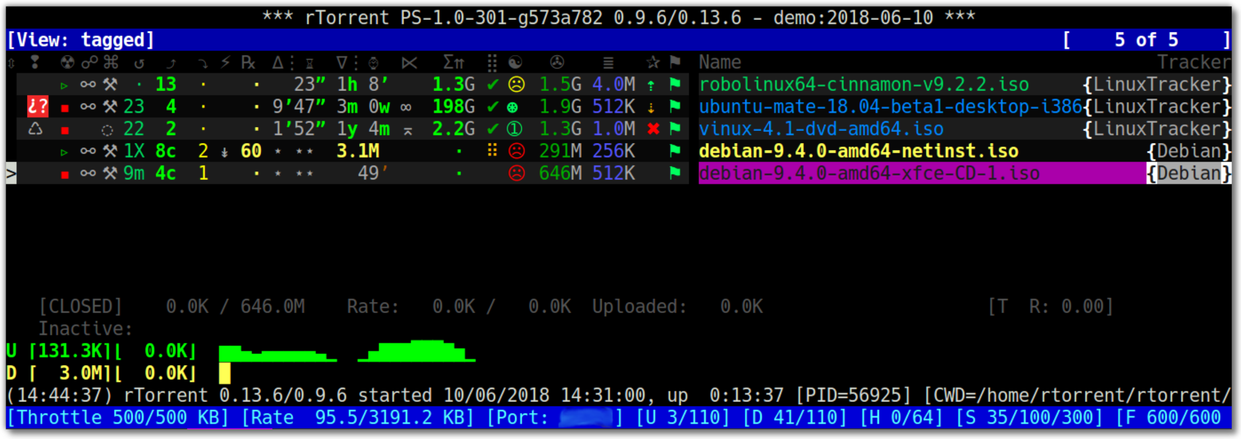
25. Trackers are scraped regularly (active items relatively often, inactive items including closed ones seldomly), so that the display of downloads / seeders / leechers is not totally outdated. The `&` key can be used to manually scrape the item in focus.
26. A watchdog for the `pyrotorque` daemon process (checks every 5 minutes, and starts it when not running *if* the *~/.pyroscope/run/pyrotorque* file exists).

With regards to using the 'right' configuration to get the above, you need the `*.rc.default` files in the `~/.pyroscope/rtorrent.d` directory provided by *pyrocore*. Standard Configuration Explained has details on these. Some more features are defined by the pimp-my-box configuration templates.

To get there, perform the *Manual Turn-Key System Setup* as described in this manual, or use the pimp-my-box project for an automatic remote installation. The instructions in the Extending your '.rtorrent.rc' section of the *pyrocore* manual only cover half of it, and you might miss some described features.

Extended Canvas Explained

The following is an explanation of the collapsed display of *rTorrent-PS* (*canvas v2*).



```
                    *** rTorrent PS-1.0-301-g573a782 0.9.6/0.13.6 - demo:2018-06-10 ***
[View: tagged]                                                                    [    5 of 5    ]
⇕ ❢  ☢ ☍ ⌘  ↺  ⤴  ⤵ ⚡ ℞  Δ ⋮ ⌛   ∇ ⋮ ⌚  ⋉   Σ⇈  ⣿ ☯   ✇    ≣     ☆ ⚑  Name                          Tracker
     ▹ ⚯ ⚒  · 13   ·     ·   23” 1h 8’    1.3G ✔ ☹  1.5G 4.0M ↑  ⚑  robolinux64-cinnamon-v9.2.2.iso   {LinuxTracker}
 ¿?  ▪ ⚯ ⚒ 23  4   ·     · 9’47” 3m 0w ∞  198G ✔ ⊛  1.9G 512K ↓  ⚑  ubuntu-mate-18.04-beta1-desktop-i386{LinuxTracker}
 ♺   ▪    ◌ 22  2   ·     · 1’52” 1y 4m ⌅  2.2G ✔ ①  1.3G 1.0M ✖  ⚑  vinux-4.1-dvd-amd64.iso           {LinuxTracker}
     ▹ ⚯ ⚒ 1X 8c  2 ⇣ 60 ⋅ ⋅⋅    3.1M        · ⣿ ☹  291M 256K    ⚑  debian-9.4.0-amd64-netinst.iso          {Debian}
>    ▪ ⚯ ⚒ 9m 4c  1     · ⋅ ⋅⋅     49’       ·   ☹  646M 512K    ⚑  debian-9.4.0-amd64-xfce-CD-1.iso        {Debian}




   [CLOSED]    0.0K / 646.0M    Rate:   0.0K /   0.0K  Uploaded:   0.0K                  [T  R: 0.00]
   Inactive:
U [131.3K][  0.0K]
D [  3.0M][  0.0K]
(14:44:37) rTorrent 0.13.6/0.9.6 started 10/06/2018 14:31:00, up  0:13:37 [PID=56925] [CWD=/home/rtorrent/rtorrent/
[Throttle 500/500 KB] [Rate  95.5/3191.2 KB] [Port: [redacted]] [U 3/110] [D 41/110] [H 0/64] [S 35/100/300] [F 600/600
```

