

Задание №1

РАСПРЕДЕЛЕННАЯ СИСТЕМА УПРАВЛЕНИЯ ВЕРСИЯМИ (GIT)

Цель работы: освоение основных приемов работы в распределенной системе управления версиями.

Задачи работы

1. Получить навыки работы с распределенной системой управления версиями.
2. Изучить основные приемы настройки программного продукта.
3. Научиться использовать распределенные системы управления версиями для работы с *web*-ресурсами.

Перечень обеспечивающих средств

Задания лабораторной работы выполняются в операционной системе *Ubuntu*.

Общие теоретические сведения

Git (произносится «гит») – распределенная система управления версиями. Проект был создан Линусом Торвальдсом для управления разработкой ядра *Linux*, первая версия выпущена 7 апреля 2005 года. На сегодняшний день его поддерживает Джунио Хамано.

Система спроектирована как набор программ, специально разработанных с учетом 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 и *Darcs*, *BitKeeper*, *Mercurial*, *Bazaar* и *Monotone*, *Git* предоставляет каждому разработчику локальную копию всей истории разработки, изменения копируются из одного репозитория в другой.

Удаленный 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 надежным методом доступа. Метод доступа по *HTTP*, несмотря на ряд ограничений, очень популярен в контролируемых сетях, потому что позволяет использовать существующие конфигурации сетевых фильтров

Настройка Git для операционных систем семейства Linux.

Для установки *Git* понадобятся библиотеки, от которых он зависит: *curl*, *zlib*, *openssl*, *expat* и *libiconv*. Например, если в используемой системе менеджер пакетов - *yum* (*Fedora*), или *apt-get* (*Debian*, *Ubuntu*), можно воспользоваться следующими командами, чтобы разрешить все зависимости:

```
$ yum install curl-devel expat-devel gettext-devel \
openssl-devel zlib-devel
```

данная команда для тех, у кого менеджер пакетов – *yum* (*Fedora*)

```
$ apt-get install libcurl4-gnutls-dev libexpat1-dev
gettext \
libz-dev libssl-dev
```

данная команда для тех, у кого менеджер пакетов – *yum* (*Debian*, *Ubuntu*)

Установив все необходимые библиотеки, можно идти дальше и скачать последнюю версию с сайта *Git*: <http://git-scm.com/download>.

Теперь скомпилируем и установим:

```
$ tar -zxf git-1.7.2.2.tar.gz
$ cd git-1.7.2.2
$ make prefix=/usr/local all
$ sudo make prefix=/usr/local install
```

После этого можно скачать *Git* с помощью самого *Git*, чтобы получить обновления:

```
$ git clone git://git.kernel.org/pub/scm/git/git.git
```

Если же у нас дистрибутив, основанный на *Debian*, например, *Ubuntu*, попробуем *apt-get*:

```
$ apt-get install git
```

Настройка *Git* для Windows

1. Загрузка и установка

Загрузим установщик с официального сайта. Загрузка начнется автоматически (рис. 3.1).



Рис. 3.1. Загрузка дистрибутива *Git* с официального сайта

Перейдем в папку *Downloads* и запустим на исполнение загруженный файл и укажем путь до каталога, в который будет установлен *Git* (рис. 3.2).

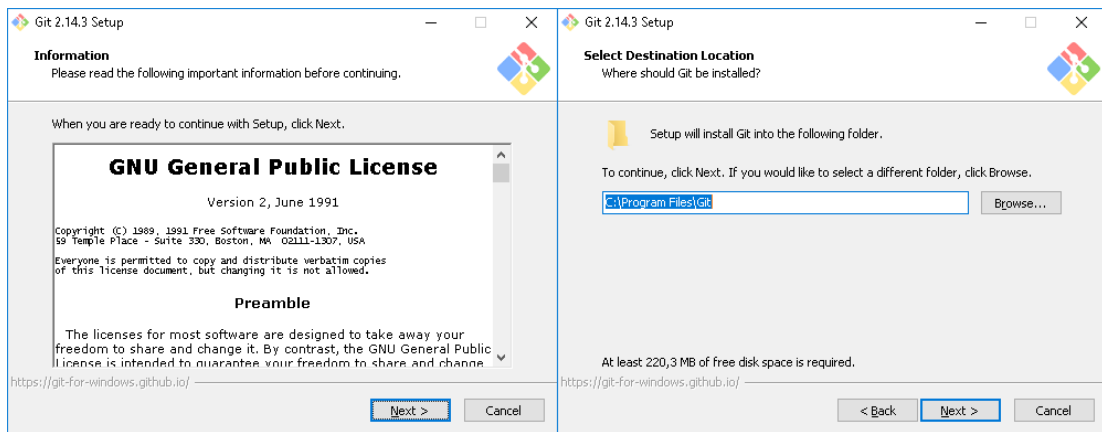


Рис. 3.2. Запуск дистрибутива

Чтобы на рабочем столе была иконка *Git*, на следующем шаге отметим галочкой *On the Desktop* (рис. 3.3).

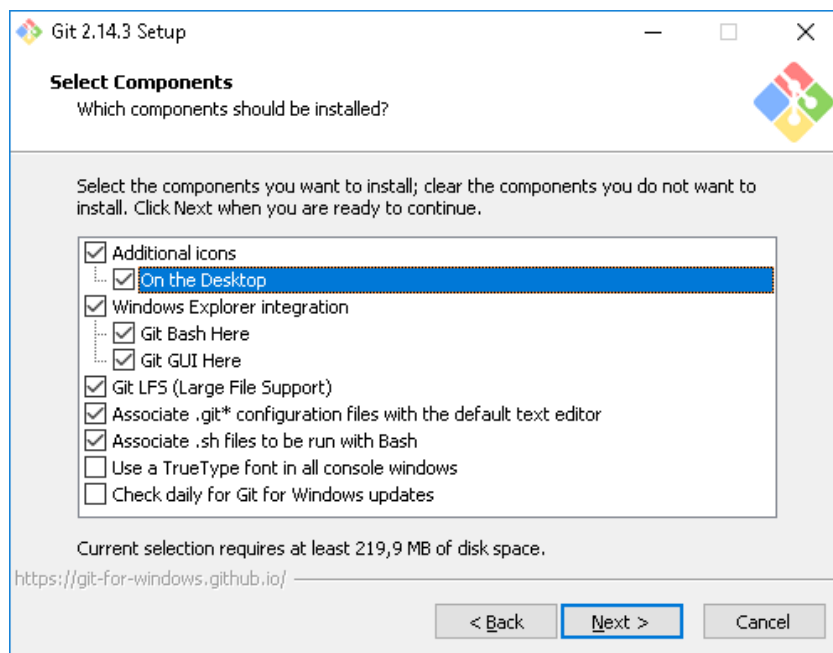


Рис. 3.3. *On the Desktop*

Введем имя директории, которая будет создана в *Start Menu*. При необходимости можно изменить путь с помощью кнопки *Browse* (рис. 3.4).

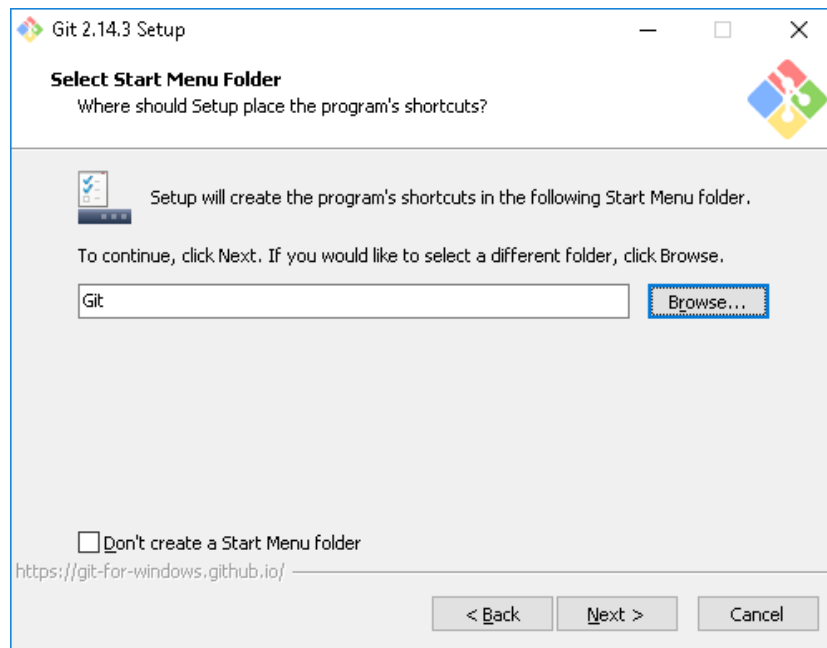


Рис. 3.4. Имя директории

Следующим шагом будет выбор одного из способов использования *git* из командной строки. На данный момент существует 3 способа использования:

1) *Use Git from Git Bash only* - использование только из командной строки *Bash*;

2) *Use Git from the Windows Command Prompt* - использование командной строки *Bash*, а также минимальный набор команд *Git* из консоли *Windows*;

3) *Use Git and optional Unix tools from the Windows Command Prompt* - использование *Git* и утилит *Unix* из командной строки *Windows*, в этом случае будут перезаписаны некоторые утилиты *Windows*, например, *find* и *sort* (рис. 3.5).

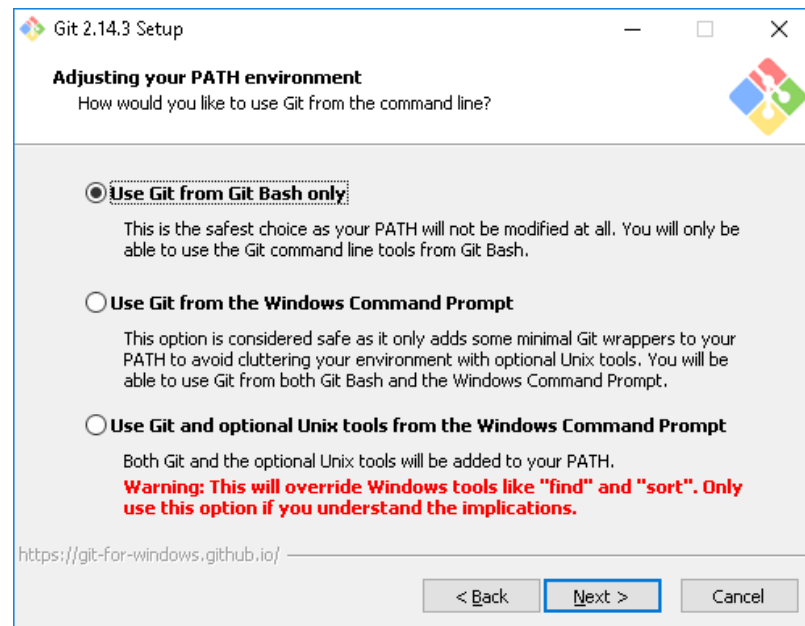


Рис. 3.5. Способы использования *git*

Выберем библиотеку, которая будет использована при подключении по протоколу *HTTPS*:

- *OpenSSL* - сертификаты сервера будут проверяться с использованием *Unix*-файла *ca-bundle.crt*;
- *Windows Secure Channel* - сертификаты сервера будут проверяться с использованием стандартной библиотеки *Windows* (рис. 3.6).

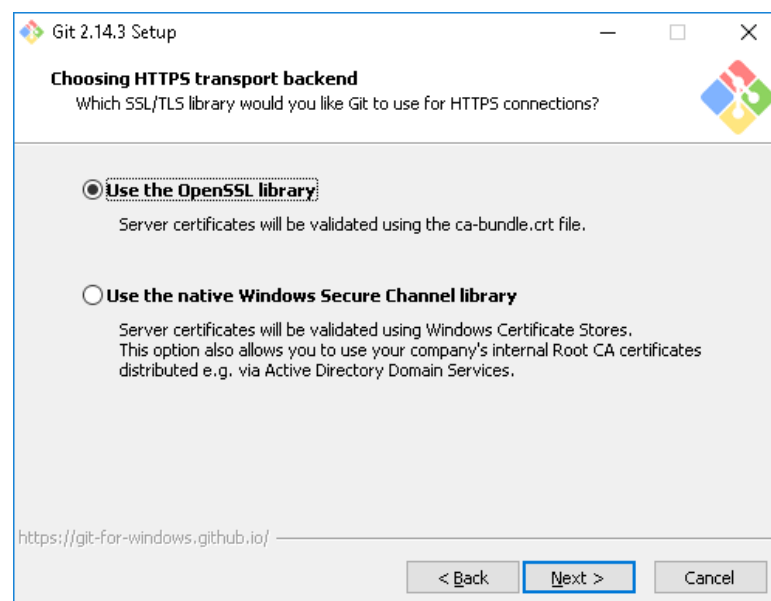


Рис. 3.6. Библиотеки

Убедимся, что мы выбрали способ обработки окончания строк «*Checkout Windows-style, commit Unix-style line endings*». Это значение гарантирует, что *Git* преобразует *LF* в *CRLF* при проверке текстовых файлов. При выполнении текстовых файлов *CRLF* также преобразуется в *LF*. Это мера совместимости для защиты новых строк в текстовых файлах, что позволяет легко работать с текстовыми файлами в *Windows* и на платформах *Unix* (рис. 3.7).

Примечание: *LF* и *CRLF* - управляющий символ для переноса строки в *Unix* и *Windows* соответственно.

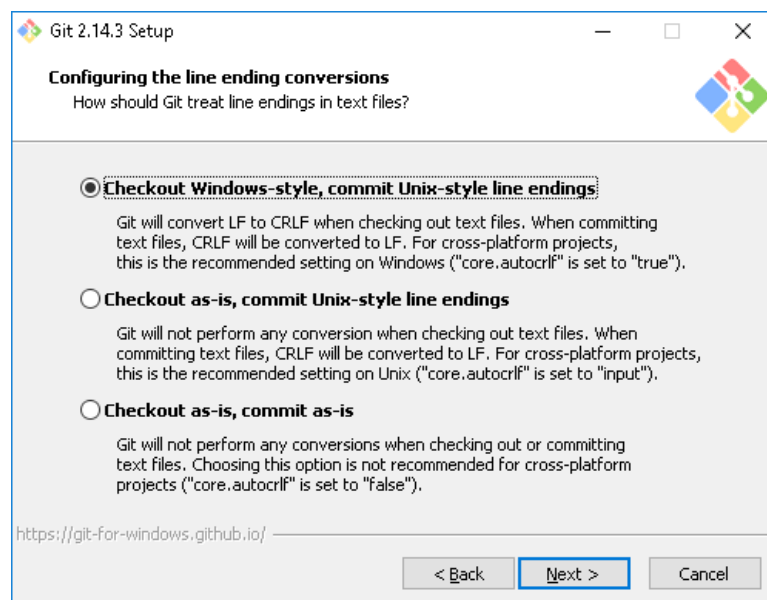


Рис. 3.7. Способы обработки окончания строк

Далее необходимо сконфигурировать используемый терминал (рис. 3.8):

- *MinTTY* - терминал *Unix*;
- *Windows* - стандартный терминал *Windows*.

