



for  
**WebDev**

- **Что такое система контроля версий**
- **Установка и настройка**
- **Создание репозитария и его состояние**
- **Фиксирование изменений**
- **История изменений**
- **Сравнение**
- **Ветки**
- **Внешние репозитарии и сервисы GitHub и BitBucket**

# ЗАЧЕМ НУЖЕН КОНТРОЛЬ ВЕРСИЙ

# ПРИЧИНЫ ИСПОЛЬЗОВАТЬ VCS

- Командная работа
- Хранение актуального кода проекта
- Откат к предыдущей версии
- Понимание что происходит

ГЛАВНАЯ ПРИЧИНА  
— С VCS ВЫ  
МОЖЕТЕ МЕНЯТЬ  
ВСЕ ЧТО ХОТИТЕ И  
КАК ХОТИТЕ В КОДЕ



# ЧТО ПОЗВОЛЯЕТ КОНТРОЛИРОВАТЬ VCS

- Что менялось
- Кто менял
- Когда менял
- Зачем менял
- Что было до этого, и до того, и еще раньше

# ЧТО ЕЩЕ ДАЕТ VSC

- Работать параллельно над разными функциями в проекте и ничего не сломать
- Работать впятером над одним проектом и не затирать изменения друг друга
- Работать над новой фичей, потом поправить критичный баг на продакшене и не накатить случайно куски недоделанной фичи

# ПЛАТА ЗА ВОЗМОЖНОСТИ VCS

- Быть более организованным
- Заменить свой процесс работы на тот, который продиктован VCS
- Тратить время на работу с VCS больше, а значит тратить меньше времени на проект

# УСТАНОВКА GIT



# git --local-branching-on-the-cheap

Search entire site...

Git is a **free and open source distributed version control system** designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

 Learn Git in your browser for free with [Try Git](#).



**About**  
The advantages of Git compared to other source control systems.

**Documentation**  
Command reference pages, Pro Git book content, videos and other material.

**Downloads**  
GUI clients and binary releases for all major platforms.

**Community**  
Get involved! Bug reporting, mailing list, chat, development and more.

**Latest source Release**  
**2.7.4**  
Release Notes (2016-03-17)  
[Downloads for Mac](#)

  
[Pro Git](#) by Scott Chacon and Ben Straub is available to [read online for free](#).  
Dead tree versions are available on [Amazon.com](#).

<https://git-scm.com>

 [Mac GUIs](#)     [Tarballs](#)  
 [Windows Build](#)     [Source Code](#)

# ОСОБЕННОСТИ GIT

- Отслеживает изменения в файлах (не сами файлы и не папки)
- Ничего не отслеживает самостоятельно
- Вы сами решаете что добавить к отслеживанию
- Все изменяemo
- Можно начинать в любой фазе проекта
- Все ходы записаны и подписаны

# ПЕРВОНАЧАЛЬНАЯ НАСТРОЙКА GIT

```
$ git config --global user.name "John Doe"  
$ git config --global user.email john@me.com
```

# РЕПОЗИТОРИЙ СОЗДАНИЕ

# ПРАВИЛА СОЗДАНИЯ РЕПОЗИТОРИЯ (РЕПО)

- **Один проект – один репозиторий**
- **Если проект новый, создайте пустую папку, и в ней создайте репо**
- **Если проект уже есть, создайте репо в корневой папке проекта**
- **В репо можно будет хранить все то, что будет в той папке где он создан и в дочерних папках**
- **Не создавайте репо внутри другого репо**

# СОЗДАНИЕ РЕПО

```
$ git init
Initialized empty Git repository in
/path/to/project/.git/
```

# ПРОВЕРКА СОСТОЯНИЯ РЕПО

```
$ git status  
On branch master
```

Initial commit

```
nothing to commit (create/copy files and use  
"git add" to track)
```

# ЧТО ЭТО ЗНАЧИТ

- Мы находимся в ветке `master`
- Коммитов еще не было
- Нечего добавлять к коммиту
- GIT любезно подсказывает нам что можно сделать
- Сейчас он предлагает создать или скопировать файлы и добавить их к коммиту командой `git add`
- Прислушивайтесь к его советам

# УДАЛЕННЫЕ РЕПОЗИТОРИИ

# ПЛЮСЫ ИСПОЛЬЗОВАНИЯ УДАЛЕННОГО РЕПО

- Начали работать дома, продолжили на работе. Никаких флешек.
- Разделили работу на двоих, каждый работает над своей частью и всегда может получить наработки партнера.
- Хотите показать друзьям/коллегам/работодателям свои наработки – просто отправьте ссылку на публичный репо.
- Кто-то предлагает улучшения в ваш проект – пусть сделает форк, внесет правки и сделает пул-реквест.
- Можно собирать замечания, и опубликовать документацию по проекту.

