Final

0.1.0

Generated by Doxygen 1.8.17

Chapter 1

doxygen-cmake-github

Demonstrates Doxygen html generation and publishing on GitHub Pages. The Doxygen files for this project can be seen `here`.

1.1 How to use

1. Point your browser to this repository (`https://github.com/semcneil/doxygen-cmake-github`)
2. Press the "Use this template" button
3. Give your repository a new name
4. Write a short (one sentence) description of what your project will do
5. Click the Create repository from template button

1.1.1 VS Code VM Instructions

1. VS Code needs the following extension added:
   (a) C/C++ from Microsoft
   (b) CMake Tools also from Microsoft
2. Connect to Host in New Window
3. Open a terminal (`ctrl+``)
4. Initialize git if you haven't already using the same email you used on your GitHub account:
   (a) `git config --global user.email "you@example.com"`
   (b) `git config --global user.name "Your Name"`
5. Navigate to the parent directory for your project
6. Clone your repository using the URL from the GitHub Code button on your repository and on VS Code either clone repository on the Welcome screen or open the Command Palette (ctrl+shift+P), type `git clone` and select `Git: Clone`
   (a) Select the parent directory for your project

   (b) Open the cloned repository either as prompted or by adding the newly created folder to your workspace by the Welcome tab's Open folder link or File -> Add Folder to Workspace
   (c) If you use the command line `git clone` the authentications for pushing to your online repository are not set up

7. If you wait a bit it should ask you which kit you want to use (at the time of this writing I typically use GCC 9.3.0)

8. Allow Intellisense if prompted

9. Edit README.md to reflect your new project

10. Edit the `project` line in the CMakeLists.txt file to have your project's name and version

11. Edit the `add_executable` line in the CMakeLists.txt file to change the name of the executable file to something relevant

12. Change the @brief, @details, @author, and @date in src/main.cpp

13. To create the PDF on a standard Ubuntu install, the following need to be added: `sudo apt install graphviz texlive-latex-base texlive-latex-recommended texlive-latex-extra`

14. Doxygen also needs installing: `sudo apt install doxygen`

15. In the terminal, change to the `build` directory (should have been automatically generated)

16. Run the following:
   (a) `make`
   (b) `make docs`
   (c) `make pdf`

17. Add the newly named PDF to git staging (`git status` -> `git add docs/yourprojectname.↩ pdf`)

18. Commit all the changes: `git commit -a -m "Initial commit"`

19. Push the changes to GitHub: `git push origin main`

20. Back at your repository on GitHub, refresh the page to show latest commit

21. In the Settings tab, scroll down to GitHub Pages

22. Select "Branch: main" as source and "/docs" as the folder and then press Save

23. Scroll back down to GitHub Pages and click the link to the published site

24. You now have a C++ repository with doxygen output hosted on GitHub Pages
   (a) The link usually doesn't work for a while (minutes to hours). This can be worked around by adding `index.html` to the end of the URL. A second commit will also fix it once the commit propagates over to GitHub Pages.
   (b) You can see the PDF file generated by Doxygen by adding the name of the PDF to the end of the URL. It will be of the form `projectname.pdf` and can be seen in the `docs` folder.
   (c) It can take a few minutes for a new `git push origin main` to propagate over to GitHub Pages

25. Edit README.md to reflect your project usage and point to the Doxygen output for your project

26. Stage the commit (`git add README.md`)

27. Commit (`git commit -a -m "Describe your changes here"`)

28. Push your changes to GitHub as before (`git push origin main`)

1.1.2 General Usage

During normal development, you will change main.cpp, maybe add more files in the src directory, make them, and run them. To update the documentation on the web do the following at a terminal prompt in your project's `build` directory:

1. `make`
2. `make docs`
3. `make pdf` Then in your project's root directory do the following:
4. Check the git status: `git status`
5. `git commit -a -m "Describe your changes since last commit"`
   (a) The `-a` flag is used to commit all the updated documentation files
   (b) VS Code also has git built into it, but the use of branches isn't as easy a workflow as the commandline offers for me (personal opinion).
6. Note that in order for numbered (ordered) lists to work across markdown and Doxygen HTML and PDF outputs they are explicitly numbered vs markdown all being 1. or Doxygen's -#.

