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

Институт Информационных технологий

Кафедра информационных технологий в системах управления

**КУРСОВАЯ РАБОТА**

по дисциплине «Технология разработки программного обеспечения АСОИУ»

**Тема курсовой работы** «Документирование утилиты CALL командного процессора CMD в публичном репозитории GitHub»

**Студент группы** ИВБО-10-17 Корляков Олег Владиславович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель курсовой работы** Сувальский Андрей Александрович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена к защите «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Допущен к защите «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Москва 2020

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

Институт Информационных технологий

Кафедра информационных технологий в системах управления

**ЗАДАНИЕ**

**на выполнение курсовой работы по дисциплине**

«Технология разработки программного обеспечения АСОИУ»

Корляков Олег Владиславович Группа ИВБО-10-17

**Тема работы:** Документирование утилиты CALL командного процессора CMD в публичном репозитории GitHub

**Исходные данные:** справочнаяинформация по утилите CALL из открытых источников

**Перечень вопросов, подлежащих разработке:**

Документ ТЗ утилиты CALL, его редактирование при помощи языка разметки markdown, загрузка отредактированного документа на веб-сервис GitHub с помощью команд git.

Отчет по курсовой работе в виде расчетно-пояснительной записки.

**Срок представления к защите курсовой работы:** до «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**Задание на курсовую работу выдал** \_\_\_\_\_\_\_\_\_\_\_\_ (Сувальский А.А.)

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**Задание на курсовую работу получил**  \_\_\_\_\_\_\_\_\_\_\_\_ (Гаврюшин И.А.)

# Оглавление

[Оглавление 3](#_Toc59721756)

[1 Введение 4](#_Toc59721757)

[1.1 Что такое контроль версий? 4](#_Toc59721758)

[1.2 Что такое Git? 5](#_Toc59721759)

[1.3 Использование Git 5](#_Toc59721760)

[2 Команда CALL 7](#_Toc59721761)

3 Основная часть……………………………………………………...........9

[4 Заключение 10](#_Toc59721762)

[Использованная литература 11](#_Toc59721763)

# Введение

Ядро Linux - это достаточно большой проект с открытым исходным кодом. Большую часть времени разработки ядра Linux (1991-2002 гг.) изменения передавались между разработчиками в виде патчей и архивов. В 2002 году проект ядра Linux начал использовать проприетарную децентрализованную СКВ BitKeeper.

В 2005 году отношения между сообществом разработчиков ядра Linux и коммерческой компанией, которая разрабатывала BitKeeper, прекратились, и бесплатное использование утилиты стало невозможным. Это сподвигло сообщество разработчиков ядра Linux (в частности Линуса Торвальдса - создателя Linux) разработать свою собственную утилиту, учитывая уроки, полученные при работе с BitKeeper. Некоторыми целями, которые преследовала новая система, были:

* скорость;
* простая архитектура;
* хорошая поддержка нелинейной разработки (тысячи параллельных веток);
* полная децентрализация.

Возможность эффективного управления большими проектами, такими как ядро Linux (скорость работы и разумное использование дискового пространства)

С момента своего появления в 2005 году, Git развился в простую в использовании систему, сохранив при этом свои изначальные качества. Он удивительно быстр, эффективен в работе с большими проектами и имеет великолепную систему веток для нелинейной разработки.

* 1. Что такое контроль версий?

Контроль версий – это возможность сохранить свое место в документе или папке и сослаться на предыдущие сохранения.

При написании этой работы, новые изменения постоянно перезаписывают предыдущие версии, но благодаря контролю версий git этого можно было бы избежать, при помощи возврата к черновикам.

* 1. Что такое Git?

Git - распределённая система контроля версий, которая позволяет сохранять все изменения, внесённые в файлы, хранящиеся в репозитории. Система спроектирована как набор программ, специально разработанных с учётом их использования в сценариях. Это позволяет удобно создавать специализированные системы контроля версий на базе Git или пользовательские интерфейсы. Например, Cogito является именно таким примером оболочки к репозиториям Git, а StGit использует Git для управления коллекцией исправлений (патчей).

Git поддерживает быстрое разделение и слияние версий, включает инструменты для визуализации и навигации по нелинейной истории разработки. Как Darcs, BitKeeper, Mercurial и Bazaar, Git предоставляет каждому разработчику локальную копию всей истории разработки, изменения копируются из одного репозитория в другой.

Удалённый доступ к репозиториям Git обеспечивается git-демоном, SSH- или HTTP-сервером. TCP-сервис git-daemon входит в дистрибутив Git и является наряду с SSH наиболее распространённым и надёжным методом доступа. Метод доступа по HTTP, несмотря на ряд ограничений, очень популярен в контролируемых сетях, потому что позволяет использовать существующие конфигурации сетевых фильтров. Ввиду всех своих преимуществ Git - незаменимый инструмент для каждого, кто занимается или собирается заниматься разработкой.

* 1. Использование Git

С Git можно работать как через командную строку, так и через графиче ский интерфейс вроде GitHub Desktop. Хотя начинающих разработчиков командная строка может отпугнуть, её всё равно лучше изучить, ведь она предоставляет больше возможностей, чем многие инструменты с интерфейсом.

Как правило, команды Git принимают вид git <команда> <аргументы>, где аргументом может быть путь к файлу. У команд также могут быть опции, которые обозначаются как -<опция> или -<однобуквенная опция>. Они позволяют более детально настроить действия команды.

Далее будет рассмотрено применение этой системы через командную строку на примере добавления технического задания на разработку утилиты CALL командного процессора CMD.

