**ECSE 321 – User Manual**

1. **Give the name of the project and the name(s) of the implementer(s)**

Our team designed a game called Bomberman, which is a strategic video game. The name of our project is called Hai- Bomb. Our team (Team 2) consists of:

* Frank Luong
* Henry-Michel Cantave
* Simon Ho
* Laurent Jacob
* Jeffrey Tichelman
* Raymond Guo

1. **Describe what the program does**

Python is a high-level programming language that can be applied to design our project. It comes with a large standard library. Our team did not use the most recent version of Python, instead we chose to use the version 2.7.6 which is the last version of the 2.x releases. The reasons why our team chose to use Python are because it is an open source and it has very good support for objects, modules and other gaming frameworks such as pygame. It is also easy to integrate with and extend using C and Java.

Pygame is a cross-platform set of Python modules to build and write video games with ease. Pygame also helps the user to easily get started and create something visible. By using the computer graphics and sound libraries, we were able to write a well-designed game. The program allows the user to smoothly control the character and complete the levels by strategically placing bombs around the map in order to kill the foes and destroy obstacles.

However, the performance of python does not scale to very large games. For example, as our team start importing large music file, it affects the speed performance of the game.

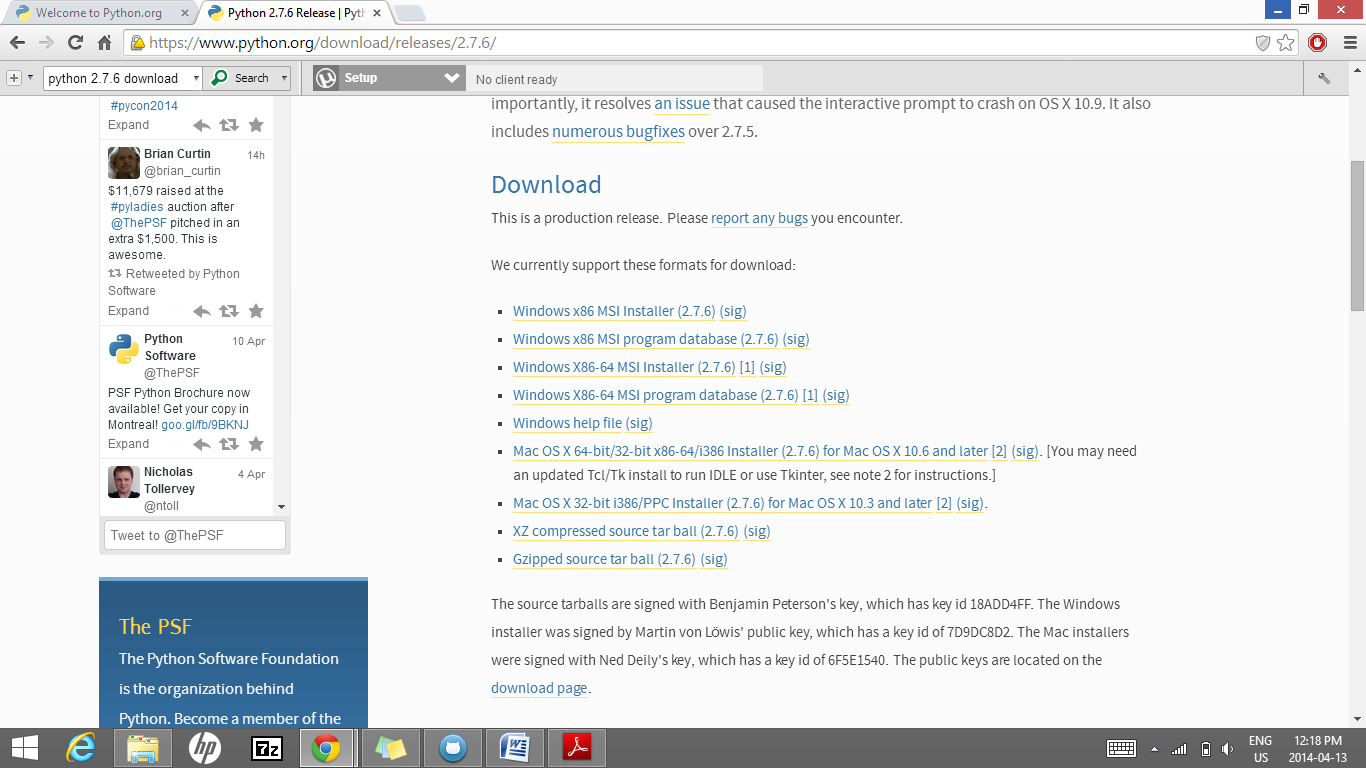
1. **Explains how to run the program.**Python version 2.7.6, the corresponding version of Pygame (py-2.7) and Github for windows has to be installed in order to play the game.