Fig. 1.3: Extended Canvas Screenshot

In case your screen looks broken compared to this, see *Setting up Your Terminal Emulator* for necessary pre-conditions and settings regarding your terminal emulator application and operating system.

In older builds, you need to remember to press the `*` key while showing a view, or change the state of new views after adding them (by calling the view.collapsed.toggle command), else you won't ever see it.

The following is an overview of the default columns, and what the values and icons in them mean.

A after the column title indicates a 'sacrificial' column, which disappear when the display gets too narrow to display all the columns. When even that does not provide enough space, columns are omitted beginning on the right side (*Name* is always included). Sacrificial columns can also be toggled using the `/` key – note they're toggled as a whole group, so other dynamic states like the column toggle are ignored.

 Message or alert indicator (= Tracker cycle complete, i.e. "Tried all trackers"; = establishing connection; = data transfer problem; = timeout; ¿? = unknown torrent / info hash; = authorization problem (possibly temporary); = tracker downtime; = DNS problems; = other)

 Item state (= started, = paused, = stopped)

 Tied item? []

 Command lock-out? (= heed commands, = ignore commands)

Number of completions from last scrape info

Number of seeds from last scrape info

Number of leeches from last scrape info

Transfer direction indicator [ ]

℞ Number of connected peers

Upload rate, or when inactive, time the download took (only after completion).

Approximate time since completion (units are «"'hdwmy» from seconds to years); for incomplete items the download rate or, if there's no traffic, the time since the item was started or loaded

Expected time of arrival – only shown on the *active* and *leeching* displays

Throttle selected for this item (∞ is the special `NULL` throttle; … for *ruTorrent*'s `thr_0...9` channels)

Σ Total sum of uploaded data

Completion status (= done; else up to 8 dots [] and , i.e. progress in 10% steps); the old `ui.style.progress.set` command is deprecated, see *Defining Your Own Columns* for the new way to get a different set of glyphs or an ASCII version

Ratio (plus color indication for < 1, — : >= the number, : >= 11); the old `ui.style.ratio.set` command is deprecated, see *Defining Your Own Columns* for the new way to get a different set of number glyphs or an ASCII version

Data size

Chunk size - this column can be toggled on / off using the _ key

Priority (= off, = low, nothing for normal, = high)

A indicates this item is on the `tagged` view

Name Name of the download item – either the name contained in the metafile, or else the value of the `displayname` custom field when set on an item

Tracker Domain of the first HTTP tracker with seeds or leeches, or else the first one altogether – note that you can define nicer aliases using the trackers.alias.set_key command in your configuration

For the various time displays to work, you need the *pyrocore* standard configuration for rtorrent.rc.

