

# Настройка desktop Ubuntu

Евгений Самарин

3 июня 2020 г.

v.1.0

# Оглавление

<b>1</b>	<b>Введение</b>	<b>3</b>
1.1	Предпосылки . . . . .	3
<b>2</b>	<b>Список программ</b>	<b>4</b>
2.1	Системные программы и инструменты . . . . .	4
2.1.1	Менеджер пакетов Synaptic . . . . .	4
2.1.2	apache2 . . . . .	4
2.1.3	PHP . . . . .	5
2.1.4	MySQL server . . . . .	5
2.1.5	Pulse audio . . . . .	6
2.2	Необходимый минимум . . . . .	7
2.2.1	git . . . . .	7
2.2.2	Postman . . . . .	7
2.2.3	Vim . . . . .	8
2.2.4	fish . . . . .	8
2.2.5	Ranger . . . . .	9
2.2.6	DBeaver Community . . . . .	9
2.2.7	JetBrains toolbar . . . . .	9
2.2.8	Менеджер виртуальных машин . . . . .	14
2.2.9	Enpass . . . . .	14
2.3	Дополнительные и полезные штуки . . . . .	15
2.3.1	Latex . . . . .	15
2.3.2	twinux . . . . .	15
2.3.3	calibre . . . . .	15
2.3.4	Rclone Browser . . . . .	15
2.3.5	Steam . . . . .	15
2.3.6	Discord . . . . .	16
2.3.7	PhotoGimp . . . . .	16

<b>3</b>	<b>Поднастройка окружения</b>	<b>17</b>
3.1	Замена местами кнопок CAPS / CTRL . . . . .	17
3.2	Перевод выделенного текста по shortcut внутри системы . .	18
3.3	Замена некоторых стандартных shortcut на более удобные .	19
3.3.1	Снимок выделенной области на экране в буфер обмена	19
<b>4</b>	<b>Красивости и прекрасности</b>	<b>20</b>
4.1	Дополнения и расширения для GNOME . . . . .	20
4.1.1	Dash to Dock . . . . .	20
4.1.2	Flickr wallpaper . . . . .	21
4.1.3	Unit . . . . .	21
4.1.4	User Themes . . . . .	21

# Глава 1

## Введение

### 1.1 Предпосылки

Я любитель новых сборок Ubuntu, как-то раз при очередном обновлении с версии 19.04 на 19.10 я столкнулся с проблемой переноса программ. Часть моих пакетов попросту была удалена из системы. И их пришлось устанавливать заново. А так как в процессе повторной установки можно что-то забыть и забытое как всегда напомнит о себе в самый неудобный момент, я буду вести список установленных мной программ. Возможно позже напишу скрипты для автоматической установки отдельных пакетов.

# Глава 2

## Список программ

### 2.1 Системные программы и инструменты



При переходе на новую сборку Ubuntu необходимо удостовериться, что все старые программы в наличии и работают должным образом. В этом могут помочь инструменты администрирования

#### 2.1.1 Менеджер пакетов Synaptic

Утилита для установки / удаления пакетов в графическом режиме, которые были установлены вручную из `deb` файлов. Поможет подчистить "хвосты" приложений, которые невозможно удалить через стандартный менеджер. Не забудь, после удаления перезагрузить компьютер. Также поможет, если ты забыл название установленного пакета. В добавок покажет всё, что установлено на твоём ПК.

#### 2.1.2 `apache2`

Хотя он предустановлен в системе, как показывает практика настройки слетают при переустановке системы. Для начала проверь статус служб `service --status-all`, если служба `apache2` не запущена - проверь настройки `/etc/apache2/apache2.conf`. В прошлый раз у меня слетел `phpmyadmin`, и из-за него не запускался `apache2`.

### 2.1.3 PHP

PHP необходим для развертывания локального web сервера, с целью тестирования приложений перед отправкой последних в PlayMarket. At that, I need different versions of a PHP. 5.6 for old projects and latest version for routine development respectively.

```
1 ## install php 5.6 vesion
2 #
3 sudo apt-get install software-properties-common
4 # add legacy ppa repository
5 sudo add-apt-repository ppa:ondrej/php
6 sudo apt-get update
7 sudo apt-get install php5.6
```

