# PRO KDE

EVERYTHING YOU NEED TO KNOW ABOUT KDE

PAUL MCQUADE

# Pro KDE

Paul McQuade

Version v0.2.1, 2020-08-08

# Table of Contents

# Licence



# Preface by Paul McQuade

I began writing this book as a way to escape from the nightmare of Covid-19. As i write i wondered what can i do to make developing software easier on people. I realise that sharing my information about coding was to way to go.

I never wrote a book before and i had very little practical knowledge about KDE Plasma and KDE Frameworks. Writing a book would force me to think.

## Dedications

I would like to thank my family - Peter, Breda, Caitriona and Caroline for their support down through the years especially when i was sick in hospital.

# Developers

## Source Code for book

This book is hosted on gitlab.com under the following link:

https://gitlab.com/paulmcquad/prokde

There's a mirror copy on github.com under the following link:

https://github.com/paulmcquad/prokde

Please fork this book today.

## Issues

If you have an issue about the book you want to raise it on gitlab.com Issues can be errata and content changes.

## Merge Requests

Pro KDE is always looking for developers. Feel free to submit patches through Gitlab Merge Requests.

In Gitlab go to: Left side Panel > Merge Requests > New merge request

## Thank You

Since this is an Open Source book, I want to say a big thank you to the developers who have send in several errata and content changes. Here are all the people who have contributed to the English version of Pro KDE as an open source project.

Thank you everyone for helping make this a better book for everyone.

# Getting Started

## About KDE

KDE is an international team co-operating on development and distribution of Free, Open Source Software for desktop and portable computing. Their community has developed a wide variety of applications for communication, work, education and entertainment. They have a strong focus on finding innovative solutions to old and new problems, creating a vibrant, open atmosphere for experimentation.

### KDE Projects

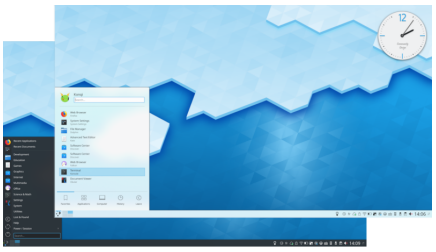
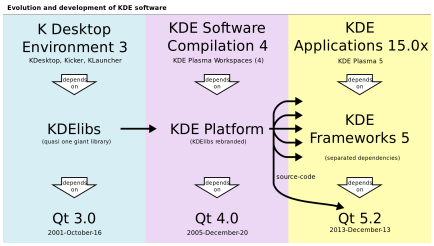
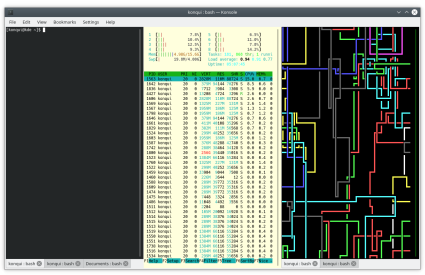
KDE Projects consists of three parts:

- KDE Plasma - A User Interface
- KDE Frameworks - A Collection of libraries and software frameworks
- KDE Applications - A list of Programs like Kate and Konsole

### Development of KDE

A few images of process in action:

*Table 1. Development of KDE*

| KDE Plasma | KDE Frameworks | KDE Applications |
| --- | --- | --- |
|  |   |  |

### THE KDE CODE OF CONDUCT

When communicating in official KDE channels please observe the KDE Code of Conduct. Our Code of Conduct presents a summary of the shared values and common sense thinking in our community. The basic social ingredients that hold our project together include:

- Be considerate
- Be respectful
- Be collaborative
- Be pragmatic

Support others in the community Get support from others in the community The Code of Conduct can be found here : http://www.kde.org/code-of-conduct/

# Setup an KDE Account

## About KDE Identity Account

In order to submit merge requests with GitLab, you'll need a KDE Identity Account.

These can be registered using the self-service Identity site. As part of this process, you will need to provide a name and email address, which has to be your own. Please note that these details will be made publicly visible on Gitlab once you have logged in there. You may therefore receive some spam as an unfortunate consequence of this.

When selecting your username, please ensure you select something which has a relation to your real name.