The scrape info and peer numbers are exact only for values below 100, else they indicate the order of magnitude using roman numerals (c = 10^2, m = 10^3, X = 10^4, C = 10^5, M = 10^6). For up-to-date scrape info, you need the Tracker Auto-Scraping configuration from *pyrocore*.

Command Extensions

The following new commands are available. Note that the links point to the Commands Reference chapter in the *rTorrent Handbook*.

*math.** Commands

- math.add
- math.avg
- math.cnt
- math.div
- math.max
- math.med

- math.min
- math.mod
- math.mul
- math.sub

*string.** Commands

- string.contains
- string.contains_i
- string.endswith
- string.equals
- string.join
- string.len
- string.lpad
- string.lstrip
- string.map
- string.replace
- string.rpad
- string.rstrip
- string.split
- string.startswith
- string.strip
- string.substr

*convert.** Commands

- convert.human_size
- convert.magnitude
- convert.time_delta

*system.** Commands

- system.client_version.as_value
- system.env
- system.has
- system.has.list
- system.has.private_methods
- system.has.public_methods
- system.random

*d.** Commands

- d.custom.as_value
- d.custom.erase
- d.custom.if_z
- d.custom.items
- d.custom.keys
- d.custom.set_if_z
- d.custom.toggle
- d.is_meta

- d.message.alert
- d.multicall.filtered
- d.tracker_alias
- d.tracker_domain
- d.tracker_scrape.complete
- d.tracker_scrape.downloaded
- d.tracker_scrape.incomplete

*network.** Commands

- network.history.auto_scale
- network.history.depth
- network.history.depth.set
- network.history.refresh
- network.history.sample

*ui.** Commands

- ui.bind_key
- ui.bind_key.verbose
- ui.canvas_color
- ui.canvas_color.set
- ui.color.*.index
- ui.color.*.set
- ui.color.alarm…title
- ui.column.hidden.list
- ui.column.hide
- ui.column.is_hidden
- ui.column.render
- ui.column.sacrificed
- ui.column.sacrificial.list
- ui.column.show
- ui.column.spec
- ui.current_view
- ui.focus.end
- ui.focus.home
- ui.focus.page_size
- ui.focus.pgdn
- ui.focus.pgup
- ui.style.progress
- ui.style.ratio

Other Commands

- array.at
- compare
- do
- import.return
- log.messages
- throttle.names
- trackers.alias.items

- trackers.alias.set_key
- value
- view.collapsed.toggle

Tips & How-Tos

Checking Details of the Standard Configuration

Not every detail of the standard configuration can be documented in this manual, because at some point it would basically duplicate the `*.rc` files in English, and also run the risk of being out of date (i.e. wrong) very fast.

To understand every nook and cranny of the features added by the config sets that get installed if you follow *Manual Turn-Key System Setup*, look into the `*.rc` files and specifically read the comments in them.

The `pyrocore` files are in the `~/.pyroscope/rtorrent.d` directory, and `pimp-my-box` ones in `~/rtorrent/rtorrent.d`.

Remember that `.rcignore` files in those directories allow to selectively disable parts of the standard configuration, like so:

```
echo >>~/rtorrent/rtorrent.d/.rcignore "disable-control-q.rc"
```

For finding specifics on added commands or paths and such, `grep` is your friend, as in this example:

```
$ grep -C1 import.rc ~/.pyroscope/rtorrent.d/*.rc ~/rtorrent/rtorrent.d/*.rc ~/
↪rtorrent/*rt*rc
rtorrent/rtorrent.rc-execute2 = (cat,(pyro.bin_dir),pyroadmin),-q,--create-import,
↪(cat,(cfg.basedir),"rtorrent.d/*.rc")
rtorrent/rtorrent.rc:import = (cat,(cfg.basedir),"rtorrent.d/.import.rc")
rtorrent/rtorrent.rc-
```

Validate Self-Signed Certs

The following helps in case you want to use trackers with `https` announce URLs that still use self-signed certificates, instead of *Let's Encrypt* like they should. Unless you disable certificate checks in the rTorrent configuration, you must add such certificates to your the system to make them valid.