# Команда CALL

**Команда CALL** в Ассемблере вызывает процедуру. Синтаксис:

CALL ИМЯ

Здесь ИМЯ может быть одним из следующих:

1. Имя процедуры
2. [Метка](http://av-assembler.ru/instructions/jmp.php#label)
3. Переменная
4. [Регистр](http://av-assembler.ru/asm/afd/asm-cpu-registers.htm)
5. Адрес (непосредственное значение)

[Флаги](http://av-assembler.ru/asm/afd/asm-flags-register.htm) при выполнении команды CALL не изменяются.

Инструкция CALL выполняет следующие действия:

* Передаёт управление процедуре.
* Записывает в стек адрес следующей команды (из [регистра IP](http://av-assembler.ru/asm/afd/asm-cpu-registers.htm) в 16-разрядном режиме или EIP - в 32-разрядном).
* В случае дальнего вызова регистр CS также записывается в стек.

Если вместо имени процедуры используется адрес, то он может быть записан в формате [СЕГМЕНТ:СМЕЩЕНИЕ](http://av-assembler.ru/asm/afd/asm-real-mode.htm). Если в качестве адреса указать только смещение, то считается, что адрес расположен в том же сегменте, что и команда CALL (при этом выполняется ближний вызов процедуры).

Как уже было сказано, команда CALL передаёт управление по адресу, который передаётся в команду в качестве параметра. При этом процессор начинает выполнять команду, расположенную по этому адресу.

Если операндом команды CALL является регистр или переменная, то её значение рассматривается как абсолютное смещение.

Если операнд - это ближняя метка в программе, то ассемблер указывает её относительное смещение.

Ну и напоследок об имени команды. Слово CALL переводится как “вызов” или “вызывать, вызвать”. То есть команда вызова.

CALL Х

по-русски будет звучать как “вызвать Х”, где Х - это, как правило, имя процедуры. Хотя, как вы теперь знаете, на самом деле эта команда не вызывает процедуру, а передаёт управление той команде, которая находится по указанному адресу Х.

В ходе выполнения задания курсовой работы были выполнены следующие пункты:

* составление документ ТЗ утилиты CALL командной строки CMD;
* редактирование с использованием языка разметки markdown;
* перенос в веб-сервис GitHub с помощью git.

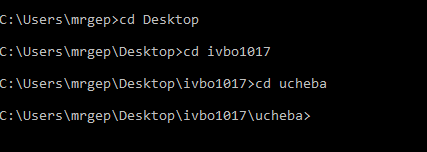
Опишем действия, производимые для добавления документа ТЗ утилиты CALL командной строки CMD, отредактированным с использованием языка разметки markdown в веб-сервис GitHub посредством консольных команд git.

# 3 Основная часть

В ходе выполнения задания курсовой работы были выполнены следующие пункты:

* составлен документ ТЗ утилиты CALL командной строки CMD;
* отредактирован с использованием языка разметки markdown;
* перенесён в веб-сервис GitHub с помощью git.

Далее опишу действия, производимые мной для добавления документа ТЗ утилиты CALL командной строки CMD, отредактированным с использованием языка разметки markdown в веб-сервис GitHub посредством консольных команд git.

1. Была установлена на персональный компьютер система контроля версий git v2.29.2.3 для операционной системы Windows с официального сайта.
2. Произведена регистрация в веб-сервисе GitHub.
3. Непосредственно на сайте GitHub был создан открытый (public) репозиторий под именем «IVBO1017». После создания GitHub предложил нам ссылку на созданный репозиторий, она понадобится нам позже.
4. В консоли был осуществлен переход к папке с нужным файлом командой:  
   
5. Осуществлена инициализация git в текущей папке командой:  
   C:\Users\mrgep\Desktop\ivbo1017\ucheba>git init
6. Для создания удаленного репозитория в локальном была введена команда, использующая ссылку из третьего пункта, после создания репозитория на сайте GitHub:

C:\Users\mrgep\Desktop\ivbo1017\ucheba>git remote add pb https://github.com/ggepardik/IVBO1017.git

1. Был выполнен вход на аккаунт GitHub в консоле.
2. Создана новая ветка в репозитории «IVBO1017» с названием утилиты варианта «CALL» командой:

C:\Users\mrgep\Desktop\ivbo1017\ucheba>git checkout -b CALL

1. Добавлен нужный файл для пуша командой:  
   git add tzCALL.md
2. Закоммичены изменения в репозитории командой:  
   git commit -m “init”
3. Запушены все закоммиченные файлы в ветку «main» репозитория «IVBO1017» командой:  
   git push -u origin main

Вот и всё, наш документ добавлен в публичный репозиторий веб-сервиса GitHub, где каждый желающий, зарегистрированный на данном сервисе, может его скачать, просмотреть, и даже, предложить свои изменения.

# Заключение

В ходе работы был составлен документ ТЗ утилиты CALL командной строки CMD, который был отредактирован с использованием языка разметки markdown. Документ был добавлен в веб-сервис GitHub с помощью консольных команд git, которые подробно были описаны в данной работе.

Ниже приведена ссылка на созданный репозиторий:  
https://github.com/ggepardik/IVBO1017

# Использованная литература

1. Межгосударственный стандарт ГОСТ 19.104-78 ЕСПД «Единая система программной документации» // Государственный комитет СССР по стандартам от 1980 г.
2. https://git-scm.com/ – официальный сайт git. Дата обращения: 01.12.2020.
3. https://github.com/ – официальный сайт GitHub. Дата обращения: 02.12.2020.