**A Developer Account is not needed to fork repositories and submit merge requests on Gitlab.**

Also note that this email address should be the same one that you use on bugs.kde.org. If you don't have an account in bugs.kde.org, please create one so that it can be given usual developer rights. Closing bug reports with keywords in commit comments only works if the email address associated with your KDE Developer account and bugs.kde.org accounts match.

## How to setup an KDE Identity Account

Register with KDE Identity Account at the following:

KDE Identity

Click Register > Accept KDE Code of Conduct > Fill in Name and Email

Identity with send an email called "Account activation on KDE Identity"

```
Something like this:

Hello {First Name, Last Name},

In order to activate your new account please follow the link below.
You will not be able to begin using your account on KDE Identity until you have
activated your account.

If you did not request this, please inform the site administrator by replying to this
email.

https://identity.kde.org/index.php=LINK

Thanks,
KDE Identity site administrators.
```

I removed the LINK details but you get the idea.

# Tools

## Core Tools

### Git/Gitlab

#### About Git

Git is a free and open source version control system designed to handle everything from small to very large projects with speed and efficiency. It provides lots of tools for figuring out where you have gone as you edit files, as well as merging your changes with those made by other developers.You can find more about git (and download it if necessary) at:

http://git-scm.com/

#### Pro Git

If this is the first time hearing of Git. There's this great book called ProGit. It can be found at:

https://git-scm.com/book

#### About Gitlab

KDE uses Gitlab and hosts their own code on their private Edition of Gitlab:

https://invent.kde.org/explore

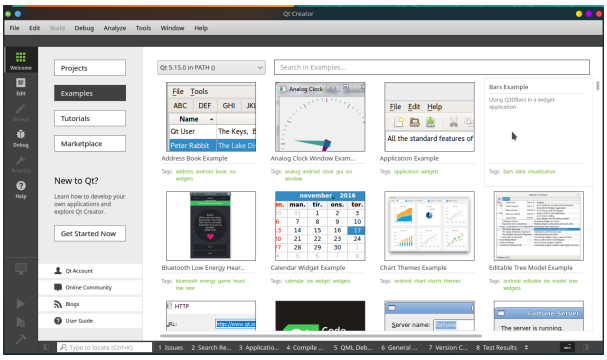
### An IDE

An integrated development environment (IDE) allows you to do project management, testing, and other activities in a convenient way alongside your coding. We recommend that you install one of the following IDEs and do your KDE development work within it.

We recommend QtCreator for its ease of use and features, especially its built-in text editor. But it's nice to know, if you're familiar with KDevelop already, that you can use that for KDE development too.

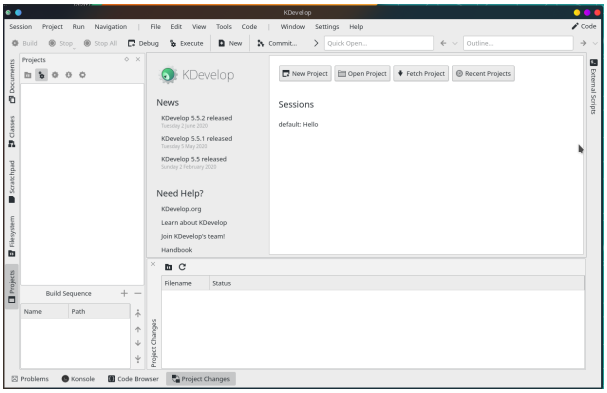
### QT Creator

Qt Creator is a cross-platform C++, JavaScript and QML integrated development environment which simplifies GUI application development. It is part of the SDK for the Qt GUI application development framework and uses the Qt API, which encapsulates host OS GUI function calls.



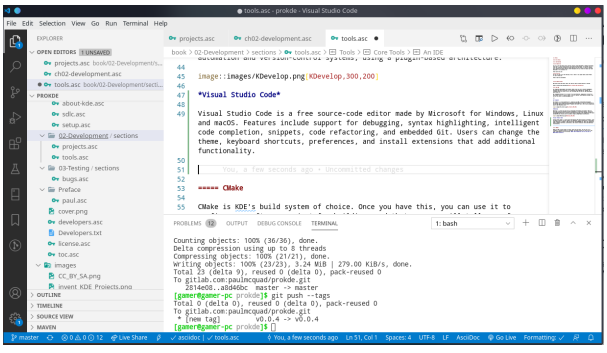

### KDevelop

KDevelop is a free and open-source integrated development environment (IDE) for Unix-like computer operating systems and Windows. It provides editing, navigation and debugging features for several programming languages, and integration with build automation and version-control systems, using a plugin-based architecture.