This script does that via an easy command call, just pass it the tracker domain or URL:

```
cat >/usr/local/sbin/load-domain-certificate <<'EOF'
#! /bin/bash
if test -z "$1"; then
    echo "usage: $0 <domainname_or_url>"
    exit 1
fi
DOMAINNAME=$(sed -re 's%^(https://)?([^/]+)(.*)$%\2%' <<<$1)
set -x
openssl s_client -servername ${DOMAINNAME} -connect ${DOMAINNAME}:443 </dev/null |␣
↪tee /tmp/${DOMAINNAME}.crt
openssl x509 -inform PEM -in /tmp/${DOMAINNAME}.crt -text -out /usr/share/ca-
↪certificates/${DOMAINNAME}.crt
grep ${DOMAINNAME}.crt /etc/ca-certificates.conf >/dev/null || echo ${DOMAINNAME}.crt␣
↪>>/etc/ca-certificates.conf
update-ca-certificates
ls -l /etc/ssl/certs | grep ${DOMAINNAME}
```

```
EOF
chmod a+x /usr/local/sbin/load-domain-certificate
```

Development Guide

This chapter contains some explanations of the project structure and activities related to development of the project.

Running Integration Tests

The `tasks.py` script defines a `test` task for *invoke*, which runs tests located in the `tests/commands/*.txt` files.

These test scripts are edited console logs of shell calls, and the test runner executes any line starting with a `$`. It then checks that any line following the command is contained in its output. The return code is checked via a `RC=n` line.

Any line starting with `#` is a comment.

The lines asserting output can contain `...` to mark omissions – any whitespace around it is ignored. That means that `foo ... bar` checks that *both* `foo` and `bar` are contained somewhere in the command's output.

Here's the sample output you want to have (no failures):

```
$ invoke test
--- Running tests in 'tests/commands/array.txt'...
.......

--- Running tests in 'tests/commands/misc.txt'...
................................................

--- Running tests in 'tests/commands/string.txt'...
....................

--- Running tests in 'tests/commands/math.txt'...
.......................................

    ALL OK.
```

And this is what a failure looks like:

```
$ invoke test -n misc
--- Running tests in 'tests/commands/misc.txt'...
.
FAIL: »!1’20”« not found in output of »rtxmlrpc convert.time_delta '' +1527903580␣
↪+1527903500«
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1’20”
RC=0
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

................................................

    1 TEST(S) FAILED.
```

This also shows how you can run only one selected test suite, using the `--name` or `-n` option.

The Build Script

The build.sh script contains all the minute details and settings to successfully build selected dependencies and the *libtorrent* / *rtorrent* source on many platforms.

And no, it is not a *Makefile*, since there's no benefit in that, for the things the script does. `[g]make` is used *within* the contributing projects.

See *Build from Source* on the 'normal' use of the script for building a binary. Other uses like building packages (with or without *Docker*) or the most recent upstream source are described in the following sections.

To **build a new extended binary after you downloaded updates** via a `git pull --ff-only`, just call `./build.sh extend` – this will apply any new patches included in that update, but not re-build all the dependencies.

If you're sure that the diff only contains source code changes (in `patches/*.cc`), only calling `make` in the *rTorrent* source directory is *way faster*. In case dependency versions changed in `build.sh`, you have to go the slowest route with `./build.sh clean_all all` to get them onto your machine.

