Code Workstation

Requirements Analysis Document

Lukas Rickard - Project Leader - l_rickard@u.pacific.edu
Jake MacMillan - Integrations Leader - j_macmillan1@u.pacific.edu

Project Scope

What will be in Code Workstation

- A text editor
    - Key shortcuts to program actions
- Buttons which will run an entered console commands
    - Customizable button pictures and colors
- Text boxes which display console command/program output
    - Resizing and docking of output boxes

What will not be in Code Workstation

- A built in debugger
- Groundbreaking concurrent efficiency
- Beautiful interface
- Built in compilers
- Syntax highlighting(Unless we find existing open-source code for it)

Requirements Summaries

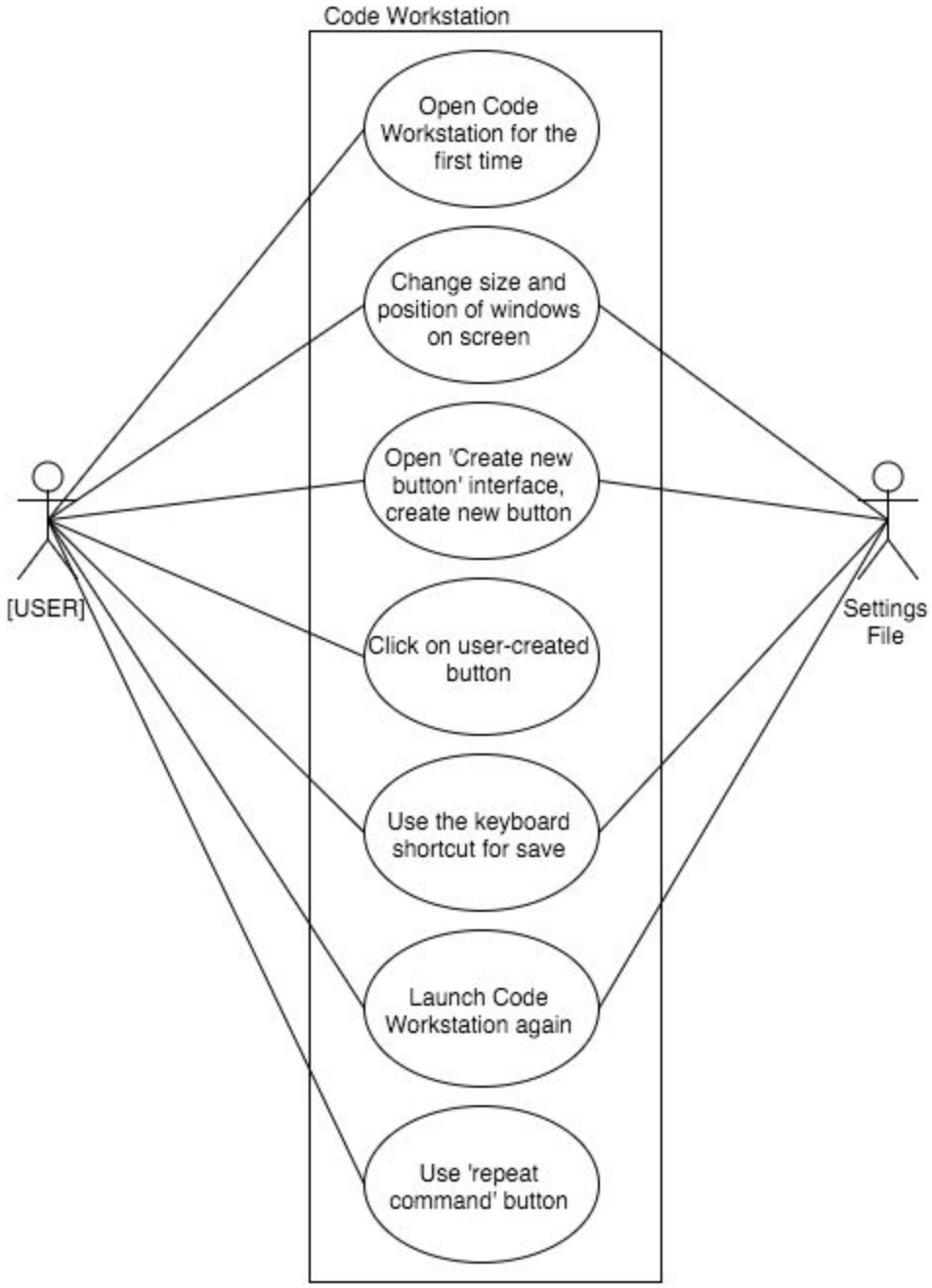
Non-functional Requirements

- Extensible Programming language (probably Python)
- Efficiency for many concurrent tasks
- Text editing needs to work how people expect it
- Responsiveness

Functional Requirements

- UI elements can be moved
- Commands can be run
- UI elements can be created

Use Case Diagram



Casual Use Case Descriptions

Starting for the first time:
[USER] has just downloaded Code Workstation (CW), and is beginning to set it up. Upon launch, CW provides the user with interface tips in a dialog window.

Changing window positions:
[USER] changes the size and positions of command boxes with mouse. Settings are saved to a file so that they can be accessed again the next time that [USER] accesses CW.

Creating a button:
[USER] is now coding in CW, and he wants to compile his code. He decides to set up a button that runs the gcc compilation command. Using the button-creation interface, he does so with ease.

Using an already-created button:
[USER] wants to use the button that he set up in Use Case 3. Upon pressing the button, the command is run and the output from that command is shown in the default command box.

Saving progress:
[USER] is done working on the project for the moment and wants to save his progress. He uses the keyboard shortcut designated for this task, and the code is saved to a file in the directory of his choice.

Starting again:
[USER] is ready to continue using Code Workstation. He opens the program back up, and it reads from the settings file to place all of his windows and buttons as he left them.

Setting up a repeating command:
[USER] wants to have a command repeat in a command box. He enters the command, then clicks on the ‘repeat’ button.

Fully Dressed Use Case Descriptions - User Stories

Title: Starting for the first time
Preconditions:

- User is starting code workstation for the first time