### Visual Studio Code

Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.



**CMake**

CMake is KDE's build system of choice. Once you have this, you can use it to configure a software project for building, and that process will tell you of any other requirements you are missing. **NOTE: KDE Frameworks can also be used in QMake-based projects.**

# Software Development Life Cycle

The 6 steps to Software Development Life Cycle are:

1. Planning - Talk to other developers.
2. Analysis - Break the problem into smaller parts.
3. Design - Put a plan together and create mockups if needed.
4. Implementation - Write code.
5. Testing & Integration - Report and fix bug issues.
6. Maintenance - Apply patches and release software.


# Who is this book for?

This book is aimed at software testers and developers who want to know more about KDE. It may also reach a wider audience but I wrote this book is to learn about KDE from a developers side.

# Implementation

## Projects

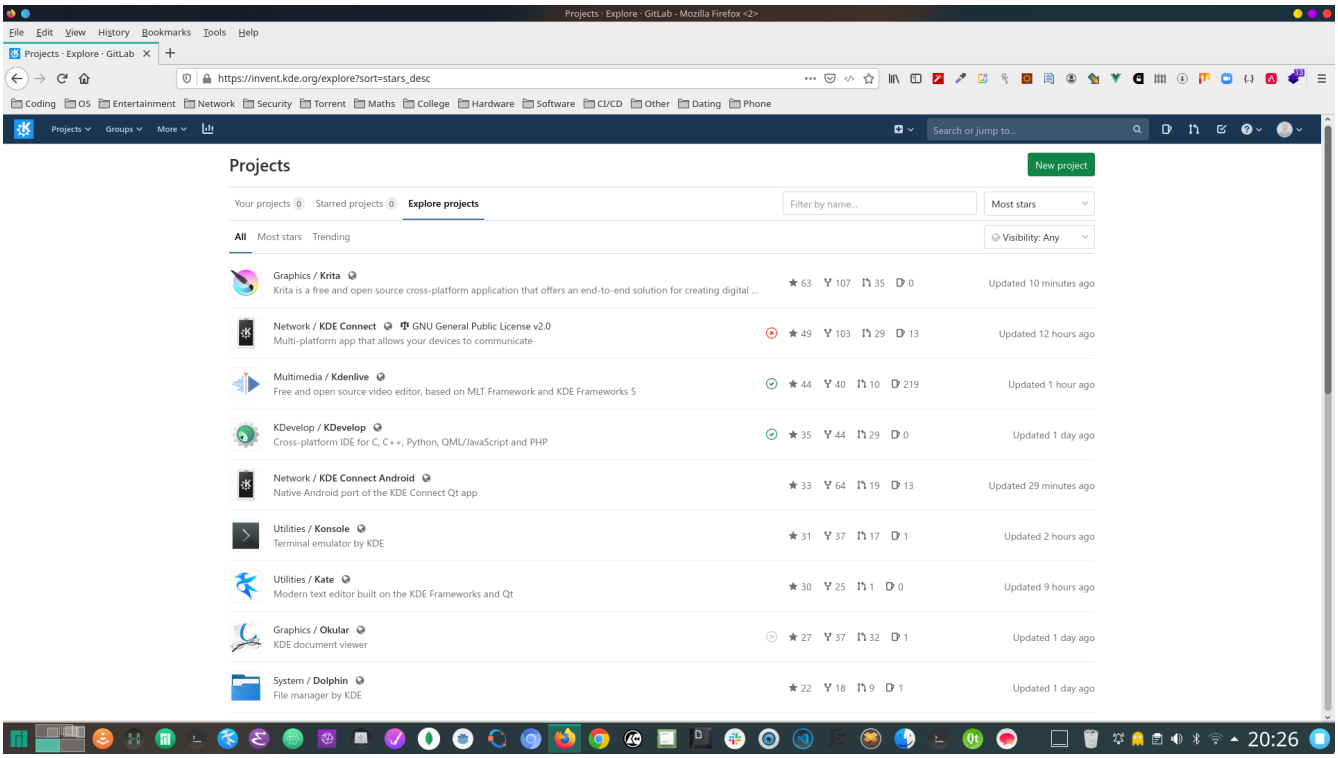
Once you setup an KDE Identity Account (see Getting Started > Setup an KDE Account). You can use your account to login to invent.kde.org/explore.