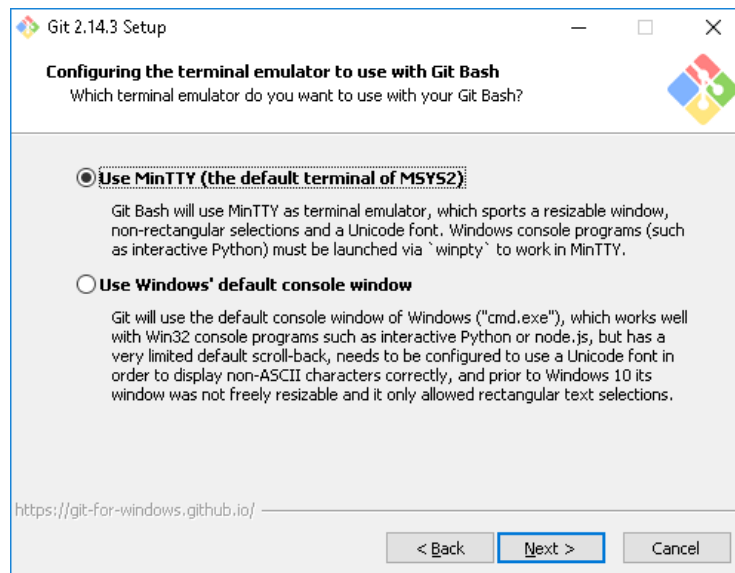


Рис. 3.8. Конфигурация используемого терминала

Отметим галочками нужные нам дополнительные функции:

- *File system caching* - кэширование файловой системы;
- *Git Credential Manager* - включить менеджер учетных данных;
- *Symbolic links* - разрешить символичные ссылки.

Нажмем кнопку *Install*, начнется процесс установки (рис. 3.9).

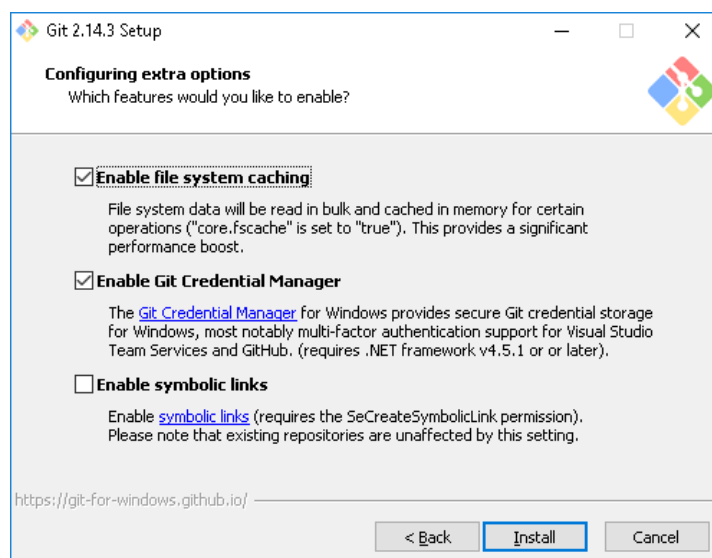


Рис. 3.9. Дополнительные функции

Подключение к удаленному репозиторию.

Откроем каталог с файлами, которые необходимо отслеживать в системе контроля версий и выложить на *GitHub*. В пустую часть каталога нажмем правой кнопкой мыши и выберем *Git Bash Here* (рис. 3.10).

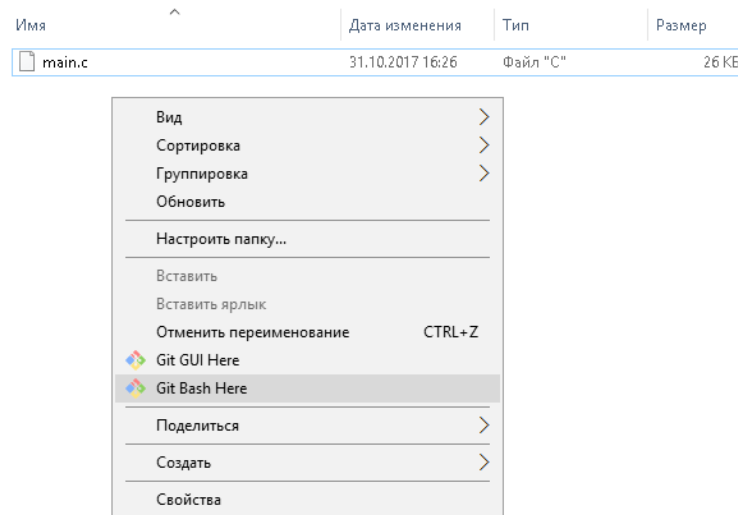


Рис. 3.10. Выборка файлов

Откроется приглашение командной строки в зависимости от настроек (рис. 3.11).

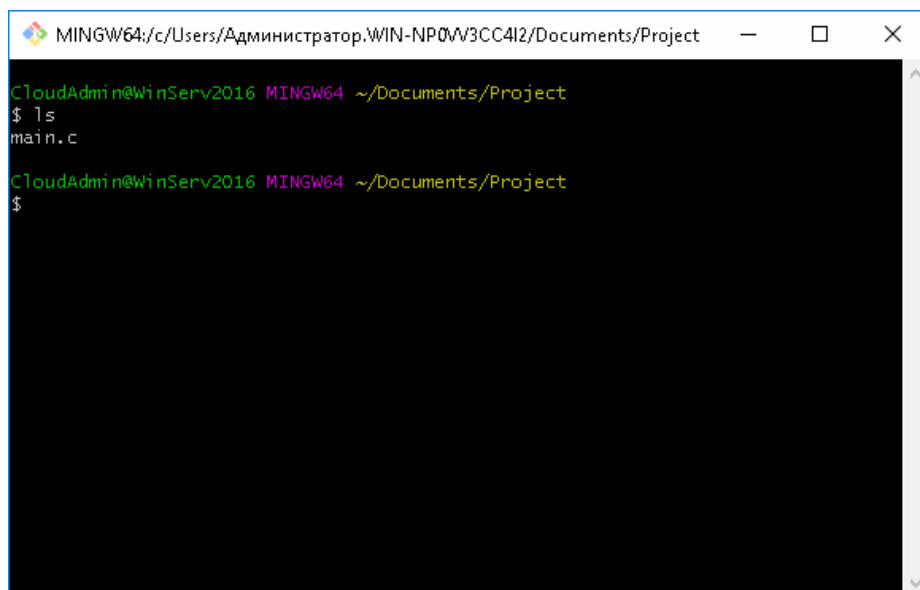


Рис. 3.11. Командная строка под *Windows*, с приглашением

Для настройки необходимо указать имя и электронную почту:

```
git config --global user.email "you@example.com"
```

```
git config --global user.name "Ваше имя"
```

Для того чтобы начать отслеживать содержимое папки в системе, выполним команды:

```
git init
```

```
git add
```

Выполним первый коммит:

```
git commit -m "Init"
```

Чтобы добавить изменения, например, на *github* выполним действие:

```
git remote add origin
```

```
https://github.com/пользователь/репозиторий.git
```

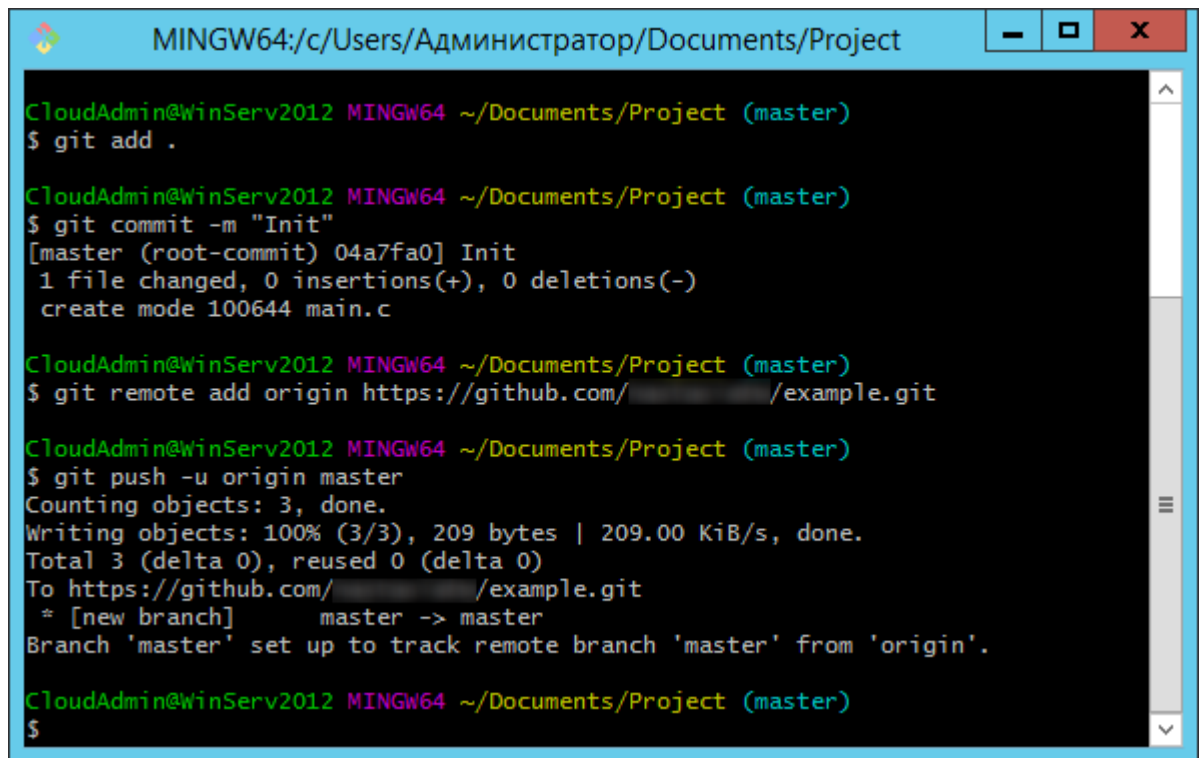
```
git push -u origin master
```

На [GitLab](#):

```
git remote add gitlab
```

```
https://server/namespace/project.git
```

```
git push -u gitlab master (рис. 3.12)
```



```
MINGW64:/c/Users/Администратор/Documents/Project
CloudAdmin@WinServ2012 MINGW64 ~/Documents/Project (master)
$ git add .

CloudAdmin@WinServ2012 MINGW64 ~/Documents/Project (master)
$ git commit -m "Init"
[master (root-commit) 04a7fa0] Init
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 main.c

CloudAdmin@WinServ2012 MINGW64 ~/Documents/Project (master)
$ git remote add origin https://github.com/_____/example.git

CloudAdmin@WinServ2012 MINGW64 ~/Documents/Project (master)
$ git push -u origin master
Counting objects: 3, done.
Writing objects: 100% (3/3), 209 bytes | 209.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/_____/example.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

CloudAdmin@WinServ2012 MINGW64 ~/Documents/Project (master)
$
```

Рис. 3.12. Ввод команд для подключения к репозиторию

Откроется окно входа (консольное или стандартное окно *Windows*). В качестве пользователя укажем логин на *GitHub*, репозиторий - название существующего репозитория (рис. 3.13).

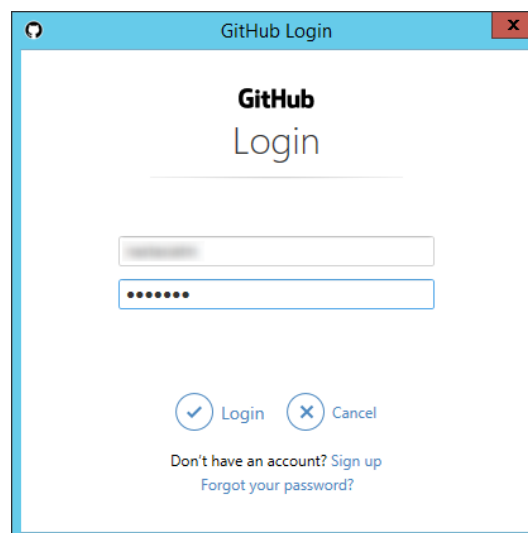


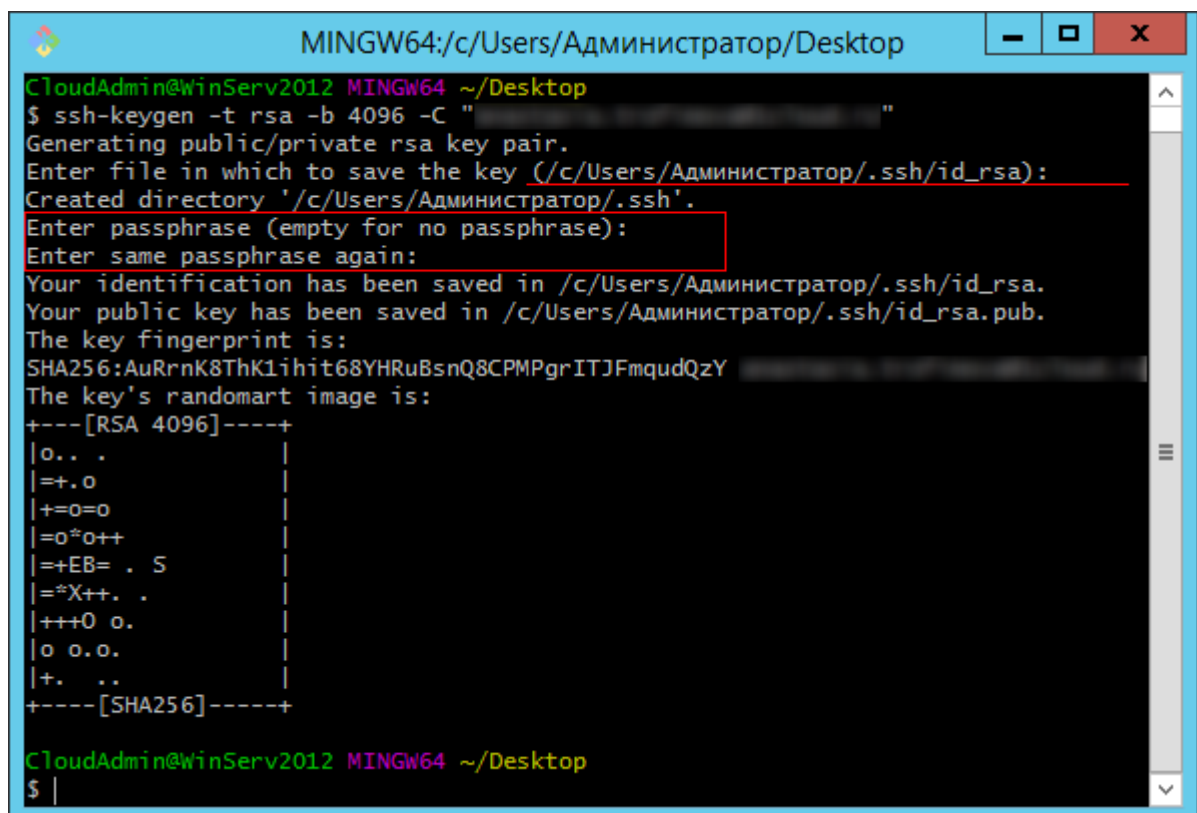
Рис. 3.13. Вход в *GitHub*

Аутентификация по SSH-ключам

Выполнить подключение к удаленному репозиторию можно по протоколу *SSH* с помощью приватного и публичного ключа. Чтобы сгенерировать пару, выполним команду в консоли *GitBash*, указав реальный почтовый адрес:

```
ssh-keygen -t rsa -b 4096 -C email@example.com
```

В процессе генерации будет предложено указать директорию и имя файла для ключа, примем значение по умолчанию, нажав *Enter*. Далее, при желании можно указать ключевую фразу в качестве дополнительной защиты, при ее отсутствии нажмем *Enter*. В результате будут сгенерированы ключи (рис. 3.14).



```
CloudAdmin@WinServ2012 MINGW64 ~/Desktop
$ ssh-keygen -t rsa -b 4096 -C "
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Администратор/.ssh/id_rsa):
Created directory '/c/Users/Администратор/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Администратор/.ssh/id_rsa.
Your public key has been saved in /c/Users/Администратор/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:AuRrnK8ThK1ihit68YHRuBsnQ8CPMPgrITJFmqudQzY
The key's randomart image is:
+----[RSA 4096]-----+
|o.. .|
|+=.o|
|+=o=o|
|=o*o++|
|+=EB= . S|
|=*X++ .|
|+++O o.|
|o o.o.|
|+ . ..|
+-----[SHA256]-----+
CloudAdmin@WinServ2012 MINGW64 ~/Desktop
$ |
```

Рис. 3.14. Генерация *ssh*-ключей

После генерации ключей необходимо добавить их к самому агенту. Для этого необходимо проделать следующие действия:

Запустим *ssh*-агент:

```
eval $(ssh-agent -s)
```

Мы увидим идентификатор процесса.