For everything else, call `./build.sh help` to get a usage summary similar to this:

```
$ ./build.sh help
Environment for building rTorrent PS-1.0-349-g12ccbe8-2018-06-30-1143 0.9.6/0.13.6
export PACKAGE_ROOT=/opt/rtorrent
export INSTALL_ROOT=/home/pyroscope
...

Usage: ./build.sh (all | clean | clean_all | download | build | check | extend)
Build rTorrent PS-1.0-... 0.9.6/0.13.6 into ~/.local/rtorrent/0.9.6-PS-1.1-dev

Custom environment variables:
    CURL_OPTS="-sLS" (e.g. --insecure)
    MAKE_OPTS="-j4"
    CFG_OPTS="" (e.g. --enable-debug --enable-extra-debug)
    CFG_OPTS_LT="" (e.g. --disable-instrumentation for MIPS, PowerPC, ARM)
    CFG_OPTS_RT=""

Build actions:
    build_all    a/k/a ‹all› - Download and build and install all deps + vanilla +␣
↪extended
    build        Build and install all components
    build_git    a/k/a ‹git› - Build and install libtorrent and rtorrent from git␣
↪checkouts
    check        Print some diagnostic success indicators
    clean_all    Remove all downloads and created files
    clean        Clean up generated files
    deps         Build all dependencies
    deps_git     Build all dependencies [GIT HEAD MODE]
    docker_deb   Build Debian packages via Docker
    download     Download and unpack sources
    env          Show build environement
    extend       Rebuild and install libtorrent and rTorrent with patches applied
    install      Install to /opt/rtorrent
    pkg2deb      Package current /opt/rtorrent installation for APT [needs fpm]
    pkg2pacman   Package current /opt/rtorrent installation for PACMAN [needs fpm]
    vanilla      Build vanilla rTorrent [also un-patches src dirs]
```

Creating a Release

- Finish `docs/CHANGES.md` and set the release date
- Run `invoke cmd_docs >docs/include-commands.rst`, and commit any additions
- Make sure every command has docs in the manual (`invoke undoc`)
- Tag the release, push tags, put changelog up on GitHub
- Build packages and upload to Bintray (using `bintray.sh`)
- Make a stable snapshot of docs under the new version
- Bump version to next release in `docs/conf.py`
- Announce to *reddit* etc.

Building the Debian Package

A Debian package for easy installation is built using fpm, so you have to install that first on the build machine, if you don't have it yet:

```
apt-get install ruby ruby-dev
gem install fpm
fpm -h | grep fpm.version
```

Then you need to prepare the install target, as follows (we assume building under the `rtorrent` user here):

```
mkdir -p /opt/rtorrent
chmod 0755 /opt/rtorrent
chown -R rtorrent.rtorrent /opt/rtorrent
```

Then, the contents of the package are built by calling `./build.sh install`, which will populate the `/opt/rtorrent` directory. When that is done, you can test the resulting executable located at `/opt/rtorrent/bin/rtorrent`.

Finally, `./build.sh pkg2deb` creates the Debian package in `/tmp`. The script expects the packager's name and email in the usual environment variables, namely `DEBFULLNAME` and `DEBEMAIL`. For a few platforms (recent Debian, Ubuntu, and Raspbian), you can find pre-built ones at Bintray.

See also:

Using Docker for Building Packages

Building git HEAD of rTorrent

You can also build the latest source of the main rTorrent project (including its `libtorrent`), with all the settings and rpath linking of the `rtorrent-ps` builds, but just like *vanilla* when it comes to applying patches (only *essential* ones are applied, like the *OpenSSL* one).

This is intended to be used for checking compatibility of patches with the head of the core project, and preparing PRs for it. You will *not get a stable system* and these builds are in no way recommended for production use.

Start by checking out the two projects as siblings of the `rtorrent-ps` workdir, leading to a folder structure like this:

```
.
- libtorrent
- rakshasa-rtorrent
- rtorrent-ps
```

As you can see, the sibling folders can have an optional `rakshasa-` prefix.

Then use these commands within `rtorrent-ps` to build all dependencies and the git HEAD code from the sibling folders:

```
INSTALL_DIR=$HOME/.local/rtorrent-git ./build.sh clean_all deps_git build_git
```

Just like with the vanilla and extended version, you'll get a 'branded' binary called `rtorrent-git`, and a symlink at `~/bin/rtorrent` will point to it.

The `INSTALL_DIR` is set explicitly, so that a release version and git HEAD can be installed and used concurrently, without any conflicts.