Here are the steps to run the program:

1. Download and install Python

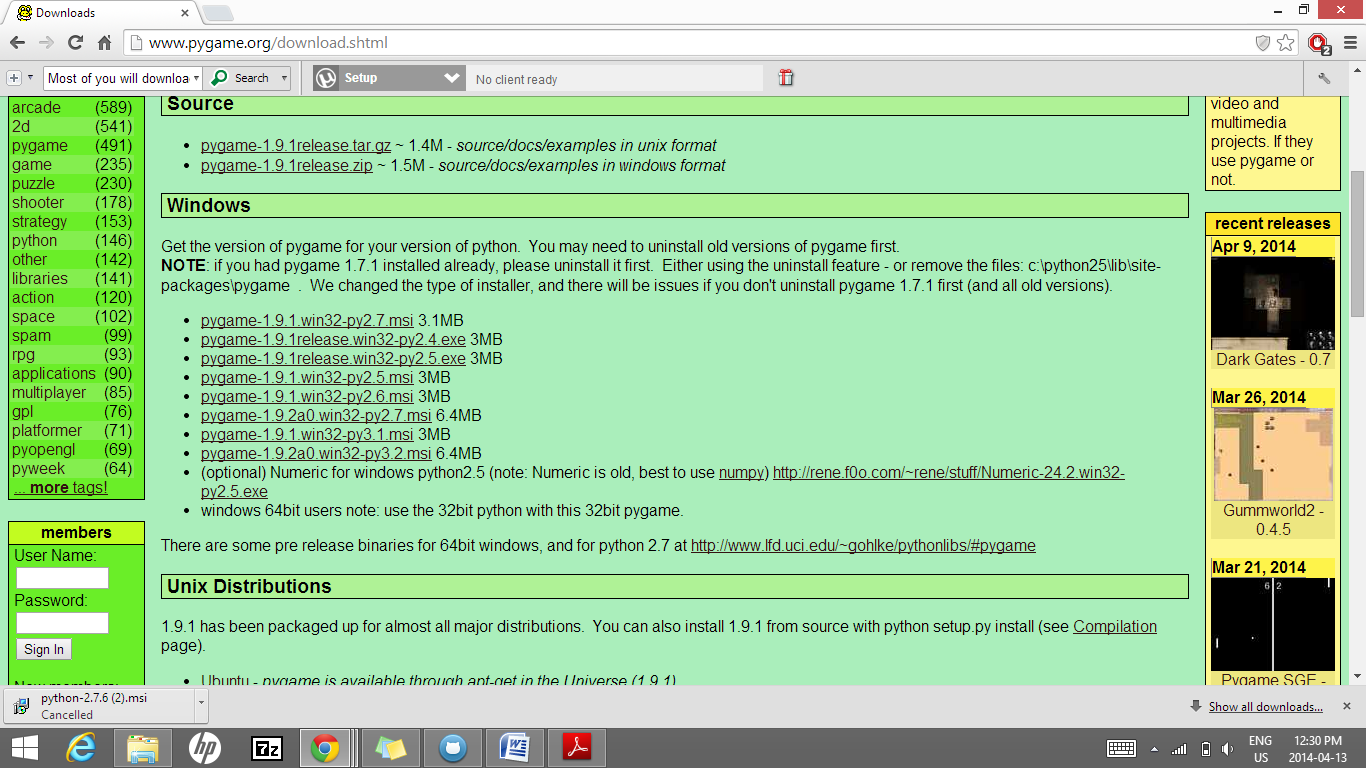
Visit the following link to download Python https://www.python.org/download/releases/2.7.6/

1. The user needs to pick the correct formats depending on his computer in order to run Python. Most of the user will download the installation package for Windows XP running on an x86 machine. (circle in red) However, you will need to use one of the other installation packages pictured below if you are using other formats. If you do not know the format of your machine, right click on My Computer and choose Properties.