Добавим ключи к *ssh*-агенту:

```
ssh-add ~/.ssh/id_rsa
```

Теперь нужно добавить ключ к удаленному серверу *Git*. Для этого заходим на сервер *Git* и в настройках добавляем *SSH*-ключ.

Например, на сервере GitHub это можно сделать следующим образом: откроем *Settings - SSH and GPG keys*, нажмем *New SSH key*. В поле *Title* введем понятное название, в поле *Key* вставим публичный ключ (содержимое файла *~/.ssh/id_rsa.pub*). Нажмем *Add SSH key* (рис. 3.15). В результате появится новый ключ.

SSH keys New SSH key

There are no SSH keys associated with your account.

Title
Example

Key
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQACQDIdUZUvaQsouLvMUqpTNMiVJj5kMolqRvVLEYqJEkF41DmzRr9Gq
wQzGrC/9QLYypopgMqECdt9aG8ARtq+WEaGlgvUL9D/nm7UwlwAM3/3udcRF3MLnyZGlgkYa7k14i437BGM
m9AyTKzzWUujAACZHRGWrHPPKVdlX8NcVutA58RZjxFyxVoe6ILpdsjzaGwAT2t2/R+BjP224jx6gffc+6s6ZMm
WIQd/+Gs1DNNKjquZ49GuesXeWnJA9PdtgQSi4KXokQgbahSunk8sy39HS2V4GAGY1aHzhJlfG5jSiGXC+njHd
3DYJ91ZFRpGBw4uMUT2VA1lkuMnc8LgA5hhMeya+7xpdK951Oa6FdpQVihgTa3cSjdnXUx+eINamkzjKWp2
R5Hx3jllcxNCVBT4SWwMG9tgCaRkKa+WzMtw9R8r4ye5XIUQtmABP+9uudzWriXToOM+XEbB28EqbEjYVxd
mljqrZ4rFiNc+wBwwF/kN72nC5XGeKQXNm7X6gM22f0N6Rta3FORySSfmtXAbd3YXMcr28vXqfHvVhEbw00o
oqyjbQnFuZSDq6ie+0Vu549S9QACO//NhAjq6W7uS6Pe1LWO3+0NulxLvWSJdCZGVgOVgSXegh6UHIrqdb8d
CQ54cWScqMoVIE3ZW/XW4+eNNbBWTFoVTpHDMKe4nw-- email@example.com

Add SSH key

Рис. 3.15. Добавление *ssh*-ключ в *Git*

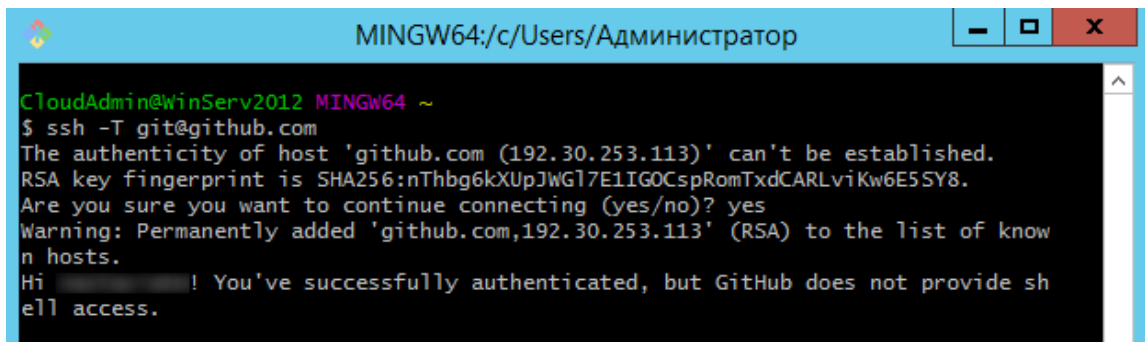
Для проверки настроек выполним команду:

```
ssh -T git@<домен или ip-адрес сервера git>
```

Например:

```
ssh -T git@github.com
```

Если все настроено верно, то мы увидите следующее сообщение (рис. 3.16).



```
CloudAdmin@WinServ2012 MINGW64 ~
$ ssh -T git@github.com
The authenticity of host 'github.com (192.30.253.113)' can't be established.
RSA key fingerprint is SHA256:nThbg6kXUpJWG17E1IGOCspRomTxdCARLviKw6E5SY8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'github.com,192.30.253.113' (RSA) to the list of known hosts.
Hi [redacted]! You've successfully authenticated, but GitHub does not provide shell access.
```

Рис. 3.16. Результат проверки

Задание

Установить и настроить *GIT* на подготовленную операционную систему.

Задание № 2

УСТАНОВКА СИСТЕМЫ УПРАВЛЕНИЯ СОДЕРЖИМЫМ (CMS)

Цель работы: освоение основных приемов работы с системой управления содержимым (CMS), на примере работы с *CMS Drupal*.

Задачи работы

1. Получить навыки работы с системой управления содержимым.
2. Изучить основные приемы настройки программного продукта.
3. Научиться использовать системы управления содержимым для работы с *web*-ресурсами.

Перечень обеспечивающих средств

Задания лабораторной работы выполняются в операционной системе *Ubuntu*.

Общие теоретические сведения

Система управления содержимым (англ. *Content management system*, *CMS*, система управления контентом) - информационная система или компьютерная программа, используемая для обеспечения и организации совместного процесса создания, редактирования и управления содержимым, иначе - контентом.

Основные функции *CMS*:

- предоставление инструментов для создания содержимого, организация совместной работы над содержимым;

- управление содержимым: хранение, контроль версий, соблюдение режима доступа, управление потоком документов;
- публикация содержимого;
- представление информации в виде, удобном для навигации, поиска.

В системе управления содержимым могут находиться самые различные данные: документы, фильмы, фотографии, номера телефонов, научные данные и так далее. Такая система часто используется для хранения, управления, пересмотра и публикации документации. Контроль версий является одной из важных возможностей, когда содержимое изменяется группой лиц.

Сейчас современное *web*-программирование сильно отличается от того, что было более 20 лет назад: от статичных файлов мы пришли к микросервисной архитектуре и платформенному вебу (рис. 4.1).

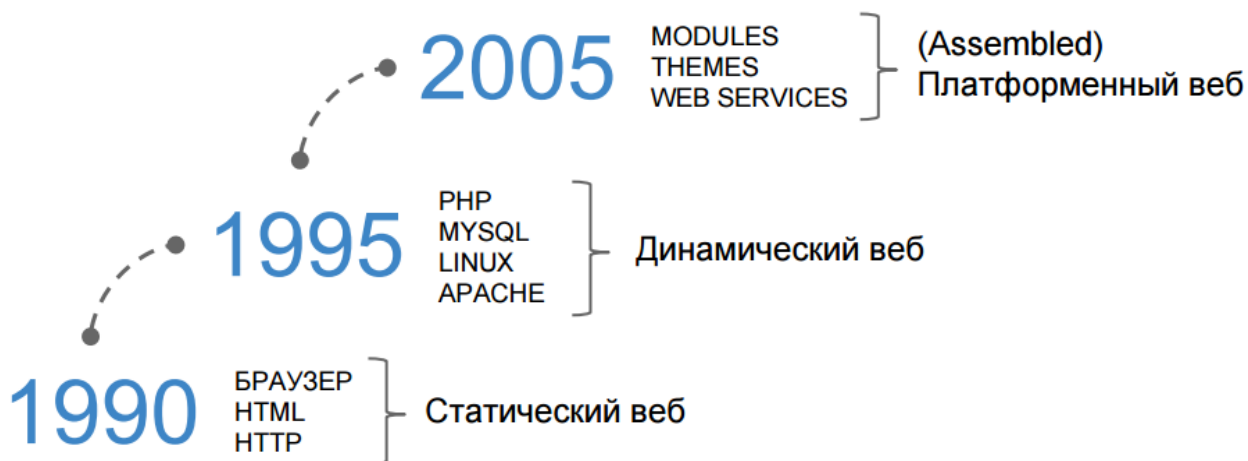


Рис. 4.1. Развитие *web* за 20 лет

За эти годы было изучено и разработано очень много инструментов, и теперь основная цель - научиться ими пользоваться, но не терять времени на то, чтобы переписывать уже имеющееся.

Drupal - одна из лучших систем в плане поддержки современных технологий, и в плане обучения, поэтому именно на этой платформе можно строить свою карьеру разработчика.

Основная концепция современного веб-программирования - микросервисная архитектура.

Архитектурный стиль микросервисов - это подход, при котором единое приложение строится как набор небольших сервисов, каждый из которых работает в собственном процессе и коммуницирует с остальными, используя легковесные механизмы, как правило *HTTP*. Эти сервисы построены вокруг бизнес-потребностей и развертываются независимо с использованием полностью автоматизированной среды. Существует абсолютный минимум централизованного управления этими сервисами. Сами по себе эти сервисы могут быть написаны на разных языках и использовать разные технологии хранения данных.

Отдельные сервисы в основном не решают бизнес-задачу целиком. Нужна система, которая будет связывать все отдельные элементы воедино, при этом реализовать свою собственную, уникальную для конкретного случая структуру взаимодействия. В качестве такой системы идеально подходит *Drupal*. На примере *Drupal 7* есть такие интеграции различных сервисов (рис. 4.2).



Рис. 4.2. Интеграция сервисов

Каждая отдельно взятая коммерческая система имеет свои плюсы и минусы, более развитые области, и менее развитые. *Drupal* - open source платформа, которая разрабатывается огромным количеством как отдельных

разработчиков, так и больших компаний, причем развитие идет сразу во всех направлениях, и это большое преимущество. *Drupal* как *CMS* можно сравнить с *Linux* в области программного обеспечения (рис. 4.3).

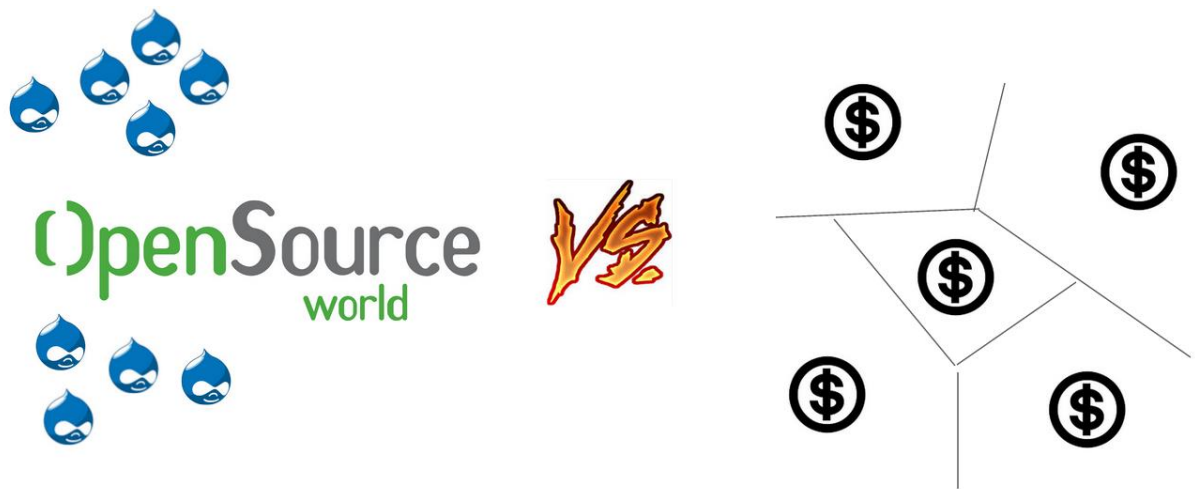


Рис 4.3. Программное обеспечение

Drupal состоит из следующих логических частей:

- ядро (Core);
- модули (Modules);
- темы (Theme);
- конфигурирование (Configure);
- контент (Content);
- процесс релиза (Deploy).

Работа с контентом в Drupal

Для того, чтобы понять, что такое контент, нужно понимать такие термины как данные, информация и суть.

Данные – что-то сырое, что пришло к нам в необработанном виде, и для нас никакой ценности пока не несет.

Информация - обработанные данные, которые без глобального контекста вероятно не помогут нам.

Суть - информация, которая подкреплена уже имеющимся контекстом, именно суть несет пользу.

Контент - это суть сайта, и это не только текст, но и все то, что используется для решения бизнес задачи.

Контент можно типизировать и структуризировать, это необходимо для более эффективной подачи (рис. 4.4).

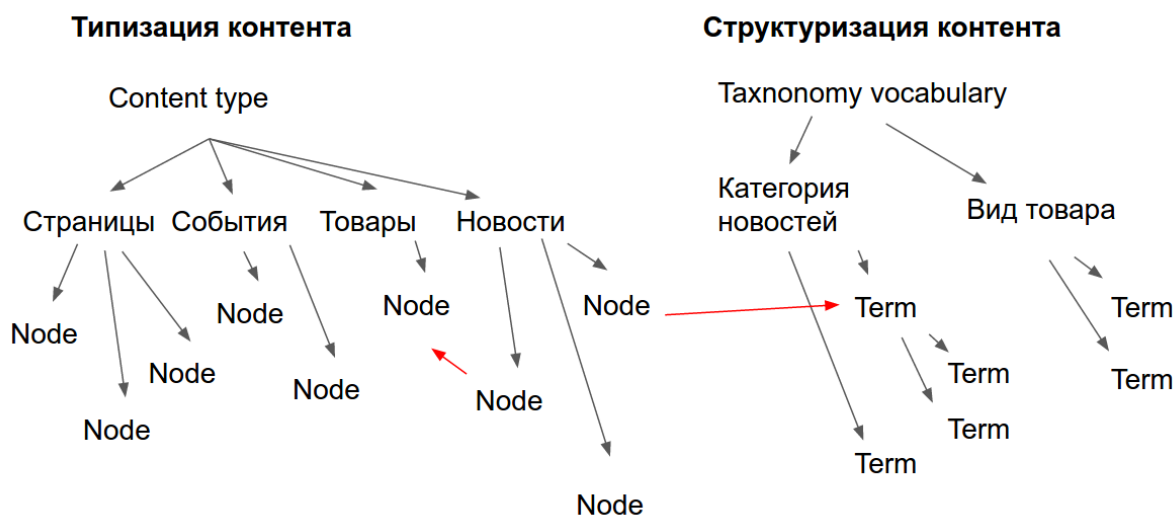


Рис. 4.4. Типизация и Структуризация контента

Типизация обусловлена необходимостью иметь для разнородной информации различный набор полей. Например, для контента типа товар нам нужна цена, но цену не имеет смысла задавать для контента типа новость.

В *Drupal* для типизации контента используется *Content type*.

<https://www.drupal.org/node/774728>

Элемент контента в *Drupal* называется *node*.

Для структуризации контента используется Таксономия.

<http://niklan.net/blog/15>

Таксономия тоже в свою очередь имеет свои типы, они называются словарями (*vocabulary*).

Элемент содержимого таксономии - *term*.

Структуризация осуществляется за счет привязки *node* к *taxonomy term*, как показано красной стрелкой справа.

Также есть возможность привязывать *node* между собой.