### 2.1.4 MySQL server

Необходим для развертывания БД для web сервера.

```
1 #install server
2 sudo apt-get update
3 sudo apt-get install mysql-server
4
5 sudo mysql_secure_installation
6 #after install you need enter root password: for example, i
   use an ovsyaniy password
7
8 #check service status
9 sudo service mysql status
10 #login to mysql server with your password
11 mysql -u root -p
12 #check what charset using by server
13 #character_set_database/server must provide utf8
14
15 show variables like 'char%';
16 #if thats not the case you need go to the /etc/mysql/conf.d/
17 #add file utf8_set.cnf
18 cd /etc/mysql/conf.d/
19 sudo touch utf8_set.cnf
20
21 #add in the file following srings
22
23 "[mysqld]" >> utf8_set.cnf
24 "character-set-server=utf8" >> utf8_set.cnf
25 "collation-server=utf8_general_ci" >> utf8_set.cnf
26
27 #then restart mysql server
28 sudo service mysql restart
```

```
29
30 #and check changes
31 mysql -u root -ppassword -e "show variables like 'char%'"
32
33 #well, you need change config mysql file also
34 #go to /etc/mysql/mysql.conf.d
35 #and add following string to mysqld.cnf file
36 "default_authentication_plugin=mysql_native_password" >>
    mysqld.cnf
```

MySQL server версии 8 и выше использует auth2.0 в качестве способа аутентификации, однако один из моих проектов использует аутентификацию по паролю. Соответственно приходится менять настройки сервера.

## 2.1.5 Pulse audio controll

Стандартный выбор контроллов для работы со звуком такой себе в ubuntu.

```
sudo apt install pavucontrol
```

## 2.2 Необходимый минимум



Включает прикладные программные продукты, позволяющие выполнять основную деятельность на ПК.

### 2.2.1 git

Удивительно, но git до сих пор не включен в базовый набор предустановленных программ для Ubuntu

### 2.2.2 Postman

По возможности хотелось бы использовать эту программу, однако на моей сборке 19.10 она не запускается, что сильно печалит. Однако существует аналог, к примеру сейчас я обкатываю insomnia Нашел на SO способ починить постман. Правда осталась висящая ссылка в списке приложений, но тем не менее.

```
1 sudo tar -xzf Postman-linux-x64-6.2.4.tar.gz
2 rm Postman-linux-x64-6.2.4.tar.gz
3 sudo mv Postman /opt/Postman
4 sudo ln -s /opt/Postman/Postman /usr/bin/postman
```

После этих манипуляций программа будет запускаться через команду терминала postman. Для удобства конечно следует создать ярлык для приложения. Ярлыки хранятся в директории `usr/share/applications`. Потребуется создать файл `Postman.desktop` со следующим содержимым:

```
1 #!/usr/bin/env xdg-open
2
3 [Desktop Entry]
4 Name=Postman
5 Name[en_IN]=Postman
6 GenericName=Postman REST Client
7 Comment=Programm for send REST API request
8 Exec=/opt/Postman/Postman %U
9 Icon=/opt/Postman/app/resources/app/assets/icon.png
10 Icon[en_IN]=/opt/Postman/app/resources/app/assets/icon.png
11 Terminal=false
12 Type=Application
13 Version=1.0
```



Не забудь сделать пере логин, иначе новый ярлык не подхватится.

### 2.2.3 Vim

Мне нравится Vim как консольный редактор. Он легкий, плагинистый и настраиваемый. Текущий конфигурационный файл:

```
1
2 " Configuration vimrc file.
3 "
4 " Date create 2019.10.03
5 "
6 " for Unix and OS/2: ~/.vimrc
7
8 call plug#begin('~/.vim/plugged')
9
10 Plug 'joshdick/onedark.vim'
11 Plug 'vim-airline/vim-airline'
12 Plug 'scrooloose/nerdtree', {'on': 'NERDTreeToggle'}
13 Plug 'udalov/kotlin-vim'
14 Plug 'Chiel92/vim-autoformat'
15 Plug 'ycm-core/YouCompleteMe'
16 Plug 'jiangmiao/auto-pairs'
17
18 call plug#end()
19
20 let g:onedark_hide_endofbuffer=1
21 colorscheme onedark
22
23 syntax on
24 set number "numbering strings
25 set mouse=a "mouse support
26 set expandtab "space instead tabulation
27 set tabstop=2 "two space tabulation
28 set hlsearch "syntax highlight
29 set incsearch "incremental search
30 set clipboard=unnamed
31
32 "map hotkeys to Nerd ctrl+N
33 map <C-n> : NERDTreeToggle<CR>
34 "map autoformat hotkeys
35 noremap <F3> :Autoformat<CR>
```

### 2.2.4 fish

Автодополнение для терминала (работает с bash и zsh). Статья про использование fish: <https://habr.com/ru/post/248881/>

## 2.2.5 Ranger

удобный консольный файловый менеджер, с поддержкой всех shortcut из vim. Хвалебные оды про ranger: <https://www.youtube.com/watch?v=0zkJG7iYTmc>

## 2.2.6 DBeaver Community

СУБД для просмотра баз данных, использую его для просмотра sql таблиц.