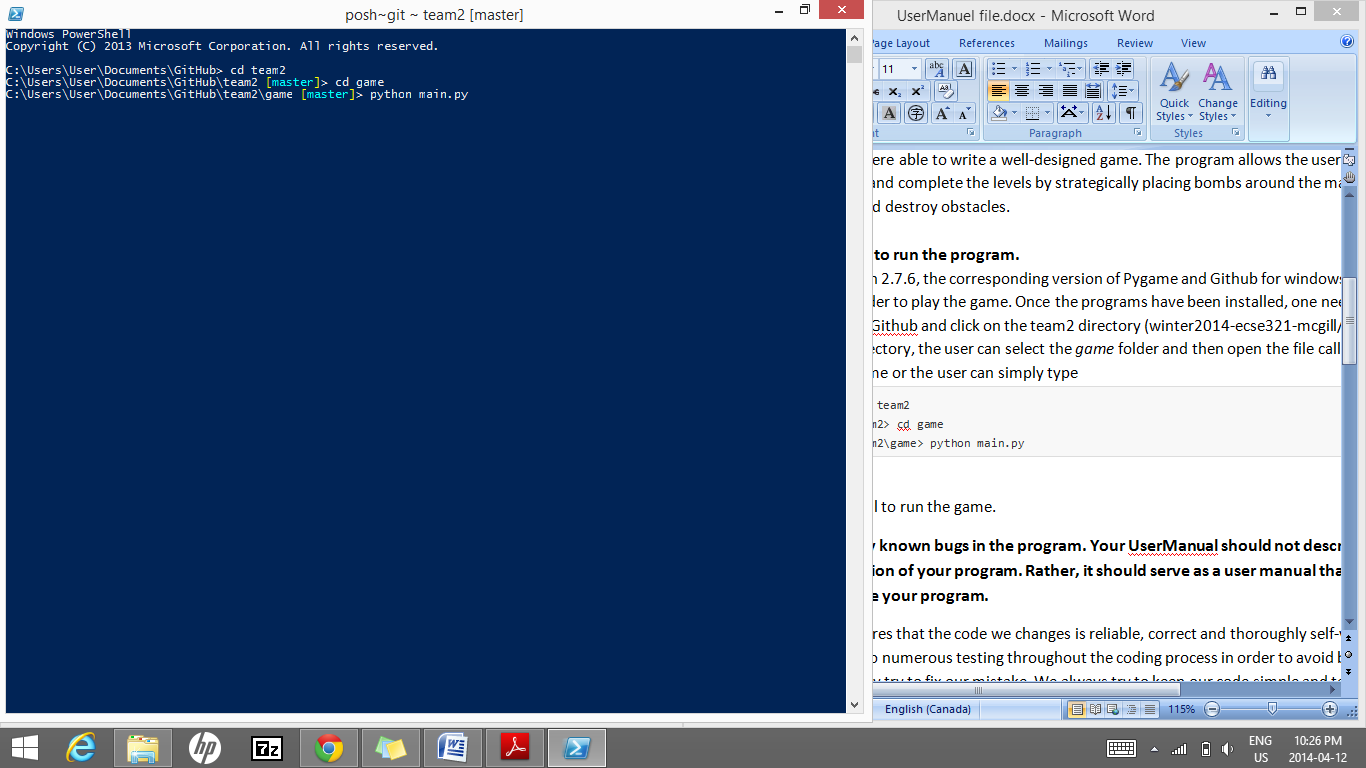


1. Click on the correct format and follow the instructions to install Python.
2. The user also needs to install pygame and here is the link to download it: <http://www.pygame.org/download.shtml>
3. The user needs to choose the correct version of pygame. Since we previously downloaded Python 2.7.6, the user needs to pick the pygame-1.9.2a0.win32-py2.7.msi (circle in red)

Note that you may need to uninstall old versions of pygame first before downloading a new one.



1. Click on the correct version of pygame and follow the instructions to install the game.
2. Once all the programs have been installed, one needs to pull a repository on Github and click on the team2 directory ([winter2014-ecse321-mcgill](https://github.com/winter2014-ecse321-mcgill)/[team2](https://github.com/winter2014-ecse321-mcgill/team2)). In the team2 directory, the user can select the *game* folder and then open the file called *main.py* to play the game or the user can simply type the following command lines on the Git Shell to run the game.

  
**Figure: How to run the game using Git shell**

1. **Describes any known bugs in the program.**

When a user already has pygame installed on his machine and tries to download another version of it, it would simply not run the desired version of pygame. The solution is to uninstall the older version and re-install the appropriate one to match your Windows and Python version.

For the code, our team usually ensures that the code we changes is reliable, correct and thoroughly self-verified. Therefore, our team do numerous testing throughout the coding process in order to avoid bugs. If a bug occurs, we immediately try to fix our mistake. We always try to keep our code simple and test groups of classes and make sure that the subsystems are correctly coded. At the end, we test the entire code in order to make sure that the game works as intended.

However, it is impossible to never have bugs in programming. One of the few minor bugs that we had was that when we place a bomb near several destructible crates, all the crates around the bomb would blow up. Fortunately, the bug was fixed immediately.