Для самопроверки рекомендуется сделать следующую структуру контента (рис. 4.5).

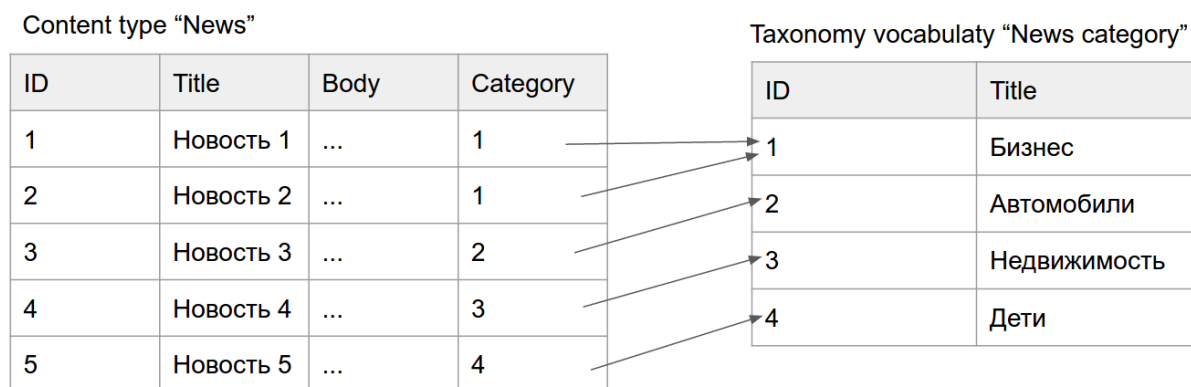


Рис. 4.5. Структура контента

На изображении схематично изображено как контент хранится в БД. Для организации такой структуры вам необходимо сделать поле типа *taxonomy term reference* у *content type* Новостей.

Еще один важный инструмент по работе с контентом – ревизионность (рис. 4.6).

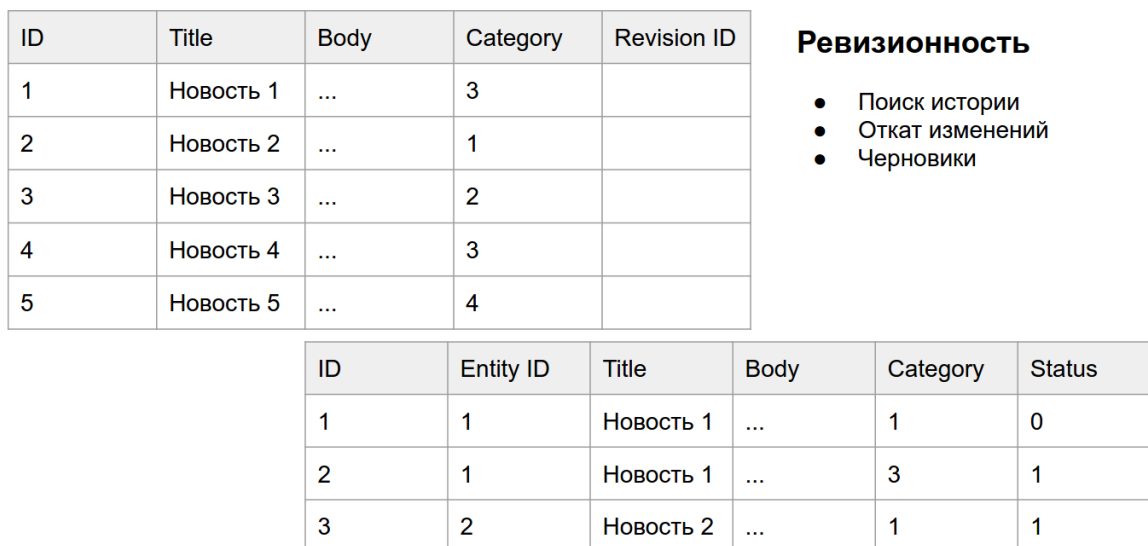


Рис. 4.6. Ревизионность

На рисунке видно, как *Drupal* хранит данные всех версий контента.

Ревизионность позволяет производить поиск по истории изменений контента, откатываться на более удачные версии, создавать черновики.

Ревизионность - мощный инструмент интернет маркетинга.

Установка Drupal

Drupal можно поставить как скачав архив с сайта <https://www.drupal.org/project/drupal>, разместив и распаковав его в корень сайта на сервере, так и воспользоваться готовыми командами специальной утилиты для консоли *Linux* - *drush*.

Базовые команды *drush*

drush dl - установка *Drupal 7 stable version*.

drush dl [имя модуля] - установка модуля
[https://www.drupal.org/project/\[имя модуля\]](https://www.drupal.org/project/[имя модуля]).

drush en [имя модуля] - включение модуля [имя модуля].

drush dis [имя модуля] - выключение модуля [имя модуля].

drush cc [имя кеша или *all* для всего] - чистка кеша.

drush sql-cli - переход в режим работы с БД

drush eval “код” - запуск *php*-кода из консоли.

Контрибные модули: */sites/all/modules/contrib/имя модуля*

Кастомные модули: */sites/all/modules/custom/имя модуля*

Кастомные темы: */sites/all/themes/имя темы*

Файлы: */sites/default/files*

Настройки для всех сайтов: */sites/default/settings.php*

Для того, чтобы было удобно работать с чистым *Drupal* рекомендуется установить и включить следующие модули:

– *admin_menu*

– *adminimal_admin_menu*

– *ckeditor*

– *ctools*

– *devel*

– *elysia_cron*

– *features*

- panels*
- panels_everywhere*
- pathauto*
- revisioning*
- rules*
- token*
- views*
- module_filter*

Также стоит отключить модуль *overlay*, чтобы не работать на сайте в административном разделе во всплывающих окнах, это не удобно и лишает вас полной кастомизации административного раздела сайта.

После того, как файлы *Drupal* лежат в корне сайта, заходим по *url* сайта и продолжаем процесс установки (рис. 4.7).

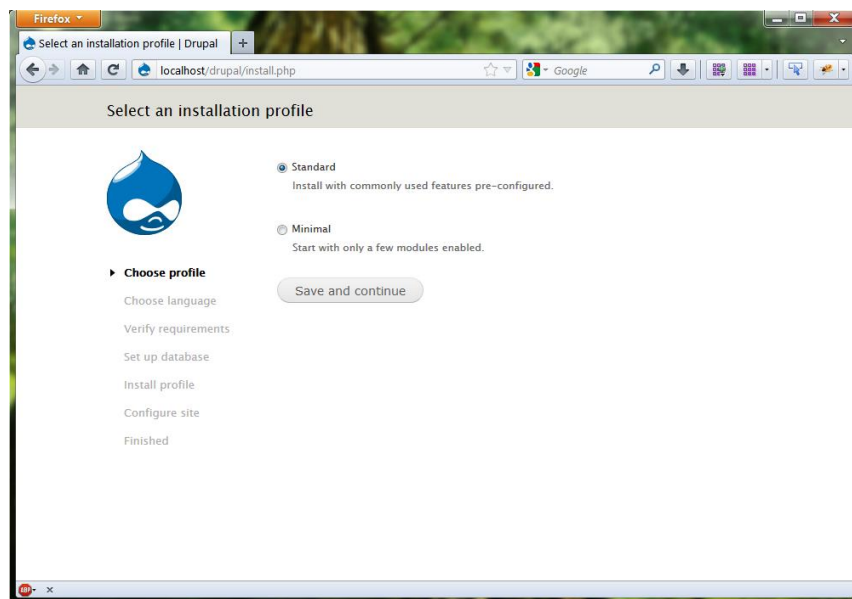


Рис. 4.7. Выбор профиля доступа

На этом этапе нужно прописать данные для вашей БД (рис. 4.8).

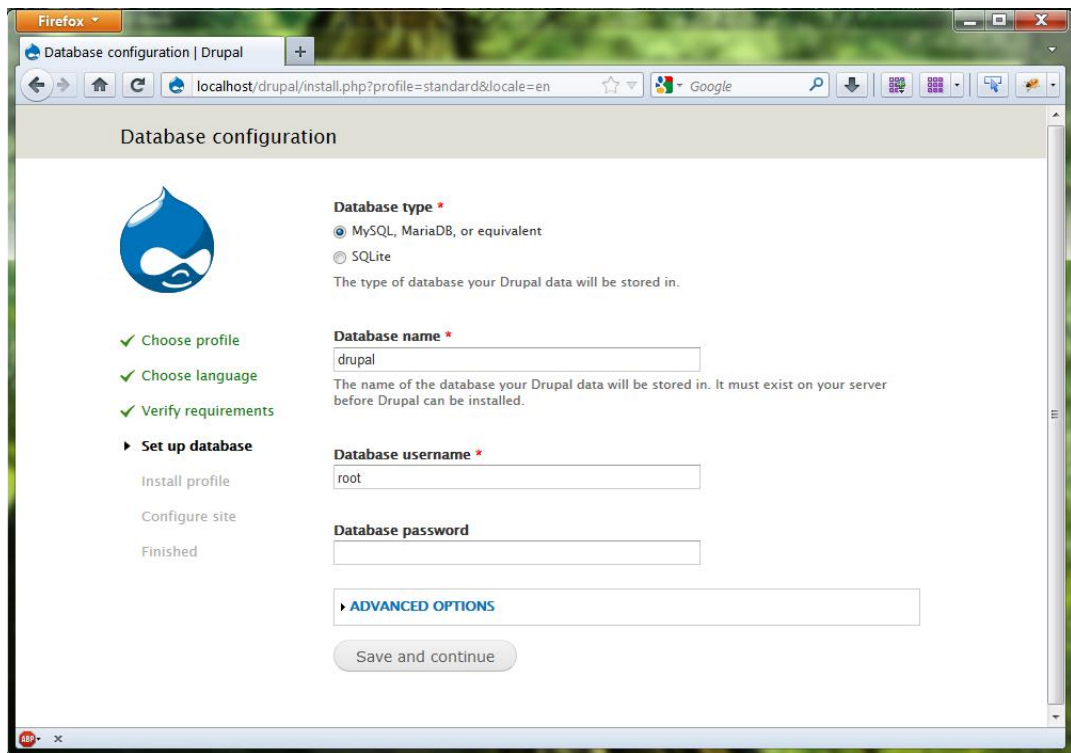


Рис. 4.8. Настройка конфигурации БД

Эти данные в дальнейшем будут записаны в *sites/default/settings.php* в виде:

```
$databases = array (  
  'default' =>  
    array (  
      'default' =>  
        array (  
          'database' => 'ИМЯ БД',  
          'username' => 'ЛОГИН',  
          'password' => 'ПАРОЛЬ',  
          'host' => 'localhost',  
          'port' => '',  
          'driver' => 'mysql',  
          'prefix' => '',  
        ),  
      ),  
    ),  
  ),  
);
```

) ;

Задание

1. Установить *Drupal*.

2. В директорию: *sites/all/modules/contrib*, если такой нет, необходимо создать, после чего скачать и распаковать в *contrib* следующие модули: *admin_menu*, *adminimal_admin_menu*, *ckeditor*, *ctools*, *devel*, *elysia_cron*, *features*, *panels*, *panels_everywhere*, *pathauto*, *revisioning*, *rules*, *token*, *views*, *module_filter*.

3. Открыть на *student.loc* страницу *Modules* и выключить *Overlay*, включить *module_filter*.

Задание № 3

МОДУЛЬ VIEWS

Цель работы: освоение основных приемов работы с основными модулями системой управления содержимым *Drupal*.

Задачи работы

1. Получить навыки работы с модульной системой.
2. Изучить основные приемы персонализации программного продукта.
3. Научиться использовать модуль views для работы с *web*-ресурсами.

Перечень обеспечивающих средств

Задания лабораторной работы выполняются в *CMS Drupal*.

Общие теоретические сведения

Основные понятия: представление, вид отображения, тема, теминг, меню, административная тема, дашборд.

Для чего на сайтах делают несколько вариантов отображения для одного и того же контента:

- конверсия;
- вовлечение;
- User-experience;
- программирование восприятия.

На примере сайта *ngs.ru* считаем количество вариантов отображения одной и той же сущности *Новость* с точки зрения программиста (рис. 5.1а, 5.1б)

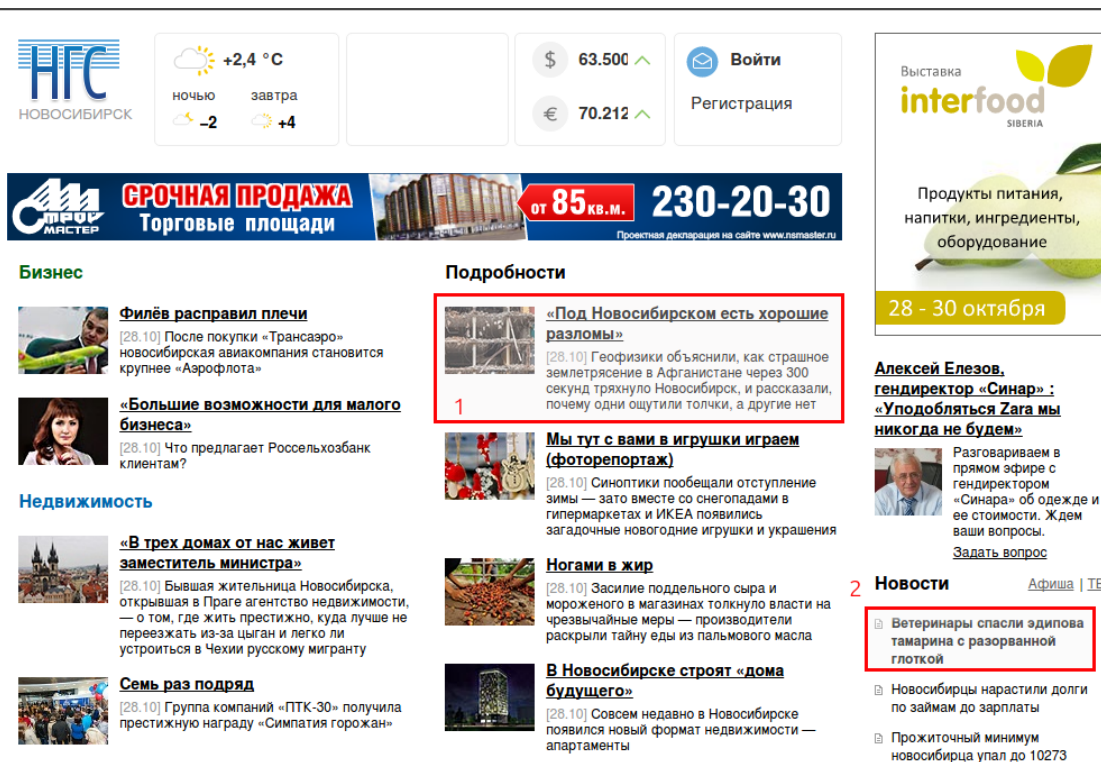


Рис. 5.1 а. Варианты отображения сущности «Новость»

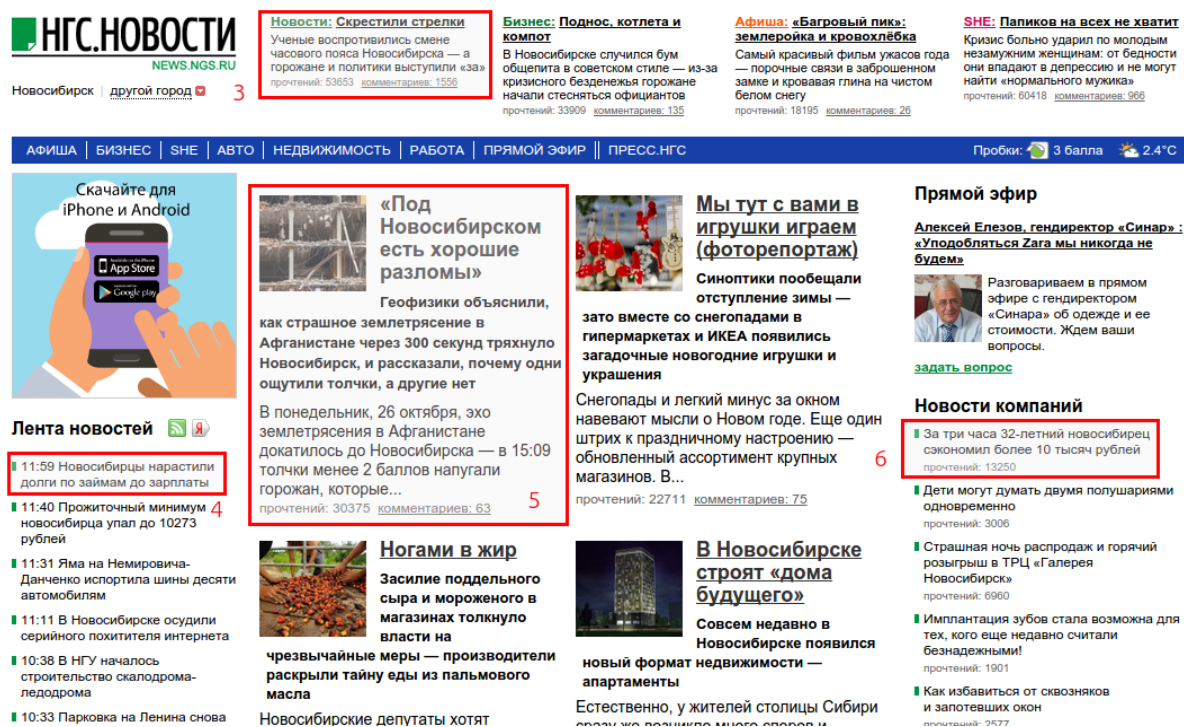


Рис. 5.1 б. Варианты отображения сущности «Новость»

+ страница самой новости и полное описание, итого 7.