Main Flow:

1. User opens Code Workstation
2. User is provided with a dialog box outlining some tips on useful interface
3. User reads over tips, begins customizing interface

Exceptional Flows:

> User has opened Code Workstation before

>> Dialog does not open

Postconditions:

User has info on how to use Code Workstation and is not lost in complexity of our software and a settings file should be created but empty

Title: Changing window positions

Preconditions:

- User is currently using Code Workstation

Main Flow:

1. User decides that he wants his primary command box to be larger on the screen
2. User drags the corner of the window outward, increasing its size
3. The window's new size is noted in Code Workstation's settings file for later reference

Exceptional Flow:

> User does not select the corner precisely enough

>> User drags their mouse away from Command box

>>> Command box is same size

Postconditions:

Settings file has been updated

Command box is larger

Title: Creating and using a button to run a command

Precondition:

- Code Workstation has started correctly and the default command box is present

Main Flow:

1. User clicks the add button button
2. A dialog appears with a command to run and an output location
3. User enters ‘ls’ into the command to run and leaves the default command box for the output location
4. User left clicks the button that has been created
5. The output of ‘ls’ command is shown in the command box

Exceptional Flow:

> User clicks the X on dialog that appears to create a new button

>> The dialog closes and nothing else happens

> User enters an invalid command into the command to run

>> User left clicks the new button

>>> The command line error displays in the default command box

Postconditions:

There is the output of the ‘ls’ command visible in the default command box

Title: Using an already created button

Preconditions:

- User has used Code Workstation before and created a button
  (The settings file is present with data on the button)

Main Flow:

1. User opens Code Workstation
2. User sees their previously created button on the button bar
3. User clicks the button
4. The command assigned but the user during their last session runs and outputs to the assigned command box

Exceptional Flow:

> User quits Code Workstation without saving progress

Postconditions:

The assigned command box contains the output of the command they assigned to the button the last time they used code workstation

Title: Saving progress

Preconditions:

- User has been using Code Workstation for a while
- User is done working for now

