

# CGAL

The Computational Geometry  
Algorithms Library

# CGAL

- The CGAL Open Source Project
  - provides easy access to efficient and reliable geometric algorithms
  - C++ library
  - Has efficient, robust, easy to use geometric data structures and algorithms
  - Easy to integrate in existing software
- Standard libraries increases productivity, as it allows software developers to focus on the application layer

# CGAL

- <https://www.cgal.org>
- Primary Developers:
  - INRIA: the French national research institute (<https://www.inria.fr/en/>)
  - Geometry Factory : company which provides robust *geometric software components* as well as *expertise* in geometric computing ( <https://geometryfactory.com/>)
  - Tel-Aviv University

# Examples of Packages available

<https://doc.cgal.org/latest/Manual/packages.html>

- Arithmetic and Algebra
  - Algebraic Foundations
  - Number Types
  - Modular Arithmetic
- Combinatorial Algorithms
  - Monotone and Sorted Matrix Search
  - Linear and Quadratic Programming Solver
- Geometry Kernels
  - 2D and 3D Linear Geometry Kernel
  - 2D Circular Geometry Kernel

# Examples of Packages available

<https://doc.cgal.org/latest/Manual/packages.html>

- Convex Hull Algorithms
  - 2D Convex Hulls and Extreme Points
  - 3D Convex Hulls
  - dD Convex Hulls and Delaunay Triangulations
- Polygons
  - 2D Polygons
  - 2D Polygon Partitioning
- Cell Complexes and Polyhedra
  - 3D Polyhedral Surface
  - Half edge Data Structures

# User manual and Reference manual & CGAL discuss forum

Cisco Webex Meetings

NITC Eduserver: Log in to the s

CGAL 5.1 - Manual: Package Overview

Example and Demo Programs

CGAL 5.1 - 2D Range and Neigh

https://doc.cgal.org/latest/Manual/packages.html

point data structure in CGAL

CGAL Version: latest

cgal.org

Top

Getting Started

Tutorials

Package Overview

Acknowledging CGAL

Search

CGAL 5.1 - Manual

Getting Started with CGAL

Tutorials

Package Overview

Developer Manual

License

Acknowledging CGAL

Refinement Relationships

Bibliography

### Polygons

#### 2D Polygons



*Geert-Jan Giezeman and Wieger Wesselink*

This package provides a 2D polygon class and operations on sequences of points, like bounding box, extremal points, signed area, simplicity and convexity test, orientation, and point location. The demo includes operations on polygons, such as computing a convex partition, and the straight skeleton.

[User Manual](#) [Reference Manual](#)

**Introduced in:** CGAL 0.9  
**BibTeX:** [cgaw-p2-20b](#)  
**License:** LGPL  
**Windows Demo:** [Operations on Polygons](#)  
**Common Demo DLLs:** [dlls](#)

#### 2D Regularized Boolean Set-Operations



*Efi Fogel, Ophir Setter, Ron Wein, Guy Zucker, Baruch Zukerman, and Dan Halperin*

This package consists of the implementation of Boolean set-operations on point sets bounded by weakly x-monotone curves in 2-dimensional Euclidean space. In particular, it contains the implementation of regularized Boolean set-operations, intersection predicates, and point containment predicates.

[User Manual](#) [Reference Manual](#)

#### 2D Boolean Operations on Nef Polygons



*Michael Seel*

A Nef polygon is any set that can be obtained from a finite set of open halfspaces by set complement and set intersection operations. Due to the fact that all other binary set operations like union,

**Introduced in:** CGAL 2.2  
**Depends on:** [2D Arrangements](#)  
**BibTeX:** [cgaw-fwzh-rbso2-20b](#)  
**License:** GPL

Generated on Sun Sep 6 2020 21:32:01 for CGAL 5.1 - Manual by [doxygen](#) 1.8.13