В *Drupal* есть уже готовые инструменты: модуль *views* и *view modes* (рис. 5.2).

Представления (views) и варианты отображения (view mode)

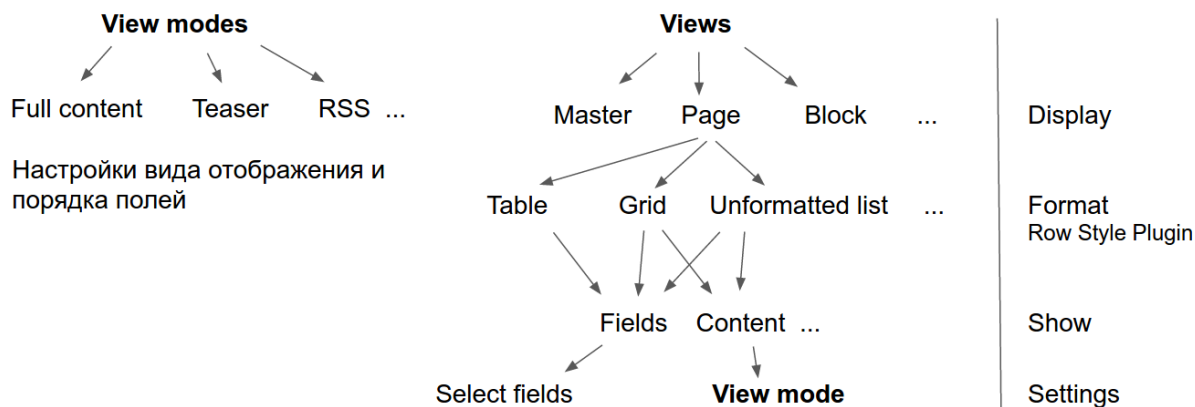


Рис. 5.2. Модули *View* и *View modes*

View mode - это характеристика *content type*. По сути *view mode* - это уже готовый *HTML node*. Задается для всего типа контента в целом (рис. 5.3).

Пример *view mode*, которые уже есть в *Drupal* по умолчанию: *Teaser* (краткое описание), *Full* (полное описание), *RSS*- тиль вывода для *RSS*-ленты.

Views - модуль для построения и вывода списка содержимого.

<https://www.drupal.org/project/views>

Display - где будет отображаться наш список (отображаться страницей по конкретному *url*, будет встроено куда-то как блок).

Format - как выводить эту информацию. Таблица - каждая запись выводится в одной строке таблицы, а столбцы - это характеристики контента. Сетка - это таблица, каждая ячейка которой является единицей контента (например, *node*). Неформатированный список - по сути Сетка с одним столбцом и множеством строк.

Show - что показывать (либо выбранный набор полей, либо уже подготовленный *view mode*).

Settings - настройки, которые зависят от выбранных ранее вариантов.

<div> Home » Administration » Structure </div> <div> <div>Views</div> <div>LIST</div> <div>SETTINGS</div> </div>					
<div> + Add new view + Add view from template + Import </div>					
VIEW NAME	DESCRIPTION	TAG	PATH	OPERATIONS	
Archive Displays: <i>Block, Page</i> In code Type: Content	Display a list of months that link to content for that month.	default	/archive	<div>Enable</div>	
Backlinks Displays: <i>Block, Page</i> In code Type: Content	Displays a list of nodes that link to the node, using the search backlinks table.	default	/node/%/backlinks	<div>Enable</div>	
Front page Displays: <i>Feed, Page</i> In code Type: Content	Emulates the default Drupal front page; you may set the default home page path to this view to make it your front page.	default	/frontpage, /rss.xml	<div>Enable</div>	
Glossary Display: <i>Page</i> In code Type: Content	A list of all content, by letter.	default	/glossary	<div>Enable</div>	
Recent comments Displays: <i>Block, Page</i> In code Type: Comment	Contains a block and a page to list recent comments; the block will automatically link to the page, which displays the comment body as well as a link to the node.	default	/comments/recent	<div>Enable</div>	

Рис. 5.3. Главная страница модуля *views*: */admin/structure/views*

Создание *views* (рис. 5.4, 5.5).

[Home](#) » [Administration](#) » [Structure](#) » [Views](#)

Add new view

LIST

SETTINGS

View name *

☐ Description

Show

Content

▼

of type

All

▼

tagged with

○

sorted by

Newest first

▼

☒ Create a page

Page title

Path

http://drupal_test.loc/

Display format

Unformatted list

▼

of

teasers

▼

with links (allow users to add comments, etc.)

▼

without comments

▼

Items to display

10

☒ Use a pager

☐ Create a menu link

☐ Include an RSS feed

☐ Create a block

Save & exit

Continue & edit

Cancel

Рис. 5.4. Создание *Views*

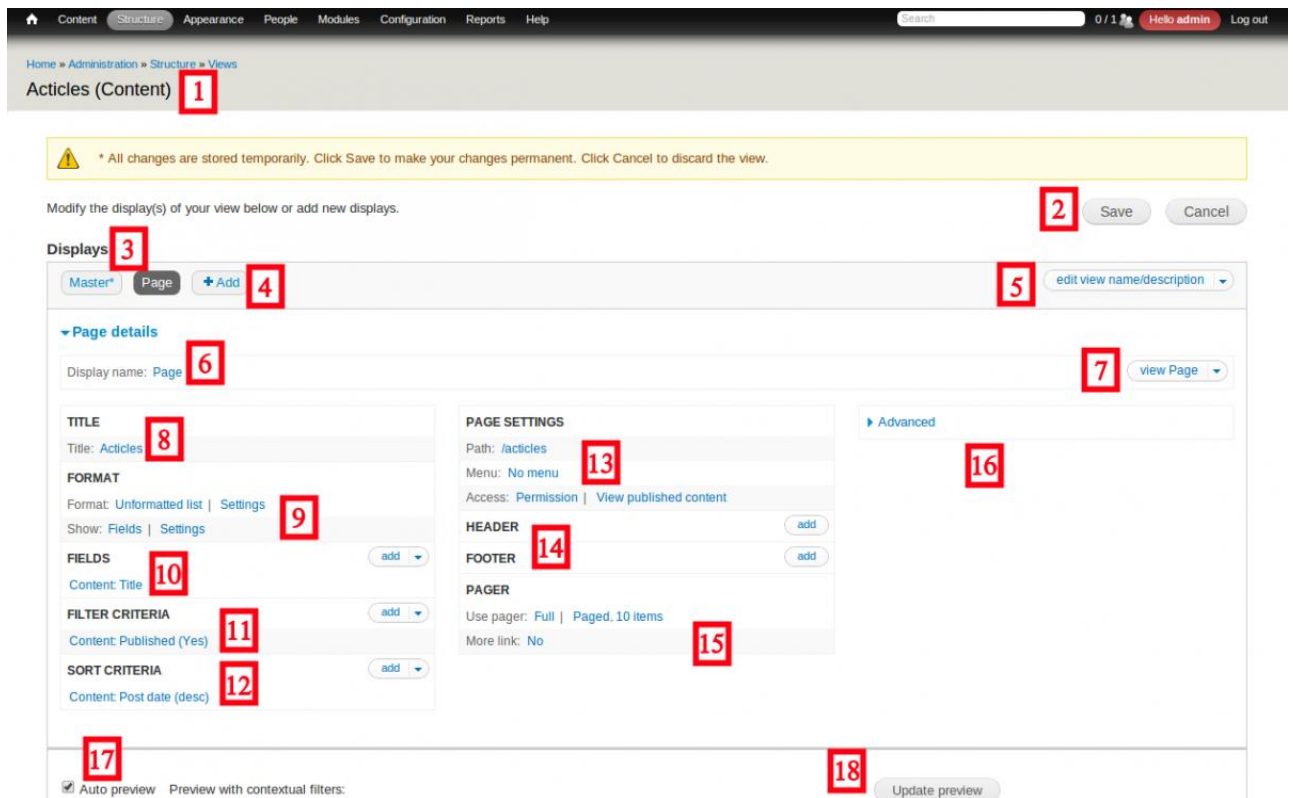


Рис. 5.5. Схема настройки *views*

1. Заголовок
2. Кнопки сохранения и отмены
3. Дисплеи
4. Создать дисплей
5. Опции (общие операции)
6. Название дисплея
7. Опции для дисплея
8. Заголовок страницы
9. Настройки формата
10. Поля
11. Критерии фильтрации
12. Критерии сортировки
13. Настройки страницы
14. Вставка Header и Footer
15. Настройка постраничной навигации

- 16. Расширенные настройки
- 17. Автоматический предпросмотр результатов
- 18. Обновить предпросмотр

Задание

Настроить отображение содержимого контента, созданного на прошлой лабораторной работе, используя модуль Views.

Для этого понадобится модуль Views.

Задание № 4

МОДУЛИ SLACK, TRELLO, ADDTHIS, DGIS_MAPS, SIMPLENEWS, LOGINZA

Цель работы: освоение основных приемов работы с основными модулями системой управления содержимым *Drupal*.

Задачи работы

1. Получить навыки работы с модулями CMS Drupal для использования сервисов slack, trello, addthis, dgis_maps, simplenews, loginza.
2. Изучить основные приемы настройки программного продукта.
3. Научиться использовать модули CMS Drupal для работы сервисов slack, trello, addthis, dgis_maps, simplenews, loginza.

Перечень обеспечивающих средств

Задания лабораторной работы выполняются в *CMS Drupal*.

Общие теоретические сведения

Slack - корпоративный мессенджер. Запущен в тестовом режиме в августе 2013 года, публичный релиз состоялся 12 февраля 2014 года. В первый день тестирования зарегистрировались 8 тысяч компаний. По данным компании на июнь 2015 Slack ежедневно используют 1,1 миллиона пользователей.

Бесплатный аккаунт slack дает возможности (применительно для системы умного дома):

- 1) хранение архива 10000 сообщений с вложениями;
- 2) возможность отправки текстовых сообщений, изображений, ссылок, кнопок и много других возможностей;
- 3) различные клиенты для всех платформ;
- 4) нет необходимости использовать VPN (спасибо РКН);
- 5) возможность отправлять сообщения в системный чат (на примере имеющего модуля телеграм) - в планах;
- 6) широчайшие возможности интеграции (хороший api).

Что реализовано модулем в настоящий момент:

- 1) подписка на системный чат в зависимости от требуемого приоритета (меню настройки);

- 2) отправка текстовых сообщений через webhook:

```
include_once(DIR_MODULES . 'slack/slack.class.php');  
$slack_module = new slack();  
$slack_module->sendMessageToAll ("test message to  
all");
```

- 2) отправка сообщений с вложением (картинка) через webhook. Картинка должна быть доступна по внешней ссылке (из локальной ссылки почему-то не работает).

```
include_once(DIR_MODULES . 'slack/slack.class.php');  
$slack_module = new slack();  
$slack_module->sendImageToAll  
("http://192.168.1.xx/img.jpg", "test image");
```

Для отправки снапшотов с камеры, пока приходится их выкладывать во временную папку по внешней ссылке.

Настройка:

Отправка сообщений через webhooks:

- a) Регистрируемся <https://api.slack.com/>

б) Создаем новое приложение *https://api.slack.com/apps/new AppName* - название приложения.

Short description - краткое описание приложения.

Describe what your app does on Slack - полное описание приложения.

Link to clear instructions on how to install your Slack app.

Link to support for your Slack app - две ссылки на страницы с описанием установки и использования данного приложения.

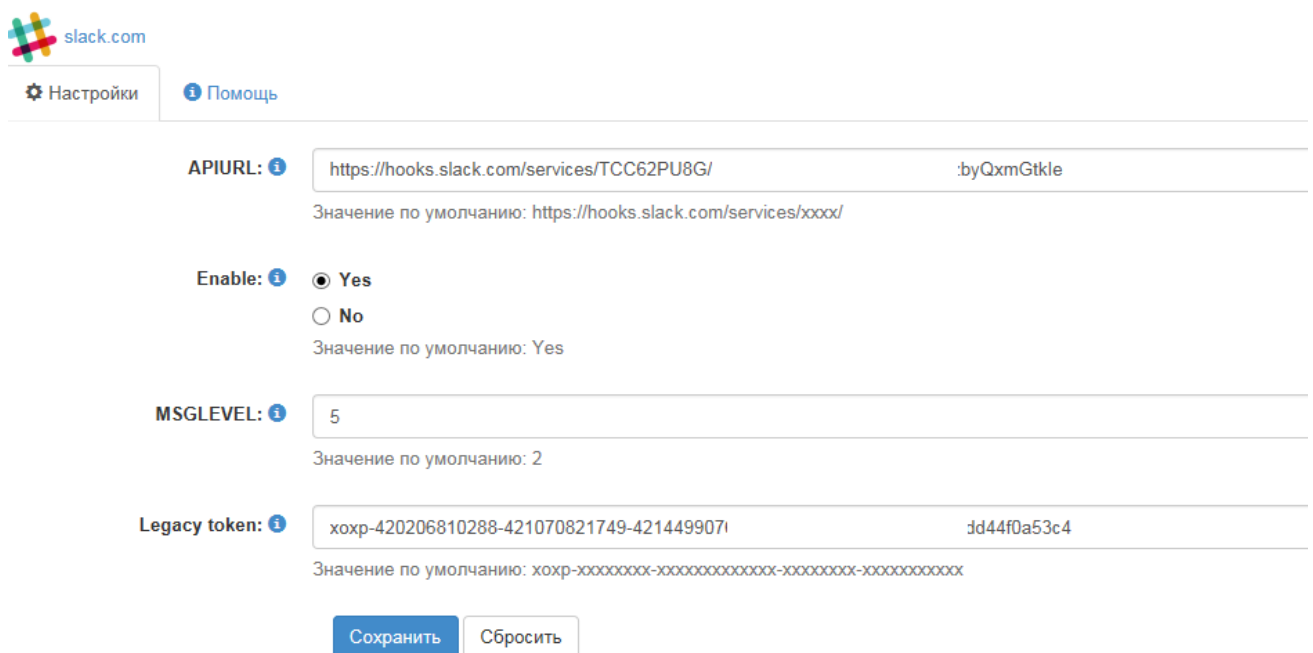
в) Получаем ссылку на *Webhook*. Для этого на вкладке *Basic Information* выбираем *Add features and functionality*, далее *Incoming Webhooks*, и включаем *Activate Incoming Webhooks*.