# ХОСТИНГ ДЛЯ РЕПО



**Bitbucket**

**GitHub**



# СРАВНЕНИЕ

- На GitHub больше пользователей, репозиториев, команд, проектов.
- GitHub более стильный модный молодежный.
- На BitBucket можно создавать приватные репозитории бесплатно и работать с командой до 5 пользователей.
- На GitHub бесплатно только публичные репозитории

# СВЯЗЫВАЕМ ЛОКАЛЬНЫЙ И УДАЛЕННЫЙ РЕПО

Если локально репо еще нет:

```
$ git clone https://uname@bitbucket.org/uname/repo.git  
$ cd ./repo/
```

Будет создана папка **repo** в текущей папке, не забудьте в нее перейти

Если локально репо уже есть:

```
$ git remote add origin https://uname@bitbucket.org/uname/repo.git  
$ git push
```

Будет подключен удаленный репо с именем **origin**. Не забудьте опубликовать туда текущие изменения.

# КОМАНДЫ ДЛЯ РАБОТЫ С УДАЛЕННЫМ РЕПО

- **remote** – список удаленных репо
- **remote add** – добавить новый
- **clone** – создать локальную копию репо
- **push** – опубликовать изменения на удаленном репо
- **pull** – получить обновления из удаленного репо

# ФИКСИРОВАНИЕ ИЗМЕНЕНИЙ

- Состояние
- Коммит
- Снимок

# В КАКИХ СОСТОЯНИЯХ БЫВАЮТ ФАЙЛЫ

- Не отслеживается
- Изменения добавлены к коммиту полностью
- Нет изменений
- Есть изменения
- Файл удален
- Изменения добавлены к коммиту частично

# НЕ ОТСЛЕЖИВАЮТСЯ

- Все вновь созданные файлы находятся в этом состоянии
- Если вы создали репо в существующем проекте, то все файлы проекта в этом состоянии
- Вы можете делать с этими файлами все что хотите (редактировать, удалять)
- Изменения в этих файлах не отслеживаются
- Добавьте файл к коммиту когда придет время

# НЕ ОТСЛЕЖИВАЮТСЯ

```
$ git status
```

```
On branch master
```

```
Untracked files:
```

```
  (use "git add <file>..." to include in  
what will be committed)
```

```
index.html
```

```
nothing added to commit but untracked files  
present (use "git add" to track)
```

```
$ git add index.html
```

# ИЗМЕНЕНИЯ КОММИТА

- Все изменения сделанные в файле добавлены к коммиту, либо новый файл добавлен к коммиту
- Новые изменения в файле не будут автоматически добавлены к коммиту
- Создайте коммит, чтобы зафиксировать изменения
- Или отмените добавление изменений к коммиту

# ДОБАВЛЕНИЕ К КОММИТУ

```
$ git status
On branch master

Changes to be committed:
  (use "git rm --cached <file>..." to
unstage)

    new file:   index.html

$ git commit -m "Создал файл дом. страницы"
```

# НЕТ ИЗМЕНЕНИЙ

- Файл не редактировался с последнего коммита
- Их нет в списке который выводит команда состояния
- Такие файлы нельзя добавить к коммиту
- Внесите изменения в файл или удалите его, чтобы перевести его в другое состояние

# НЕТ ИЗМЕНЕНИЙ

```
$ git status
On branch master
nothing to commit, working directory clean
```

# ЕСТЬ ИЗМЕНЕНИЯ

- В файл внесены правки с момента последнего коммита
- Эти правки не добавлены к коммиту или добавлены частично
- Можно увидеть какие правки были внесены
- Добавьте файл к коммиту либо отмените изменения, чтобы изменить его состояние

# ЕСТЬ ИЗМЕНЕНИЯ

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what
will be committed)
  (use "git checkout -- <file>..." to
discard changes in working directory)

modified:   index.html

no changes added to commit (use "git add"
and/or "git commit -a")
```

# ДОБАВЛЯЕМ ФАЙЛЫ К КОММИТУ