10:02 AM  
10/1/2020

# Demo and Example Folders

- The best way to illustrate the functionality provided by the library is through programs that users can compile, run, copy and modify.
- Thus every package should contain some of these programs.
- In CGAL we distinguish between two types of programs: those that provided graphical output (demos) and those that do not (examples).
- <https://www.cgal.org/FAQ.html>

Navigation: [Up](#), [Table of Contents](#), [Short Table of Contents](#), [Package Overview](#), [Bibliography](#), [Index](#), [Title Page](#)

# Chapter 19

## Example and Demo Programs

The best way to illustrate the functionality provided by the library is through programs that users can compile, run, copy and modify to their hearts' content. Thus every package should contain some of these programs. In CGAL we distinguish between two types of programs: those that provided graphical output (demos) and those that do not (examples).

In this chapter we provide guidelines for the development of these programs and their inclusion in the documentation. See Sections 4.3 and 4.4 for a description of the directory structure required for example and demo programs, respectively. Note in particular that each directory should contain a `README` file that explains what the programs do and how one interacts with them.

### 19.1 Coding conventions

Remember that these programs are likely to be a user's first introduction to the library, so you should be careful to follow our coding conventions (Chapter 6) and good programming practice in these programs. In particular:

- Include a comment that gives the name of the file relative to the `CGALROOT` directory, such as:

```
//  
// file : examples/Generator/random_polygon_ex.C  
//
```

- Do **not** use the commands

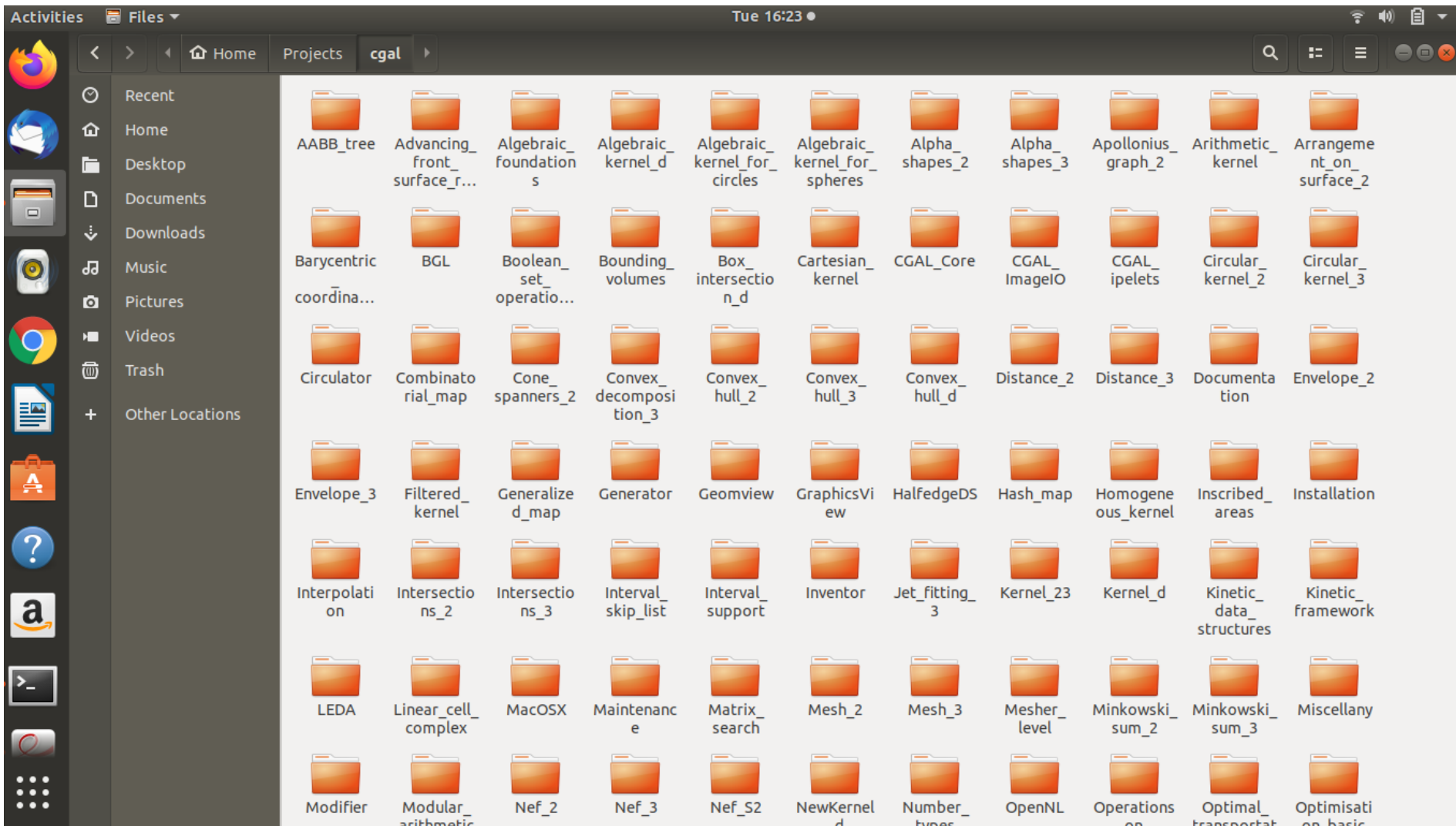
```
using namespace CGAL;  
using namespace std;
```

