For setting up **graphics.h** with MinGW on a Windows system, the steps are a bit different. Here’s how you can do it:

Step 1: Install MinGW and Set Up Environment

1. **Download and Install MinGW**:

- Go to the MinGW website and download the installer.
- Run the installer and select the **mingw32-gcc-g++** package for installation.
- Add MinGW to your system’s PATH environment variable. This allows you to use **gcc** and **g++** from the command line.

2. **Install the WinBGIm Library**:

- Download the WinBGIm library from this link.
- Extract the contents of the downloaded file.
- Copy **graphics.h** and **winbgim.h** to the MinGW include directory (e.g., **C:\MinGW\include**).
- Copy **libbgi.a** to the MinGW lib directory (e.g., **C:\MinGW\lib**).

Step 2: Compile and Run Your Program

1. **Write Your Program**:

- Example code to test:

```
#include <graphics.h>
int main() {
        int gd = DETECT, gm;
        initgraph(&gd, &gm, "");
        circle(50, 50, 30);
        getch();
        closegraph();
        return 0;
}
```

2. **Compile Your Program**:

- Open Command Prompt and navigate to the directory where your program is saved.
- Compile the program using the following command:

```
gcc -o myprogram myprogram.c -lbgi -lgdi32 -lcomdlg32 -luuid -loleaut32 -lole32
```

3. **Run Your Program**:

- Execute the compiled program:

```
./myprogram
```

Summary

- Step 1: Install MinGW and set up the WinBGIm library.

- Step 2: Compile and run your graphics program.

To use the **graphics.h** library with MinGW on Ubuntu, you need to follow these steps:

1. **Install Required Packages**:

```
sudo apt-get install build-essential libsdl-image1.2-dev guile-2.0-dev libsdl1.2debian libart-2.0-dev
libaudiofile-dev libesd0-dev libdirectfb-dev libfreetype6-dev libxext-dev x11proto-xext-dev libaa1-dev
libslang2-dev libasound2-dev
```

2. **Download and Install libgraph**:

•Download the **libgraph** package from this link.

•Extract the downloaded file:

```
tar -xvzf libgraph-1.0.2.tar.gz
```

•Navigate to the extracted directory and install:

```
cd libgraph-1.0.2
./configure
make
sudo make install
sudo cp /usr/local/lib/libgraph.* /usr/lib
```

3. **Include and Use graphics.h in Your Code**:

•Example code to test:

```
#include <graphics.h>
int main() {
        int gd = DETECT, gm;
        initgraph(&gd, &gm, NULL);
        circle(50, 50, 30);
        delay(5000);
        closegraph();
        return 0;
}
```

Analsysis for UNIX operations:

Step 1: Install Required Packages

This step installs various development tools and libraries that are essential for compiling and linking programs on Ubuntu. These packages include:

•build-essential: A package that includes the GCC compiler and other essential tools for building software.

•libsdl-image1.2-dev, libsdl1.2debian: Libraries for handling graphics and images.

•libfreetype6-dev: A library for font rendering.

•libxext-dev, x11proto-xext-dev: Libraries for X Window System extensions.

•Other libraries: These provide additional functionalities like audio support, framebuffer, and more.

Step 2: Download and Install libgraph

This step specifically deals with the **libgraph** library, which is a graphics library that provides the **graphics.h** header file. The steps include:

•Downloading: Getting the **libgraph** package from a repository.

•Extracting: Unpacking the downloaded file.

•Configuring and Installing: Running the configuration script and installing the library to make it available for use in your programs.

Differences

•Purpose: Step 1 sets up the general development environment with necessary tools and libraries, while Step 2 focuses on installing a specific graphics library (**libgraph**).

•Scope: Step 1 is broader, covering a range of libraries and tools needed for various development tasks. Step 2 is narrower, targeting the installation of a single library required for graphics programming.