### Choosing a Project

When you come to KDE as a developer, you may already have a favorite project and know how you want to contribute. But it's worth looking over the various projects listed in this chapter, to find out all the ways you may be able to help. And even if you're really only interested in one project, it's useful to know what others are active because your work may interact with them.

### KDE Projects

These are general components underlying the applications and other visible parts of KDE. The team is working hard to make the libraries modular, clarify the dependencies, simplify, and increase the quality and stability.



# The QT Framework

To start developing on the KDE Development Platform you will need to get familiar with the Qt framework, which is one of building blocks of KDE development.

Qt (pronounced officially as cute) is a cross-platform application framework based on C++, that is widely used for developing application software with a graphical user interface (GUI). Thus, it is largely a widget toolkit, but is also used for developing non-GUI programs such as command-line tools and consoles for servers.

Besides the KDE Development Platform, Qt is most notably used in Autodesk Maya, Adobe Photoshop Elements, OPIE, Skype, VLC media player, VirtualBox, and Mathematica, and by the European Space Agency, DreamWorks, Google, HP, Lucasfilm, Panasonic, Philips, Samsung, Siemens, Volvo, and Walt Disney Animation Studios.

# Advantages of QTs

**Writing code once to target multiple platforms**

Qt allows you to write advanced applications and UIs that you can deploy across different desktops and embedded operating systems without rewriting the source code, saving time and development cost.

**Creating amazing user experiences**

Whether you prefer C++ or JavaScript, Qt provides the building blocks for modern, interactive interfaces: a broad set of customizable widgets, graphics canvas, style engines, and more. You can incorporate 3D graphics, multimedia audio or video, visual effects, and animations to set your application apart from the competition.

**Doing more (and faster!) with less**

Qt is fast to learn and to use, particularly when used together with the new Qt Creator cross-platform IDE. And Qt's modular class library provides much of the necessary infrastructure for interactive applications.

**Blending web and native code in a single application**

Qt's integration with the WebKit web rendering engine means that you can quickly incorporate content and services from the Web into your native application, and can use the web environment to deliver your services and functionality. To learn how to use Qt, we recommend the tutorials here:

https://doc.qt.io/

QT Developers site:

https://www.qt.io/developers

# KDE API

KDE has a set of API's that can be found at:

https://api.kde.org/

### API

An Application Programming Interface (API) is a set of functions exposed by a program or library that can be invoked by other programmers. An API greatly extends a program by allowing third-party developers to add new functionality. It is an ideal way to allow new features to be added without the need to modify the existing, core code.

### Build Prerequisites

This section details the software requirements you must install on your system before you can start building the KDE platform. For most of these requirements, it is best to use the packages provided by your operating system distribution, but in some cases you will need to build these requirements yourself.

# Fork a Project

Once you have made some local changes that you would like to submit to KDE, you need to create a personal fork of the project and push your changes to the forked copy.

Navigate to https://invent.kde.org/kde and locate the project.

# Plasma

Plasma is a group on Invent with many different projects that deal with the graphical workspaces environment or User interface(UI). It describes the group as:

Desktop environment for a wide range of devices and user needs:

### Plasma Projects

The top three Plasma projects are:

KWin - Easy to use, but flexible, X Window Manager and Wayland Compositor

plasma-desktop - Plasma for the Desktop

plasma-workspace - Various components needed to run a Plasma-based environment

For more Plasma Projects see:

invent.kde.org/plasma

# Testing & Integration

## Issue Reporting

If you've found a problem with a piece of KDE software or have an idea for an improvement, report it to the developers! QA is critically important to ensure quality, and you can be involved in an effort to make sure that our users are happy with the final products. Find out more about issue reporting.