1.2 References

1. https://www.doxygen.nl/manual/docblocks.html
2. https://stackoverflow.com/questions/44212101/cmake-how-to-have-add-custom-command-r
3. Very useful overview: https://caiorss.github.io/C-Cpp-Notes/Doxygen-documentation.↩ html
4. https://devblogs.microsoft.com/cppblog/clear-functional-c-documentation-with-sphinx
5. https://vicrucann.github.io/tutorials/quick-cmake-doxygen/
6. https://medium.com/practical-coding/c-documentation-with-doxygen-cmake-sphinx-breat
7. https://stackoverflow.com/questions/18590445/cmake-custom-command-to-copy-and-renam

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 TrieNode Struct Reference

Public Member Functions

- TrieNode ()

Public Attributes

- unordered_map< char, TrieNode ∗ > child
- bool lastLetter

4.1.1 Detailed Description

Definition at line 14 of file final.cpp.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 TrieNode()

```
TrieNode::TrieNode ( )  [inline]
```

Definition at line 18 of file final.cpp.

```
    {
        for (char i = ’a’; i <= ’z’; i++)
            child[i] = NULL;
        lastLetter = false;
    }
```

4.1.3 Member Data Documentation

4.1.3.1 child

```
unordered_map<char,TrieNode*> TrieNode::child
```

Definition at line 16 of file final.cpp.

4.1.3.2 lastLetter

```
bool TrieNode::lastLetter
```

Definition at line 17 of file final.cpp.

The documentation for this struct was generated from the following file:

- /home/benson/CPTR227/Final/src/final.cpp

Chapter 5

File Documentation

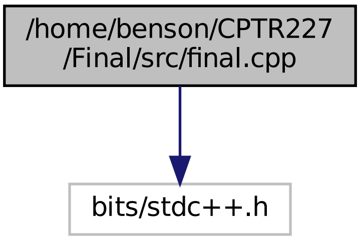
5.1 /home/benson/CPTR227/Final/README.md File Reference

5.2 /home/benson/CPTR227/Final/src/final.cpp File Reference

This is a test of CMake, doxygen, and GitHub.

```
#include <bits/stdc++.h>
```

Include dependency graph for final.cpp:



Classes

- struct TrieNode

Functions

- void AddContact (string s)
- void showContactsTil (TrieNode ∗currentNode, string pre)
- void showContacts (string str)
- void insertInDirectory (string contacts[ ], int n)
- int main ()

Variables

- string contactList [ ] = {"Seth", "Scott","Benson"}
- TrieNode ∗ root = NULL

5.2.1 Detailed Description

This is a test of CMake, doxygen, and GitHub.

This is the long brief at the top of main.cpp.

Author

Benson Nyakango and Scott Gillis

Date

5/5/2021

5.2.2 Function Documentation

5.2.2.1 AddContact()

```
void AddContact (
             string s )
```

Definition at line 28 of file final.cpp.

```
{
    int length = s.length();
    TrieNode *head = root;
    for (int i = 0; i < length; i++)
    {
        TrieNode *nextNode = head->child[s[i]];
        if (nextNode == NULL)
        {
            nextNode = new TrieNode();
            head->child[s[i]] = nextNode;
        }

        head = nextNode;
        if (i == length - 1)
            head->lastLetter = true;
    }
}
```

5.2.2.2 insertInDirectory()

```
void insertInDirectory (
             string contacts[],
             int n )
```

Definition at line 91 of file final.cpp.

```
{
    root = new TrieNode();
    for (int i = 0; i < n; i++)
        AddContact(contacts[i]);
}
```