Ссылка *Webhook* имеет формат:
https://hooks.slack.com/services/T00000 ... XXXXXXXXXXXX

Эту ссылку добавляем на странице *Настройки* модуля.

г) настраиваем необходимый приоритет отправляемого системного чата.

Меню настройки (рис. 6.1, 6.2).



The screenshot shows the Slack settings interface for an incoming webhook. At the top, there is a header with the Slack logo and the text 'slack.com'. Below the header, there are two tabs: 'Настройки' (Settings) and 'Помощь' (Help). The main content area contains several configuration fields:

- APIURL:** A text input field containing 'https://hooks.slack.com/services/TCC62PU8G/:byQxmGtle'. Below it, a note says 'Значение по умолчанию: https://hooks.slack.com/services/xxxx/'.
- Enable:** A radio button selection with 'Yes' selected and 'No' as an option. Below it, a note says 'Значение по умолчанию: Yes'.
- MSGLEVEL:** A text input field containing '5'. Below it, a note says 'Значение по умолчанию: 2'.
- Legacy token:** A text input field containing 'хохр-420206810288-421070821749-4214499071jd44f0a53c4'. Below it, a note says 'Значение по умолчанию: хохр-xxxxxxxx-xxxxxxxxxxxx-xxxxxxxx-xxxxxxxx'.

At the bottom of the form, there are two buttons: 'Сохранить' (Save) and 'Сбросить' (Reset).

Рис. 6.1. Меню настройки *Slack*

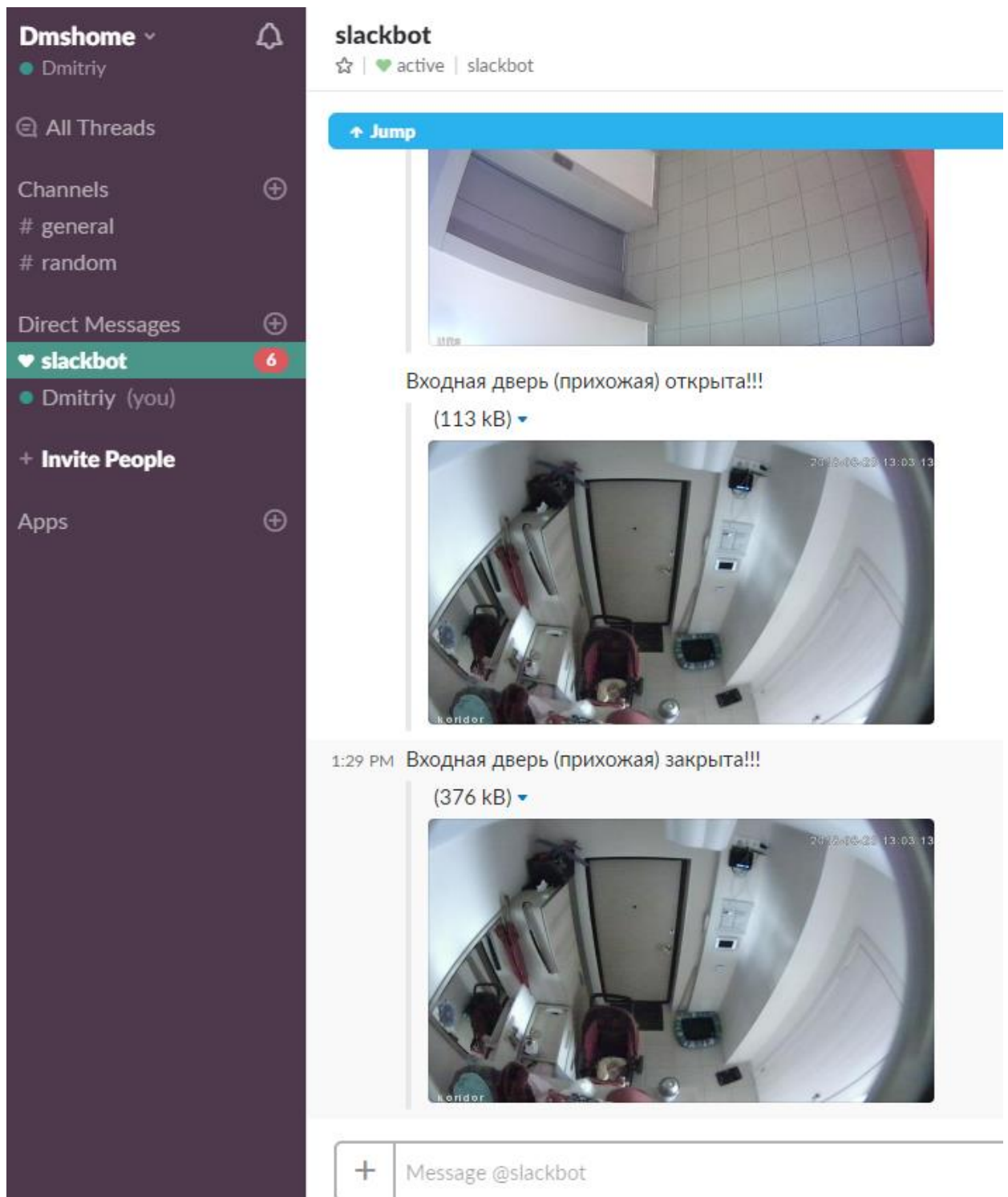


Рис. 6.2. Чат с программой

Trello - облачная программа для управления проектами небольших групп, разработанная *Fog Creek Software* (рис. 6.3).

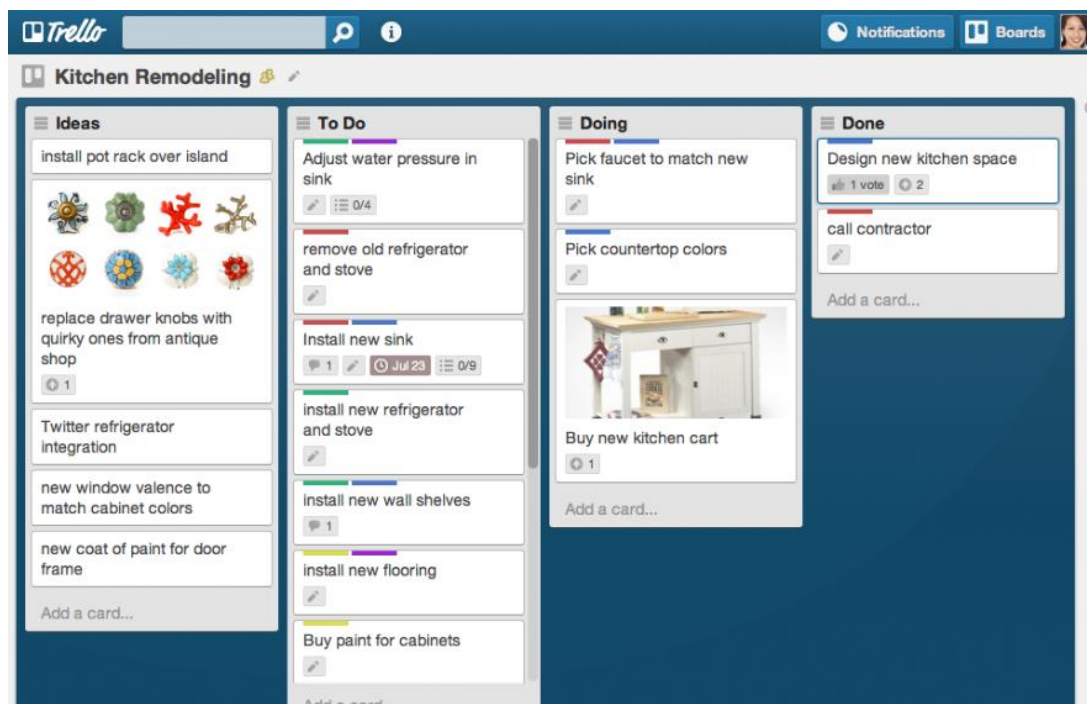


Рис. 6.3. Модуль управления проектами *Trello*

AddThis.com представляет собой набор инструментов, позволяющий пользователям делиться контентом с сайтами социальных сетей, используя *AddThis*.

AddThis запущен в сентябре 2006 года, является модулем № 1 для создания закладок и публикации в Интернете. *AddThis* распространяет контент по сети, облегчая вашим посетителям возможность делать закладки и делиться ими с другими людьми. Эта простой, но мощный модуль прост в установке и предоставляет ценную аналитику о закладках и активности пользователей.

Обратите внимание, что этот модуль поддерживает только часть функций, предоставляемых *AddThis*.

Модуль *2gis maps* добавляет свой тип поля для работы с картой 2ГИС, через который можно добавлять карты в объекты. Кроме того, он добавит блок, в котором показывается карта.

Используется *2GIS Maps API* и для каждой карты можно определить центр и масштаб, добавлять и удалять маркеры (рис. 6.4).

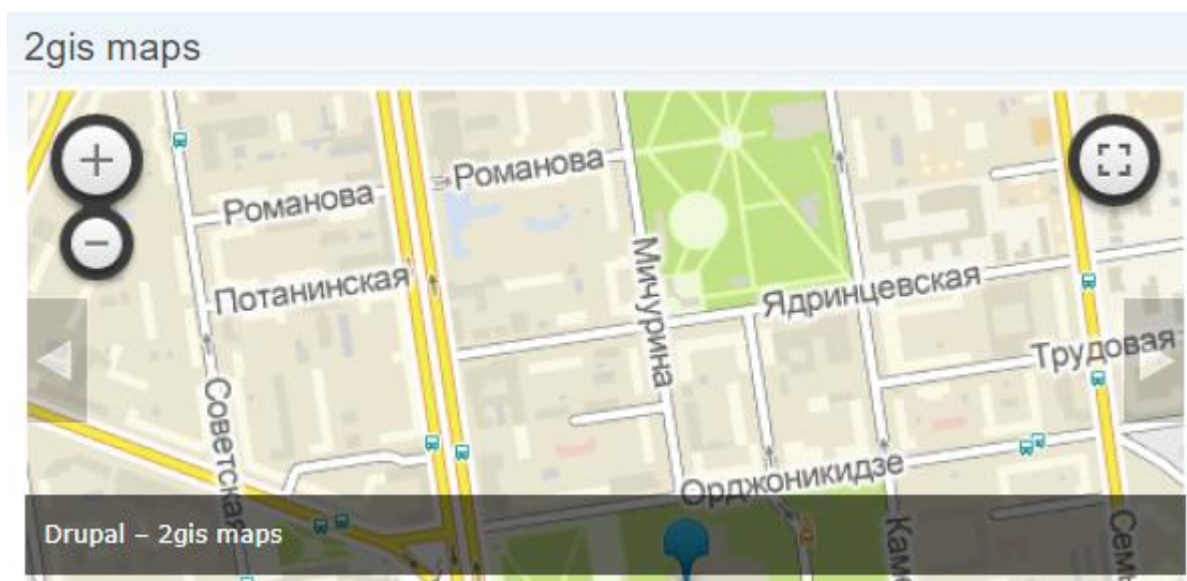


Рис. 6.4. Модуль *2gis maps*

Модуль *Simplenews* публикует и рассылает информационные бюллетени по спискам подписчиков.

Информационные письма можно рассылать как анонимным, так и аутентифицированным пользователям в вашей системе. Письмо в формате *HTML* можно отправить с помощью модуля *Mime Mail* (рис. 6.5).

Home » Administration » Content

SHOW ONLY NEWSLETTERS WHICH

Subscribed to:

UPDATE OPTIONS

<input type="checkbox"/>	TITLE	NEWSLETTER CATEGORY	CREATED	PUBLISHED	SENT	SUBSCRIBERS	OPERATIONS
<input type="checkbox"/>	French Newsletter	Drupal 7 newsletter	Fri, 01/06/2012 - 16:04		Not yet sent	35	edit
<input type="checkbox"/>	English Newsletter	Drupal 7 newsletter	Fri, 01/06/2012 - 16:03		Not yet sent	35	edit
<input type="checkbox"/>	Newsletter	Drupal 7 newsletter	Wed, 01/04/2012 - 12:04		Not yet sent	35	edit
<input type="checkbox"/>	Newsletter	Drupal 7 newsletter	Wed, 01/04/2012 - 12:04		Not yet sent	35	edit
<input type="checkbox"/>	Another newsletter	Drupal 7 newsletter	Thu, 12/15/2011 - 17:20		Not yet sent	35	edit
<input type="checkbox"/>	New product	Drupal 7 newsletter	Wed, 12/14/2011 - 15:09			35	edit
<input type="checkbox"/>	New test	Drupal 7 newsletter	Wed, 12/14/2011 - 14:52			35	edit
<input type="checkbox"/>	really unpublished	Test	Thu, 12/08/2011 - 14:21			0	edit

Рис. 6.5. Модуль информационной рассылки *simplenews*

Loginza – это система идентификации, обеспечивающая единый доступ к популярным *web*-сервисам, которая представляет собой интерактивный виджет на *JavaScript*.

Данная система предоставляет широкий список вариантов аутентификации через учетные записи распространенных *web*-порталов и сервисов, таких Яндекс, *Google* и многие другие (рис. 6.6).

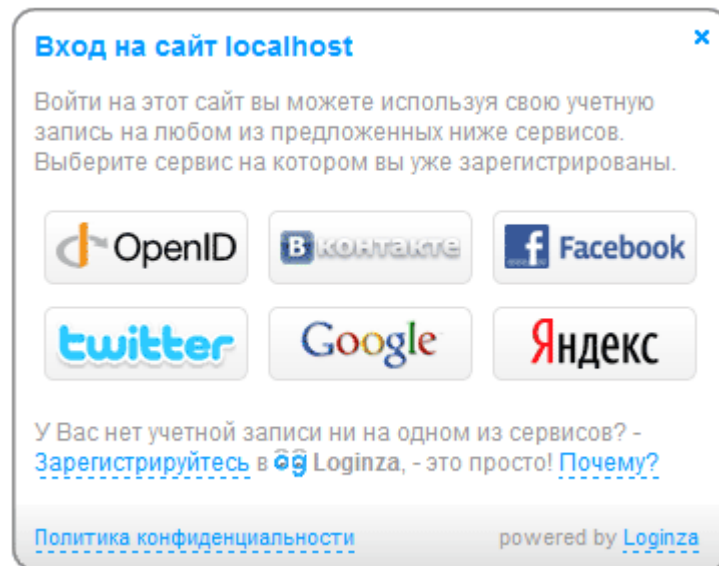


Рис. 6.6. Виджет *Loginza*

Задание

Настроить имя сайта: *Drupal First Test*. Слоган: *FirstD*.

Создать словарь таксономии *News Category*.

1. Создать контент тайп *News* с филдом *category* и привязкой к терму из задачи 1. Комментарии включены.
2. Настроить вывод категории новости на детальной странице новости.
3. Сделать красивые *URL* для новостей вида *news/title*.
4. Зарегистрироваться в *slack* и сделать пересылку новостей в приватный чат.
5. Зарегистрироваться на *trello*, сделать пересылку новости в столбец *New news*.