### Issue reporting involves responsibility

Before you report a Bugzilla ticket, please make sure that you are:

- using relatively recent versions of KDE software (≤ 1 year old) or an LTS supported version
- willing to do whatever is necessary to get the bug to happen again (if reporting a bug)
- willing to correspond with KDE developers about the issue, potentially over an extended period of time
- prepared to gather and report a lot of additional information
- ready to potentially use a terminal program to execute command line debugging tools

If those points all apply to you, then we can move on! Let's see now make sure that the issue you're experiencing is something that can be reported.

### Step 0: Is it a security issue?

Do not file bug reports for security issues, as this makes the vulnerability public and could expose users to exploits. Instead, send an email to security@kde.org. For more information, see https://kde.org/info/security/.

### Step 1: Make sure it's a valid bug or feature request

Real issues involve any of the following:

- Crashes
- Broken functionality
- Anything that causes data loss
- Design issues, such as faulty paddings/margins, misplaced content, lack of consistency with the rest of the desktop, incorrect icon/image rendering, or other user interface anomalies
- Translation problems, where certain words or expressions are inappropriate for a given context, or a part of a program that was not properly translated or is lacking translation
- Typographic errors (typos), unclear instructions or general communication, inappropriate grammar or style, or not conveying a program's functionality correctly
- Usability issues, such as inappropriate default window sizes, excessive or non-optimal amount of clicks to achieve a task, and others

- Accessibility issues, such as text or software itself that cannot be used with a text-to-speech interface or screen reader
- Request for missing features or ideas about how something that's already working could be further improved

If your issue is on this list, **proceed onto step 2**.

By contrast, here are some examples of issues that are almost never real issues:

- Vague and subjective matters of aesthetics or personal preference (e.g. "The program is ugly")
- Design choices too broad to be changed by a Bugzilla ticket (e.g. "The program should work more like this other program that I'm used to")
- Anything caused by user or vendor configuration choices (e.g. "The program disappeared after I did a system update")
- Support questions ("How do I change such-and-such behavior of the program?")
- Development assistance requests ("Can you help me compile the program?")

Do not file **Bugzilla tickets for these kinds of issues**. Instead, you might try bringing up the matter on the KDE forum or another online discussion group of your choice.

Here are examples of issues that *may* be valid, but require some more investigation:

- "The program is really slow" (It could be a bug in the software, or it could be an issue with your computer or its configuration)
- "Some of the program's icons are inconsistent" (If you're using a 3rd-party icon theme, that could be causing the problem)
- "The program did something unexpected in response to an action that I took" (this could be a bug or a usability issue, or it could just be that the program's operation is unfamiliar to you)
- "The label for one of the program's user interface controls doesn't clearly indicate what it does" (It could be a badly-worded or badly-translated piece of text, or it could be a 3rd-party widget theme mangling the user interface)

If your issue is on this list, **proceed to Step 2**, but be prepared for the issue to be caused by configuration or hardware issues, or simply be an example of unfamiliar behavior.

### Step 2: Make sure it hasn't already been fixed

# Bug triaging

## Summary

The KDE Bugsquad keeps track of incoming bugs in KDE software, and goes through old bugs. They verify that a bug exists, whether it is reproducible, and that the reporter has given enough information. Their goal is to save developers from doing this, **which helps them fix bugs more quickly and do more work on KDE software.**

Getting involved in the Bugsquad is a good place to start. An existing member will help you out and mentor you. One of the great things about the Bugsquad and bug triaging is that **you do not need any programming knowledge!**

That said, experience has shown that members of this team learn a lot about KDE and Qt programming during the course of dealing with bug reports, and many move on to developing the software itself. If you are just starting to learn programming, bug triaging is a great way to gain familiarity with the components and give practical support to the KDE community.