Using Docker for Building Packages

The `docker_deb` build action uses `Dockerfile.Debian` to compile and package *rTorrent-PS* on a given *Debian* or *Ubuntu* release.

`docker_deb` takes an optional `‹distro›:‹codename›` argument, and defaults to `debian:stretch`. You can also use `all`, `stable`, or `oldstable` to name classes of distributions, defined in the related docker_distros_* lists at the start of build.sh.

Any additional arguments are passed on to the underlying `docker build` command. Since `docker_deb` takes arguments, you cannot call any further actions after it, in the same `build.sh` call.

Note: You need Docker version `17.06` or higher to use this.

Trouble-Shooting Guide

Reporting Problems

If you have any trouble during *rtorrent-ps* installation and configuration, join the pyroscope-users mailing list or the inofficial `##rtorrent` channel on `irc.freenode.net`. IRC will generally provide a faster resolution.

If you are sure there is a bug, then open an issue on *GitHub*. Report any problems that are clearly rooted in the *rTorrent* core to the upstream issue tracker.

Make sure that nobody else reported the same problem before you, there is a search box you can use (after the **Filters** button). Please note that the *GitHub* issue tracker is not a support platform, use the mailing list or IRC for that.

Note: Please **describe your problem clearly**, and provide any pertinent information. What are the **version numbers** of software and OS? What did you do? What was the **unexpected result**? If things worked and 'suddenly' broke, **what did you change**?

On IRC, don't ask if somebody is there, just describe your problem. Eventually, someone will notice you – IRC is a global medium, and people *do* live in different time zones than you.

Put up any logs on 0bin or any other pastebin service, and **make sure you removed any personal information** you don't want to be publically known. Copy the pastebin link into IRC or into your post.

Common Problems & Solutions

Please open an issue on *GitHub* if you think that you have a problem that happens a lot, or you know several other people have the same problem, and **it's not already mentioned below**.

Error in option file: …/05-rt-ps-columns.rc:…: Invalid key

You combined a brand-new *pimp-my-box* configuration with an older version of *rTorrent-PS*.

Solution 1 (preferred)

Update to a recent build *rTorrent-PS*.

Also make sure your `~/rtorrent/rtorrent.rc` is the newest one with the line…

```
method.insert = pyro.extended, const|value, (system.has, rtorrent-ps)
```

This auto-detects the presence of *rTorrent-PS*, but only works with builds from June 2018 onwards.

Solution 2

Replace this line in `~/rtorrent/rtorrent.rc`…

```
method.insert = pyro.extended, const|value, (system.has, rtorrent-ps)
```

with that one…

```
method.insert = pyro.extended, const|value, 1
```

Startup Failure: 'your terminal only supports 8 colors'

See the *Setup & Configuration* chapter for detailed help on proper terminal setup.

If all else fails or you're in a rush, you can switch to the 8-color theme by calling the `echo` command as shown and then start *rTorrent-PS* again:

```
echo default-8 >~/.pyroscope/color-schemes/.current
~/rtorrent/start
```

If you don't use the standard configuration (where theme support comes from), then add the `ui.color.*` commands from this configuration snippet to `rtorrent.rc`, which does the same thing.

Startup Failure: 'libxmlrpc_*.so cannot open shared object file'

On newer systems, `RPATH` is replaced by `RUNPATH` with consequences regarding the search path for *transitive* library dependencies (like that of `libxmlrpc` to the other `libxmlrpc_*` libraries). In the end, those transitive dependencies cannot be resolved without some extra config.

The solution is to use the provided start script, which explicitly sets `LD_LIBRARY_PATH` from any `RPATH` or `RUNPATH` found in the executable. Or if you use a systemd unit, use an `Environment` directive to set the library path, e.g. `Environment=LD_LIBRARY_PATH=/opt/rtorrent/lib`.

Change History

List of Releases

2018-07-01 v1.1 "Design Your Canvas"

Read the related pyrocore update to 0.6.x instructions, which will update configuration snippets in the `rtorrent.d` directories. If you used those snippets before, their update is **required** to run the new version – else you'll get visual defects at the very minimum, but worse stuff might happen.