## 2.2.7 JetBrains toolbar

Нужен для обновления и установки моих рабочих инструментов: Android Studio, IntelliJ Idea. Для android studio требуется небольшая дополнительная настройка:

Установка плагинов: я выбрал для себя IdeaVim, ADB Wifi, MultiHighlights и DraculaTheme. В Vim используются сокращения, которые также используются в IDE, их необходимо разрешить:

- \* `ctrl+c` - в vim используется не для копирования текста, а для сброса режимов до режима "command"/"normal". Копирование текста следует производить командой "+y (в "визуальном" режиме для копирования в системный буфер) или командой y (для копирования внутри текущего редактора). Вставку текста следует производить или командой p или командой `ctrl+r` + (для системного буфера - **работает в режиме вставки i**). Однако в vim существует также комбинация `ctrl+[`, которая по сути делает тоже самое, поэтому следует использовать её. А в ide вне текста всё равно работает комбинация `ctrl+c`, поэтому не имеет смысла использование комбинацию vim, оставим системный буфер в покое.

*следует установить это значение в IDE.*

- \* `ctrl+[` - как я писал выше, эта комбинация нам жизненно необходима, чтобы не тянуться бедным мизинчиком до клавиши `esc`, ide в свою очередь использует это сочетание, чтобы прыгнуть в начало блока кода, тем более мы ничего не теряем, об этом ниже.

*следует установить это значение в VIM.*

- \* `ctrl+]` - раз уж мы переопределили предыдущее сочетание, не имеет смысла оставлять это за ide, как бонус в vim мы получим переход по ключевому слову под курсором. В данном контексте это будет

дублированием комбинации `ctrl+b`. В `ide` существует замена - сочетание клавиш `ctrl+shift+m`

*следует установить это значение в VIM.*

- \* `ctrl+v` - в `vim` используется не вызова режима `visual block`, позволяет выделять часть текста блоком. Такого же эффекта можно добиться с помощью `alt +` мышь, однако вся соль состоит в том чтобы отказаться от мышки.  
*следует установить это значение в VIM.*
- \* `ctrl+e` - в `vim` используется для прокрутки экрана вниз на одну строку, однако в `ide` есть более полезное назначение - открытие списка последних редактированных файлов.  
*следует установить это значение в IDE.*
- \* `ctrl+u` - в `vim` используется для прокрутки на 1/2 экрана вверх. В `ide` используется, для перехода у `super` классу.  
*следует установить это значение в IDE.*
- \* `ctrl+d` - в `vim` используется для прокрутки на 1/2 экрана вниз. В `ide` используется, для дублирования текущей строки.  
*следует установить это значение в VIM.*
- \* `ctrl+y` - в `vim` используется для прокрутки экрана вверх на одну линию, в `ide` для удаления строки.  
*следует установить это значение в VIM.*
- \* `ctrl+f` - в `vim` используется для постраничной прокрутки экрана вниз, в `ide`, для вызова поисковой строки по файлу.  
*Т.к. в vim существует собственный способ поиска, следует установить это значение в VIM.*
- \* `ctrl+b` - в `vim` используется для постраничной прокрутки экрана вверх, в `ide`, для перехода к реализации базового класса.  
*следует установить это значение в IDE.*
- \* `ctrl+g` - в `vim` используется чисто в информативных целях, для вывода названия текущего редактируемого файла, а также позиции на которой находится курсор. В `ide`, для перехода к указанной строке/столбцу. Я не вижу смысла ни в первом ни во втором (особенно для полноценной ОС).  
*следует установить это значение в VIM, только ради создания привычки.*