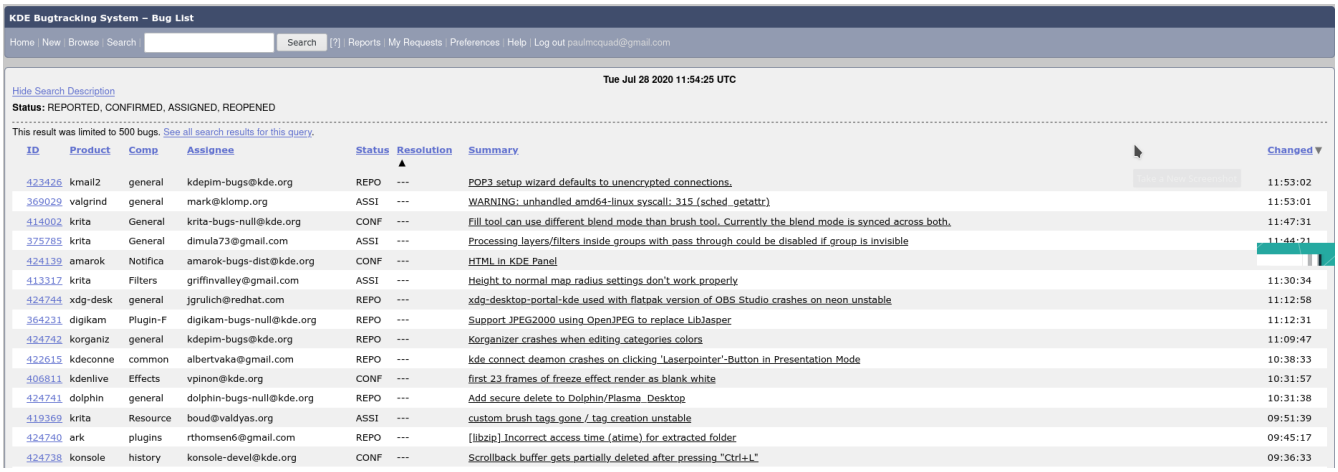
## Live chat

For live chat, we have the #kde-bugs IRC channel on Freenode. Please feel free to stop by! You can also connect via Matrix if you prefer.

# Reporting Bugs

Report bugs to the the KDE Bugtracking System at:

https://bugs.kde.org/




# Logs

## Build Logs

When building KDE for the first time it can be a little hard to see the problem.

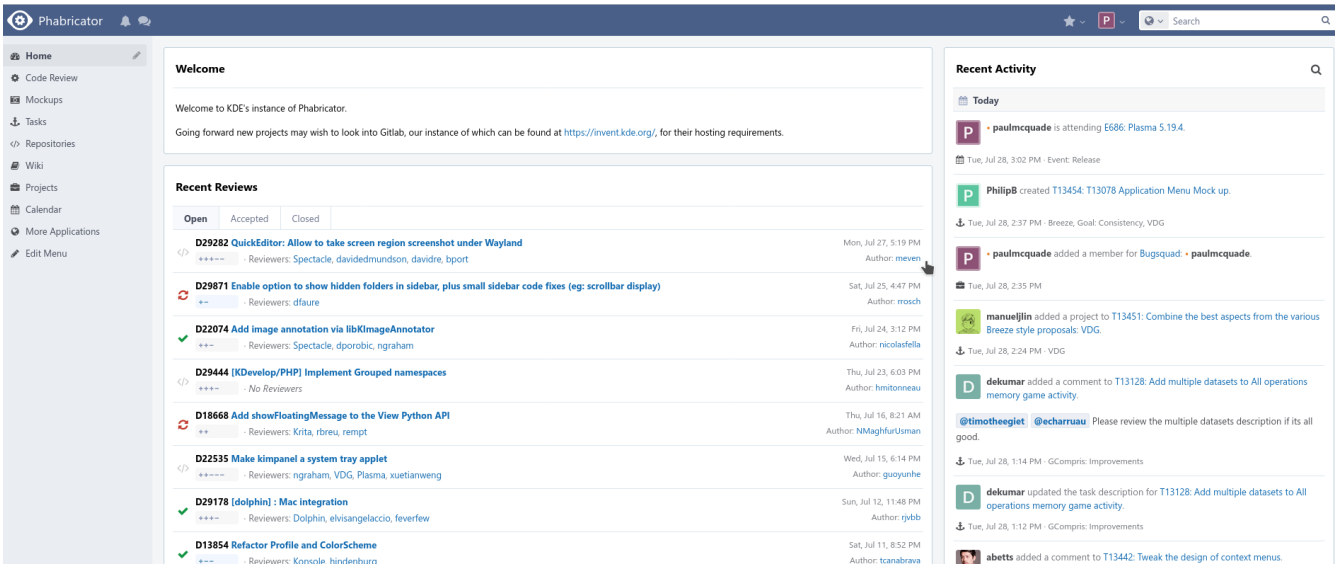
If you use the tool kdesrc-build it will build to a log file directory.