If you do not have a `~/rtorrent/rtorrent.d` directory created by the helper scripts, then you just have to know yourself what to do (i.e. browse through `git` diffs) – those scripts can only handle standard situations and setups.

- Docs: Much improved (moved to *Read the Docs* and using *Sphinx*)
- UI: Added responsive *canvas v2* with full customization (issue #60)
  - New `ui.column.render` multi-command
  - Added `ui.column.hide` and related commands
  - Added 3 new `convert.*` commands
  - Added `ui.color.custom1...9` and `ui.color.*.index` commands
  - New default columns: ℞
  - New alert indicators for "tracker down" and "DNS problems"
  - 'Tagged' indicator () now in its own column
  - Built-in views are collapsed by default now
- UI: `Info` details panel is now active by default (not `Peer list`)
- UI: Made `*` key a built-in keyboard shortcut [@chros73]
- UI: Key to toggle sacrificial columns manually (bound to `/`)
- Command: Added `string.*` command group (issue #59)
- Command: Added `math.*` command group [@chros73] (issue #65)
- Command: Added `value` conversion command

- Command: Added `array.at` command (issue #60)
- Command: Added `d.custom.if_z` and more `d.custom.*` commands (issue #101)
- Command: Added `d.is_meta` command [@chros73]
- Command: Added `d.tracker_alias` command (issue #97)
- Command: Added `d.multicall.filtered`
- Command: Added `system.has` and 3 other related commands (issue #82)
- Command: Added `system.client_version.as_value` command
- Command: Added `do` command (issue #98)
- Command: Added `import.return` private command
- Command: Added `throttle.names` (issue #65)
- Command: Added `ui.bind_key.verbose` flag (issue #55)
- Command: Added `event.view.hide` and `event.view.show` events
- Fix: Prevent filtering of `started` and `stopped` views [@chros73] (issue #36)
- Fix: `system.file.allocate.set=[0|1]` semantics [@chros73] (issue #41)
- Fix: `log.messages` – restored lost patch for message writing (issue #78)
- Fix: Properly close XMLRPC log, i.e. only *once* (issue #94)
- Fix: `start` handles platforms that use `RUNPATH` instead of `RPATH`
- Patch: `catch` command is silent when first command is `false=`
- Build: `bootstrap` script – switch to Python3 (issue #84)
- Build: Improved git build process
- Build: `all` now really does it all, and added a few more actions
- Build: Allow installation of several concurrent patch versions
- Build: Apply patches for OpenSSL 1.1 (on platforms using it)
- Build: Added `-std=c++11|0x` option (needed for `algorithm::median` patch)
- Build: Fixed `CXXFLAGS` for GCC 6+
- Build: Dropped *rTorrent* 0.8.x/0.9.2 support (issue #26)
- Build: Fedora 26 support (if you use this: it's unmaintained, unless people open PRs)
- Tasks: Added integration tests for commands – `invoke test [-n ‹name›]` (issue #84)
- Tasks: New `cmd_docs` task to generate extension command index
- Docker: Run `pkg2deb` for several Debian / Ubuntu LTS versions
- Docker: Run an *EMPHEMERAL* rTorrent-PS instance in a *Debian Stretch* container

NOTE: Support for *rTorrent* versions before 0.9.6 and non-C++11 compilers will end soon!

2017-03-12 v1.0 "First version-tagged release"

- Improved build script
- `system.random = [[<lower>,] <upper>]` command
- UI: tracker alias mappings
  - `trackers.alias.set_key=«domain»,«alias»` and `trackers.alias.items=` commands
- New Arch AUR PKGBUILDs, and `pkg2pacman` action by @xsmile (#48)
- More flexible installation options (#43)
- Easier manual setup with `make-rtorrent-config.sh` script

CHAPTER 2

Indices & Tables

- genindex
- modindex
- search