```
$ git add index.html
$ git add css/main.css css/font.css
$ git add images/
```

# ДОБАВЛЯЕМ ФАЙЛЫ К КОММИТУ

- Внимательно следите что добавляете к коммиту именно те файлы которые нужны
- Избегайте случаев добавления разом всей папки, или добавления вообще всех файлов
- Проверяйте что добавилось к коммиту после добавления используя `git status`

# ПРОВЕРЯЕМ СОСТОЯНИЕ

```
$ git status
Changes to be committed:
  (use "git reset HEAD <file>..." to
unstage)

  new file:    css/fonts.css
  new file:    css/main.css
  modified:   index.html
```

ВСЕГДА  
ПРОВЕРЯЙТЕ  
СОСТОЯНИЕ  
ПЕРЕД КОММИТОМ



# ОТМЕНА ДОБАВЛЕНИЯ ИЗМЕНЕЙ К КОММИТУ

```
$ git reset HEAD -- css/
```

# ОТМЕНА ДОБАВЛЕНИЯ ИЗМЕНЕНИЙ К КОММИТУ

```
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to
unstage)
  modified:   index.html

Untracked files:
  (use "git add <file>..." to include in
what will be committed)
  css/
```

# ОТМЕНА ДОБАВЛЕНИЯ ИЗМЕНЕЙ К КОММИТУ

- Если все же в коммит просочились файлы которые не нужны, просто удалите их из коммита
- Отменить можно один файл, несколько файлов, папку
- Если нужно отменить всё воспользуйтесь командой `git reset`

# ФИКСАЦИЯ ИЗМЕНЕНИЙ К КОММИТУ

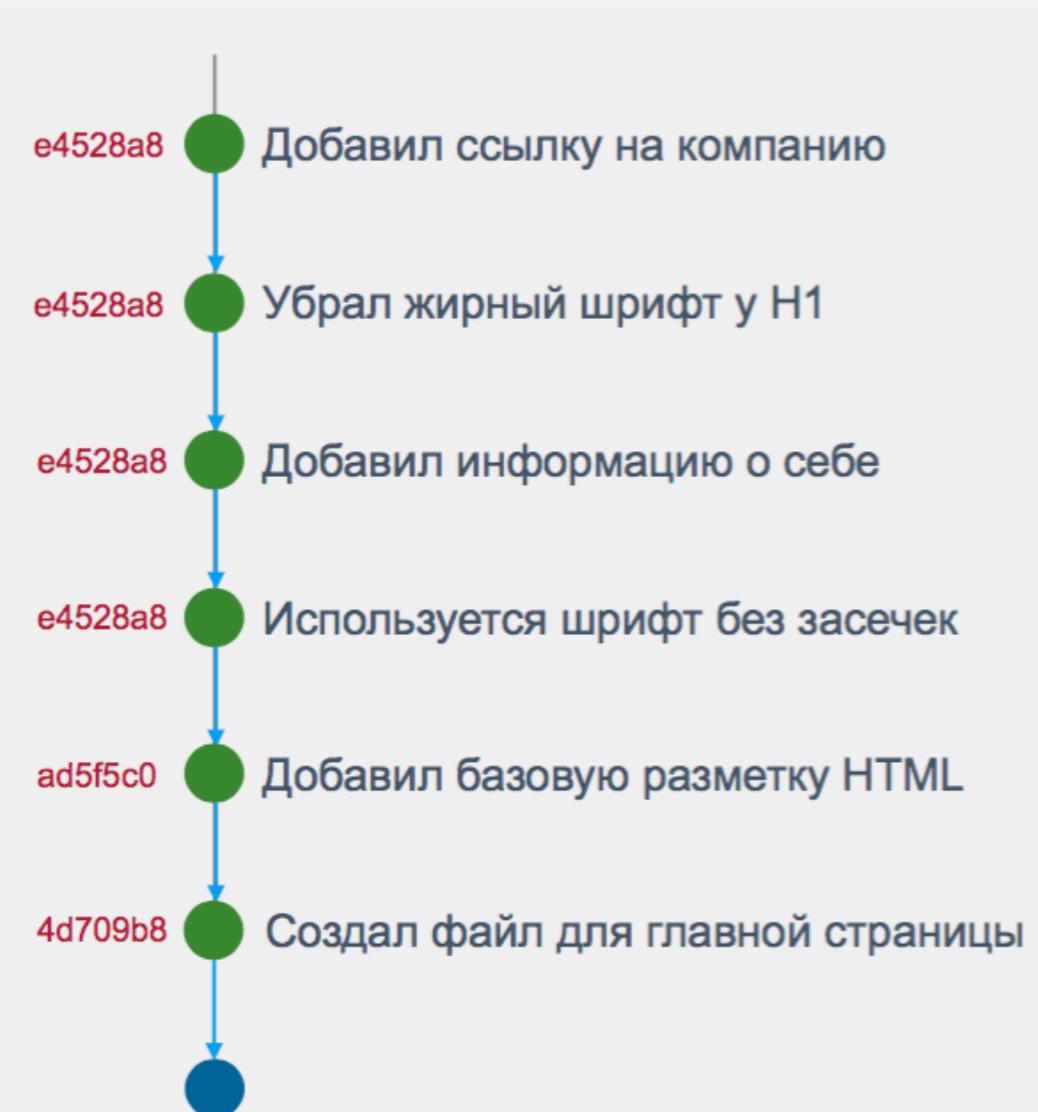
```
$ git commit -m "Создал файл дом. страницы"  
$ git status  
On branch master  
nothing to commit, working directory clean
```

# ФИКСАЦИЯ ИЗМЕНЕНИЙ К КОММИТУ

- Коммит – основная единица контроля версий
- Страйтесь делать коммит по каждому осмысленному изменению
- Например по каждому замечанию к домашней работе

# ИСТОРИЯ ИЗМЕНЕНИЙ

# ИСТРОИЯ ИЗМЕНИЙ



# ИСТРОИЯ ИЗМЕНИЙ

- Все коммиты идут один за другим
- Вместе с коммитом фиксируется дата создания
- Последние коммиты выводятся сверху
- Есть несколько способов вывода истории

# ПРОСМОТР ИСТОРИИ