Following the guide here - https://community.kde.org/Get_Involved/development

The Log directory will be at ~/kde/src/log/ and it will be timestamped.

# KDE Bugsquad

The KDE Bugsquad keeps track of incoming bugs in KDE software, and goes through old bugs. They verify that a bug exists, whether it is reproducible, and that the reporter has given enough information. Their goal is to save developers from doing this, which helps them fix bugs more quickly and do more work on KDE software.




## Join the Bugsquad

To join the Bugsquad, become a member of our project on https://phabricator.kde.org/. We organize Bug Days, where we focus on a specific product, with the goal of reviewing all open bugs by the end of the day. These meetings occur primarily on #kde-bugs.

## Bugsquad Calendar

The following is a link to the Bugsquad Calendar:

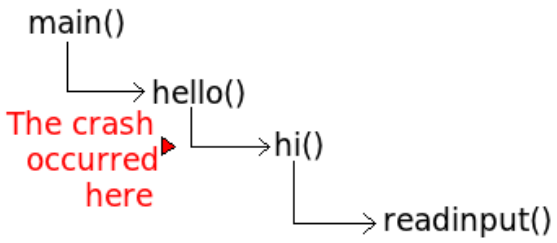
Bugsquad Calendar

# Reading Backtraces

A backtrace (also called a stack trace or stack traceback) is a report of how the program has called different functions as it goes along. It is commonly used during interactive and post-mortem debugging. It can also be displayed to the user of a program as part of an error message, which a user can report to a programmer.

Each function puts a stack frame on the stack containing its arguments and other information it needs to run. The active stack frames reflect a certain point in time during the execution of a program. A stack trace allows you to track the sequence of nested functions called up to the point where the stack trace is generated. In a post-mortem scenario , the stack trace goes up to, and includes , the function where the failure occurred. Be aware, however, that the function where the failure occurred might not be responsible for the failure; an error could well have been embedded in a higher function (for instance, by passing an incorrect value to the function where the program failed).

The following figure illustrates a stack frame, where main() called hello(), which called hi(), which called readinput(). A stack trace is likely to work down from the last call to the first, so that readinput() might appear first.




Backtraces are essential. They may look meaningless to you, but they might actually contain a wealth of useful information. A backtrace describes which functions were called prior to the crash, so that developers may track down in which function the mess started. Exact memory addresses can also help locate problematic data, such as in a core dump (a file left behind when a program fails , containing the contents of live memory at the time of the failure). But producing good backtraces has a downside: libraries and executables occupy much more disk space than their optimized counter parts that can’t provide the information to produce a backtrace.

A Backtrace looks like this:

```
Using host libthread_db library "/lib/libthread_db.so.1".
[Thread debugging using libthread_db enabled]
[New Thread -1232783168 (LWP 7604)]
[KCrash handler]
#6 0x0806be76 in TreeM apItem::parent (this=0x0)
at /home/bram/KDE/kde3/kdeaddons/konq-plugins/fsview/treemap.h:285
#7 0x08065fea in TreeM apItemList::compareItems (this=0xbfec04a8, item1=0x0,
item2=0x0)
at /home/bram/KDE/kde3/kdeaddons/konq-plugins/fsview/treemap.cpp:720
#8 0xb7281619 in Q G List::operator== () from /usr/qt/3/lib/libqt-mt.so.3
#9 0x0806d498 in Q PtrList<TreeM apItem>::operator== (this=0xbfec04a8,
list=@ 0xbfec0468) at /usr/qt/3/include/qptrlist.h:74
#10 0x08062e18 in TreeM apWidget::mousePressEvent (this=0xbfec03ac,
e=0xbfebff1c)
at /home/bram/KDE/kde3/kdeaddons/konq-plugins/fsview/treemap.cpp:1840
#11 0xb7004a63 in Q Widget::event () from /usr/qt/3/lib/libqt-mt.so.3
#12 0xb6f6bca7 in Q Application::internalNotify ()
from /usr/qt/3/lib/libqt-mt.so.3
#13 0xb6f6ca88 in Q Application::notify () from /usr/qt/3/lib/libqt-mt.so.3
32
#14 0xb7725a84 in KApplication::notify (this=0xbfec055c, receiver=0xbfec03ac,
event=0xbfebff1c)
at /home/bram/KDE/kde3/kdelibs/kdecore/kapplication.cpp:550
#15 0xb6f0bfd2 in Q ETWidget::translateM ouseEvent ()
from /usr/qt/3/lib/libqt-mt.so.3
#16 0xb6f0b8b0 in Q Application::x11ProcessEvent ()
from /usr/qt/3/lib/libqt-mt.so.3
#17 0xb6f1b761 in Q EventLoop::processEvents () from /usr/qt/3/lib/libqt-mt.so.3
#18 0xb6f82831 in Q EventLoop::enterLoop () from /usr/qt/3/lib/libqt-mt.so.3
#19 0xb6f826b6 in Q EventLoop::exec () from /usr/qt/3/lib/libqt-mt.so.3
#20 0xb6f6b72f in Q Application::exec () from /usr/qt/3/lib/libqt-mt.so.3
#21 0x0805181e in main (argc=134673960, argv=0xffffffff)
at /home/bram/KDE/kde3/kdeaddons/konq-plugins/fsview/main.cpp:55
```