We discourage the use of these as they introduce more names than are necessary and may lead to more conflicts than are necessary.

- As of release 2.3, you can include only the kernel include file (e.g., `Cartesian.h` or `Homogeneous.h`) to get all kernel classes as well as the `basic.h` file. All example and demo programs should do this. For example, you should have simply:



# CGAL examples Folder



# Tools for display

- OpenGL
- QT
- QGLViewer

# CGAL Installation

- [https://www.cgal.org/FAQ.html#how\\_to\\_install](https://www.cgal.org/FAQ.html#how_to_install)
- Two script files
- `cgal-depends.sh`
- `cgal.sh`

# cgal-depends.sh

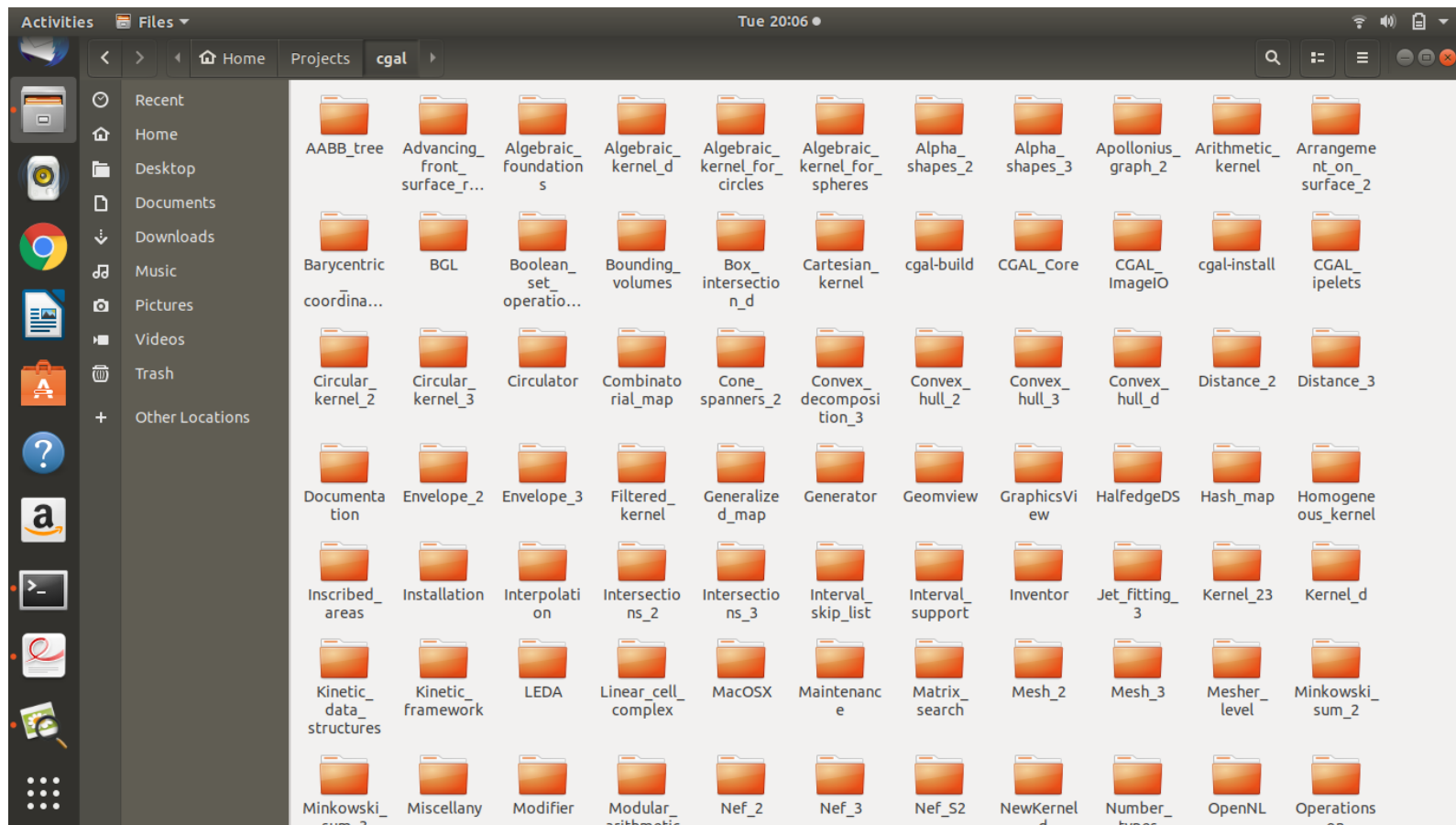
- `#!/bin/bash`
- `sudo apt install -y libcgal-dev libcgal-qt5-dev`
- `sudo apt install -y libgmp-dev libmpfr-dev  
libeigen3-dev`
- `sudo apt install -y qtbase5-dev unzip`
- `sudo apt install -y mesa-utils`
- `sudo apt install -y cmake cmake-gui gcc gdb  
build-essential`

# cgal.sh

- `#!/bin/bash`
- `#Download CGAL from this link`
- `wget https://github.com/CGAL/cgal/releases/download/v5.6/CGAL-5.6.zip`
- `unzip CGAL-5.6.zip`
- `#extract the archive`
- `cd CGAL-5.6`
- `mkdir build`
- `cd build`
- `cmake .. -DCMAKE_BUILD_TYPE="Release" -DWITH_examples=ON -DWITH_demos=ON -DCMAKE_INSTALL_PREFIX="$HOME/.local"`
- `make install`
- `#make examples -j$(nproc)`
- `#make demos -j$(nproc)`
- `# wayland fix, ask students to use Xorg`
- `echo "QT_QPA_PLATFORM=xcb" >> ~/.bashrc`