- \* `ctrl+o` - в vim используется для перехода к предыдущей позиции курсора в стеке переходов, в ide, для вызова списка override методов.  
*В IDE сочетание поважнее, следует установить это значение в IDE.*
- \* `ctrl+i` - в vim используется для перехода к следующей позиции курсора в стеке переходов, в ide, для вызова списка расширяемы методов.  
*В IDE сочетание поважнее, да и аналог есть (`ctrl+shift+стрелки`). Следует установить это значение в IDE.*
- \* `ctrl+t` - в vim используется для перехода вниз по стеку меток, в ide, для работы с git - производит pull request. Стеком меток в vim я не пользуюсь, а вот git мне точно нужен.  
*следует установить это значение в IDE.*
- \* `ctrl+k` - в vim используется в режиме вставки для ввода символов языков, отличных от латиницы, в которой vim работает по умолчанию. В ide используется, для работы с git - производит commit request.  
*следует установить это значение в IDE.*
- \* `ctrl+l` - в vim используется для перерисовки экрана, в ide, для перехода к следующему найденному фрагменту. Я не буду пользоваться ни первым ни вторым.  
*следует установить это значение в VIM.*
- \* `ctrl+r` - в vim одно из самых полезных сочетаний - работает как Redo, для восстановления отменённых изменений (необходимо для восстановления данных из системного буфера обмена в режиме "вставки"). В ide используется, для замены текста.  
*следует установить это значение в VIM.*
- \* `ctrl+w` - в vim используется для перемещения между split окнами, в IDE для расширения выделенной области. **Обе фишки полезные не знаю, что выбрать.**  
*ещё подумаю, пока следует установить это значение в VIM.*
- \* `ctrl+x` - не разобрался для чего это нужно, приведу статью из учебника.

*"Enter CTRL-X mode. This is a sub-mode where commands can be given to complete words or scroll the window. See*

*/i\_CTRL-X/ and /ins-completion/. {not in Vi}"*

Информация с сайта:

<http://vimdoc.sourceforge.net/htmldoc/insert.html>

Есть негласное правило, если не знаешь как работает фича - не используй её. *следует установить это значение в IDE.*

- \* **ctrl+a**, **ctrl+shift+2** - не разобрался, что делают эти hotkey.  
*следует установить это значение в IDE.*
- \* **ctrl+p** - в vim используется в режиме вставки как autocomplete, в ide используется, для отображения параметров методов. По понятным причинам ide лучше.  
*следует установить это значение в IDE.*
- \* **ctrl+s** - в vim используется для "заморозки" терминала при этом работа приложения не завершается, все изменения продолжают происходить (выйти из режима можно комбинацией **ctrl+q**), в ide используется для сохранения изменений. Эта фича работает только в консоли.  
*следует установить это значение в IDE.*
- \* **ctrl+n** - в vim используется для авто подстановки слов, в ide используется в различных ситуациях, по умолчанию открывает строку поиска, хотя также используется для создания нового файла, впрочем как и комбинация **alt+insert**. По этому проще убрать комбинацию с создания файла в keymap.  
*следует установить это значение в IDE, всё равно не получится вызвать vim tar для этой комбинации, будет работать только в консоли.*
- \* **ctrl+m** - в vim используется для перехода к первому не пробельному символу в следующей строке, в ide используется для быстрого создания функции из выражения, я этой штукой не пользуюсь.  
*следует установить это значение в VIM.*
- \* **ctrl+h** - в vim используется как альтернатива backspace, в ide для просмотра иерархии класса.  
*следует установить это значение в IDE.*
- \* **ctrl+2** - в vim используется в режиме вставки, для вставки последней введенной последовательности символов. Не нужная на мой взгляд фича.  
*следует установить это значение в IDE.*

- \* `ctrl+shift+6` - в `vim` используется для перехода в предыдущий файл, на позицию курсора. Есть альтернатива в IDE.  
*следует установить это значение в IDE.*

Для `MultiHighlights` нужно добавить `hotkeys`. Для закрепления выделения за словом использую сочетания клавиш `CTRL+Quote`, для снятия выделения со всех слов - `ALT+CTRL+Quote`.

Также следует добавить несколько `LiveTemplates`. Область видимости включаем `Kotlin`. Переменная `date` во всех случаях настроена след образом: `name=date, expression=date("yyyy.MM.dd"), default=, skip if defined=true`. Все элементы создаем в списке `AndriodComments`

Шаблоны для быстрого создания описаний:

Аббревиатура шаблона: `descm` Описание: `add method description`

```
1 /** $date$ v1 $description$ */
```

Аббревиатура шаблона: `desc` Описание: `add JavaDoc description`

```
1 /**
2  * $description$
3  *
4  * @author EvgenySamarin [GitHub](https://github.com/
5  *         EvgenySamarin)
6  * @since $date$ v1
7  */$END$
```

Шаблоны для быстрого создания `todo` напоминаний.