# Appendix A: Getting Help

## KDE IRC Channels

The main IRC Channels on Freenode are:

*Table 2. Developer/Support Channels*

| **Channels:** |
| --- |
| **Developer** |
| #kde - Main channel for users of KDE software |
| #kde-bugs - For KDE BugSquad |
| #kde-devel - For general KDE development |
| #kde-docs - Official KDE Documentation Project |
| #kde-quality - For KDE quality assurance |
| **Project** |
| #plasma - Plasma team |
| **Support** |
| #kde-chat - For Off-Topic discussions |
| #kde-i18n - Localization team |
| #kde-promo - Promotion and communication team |
| #kde-soc - The channel for Google Summer of Code and Code In students |
| #kde-vdg - KDE V Design Group channel |
| #kde-women - Women who use KDE software |
| #kde-www - The channel for discussion regarding the KDE community web sites |

*Table 3. Communities by Country*

| |
|---|
| #kde-ar - KDE software in the Argentine |
| #kde-be - KDE software in Belgium |
| #kde-brasil - KDE software in Brazil |
| #kde-cn - KDE software for Chinese speakers |
| #kde-de - KDE software in Germany |
| #kde-el - KDE software in Greece |
| #kde-es - KDE software in Spain |
| #kde-fi - KDE software in Finland |
| #kde-fr - KDE software for French speakers |
| #kde-gl - KDE software for Galician speakers |
| #kde-in - KDE software in India |
| #kde-ir - KDE software in Iran |
| #kde-italia - Italian user support |
| #kde-latam - KDE software in Latin America |
| #kde-nl - KDE software in the Netherlands |
| #kde-pt - KDE software in Portugal |
| #kde.hu - KDE software for Hungarian speakers |
| #kde_ru - KDE software for Russian speakers |
| #kdehispano - KDE software for Spanish speakers |
| **More IRC Channels found:** |
| IRC_Channels |

# KDE Mailing Lists

The KDE mailing lists are one of the main communication channels in the KDE Community. This is a list of general mailing lists to give a quick overview. For application specific lists, please see the application's page. For a complete overview, see the full list of all mailing lists at:

KDE Mailing Lists

## General User Lists

- kde - for KDE users with questions strictly about KDE applications (subscribe kde, archive of kde)
- kde-announce - announcements of new KDE releases, security announcements, and notification of new KDE applications (very low traffic) (subscribe kde-announce, archive of kde-announce)

## General development lists

- kde-devel - for application developers (both applications in central KDE packages and contributed applications (subscribe kde-devel, archive of kde-devel)

## Project development lists

- Plasma - plasma-devel@kde.org

# Appendix B: Fixing Bugs

## KDE Bugtracking System

File a bug:

https://bugs.kde.org/

Search for bugs at:

https://bugs.kde.org/query.cgi

## KDE Bugsquad

Search KDE Bugsquad for Comfirmed bugs:

https://phabricator.kde.org/

Become a member of the KDE Bugsquad at:

https://phabricator.kde.org/project/update/290/join/

# Index