6. Добавить возможность делиться новостью в социальных сетях, используя модуль *addthis*.

7. Добавить возможность показывать на новости метки на карте *2gis*, используя модуль *dgis_maps*. Поле не обязательное вывод только на детальном описании (*full view mode*).

8. Сделать подписку *News*, зарегистрировать пользователя с *email drupaltest@yopmail.com*, подписать его и отправить рассылку.

9. Сделать авторизацию через соцсети, используя модуль *loginza*, авторизоваться.

Для этого понадобятся следующие модули: *slack*, *trello*, *addthis*, *dgis_maps*, *simplenews*, *loginza*.

Лабораторная работа № 5

SWAPI

Цель работы: Разработать систему интеграции своего сайта с сайтом SWAPI с помощью средств API.

Задачи работы

4. Развернуть личный сайт с системой CMS drupal на хостинге pantheon.io;
5. Научиться клонировать сайт с помощью средств Git;
6. Создать свой собственный сервис для обработки и вывода получаемой информации с сайта SWAPI;
7. Организовать работу сервиса на своем личном сайте;.

Перечень обеспечивающих средств

Задания лабораторной работы выполняются в операционной системе Ubuntu.

Общие теоретические сведения

1. Необходимо зарегистрироваться и выделить место для нашего сайта на бесплатном веб-хостинге <https://pantheon.io/>. Pantheon.io имеет уже встроенные возможности для создания сайта на базе Drupal 8.

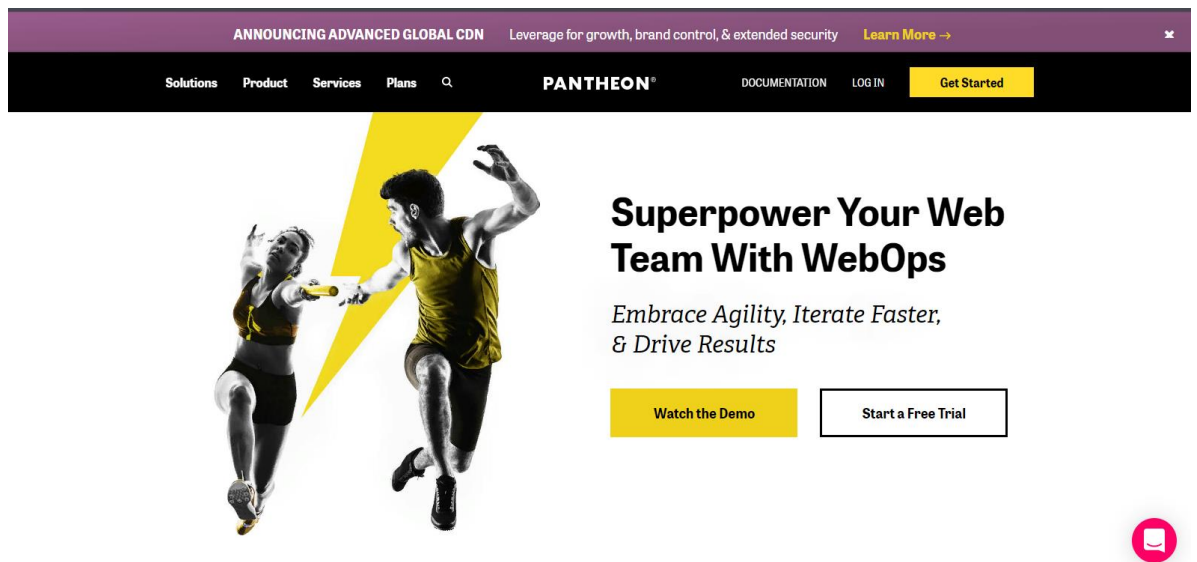


Рисунок 1 – Главная страница сайта pantheon.io

2. Далее мы попадаем в наш Dashboard, где нам предлагают создать наш сайт или сделать миграцию сайта со стороннего ресурса.

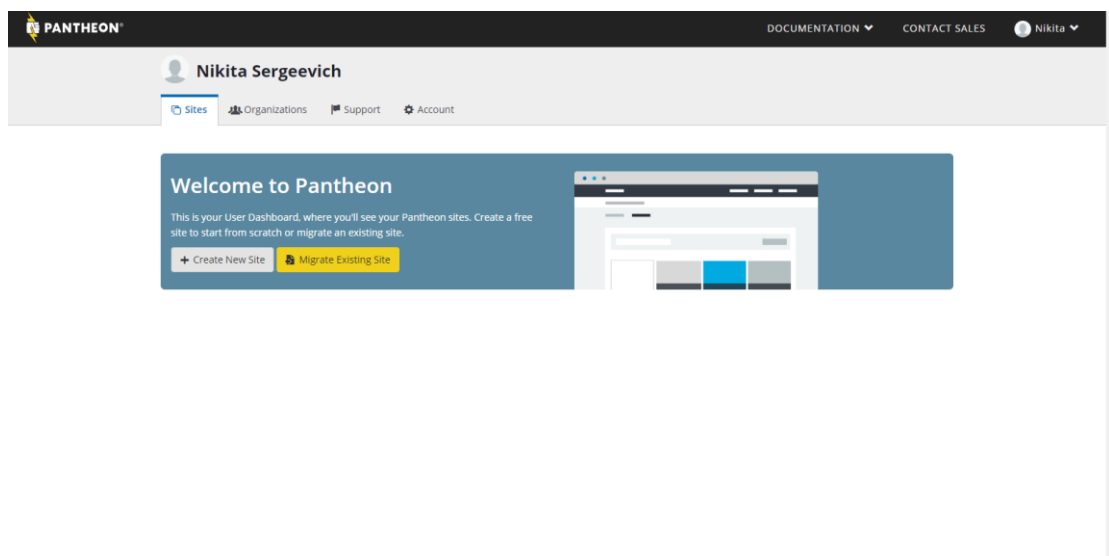


Рисунок 2 – Dashboard пользователя pantheon.io

3. Нажимаем Create new site, нас перекидывает на страницу для создания своего сайта. Вводим имя нашего сайта и выбираем регион.

Create Your Pantheon Site

Name Your Pantheon Site

Site names cannot be changed once created.

Your Pantheon Development Site URL

Choose a Region for the Site

- ☒ United States
- ☐ European Union
- ☐ Australia
- ☐ Canada

For Service Offerings where customers select a specific region as the data location, Pantheon utilizes a Google Cloud Platform data center located in that region to store all data collected by Subscribers from the End Users. All data transfers from that Google Cloud Platform data center are made in compliance with GDPR and other relevant data transfer regulations.

Continue

Рисунок 3 – Создание веб-сайта на хостинге pantheon.io

4. Далее нам предлагают выбрать информационную систему нашего сайта - CMS. Так как мы используем в нашей работе Drupal, то выбираем 8 версию и жмем на кнопку Deploy.

Choose Your CMS

Start with a vanilla CMS installation.



WordPress

Level up your WordPress workflow with version control and our Dev/Test/Live environments, so you can develop without fear of breaking your live site.

Deploy



Drupal 8

Use our professional workflow hooks for modern web development with the #1 enterprise open-source CMS.

Deploy



Drupal 7

Maintaining Drupal 7 sites can be a pain - our 1-click core updates and multiple development environments make it safe and easy.

Deploy

Рисунок 4 – Выбор CMS сайта

5. После установки CMS и создания сайта нас перекидывает в dashboard сайта. Перед нами представлены все инструменты для взаимодействия сайта.

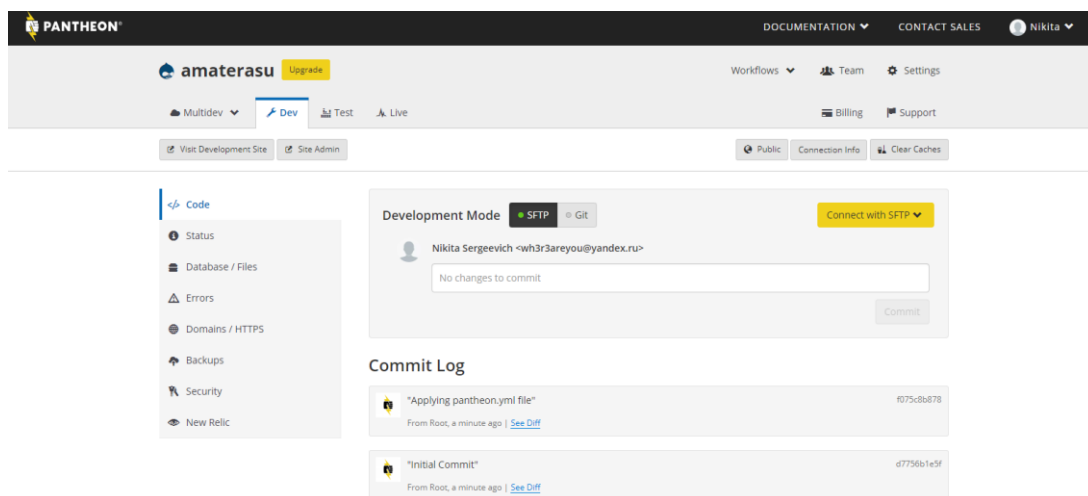


Рисунок 5 – Dashboard сайта

6. Далее необходимо установить drupal на нашем сайте. Нажимаем на кнопку Visit Development Site и происходит переход на наш сайт с персональной ссылкой.

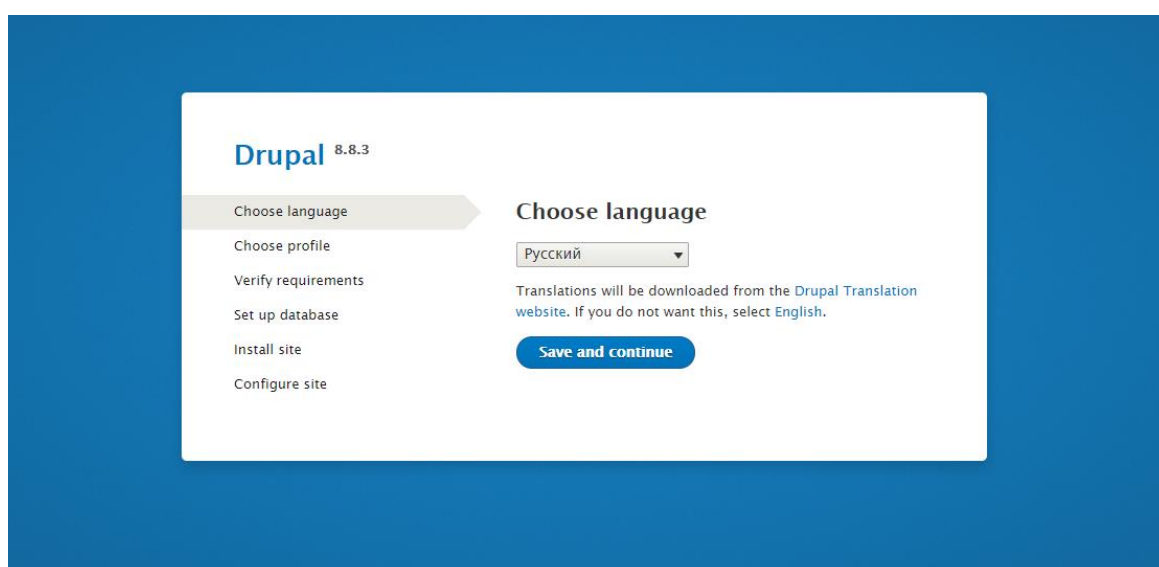


Рисунок 6 – Установка drupal 8

7. После установки нашего сайта заполняем нужные поля и нажимаем сохранить и продолжить. Теперь наш сайт установлен.

The screenshot shows the 'Настройка сайта' (Site Configuration) step of the Drupal 8.8.3 installation. On the left is a vertical menu with the following items: 'Выберите язык' (Select language), 'Выбор профиля' (Select profile), 'Проверка соответствия требованиям' (Check requirements), 'Установка базы данных' (Install database), 'Установка сайта' (Install site), 'Установка переводов' (Install translations), 'Настройка сайта' (Site configuration), and 'Завершение переводов' (Finish translations). The 'Настройка сайта' item is highlighted with a grey arrow pointing to the right.

The main content area is titled 'Настройка сайта' and contains two status messages in boxes. The first is a green box with a checkmark icon stating: 'Импортирован один файл перевода. 8456 переводов добавлено, 0 переводов обновлено и 0 переводов удалено.' (One translation file imported. 8456 translations added, 0 translations updated and 0 translations removed). The second is an orange box with a warning icon stating: '5 строк перевода было пропущено из-за недопустимого или повреждённого HTML. Подробности в журнале.' (5 lines of translation were skipped due to invalid or corrupted HTML. Details in the log).

Below these messages is the 'ИНФОРМАЦИЯ О САЙТЕ' (Site Information) section. It includes the following fields and labels:

- Название сайта *** (Site name *): A text field containing the value 'amaterasu'.
- Адрес электронной почты сайта *** (Site email address *): An empty text field.
- A descriptive paragraph: 'Автоматизированные электронные сообщения, например информация о регистрации, будут отправлены с этого адреса. Для предотвращения попадания таких электронных писем в спам используйте адрес, принадлежащий к домену вашего сайта.' (Automated electronic messages, such as registration information, will be sent from this address. To prevent such electronic messages from ending up in spam, use an address belonging to your site's domain.)
- УЧЁТНАЯ ЗАПИСЬ ОБСЛУЖИВАНИЯ САЙТА** (Site maintenance user account):
- Имя пользователя *** (Username *): An empty text field.
- A descriptive paragraph: 'Допускаются некоторые спецсимволы, среди которых пробел, точка (.), дефис (-), одинарная кавычка ('), подчёркивание (_) и знак @.' (Some special characters are allowed, including space, period (.), hyphen (-), single quote ('), underscore (_), and the @ symbol.)
- Пароль *** (Password *): An empty text field.

Рисунок 7 – Завершение установки drupal

8. После завершения установки перед нами появляется наш сайт на основе CMS drupal.

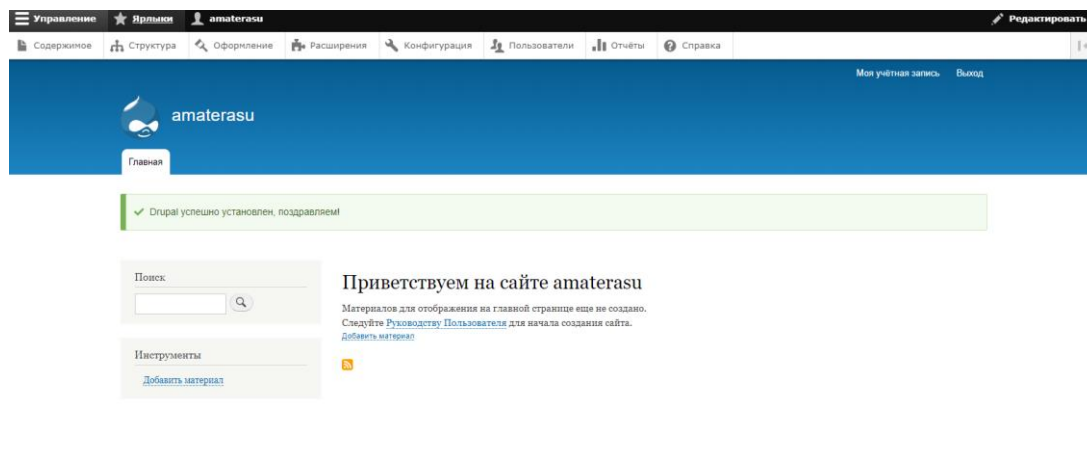


Рисунок 8 – Созданный сайт на CMS drupal

9. Теперь нам необходимо создать отдельную страницу (Node) чтобы воспользоваться интерфейсом интеграции сайта с API SWAPI. Переходим в административной панели в вкладку Содержимое и там мы попадаем в панель управление содержимым сайта.