```
$ git log
commit bd5b599b339d565f0d7c4189c6ed7b8c53bfe579
Author: Dima Fitiskin <dfitiskin@gmail.com>
Date:   Sat Mar 12 01:18:54 2016 +0300
```

Реализован базовый функционал роутера и прототипа контроллера. Доработан запрос, чтобы он мог хранить параметры и сущности

```
commit 5ca51603d11133deecef39b57f9466310699b88d
Author: Dima Fitiskin <dfitiskin@gmail.com>
Date:   Sat Feb 20 10:29:35 2016 +0300
```

Вынес стили из шаблона

:

# ПРОСМОТР ИСТОРИИ

```
$ git log --oneline
```

**bd5b599** Реализован базовый функционал роутера и прототипа контроллера. Доработан запрос, чтобы он мог хранить параметры и сущности

**5ca5160** Вынес стили из шаблона

**5fc9d3d** Редактирование параметров компании

**086f8f0** Добавил возможность просмотра акций и правил в админку

**d599b4f** Вынес функционал по работе с правилами в отдельный модуль

**af5cbaf** Условия бесплатной доставки создаются по правилам

**6f78d5e** Используем название группы для названия отбора и названия варианта, если не задан заголовок и короткое название (fix #100)

**39e1301** Реализована функция задания типа группы в админке Группа/Группа с фильтрами/Фильтр и настройки групп и фильтров (fix #104)

:

# ПЕРЕИМЕНОВАНИЕ УДАЛЕНИЕ

ОЧЕНЬ ЧАСТО ПО ХОДУ  
ПРОЕКТА МЫ ДЕЛАЕМ  
РЕФАКТОРИНГ —  
ПЕРЕИМЕНОВЫВАЕМ И  
УДАЛЯЕМ ФАЙЛЫ. КАК  
ФИКСИРОВАТЬ ЭТИ  
ИЗМЕНЕНИЯ?



# ПЕРЕИМЕНОВАНИЕ И УДАЛЕНИЕ

- **rm** – удалить файл в консоле
- **git rm** – удалить файл и зафиксировать удаление в текущий коммит
- **mv** – переименовать/переместить файл
- **git mv** – переименовать и зафиксировать изменения в текущий коммит

# ПРАКТИЧЕСКОЕ ЗАДАНИЕ

1. Создать удаленный репозиторий на GitHub
2. Пригласить коворкера rakashnath
3. Сверстать макет **Homework.psd**
4. Для каждого блока сделать отдельный коммит
5. Опубликовать изменения на GitHub
6. Скинуть ссылку на репо в Telegram