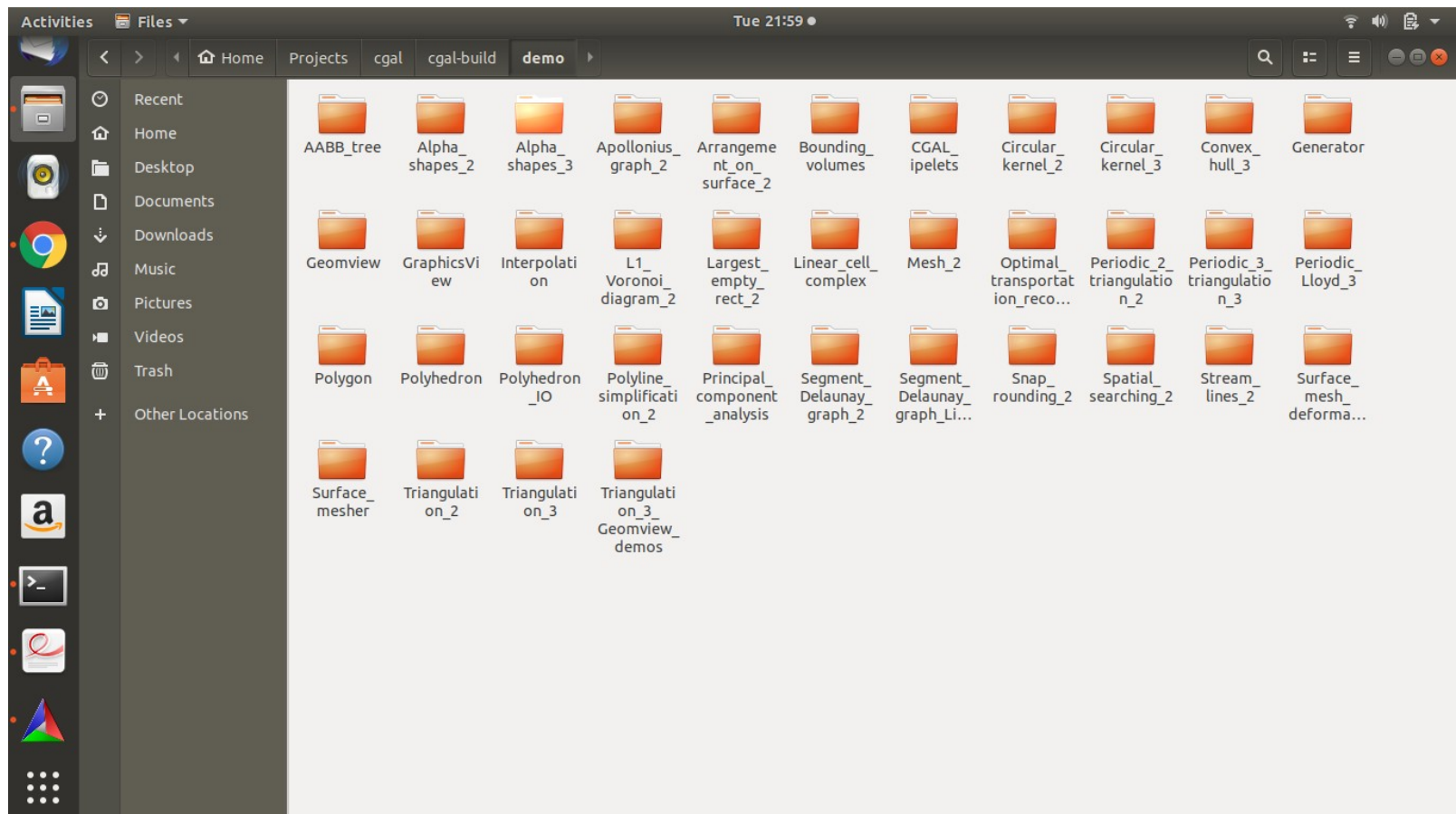
# Run a CGAL demo - GraphicsView

## 1. Open your CGAL folder



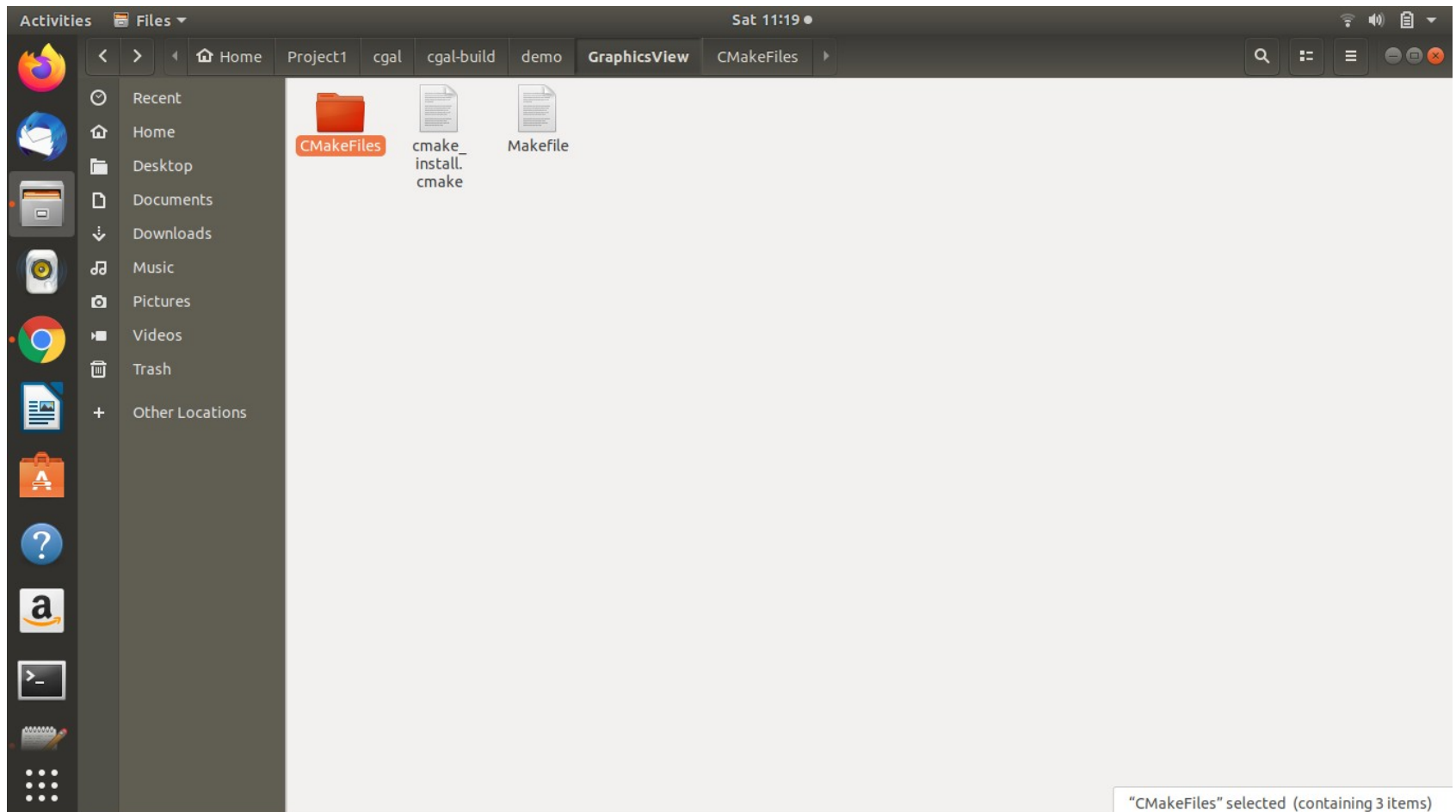
# Run a CGAL demo - GraphicsView

## 