**Лабораторная работа 5**

**Изменение оригинального репозитория**

**Цели:** внести некоторые изменения в оригинальный репозиторий, чтобы затем попытаться извлечь и слить изменения из удаленной ветки в текущую

**1.Внесите изменения в оригинальный репозиторий hello**

**ВЫПОЛНИТЕ:**

cd ../hello

# (You should be in the original hello repository now)

**Примечание: Сейчас мы находимся в репозитории *hello***

Внесите следующие изменения в файл README:

**ФАЙЛ: *README***

This is the Hello World example from the git tutorial.

(changed in original)

Теперь добавьте это изменение и сделайте коммит

**ВЫПОЛНИТЕ:**

git add README

git commit -m "Changed README in original repo"

**2. Далее**

Теперь в оригинальном репозитории есть более поздние изменения, которых нет в клонированной версии. Далее мы извлечем и сольем эти изменения в клонированный репозиторий.

# Извлечение изменений

### Цели: научиться извлекать изменения из удаленного репозитория.

#### ВЫПОЛНИТЕ:

cd ../cloned\_hello

git fetch

git hist –all

**Примечание: Сейчас мы находимся в репозитории cloned\_hello**

#### РЕЗУЛЬТАТ:

$ git fetch

From /Users/alex/Documents/Presentations/githowto/auto/hello

6e6c76a..2faa4ea master -> origin/master

$ git hist --all

\* 2faa4ea 2011-03-09 | Changed README in original repo (origin/master, origin/HEAD) [Alexander Shvets]

\* 6e6c76a 2011-03-09 | Updated index.html (HEAD, origin/style, master) [Alexander Shvets]

\* 1436f13 2011-03-09 | Hello uses style.css [Alexander Shvets]

\* 59da9a7 2011-03-09 | Added css stylesheet [Alexander Shvets]

\* 6c0f848 2011-03-09 | Added README [Alexander Shvets]

\* 8029c07 2011-03-09 | Added index.html. [Alexander Shvets]

\* 567948a 2011-03-09 | Moved hello.html to lib [Alexander Shvets]

\* 6a78635 2011-03-09 | Add an author/email comment [Alexander Shvets]

\* fa3c141 2011-03-09 | Added HTML header (v1) [Alexander Shvets]

\* 8c32287 2011-03-09 | Added standard HTML page tags (v1-beta) [Alexander Shvets]

\* 43628f7 2011-03-09 | Added h1 tag [Alexander Shvets]

\* 911e8c9 2011-03-09 | First Commit [Alexander Shvets]

На данный момент в репозитории есть все коммиты из оригинального репозитория, но они не интегрированы в локальные ветки клонированного репозитория.

В истории выше найдите коммит «Changed README in original repo». Обратите внимание, что коммит включает в себя коммиты «origin/master» и «origin/HEAD».

Теперь давайте посмотрим на коммит «Updated index.html». Вы увидите, что локальная ветка master указывает на этот коммит, а не на новый коммит, который мы только что извлекли.

Выводом является то, что команда «git fetch» будет извлекать новые коммиты из удаленного репозитория, но не будет сливать их с вашими наработками в локальных ветках.

## 1 Проверьте README

Мы можем продемонстрировать, что клонированный файл README не изменился.

#### ВЫПОЛНИТЕ:

cat README

#### РЕЗУЛЬТАТ:

$ cat README

This is the Hello World example from the git tutorial.

Как видите, никаких изменений.

# Слияние извлеченных изменений

### Цели: Научиться перемещать извлеченные изменения в текущую ветку и рабочий каталог.

## 1 Слейте извлеченные изменения в локальную ветку master

#### ВЫПОЛНИТЕ:

git merge origin/master

#### РЕЗУЛЬТАТ:

$ git merge origin/master

Updating 6e6c76a..2faa4ea

Fast-forward

README | 1 +

1 files changed, 1 insertions(+), 0 deletions(-)

## 2 Еще раз проверьте файл README

Сейчас мы должны увидеть изменения.

#### ВЫПОЛНИТЕ:

cat README

#### РЕЗУЛЬТАТ:

$ cat README

This is the Hello World example from the git tutorial.

(changed in original)

Вот и изменения. Хотя команда «git fetch» не сливает изменения, мы можем вручную слить изменения из удаленного репозитория.

## 3 Далее

Теперь давайте рассмотрим объединение fetch & merge в одну команду.

# Извлечение и слияние изменений

### Цели: узнать о том, что команда git pull эквивалентна комбинации git fetch и git merge.

### Обсуждение

Мы не собираемся опять проходить весь процесс создания нового изменения и его извлечения, но мы хотим, чтобы вы знали, что выполнение:

git pull

действительно эквивалентно двум следующим шагам:

git fetch

git merge origin/master

# Добавление ветки наблюдения

### Цели: научиться добавлять локальную ветку, которая отслеживает изменения удаленной ветки.

Ветки, которые начинаются с remotes/origin являются ветками оригинального репозитория. Обратите внимание, что у вас больше нет ветки под названием style, но система контроля версий знает, что в оригинальном репозитории ветка style была.