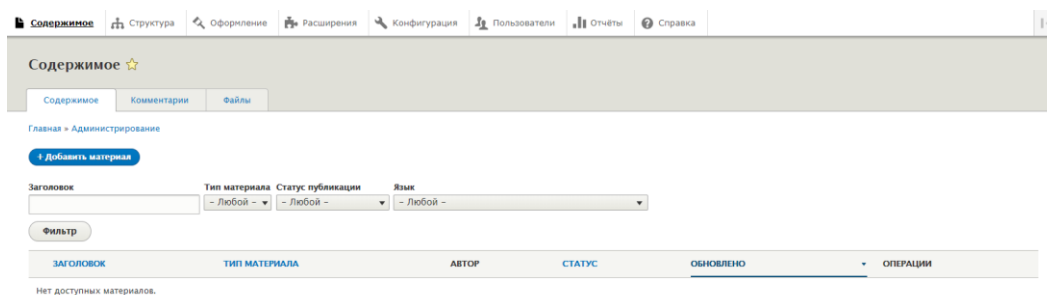


Рисунок 9 – Содержимое сайта

10. Теперь нам необходимо добавить страницу с интерфейсом интеграции. Нажимаем добавить материал и выбираем пункт Страница.

Вводим название страницы в заголовок и в настройки меню создать ссылку в меню.

Создание материала Страница ☆

Главная » Добавить материал

Заголовок *

SWAPI

Содержимое (Редактировать анонс)

В I [Rich Text Editor Icons] Формат... [Source Icon] Источник

Текстовый формат Базовый HTML [О текстовых форматах]

☒ Опубликовано

Сохранить Предпросмотр

Последнее сохранение: Еще не сохранено

Автор: amaterasu

Сообщение в журнал о редакции

Кратко опишите внесенные вами изменения.

НАСТРОЙКИ МЕНЮ

☒ Создать ссылку в меню

Название ссылки меню

SWAPI

Описание

Описание, показываемое при наведении мыши на ссылку меню.

Родитель

<Основная навигация>

Вес

1

Ссылки меню с меньшим весом отображаются перед ссылками с большим весом.

СИНОНИМ URL

Рисунок 10 – Создание страницы

Готовая страница создана и выглядит таким образом.

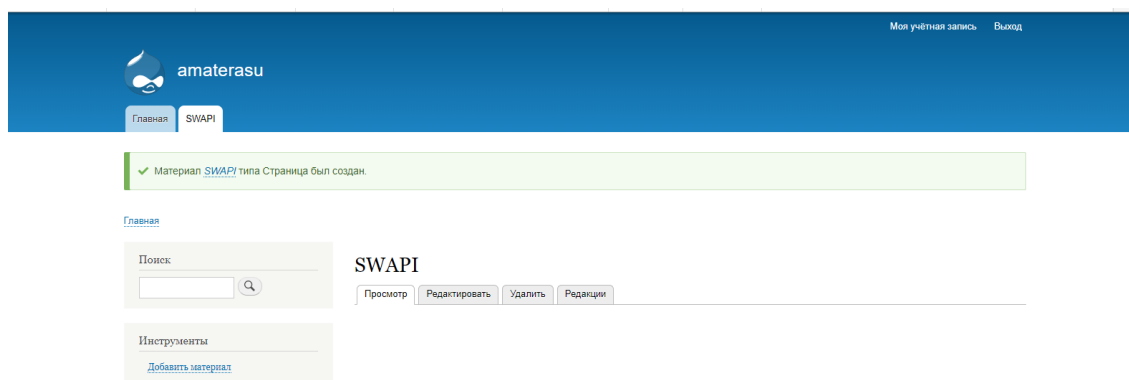


Рисунок 11 – Готовая страница

11. Теперь необходимо скопировать наш сайт с хостинга на компьютер. Для этого переходим на dashboard сайта на pantheon.io. Переходим в режим GIT разработки. С помощью консольного приложения GIT BUSH мы будем копировать файлы сайта с FTP на наш компьютер.

Жмем на кнопку Clone with Git. И перед нами появляется ссылка, с помощью которой мы и будем клонировать наш сайт на компьютер.

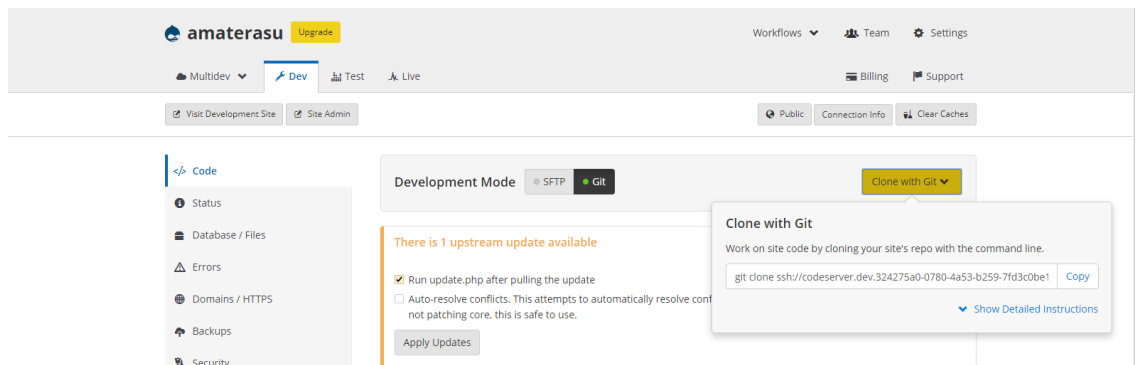


Рисунок 12 – Клонирование сайта

12. Создаем на компьютере папку и запускаем в ней консольное приложение Git Bush. Копируем полностью ссылку из раздела Clone with Git. И вставляем в git bush.

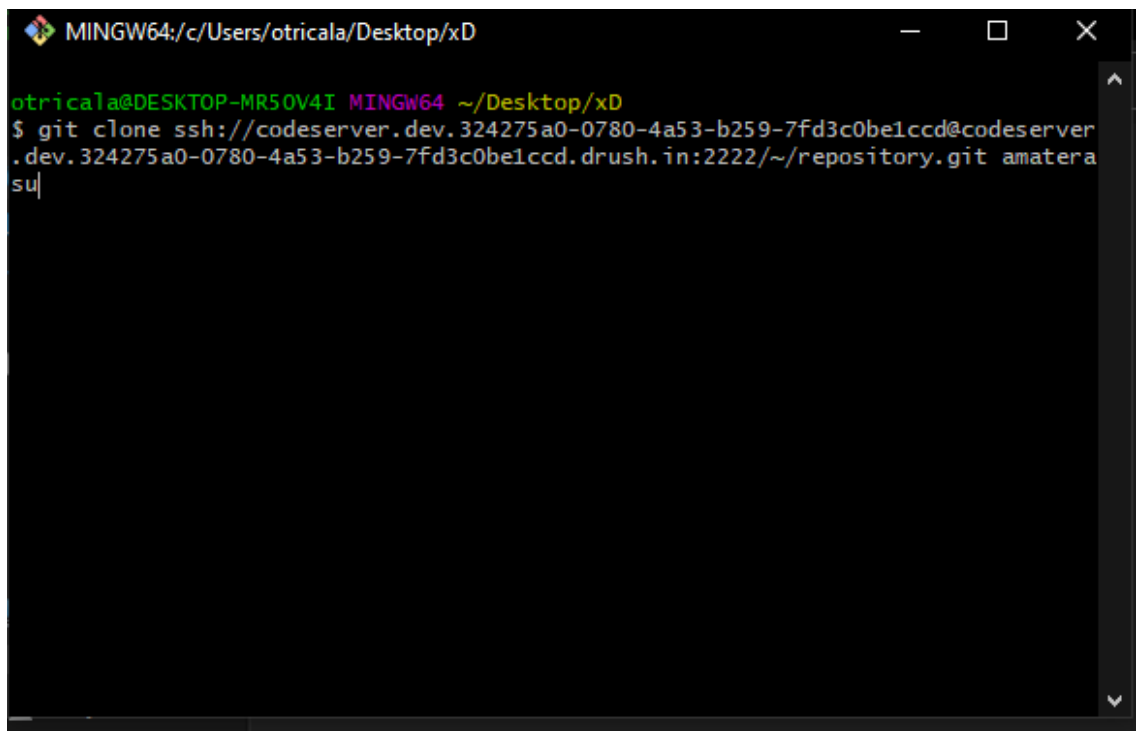
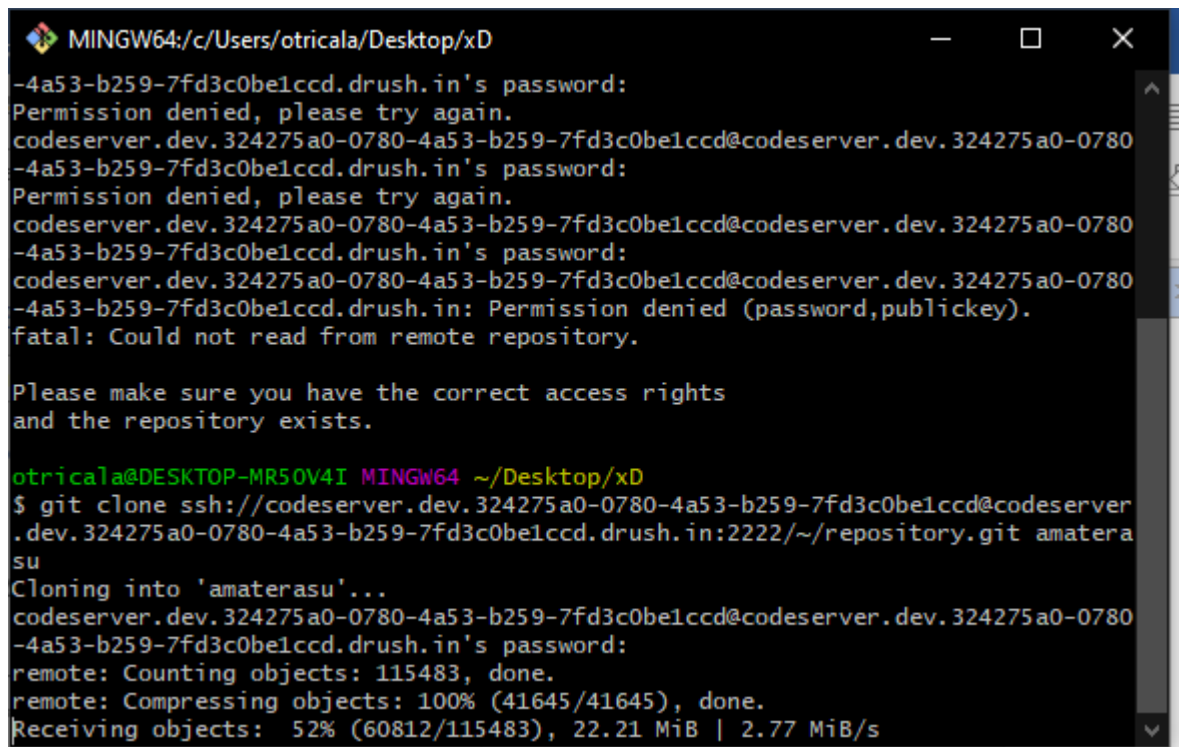


Рисунок 13 – Работа с Git Bush



```
MINGW64:/c/Users/otricala/Desktop/xD
-4a53-b259-7fd3c0be1ccd.drush.in's password:
Permission denied, please try again.
codeserver.dev.324275a0-0780-4a53-b259-7fd3c0be1ccd@codeserver.dev.324275a0-0780
-4a53-b259-7fd3c0be1ccd.drush.in's password:
Permission denied, please try again.
codeserver.dev.324275a0-0780-4a53-b259-7fd3c0be1ccd@codeserver.dev.324275a0-0780
-4a53-b259-7fd3c0be1ccd.drush.in's password:
codeserver.dev.324275a0-0780-4a53-b259-7fd3c0be1ccd@codeserver.dev.324275a0-0780
-4a53-b259-7fd3c0be1ccd.drush.in: Permission denied (password,publickey).
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

otricala@DESKTOP-MR50V4I MINGW64 ~/Desktop/xD
$ git clone ssh://codeserver.dev.324275a0-0780-4a53-b259-7fd3c0be1ccd@codeserver
.dev.324275a0-0780-4a53-b259-7fd3c0be1ccd.drush.in:2222/~/repository.git amatera
su
Cloning into 'amaterasu'...
codeserver.dev.324275a0-0780-4a53-b259-7fd3c0be1ccd@codeserver.dev.324275a0-0780
-4a53-b259-7fd3c0be1ccd.drush.in's password:
remote: Counting objects: 115483, done.
remote: Compressing objects: 100% (41645/41645), done.
Receiving objects: 52% (60812/115483), 22.21 MiB | 2.77 MiB/s
```

Рисунок 14 – Загрузка сайта на компьютер

13. Далее перед нами находится наша файловая система сайта.

Имя	Дата изменения	Тип	Размер
.git	21.03.2020 16:26	Папка с файлами	
core	21.03.2020 16:26	Папка с файлами	
drush	21.03.2020 16:26	Папка с файлами	
modules	21.03.2020 16:26	Папка с файлами	
profiles	21.03.2020 16:26	Папка с файлами	
sites	21.03.2020 16:26	Папка с файлами	
themes	21.03.2020 16:26	Папка с файлами	
vendor	21.03.2020 16:26	Папка с файлами	
.csslintrc	21.03.2020 16:25	Файл "CSSLINTRC"	2 КБ
.drush-lock-update	21.03.2020 16:25	Файл "DRUSH-LO...	1 КБ
.editorconfig	21.03.2020 16:25	Файл "EDITORCO...	1 КБ
.eslintignore	21.03.2020 16:25	Файл "ESLINTIGN...	1 КБ
.eslinttrc.json	21.03.2020 16:25	Файл "JSON"	1 КБ
.gitattributes	21.03.2020 16:25	Текстовый докум...	4 КБ
.gitignore	21.03.2020 16:25	Текстовый докум...	2 КБ
.ht.router.php	21.03.2020 16:25	Файл "PHP"	3 КБ
.htaccess	21.03.2020 16:25	Файл "HTACCESS"	8 КБ
autoload.php	21.03.2020 16:25	Файл "PHP"	1 КБ
composer.json	21.03.2020 16:25	Файл "JSON"	3 КБ
composer.lock	21.03.2020 16:25	Файл "LOCK"	131 КБ
example.gitignore	21.03.2020 16:26	Текстовый докум...	2 КБ
index.php	21.03.2020 16:26	Файл "PHP"	1 КБ
INSTALL.txt	21.03.2020 16:25	Текстовый докум...	1 КБ
LICENSE.txt	21.03.2020 16:25	Текстовый докум...	18 КБ
pantheon.upstream.yml	21.03.2020 16:26	Файл "YML"	1 КБ
pantheon.yml	21.03.2020 16:26	Файл "YML"	1 КБ
README.txt	21.03.2020 16:25	Текстовый докум...	6 КБ
robots.txt	21.03.2020 16:26	Текстовый докум...	2 КБ
update.php	21.03.2020 16:26	Файл "PHP"	1 КБ
web.config	21.03.2020 16:26	Файл "CONFIG"	5 КБ

Рисунок 15 – Файловая система сайта

Задание

Разработать систему интеграции своего сайта с сайтом SWAPI с помощью средств API.