5.2.2.3 main()

```
int main ( )
```

Definition at line 97 of file final.cpp.

```
98 {
99     int n = sizeof(contactList)/sizeof(string);
100     insertInDirectory(contactList, n);
101     string Search;
102     cout « "Search: ";
103     cin » Search;
104     cout « "Results: " « Search « endl;
105     showContacts(Search);
106     return 0;
107 }
```

5.2.2.4 showContacts()

```
void showContacts (
               string str )
```

Definition at line 59 of file final.cpp.

```
60 {
61     TrieNode *previousNode = root;
62     string pre = "";
63     int length = str.length();
64     int i;
65     for (i=0; i<length; i++)
66     {
67         pre += (char)str[i];
68         char lastChar = pre[i];
69         TrieNode *currentNode = previousNode->child[lastChar];
70         if (currentNode == NULL)
71         {
72             cout « "No result found "«" " « pre
73                  « endl;
74             i++;
75             break;
76         }
77 
78         cout « "Suggesions for" «" "« pre«" "
79              « "are "« endl;
80         showContactsTil(currentNode, pre);
81         previousNode = currentNode;
82     }
83     for (; i<length; i++)
84     {
85         pre += (char)str[i];
86         cout « "No contact found for"«" " « pre
87              « endl;
88     }
89 }
```

5.2.2.5 showContactsTil()

```
void showContactsTil (
               TrieNode * currentNode,
               string pre )
```

Definition at line 47 of file final.cpp.

```
48 {
49     if (currentNode->lastLetter)
50         cout « pre « endl;
51    for (char i = ’a’; i <= ’z’; i++)
52     {
53         TrieNode *next = currentNode->child[i];
54         if (next != NULL)
55             showContactsTil(next, pre + (char)i);
56     }
57 }
```

5.2.3 Variable Documentation

5.2.3.1 contactList

```
string contactList[] = {"Seth", "Scott","Benson"}
```

Definition at line 13 of file final.cpp.

5.2.3.2 root

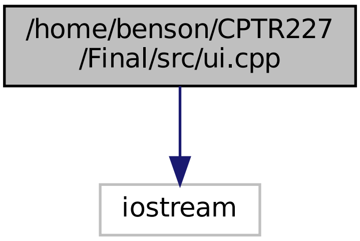
```
TrieNode* root = NULL
```

Definition at line 26 of file final.cpp.

5.3 /home/benson/CPTR227/Final/src/ui.cpp File Reference

```
#include <iostream>
```

Include dependency graph for ui.cpp:



Functions

- int menu ()
- int main ()

Variables

- int k =0

5.3.1 Function Documentation

5.3.1.1 main()

```
int main ( )
```

Definition at line 25 of file ui.cpp.

```
{
    string Fname[100];
    string Pnumber[100];
    int check=0;
    int Total_contacts=0;
    check=menu();
    do
    {
        // Add contacts
        if(check==1)
        {
            cout«"\t\t\t\t\t\t Name: ";
            cin»Fname[k];
            cout«"\t\t\t\t\t\t Phone no: ";
            cin»Pnumber[k];
            k++;
            Total_contacts++;
        }

        //Display contacts
        else if (check==2)
        {
            int check2=0;
            for(int i=0; i<100;i++)
            {
                if(Fname[i]!="\0")
                cout«"\t\t\t\t\t Contact Name: "«Fname[i]« "     Phone # of contact: "«Pnumber[i]«endl;
                check2++;
            }
            if(check2==0)
            {
                cout«"\t\t\t\t\t";
            }
        }
        //Search by Number
        else if(check==3)
        {
            string temp;
            cout«"\t\t\t\t\tNumber: ";
            cin»temp;
            int check2=0;
            for(int i=0;i<100;i++)
            {
                if(temp==Pnumber[i])
                {
                    cout«"\t\t\t\t\tNumber is Found\n";
                    cout«"\t\t\t\t\tName : "«Fname[i]«"       Phone : "«Pnumber[i]«endl;
                    check2++;
                }
            }
            if(check2==0)
            {
                cout«"\t\t\t\t\t This Number is Not found in your contact list\n";
            }
        }
        //Search By Name
        else if(check==4)
        {
                string temp;
            cout«"\t\t\t\t\tName : ";
            cin»temp;
            int check2=0;
            for(int i=0;i<100;i++)
            {
                if(temp==Fname[i])
                {
                    cout«"\t\t\t\t\tName is Found\n";
                    cout«"\t\t\t\t\tName : "«Fname[i]«"       Phone : "«Pnumber[i]«endl;
                    check2++;
                }