## 1 Добавьте локальную ветку, которая отслеживает удаленную ветку.

#### ВЫПОЛНИТЕ:

git branch --track style origin/style

git branch -a

git hist --max-count=2

#### РЕЗУЛЬТАТ:

$ git branch --track style origin/style

Branch style set up to track remote branch style from origin.

$ git branch -a

style

\* master

remotes/origin/HEAD -> origin/master

remotes/origin/style

remotes/origin/master

$ git hist --max-count=2

\* 2faa4ea 2011-03-09 | Changed README in original repo (HEAD, origin/master, origin/HEAD, master) [Alexander Shvets]

\* 6e6c76a 2011-03-09 | Updated index.html (origin/style, style) [Alexander Shvets]

Теперь мы можем видеть ветку style в списке веток и логе.

# Чистые репозитории

### Цели: научиться создавать чистые репозитории.

Чистые репозитории (без рабочих каталогов) обычно используются для расшаривания.

Небольшое пояснение, что же все-таки означает «чистый репозиторий». Обычный git-репозиторий подразумевает, что вы будете использовать его как рабочую директорию, поэтому вместе с файлами проекта в актуальной версии, git хранит все служебные, «чисто-репозиториевские» файлы в поддиректории .git. В удаленных репозиториях нет смысла хранить рабочие файлы на диске (как это делается в рабочих копиях), а все что им действительно нужно — это дельты изменений и другие бинарные данные репозитория. Вот это и есть «чистый репозиторий».

## 1 Создайте чистый репозиторий

#### ВЫПОЛНИТЕ:

cd ..

git clone --bare hello hello.git

ls hello.git

**Примечание: Сейчас мы находимся в рабочем каталоге**

#### РЕЗУЛЬТАТ:

$ git clone --bare hello hello.git

Cloning into bare repository hello.git...

done.

$ ls hello.git

HEAD

config

description

hooks

info

objects

packed-refs

refs

Как правило, репозитории, оканчивающиеся на «.git» являются чистыми репозиториями. Мы видим, что в репозитории hello.git нет рабочего каталога. По сути, это есть не что иное, как каталог .git нечистого репозитория.

# Добавление удаленного репозитория

### Цели: добавить чистый репозиторий в качестве удаленного репозитория к нашему оригинальному репозиторию.

Давайте добавим репозиторий hello.git к нашему оригинальному репозиторию.

#### ВЫПОЛНИТЕ:

cd hello

git remote add shared ../hello.git

**Примечание: Сейчас мы находимся в репозитории hello.**

# Отправка изменений

### Цели: научиться отправлять изменения в удаленный репозиторий.

Так как чистые репозитории, как правило, расшариваются на каком-нибудь сетевом сервере, нам необходимо отправить наши изменения в другие репозитории.

Начнем с создания изменения для отправки. Отредактируйте файл README и сделайте коммит

#### ФАЙЛ: *README*

This is the Hello World example from the git tutorial.

(Changed in the original and pushed to shared)

#### ВЫПОЛНИТЕ:

git checkout master

git add README

git commit -m "Added shared comment to readme"

Теперь отправьте изменения в общий репозиторий.

#### ВЫПОЛНИТЕ:

git push shared master

Общим называется репозиторий, получающий отправленные нами изменения. (Помните, мы добавили его в качестве удаленного репозитория в предыдущем уроке.)

#### РЕЗУЛЬТАТ:

$ git push shared master

To ../hello.git

2faa4ea..79f507c master -> master

**Примечание:** Мы должны были явно указать ветку master для отправки изменений. Это можно настроить автоматически, но я все время забываю нужные команды. Для более простого управления удаленными ветками переключитесь в «Git Remote Branch».

# Извлечение общих изменений

### Цели: научиться извлекать изменения из общего репозитория.

Быстро переключитесь в клонированный репозиторий и извлеките изменения, только что отправленные в общий репозиторий.

#### ВЫПОЛНИТЕ:

cd ../cloned\_hello

**Примечание: Сейчас мы находимся в репозитории cloned\_hello.**

Продолжите с…

#### ВЫПОЛНИТЕ:

git remote add shared ../hello.git

git branch --track shared master

git pull shared master

cat README

# 50. Размещение ваших git репозиториев

### Цели: научиться настраивать git сервер для совместного использования репозиториев.

Есть много способов расшаривать git репозитории по сети. Вот быстрый способ.

## 1 Запуск git сервера

#### ВЫПОЛНИТЕ:

# (From the work directory)

git daemon --verbose --export-all --base-path=.

Теперь в отдельном окне терминала перейдите в ваш рабочий каталог

#### ВЫПОЛНИТЕ:

# (From the work directory)

git clone git://localhost/hello.git network\_hello

cd network\_hello

ls

Вы увидите копию проекта hello.

## 2 Отправка в Git Daemon

Если вы хотите разрешить отправку (push) в репозиторий Git Daemon, добавьте метку --enable=receive-pack к команде git daemon. Будьте осторожны, этот сервер не производит аутентификацию, поэтому любой сможет отправлять изменения в ваш репозиторий.

# Расшаривание репозиториев

### Цели: научиться расшаривать репозитории по WIFI.

Посмотрите, запущен ли git daemon у вашего соседа. Обменяйтесь IP-адресами и проверьте, сможете ли вы извлекать изменения из репозиториев друг друга.