Main Flow:

5. User uses the keyboard shortcut assigned to saving progress (defaultly ctrl + s)
6. Code Workstation saves the currently selected file to the chosen directory
7. Code Workstation saves any unsaved settings to the settings file
8. User quits Code Workstation

Exceptional Flow:

> User quits Code Workstation without saving progress

Postconditions:

Settings file and code are saved on the machine

Title: Starting again

Precondition:

- User has already used Code Workstation, saved, and exited the session

Main Flow:

1. User re-launches Code Workstation
2. User finds all settings in the same configuration as they were when he logged off
3. The code from the last time User was using Code Workstation is also present and available to be worked on again

Exceptional Flow:

> User had deleted the Code Workstation settings file

>> User is greeted with tips when Code Workstation starts

>>> Previous settings are not present

Postconditions:

User continues using Code Workstation without a hitch

Title: Using the command box to run a command constantly

Precondition:

- User has downloaded and started Code Workstation

Main Flow:

1. User clicks a command box
2. User types a command into the command section of the command box
3. User toggles the refresh button
4. User clicks the ‘1s’ tick box
5. The command box body clears and repopulates with the output of the command

Exceptional Flow:

> User does not select a time interval

>> 30 seconds is the default selection

> User enters an invalid command

>> User does step step 3 and 4

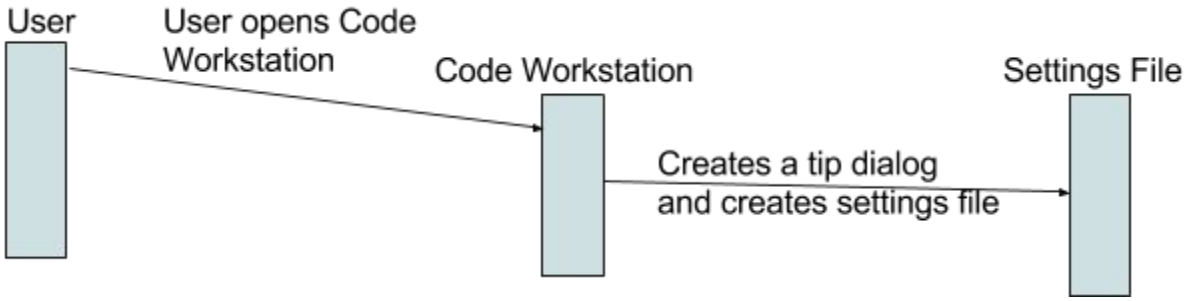
>>> The command box clears and refreshes an error every second

Postcondition:

The entered command is running and displaying in the command box every 1 second

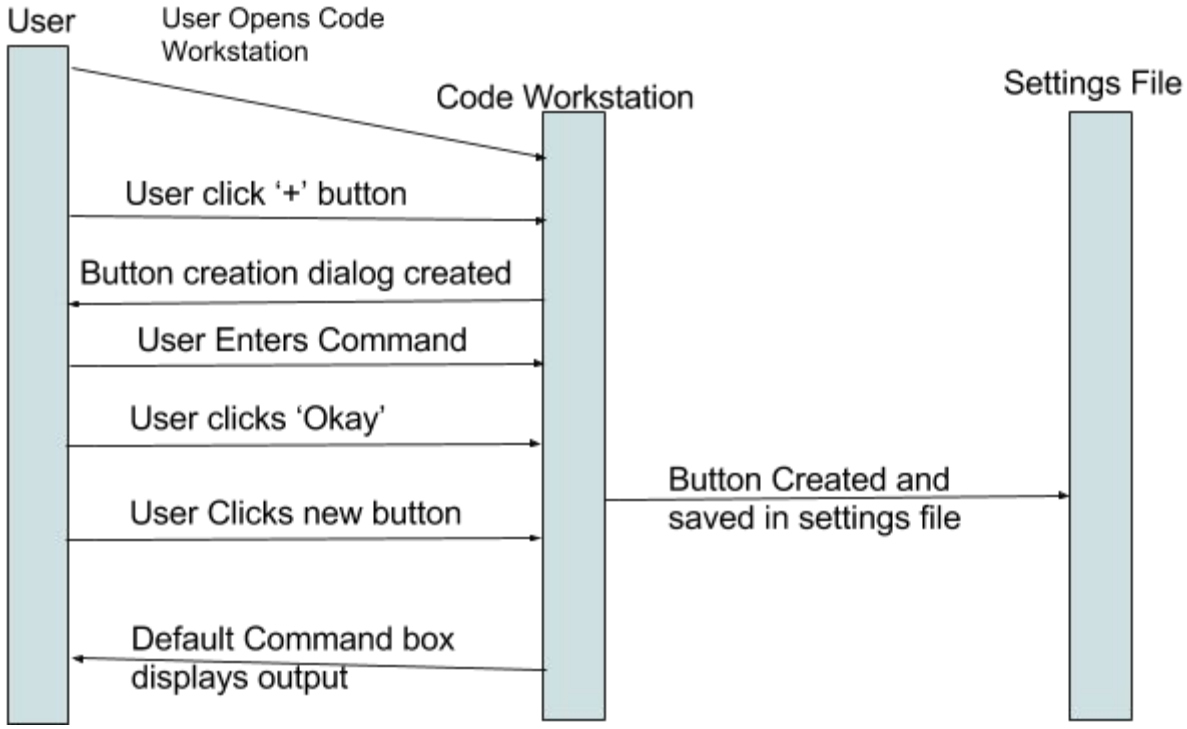
Sequence Diagrams

Sequence 1 - Starting for the first time



Sequence 2 - Changing window positions

Sequence 3 - Creating a button

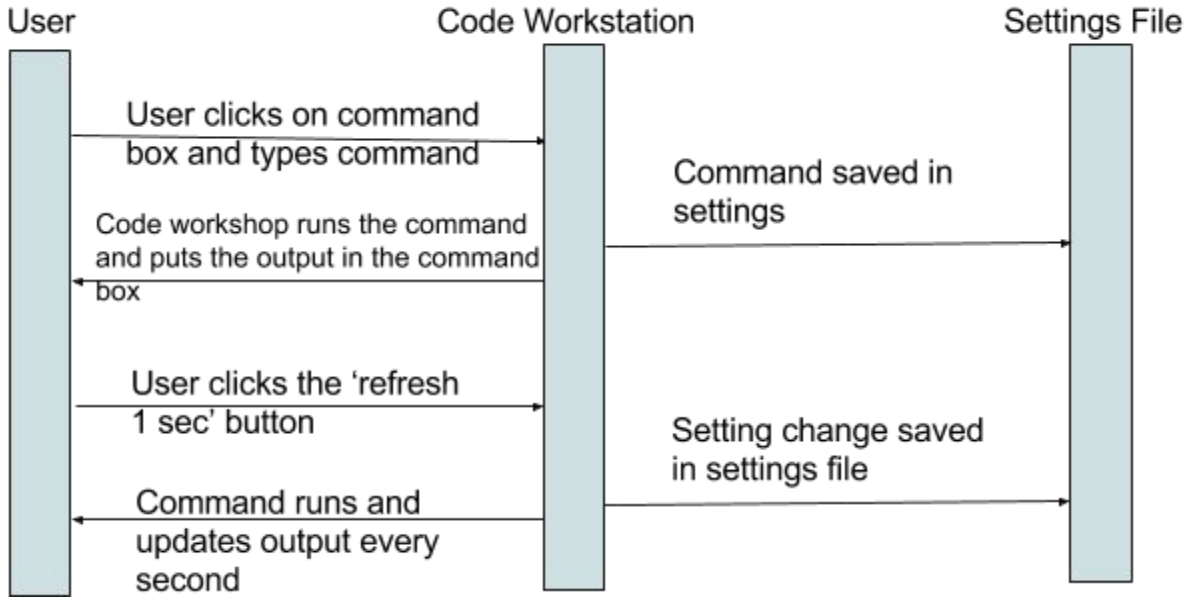


Sequence 4 - Using an already created button

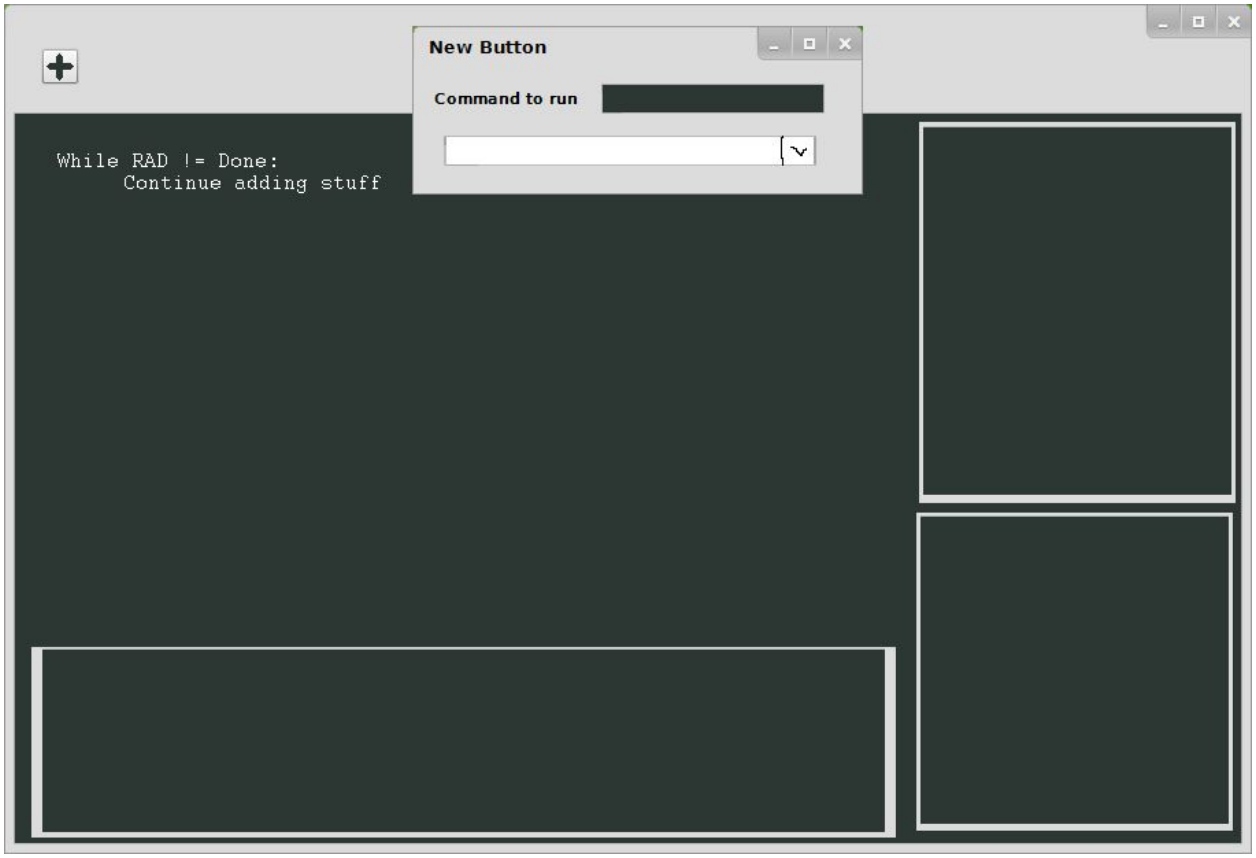
Sequence 5 - Saving Progress

Sequence 6 - Starting again

Sequence 7 - Using the command box to run a command constantly



User Interface Sketches



Glossary

Shell - A operating system interface or interface language
Console Command - A command run in a shell console
Linux - An open source operating system with many different distributions

References

Only self knowledge so far but might as well list some places relevant to the project (I will probably take these out in the final version unless they are wanted)

http://www.scintilla.org/SciTE.html
https://wiki.python.org/moin/PyQt
https://riverbankcomputing.com/software/pyqt/intro