```

```
           }
           if(check2==0)
           {
                cout«"\t\t\t\t\t This name is Not found in your contact list\n";
            }
        }
        // Update
        else if(check==5)
        {
            string temp,temp2,temp3;
            cout«"\t\t\t\t\tName : ";
            cin»temp;
            int check2=0;
            for(int i=0;i<100;i++)
            {
                if(temp==Fname[i])
                {
                    cout«"\t\t\t\t\tNew Name : ";
                    cin»temp2;
                    cout«"\t\t\t\t\tNew Number : ";
                    cin»temp3;
                    Fname[i]=temp2;
                    Pnumber[i]=temp3;
                    check2++;
                    cout«"\t\t\t\t\tUpdated Successfully ";
                }

            }
                if(check2==0)
            {
                cout«"\t\t\t\t\t This name is Not found in your contact list\n";
            }
        }
        // Delete
        else if(check==6)
        {
                string temp;
            cout«"\t\t\t\t\tFor Delete Enter Name : ";
            cin»temp;
            int check2=0;
            for(int i=0;i<100;i++)
            {
                if(temp==Fname[i])
                {
                    cout«"\t\t\t\t\tDeleted Successfully\n";
                    cout«"\t\t\t\t\tName : "«Fname[i]«"       Phone : "«Pnumber[i]«endl;
                    Fname[i]="\0";
                    Pnumber[i]="\0";
                    check2++;
                    Total_contacts--;
                }
            }
            if(check2==0)
            {
                cout«"\t\t\t\t\t This name is Not found in your contact list\n";
            }
        }
        // Deletes all the contacts
        else if(check==7)
        {

                    cout«"\t\t\t\t\t All Deleted Successfully\n";
                    for(int i=0;i<k;i++)
                    {
                        Fname[i]="\0";
                        Pnumber[i]="\0";
                    }
                    k=0;
                    Total_contacts=0;
        }

        check=menu();

    }while(check!=8);

}
```

5.3.1.2 menu()

```
int menu ( )
```

Definition at line 173 of file ui.cpp.

```
{
    cout«"\t\t\t\t\t--------------------------------------------\n";
    cout«"\t\t\t\t\t|          Phonebook Program               |\n";
    cout«"\t\t\t\t\t--------------------------------------------\n";
    cout«"\t\t\t\t\t|       1.   Add Contacts                  |\n";
    cout«"\t\t\t\t\t|       2.   Display All Contacts          |\n";
    cout«"\t\t\t\t\t|       3.   Search by Number              |\n";
    cout«"\t\t\t\t\t|       4.   Search by Name                |\n";
    cout«"\t\t\t\t\t|       5.   Modify                        |\n";
    cout«"\t\t\t\t\t|       6.   Delete                        |\n";
    cout«"\t\t\t\t\t|       7.   Delete All                    |\n";
    cout«"\t\t\t\t\t|       8.   Exit                          |\n";
    cout«"\t\t\t\t\t--------------------------------------------\n";

    int a;
    cin»a;
    return a;
}
```

5.3.2 Variable Documentation

5.3.2.1 k

```
int k =0
```

Definition at line 22 of file ui.cpp.

Index