Аббревиатура шаблона: `todo` Описание: `adds // TODO`

```
1 /** TODO [$date$] $todo$ */
```

Далее добавляем шаблоны по аналогии с `todo` для остальных напоминаний.

`ADB Wifi` не требует настройки, зато требует определённых манипуляций для работы.

1. Подключаем девайс по `USB`
2. Открываем терминал в папке с `adb` по пути `SDK-PATH/sdk/platform-tools`
3. Набираем `adb tcpip 5555`
4. Набираем `adb connect DEVICE-IP`
5. Отключаем `USB`

Теперь можно собирать программы на устройство без необходимости подключения по кабелю.

### **2.2.8 Менеджер виртуальных машин**

Очень редко нужен, последний раз использовал, чтобы подобрать для себя VDS сервер. Однако, как показывает практика пакет нужен для работы эмулятора Android.

### **2.2.9 Enpass**

Для хранения паролей и другой конфиденциальной информации.

## 2.3 Дополнительные и полезные штуки



Программы и пакеты которые не критичны для работы, но их присутствие скрашивает серые будни линусоида.

### 2.3.1 Latex

*Thanks, YouTube channel Диджитализируй for show me this perfect PDF creating product.* В качестве редактора используем **TexStudio**. Кстати, эту статью я пишу как раз с использованием этой студии, эдакая работа с текстом для программистов. Тебе не нужно думать над подбором стилей текста и тому подобного. Ты просто пишешь текст а он выглядит потрясающе. Подобный документ выходит после компиляции исходного текста, да да - нужно компилировать набранный текст.

```
1 sudo apt-get install texlive texlive-full
2
3 # if you need work with graphics
4 sudo apt-get install imagemagick
5 sudo apt-get install texstudio
```

### 2.3.2 twinux

Twitter client - для написания постов и чтения новостей.

### 2.3.3 calibre

Программа для каталогизации книг. Эдакая личная книжная полка

### 2.3.4 Rclone Browser

Для синхронизации электронной книжки "по воздуху" через Dropbox, а также для связи с другими облаками: Google, Yandex, OneDrive

### 2.3.5 Steam

Для игрушек, что поделать - все мы не безгрешны.



### **2.3.6 Discord**

Для тех же игрушек, чтобы общаться с друзьями.

### **2.3.7 PhotoGimp**

Аналог Photoshop для Linux.

## Глава 3

# Поднастройка окружения

### 3.1 Замена местами кнопок CAPS / CTRL

Я пользователь VIM и мне часто приходится нажимать `ctrl+[`, стоит пожалеть свой мизинец и не тянуться в судорожных попытках к клавише `ctrl-` - это облегчит жизнь и написание кода. За основу взял статью Хабра: <https://habr.com/ru/post/222285/> о перенастройке клавиш в unix системах. Первое что нам потребуется - утилита `xkbcomp`. Выполняем чтение текущих настроек системы `setxkbmap -layout us,ru -print`. Записываем то что вывелось в файл `my` и размещаем его в директорию `~/.config/xkb/my`. Теперь открываем файл на редактирование, в нём будем прописывать конфигурации биндинга клавиш. Утилита `xkbcomp` из коробки уже имеет ряд стандартных конфигов, которые можно включить прописав `include "capslock(escape)"` в блок `xkb_symbols "my" { your_code_here }`, где `capslock` - название конфигурационного файла из списка `/usr/share/X11/xkb/symbols`, а `escape` - название конфигурационного параметра. В директории `symbols` все параметры любезно описаны разработчиками и готовы к употреблению. Итак, мой файл конфигурации выглядит след образом:

```
1 xkb_keymap {
2     xkb_keycodes { include "evdev+aliases(qwerty)" };
3     xkb_types     { include "complete" };
4     xkb_compat    { include "complete" };
5     xkb_geometry  { include "pc(pc105)" };
6
7     xkb_symbols "my" {
8         include "pc+us+ru:2+inet(evdev)"
9         include "capslock(escape)"
10
11         replace key <CAPS> { [ Control_L ] };
12         replace key <LCTL> { [ Caps_Lock ] };
```

```
13     };
14 };
```

Запустить конфигурацию можно командой:  
`xkbcomp $HOME/.config/xkb/my $DISPLAY`

**Важно:** если что-то пойдет не так, как задумывалось, сбросить настройки клавиш можно вызвав команду `setxkbmap` без параметров.

