|  |  |
| --- | --- |
|  |  |
| МИНОБРНАУКИ РОССИИ |  |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |  |

**Институт информационных технологий**

КАФЕДРА КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

**ОТЧЁТ**

**ПО ДОМАШНЕЙ РАБОТЕ №5**

**по дисциплине**

«Конфигурационное управление»

Выполнил студент группы ИКБО-24-21 *Елков Е.М.*

Принял *Емельянов А.М.*

Практическая работа выполнена «\_\_\_»\_\_\_\_\_\_\_2022г.

«Зачтено» «\_\_\_»\_\_\_\_\_\_\_2022г.

Москва 2022

# Домашняя работа №5: «Визуализатор коммитов для git»

Разработать CLI-утилиту для выдачи информации по локальному git-репозиторию. Назначение утилиты – визуализатор графа изменений дерева (tree: файлы и папки) проекта с выводом в формате dot (graphviz).

Данные извлекаются из папки .git без использования сторонних программ и библиотек. То есть нельзя из программы вызывать команды git для получения данных (например "git log" и "git cat-file"). Но можно и нужно пользоваться сведениями отсюда: <https://git-scm.com/book/ru/v2/Git-%D0%B8%D0%B7%D0%BD%D1%83%D1%82%D1%80%D0%B8-%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D1%8B-Git>

Обязательно привести пример работы утилиты.

1. Постановка задачи

Дано:

Git-репозиторий

Результат:

Графическое представление графа коммитов

1. Подход к решению

Исходя из условия задачи, самым оптимальном языком программирования для её реализации будет python. Приведём код решения.

Вывод программы вставим на сайт «https://dreampuf.github.io/GraphvizOnline/», который производит отображение искомого графа.

1. Код:

import os  
import zlib  
  
import graphviz  
  
  
def clearMessage(message):  
 # b'Adding readme'  
 return message.decode("utf-8")  
  
  
def clearParents(dt):  
 *"b'parent 2ac228697173ab1e6381ed608c73fd022af11a4b'"* # каждую запись такого вида разбиваем по пробелам, берем хеш как второй элемент  
 # добавляем в список, предварительно декодировав его  
 # возвращаем список  
 return [parent.split(b" ")[-1].decode("utf-8") for parent in dt if parent.startswith(b"parent")]  
  
  
def deserialize(data):  
 # нас интересуют следующие поля:  
 # parent, message  
 dt = data.split(b"\n") # Разбиваем на строки  
 # Удаляем пустые строку  
 dt.remove(b"")  
 dt.remove(b'')  
 # информация с сообщением - последнее поле  
 roughMessage = dt[-1]  
 # оно выглядит так:  
 # b'Adding readme'  
 parents = clearParents(dt) # чистые хеши родителей  
 message = clearMessage(roughMessage) # чистый текст сообщения  
 return [parents, message]  
  
def buildGraph(path\_to\_rep, graph):  
 digraph = graphviz.Digraph(comment=f'Git Log for {path\_to\_rep}')  
 digraph.attr('node', shape='octagon')  
 for node in graph:  
 for parent in node[1]:  
 digraph.edge(parent, node[0], label=node[2])  
 digraph.render(f'{path\_to\_rep}.gv', view=True)  
  
  
unpack = zlib.decompressobj()  
graph = []  
path\_to\_rep = "D:\\ConfigPr5\\ThirdTask\\bare\_repo\\.git" # Путь к репозиторию  
os.chdir("D:\\ConfigPr5\\ThirdTask\\bare\_repo\\.git") # Переходим в папку с репозиторием  
for path, directories, files in os.walk("../../Downloads"): # Проходим по всем файлам  
 if files:  
 for file in files:  
 with open(path + "\\" + file, "rb") as f:  
 data = zlib.decompress(f.read())  
 if data[:6] == b"commit":  
 commit\_id = path[-2:] + file

parents, message = deserialize(data)  
 graph.append([commit\_id, parents, message])  
 f.close()  
os.chdir("D:\\ConfigPr5\\HomeWork")  
buildGraph(path\_to\_rep, graph)

1. Тестирование
   1. Для тестирования работы программы создадим тестовый git-репозиторий. Отобразим его с помощью визуализатора коммитов в среде разработки PyCharm (рисунок 1):

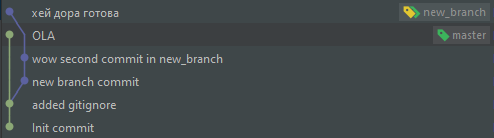


Рисунок 1 – данный git-репозиторий

* 1. Запустим программу и скопируем полученный вывод на сайт «https://dreampuf.github.io/GraphvizOnline/», и проверим соответствие полученного графа требуемому (рисунок 2):

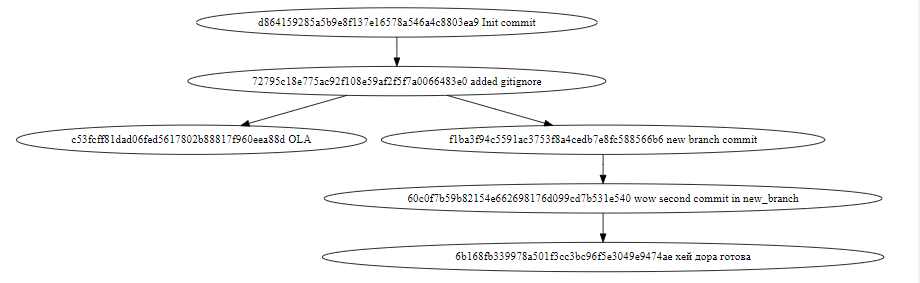


Рисунок 2 – полученный в результате работы программы граф

Исходя из результатов тестирования, можно сделать вывод, что программа работает корректно.

1. ВЫВОДЫ

В ходе выполнения данной работы были изучены методы работы со строками, содержимым файлов и папок в структуре git. Также была изучена сама структура git и принципы его работы.