2. Go to cgal-build/demo



# Run a CGAL demo - GraphicsView

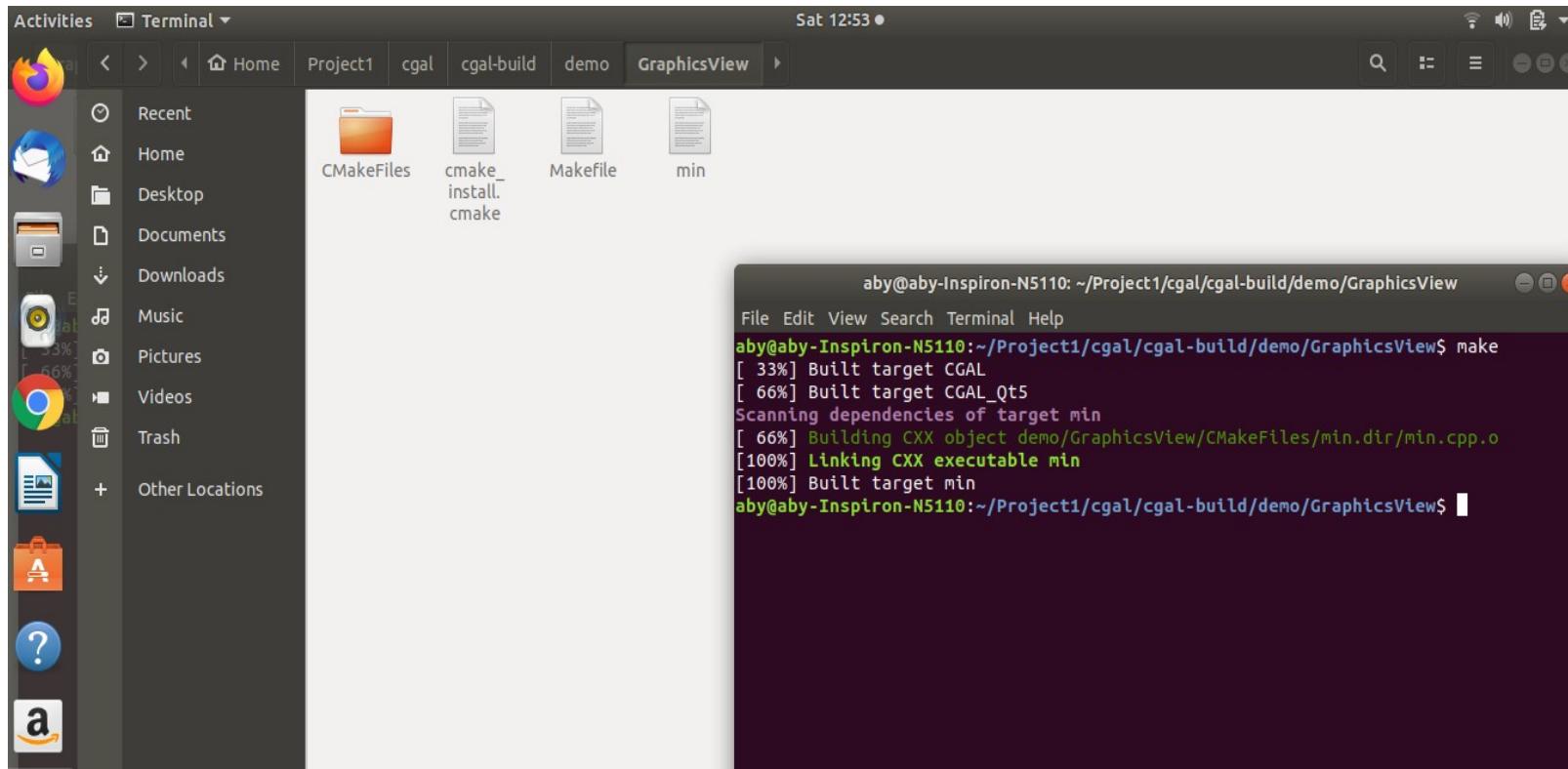
## 3. Open folder GraphicsView





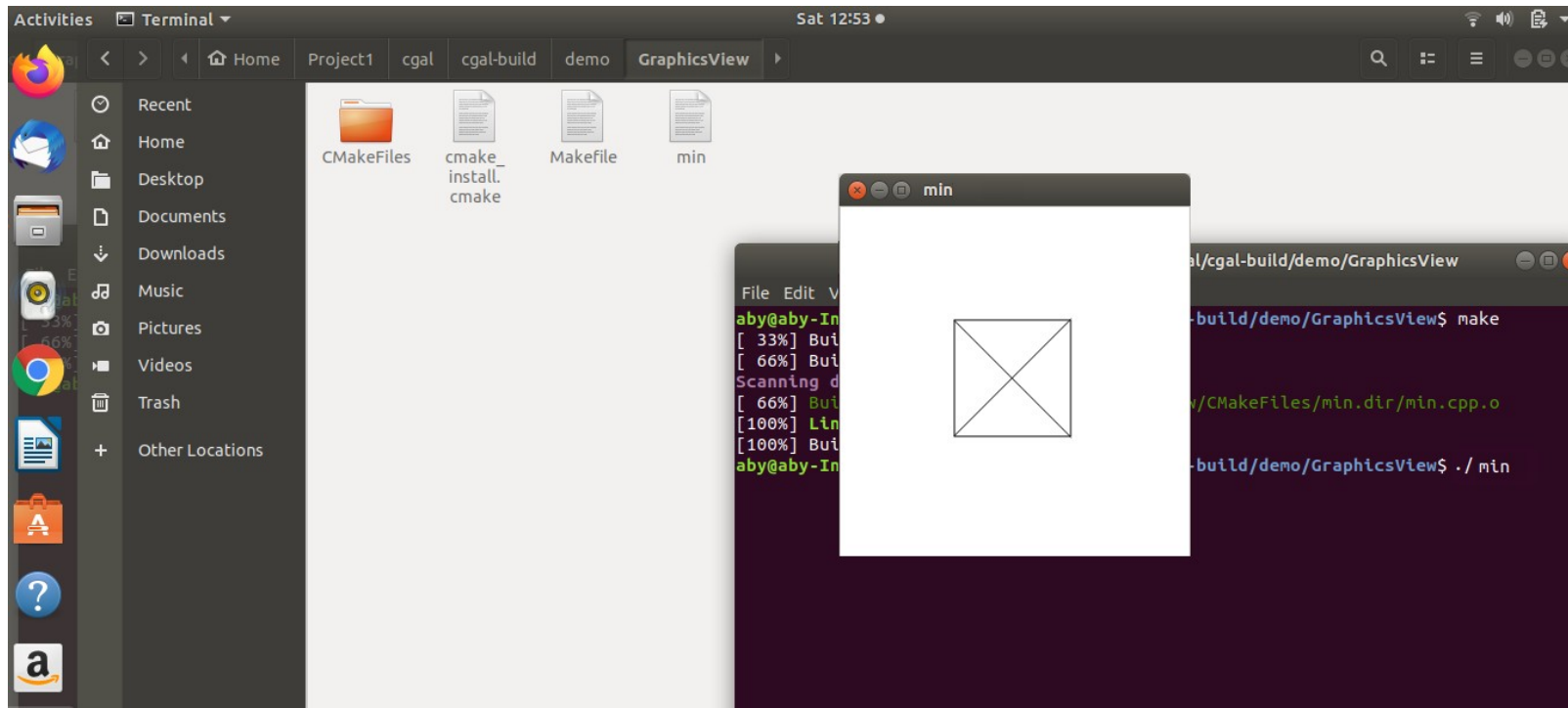
# Run a CGAL demo - GraphicsView

4. Open terminal inside this folder and create the executable by make command.
  - An executable file named min will be created inside the folder



# Run a CGAL demo - GraphicsView

5. Run the executable either by clicking it or by typing `./min` in the terminal



-This demo draws a rectangle and two diagonals of it in a QT window.

**THANK YOU**