В общем на новых системах Gnome этот способ работает криво. будем довольствоваться заменой левого контрола на caps и наоборот. Необходимо скачать `gnome-tweak-tools` (дополнительные настройки gnome) и в разделе клавиатура выбрать "поменять местами ctrl и capslock". И конфиги писать не нужно.

## 3.2 Перевод выделенного текста по shortcut внутри системы

Будет работать при наличии интернета, так как используется запрос к google translate API. Итак, устанавливаем пакеты `libnotify-bin` (благодаря ему скрипт сможет отображать уведомления на рабочем столе), `wget` (для получения перевода от Google) и `xsel` (нужен для извлечения выделенного текста)

```
sudo apt-get install libnotify-bin wget xsel
```

Создаем файл с любым названием в домашнем каталоге, например "notitrans" внутри файла пишем скрипт:

```
1  #!/usr/bin/env bash
2
3  text="$(xsel -o)"
4
5  translate="$(wget -U "Mozilla/5.0" -qO - "http://translate.
    googleapis.com/translate_a/single?client=gtx&sl=auto&tl=ru
    &dt=t&q=$(echo $text | sed "s/[\"'<>]//g")" | sed "s
    /,,0],,..*/g" | awk -F"'"' '{print $2, $6}')"
6
7  echo -e "Original text:" "$text"'\n' > /tmp/notitrans
8
```

```

9 formatstring=$(echo $translate | sed 's/21791
    a8d07c2303bc1a3f9824e6ec4be//g' | sed 's/en en en//')
10 echo "Translation: " "$formatstring" >> /tmp/notitrans
11
12 zenity --text-info --title="Translation" --filename=/tmp/
    notitrans

```

В тексте скрипта, приведенном выше, меняем «tl=en» на обозначение языка, в котором хотелось бы получать результат перевода

Теперь делаем файл исполняемым:

```
chmod +x ~/notitrans
```

Перемещаем скрипт в каталог /usr/local/bin/:

```
sudo mv ~/notitrans /usr/local/bin/
```

Теперь биндим удобные клавиши для перевода: В GNOME (и Unity) это можно сделать перейдя в System Settings > Keyboard > Shortcuts > Custom Shortcuts и нажав «+» для добавления нового сочетания клавиш. В качестве shortcut я выбрал ctrl+shift+>. Готово, наслаждаемся встроенным в систему переводчиком.

### 3.3 Замена некоторых стандартных shortcut на более удобные

Замена shortcut не многим отличается от способа описанного в предыдущем параграфе:

В GNOME (и Unity) это можно сделать перейдя в System Settings > Keyboard > Shortcuts в поиске набираем нужную нам команду.

#### 3.3.1 Снимок выделенной области на экране в буфер обмена

Стандартное сочетание ctrl+shift+PrintScreen не удобно пальцы страдают, меняем на ctrl+shift+S

## Глава 4

# Красивости и прекрасности

### 4.1 Дополнения и расширения для GNOME

На текущий момент для Ubuntu 20.04 у меня есть ряд установленных расширений, которые влияют на внешний вид системы. Плата за это небольшая нагрузка на CPU и RAM так что ставить или нет это всегда спорный вопрос.

Для начала нам потребуется установить `gnome-tweak-tools`

```
sudo apt-get install gnome-tweak-tool
```

Далее нам потребуется установить `gnome-shell`

```
sudo apt-get install gnome-shell
```

Все расширения для `gnome tweak tools` берутся с сайта: <https://extensions.gnome.org/>. Потребуется установить плагин для браузера - для того, чтобы устанавливать расширения в 1 клик - удобно и быстро.

#### 4.1.1 Dash to Dock

Замена стандартной панели Ubuntu на OSX подобную. В `Gnome tweak` не забудь отключить стандартный Ubuntu Dock, чтобы не было конфликтов. Настраиваем следующим образом:

- Расположение на экране - снизу
- Интеллектуальное скрывание - да
- Ограничение дока - 90%

- Размер иконок - 64

### 4.1.2 Flickr wallpaper

Случайный фон с известного сайта, который будет динамически меняться

### 4.1.3 Unit

Экономит место на экране ноутбука, за счет title appbar Настраиваем следующим образом:

- Extend top bar left box - false
- Notification messages position - center
- Hide app menu application icon - false
- Top bar windows buttons position - first
- Остальное по умолчанию

### 4.1.4 User Themes

Самая мякотка - позволяет устанавливать тему рабочего стола - я выбрал Flat-Remix-Dark-FullPanel. Настраивается на вкладке внешний вид, в gnome tweak tools. Поменять можно на экране входа в систему.