

Системы контроля версий



Кирилл Корняков (Itseez, ННГУ)
Сентябрь 2015

Содержание

1. Введение

2. Git

- *Индивидуальная работа*

3. Workflow

- *Распределенная работа*

4. Демонстрация

Будни студентов

- Вчера программа работала, я стал ее улучшать, она почему-то сломалась...
 - *Как откатить все назад?*
- Ранее я реализовал очень полезный класс, но потом удалил его. Сейчас он мне снова нужен.
 - *Есть ли возможность восстановить мой код?*
- Мы с Васей работали над разными частями программы.
У него все было неправильно, поэтому я улучшил его код.
Оказалось, что он тоже поменял мой код! Сейчас мы не можем понять,
 - *Кто, что и где поменял, и как это собрать воедино?*

Будни программистов

- Старшим инженером был реализован и отлажен программный компонент. Потом младший инженер делал какие-то мелкие правки. И все поломал! Сейчас в программе ошибка, но чтобы ее найти
 - *Нужно знать, какие участки кода он менял!*
- Мы втроем пишем большую систему. Каждый реализует свой модуль.
 - *Где последняя и полная версия системы?*
- Месяц назад мы выпустили gold release. Сейчас мы активно работаем над новым релизом, который должен выйти через 3 месяца. Однако, заказчик нашел критическую ошибку и просит выдать ему обновление.
 - *Как выдать стабильное обновление, если система уже наполовину разобрана?*

Семь бед — один ответ!

Системы контроля версий — это программные системы, хранящие несколько версий документов, и позволяющие вернуться к более ранним версиям.

Типичные функции

1. Возможность получения прошлых версий файлов

- *Получение истории индивидуального файла*
- *Получение среза всего проекта (snapshot)*

2. Централизованное хранение исходного кода

- *Актуальное "официальное" состояние проекта*
- *Удаленный доступ к проекту*
- *Разграничение прав доступа*
- *Использование в качестве запасного хранилища экспериментальных веток (backup)*

Машина времени и сетевое хранилище в одном флаконе!

Патчи

Патч (англ. patch — заплатка) — информация, предназначенная для автоматизированного внесения определённых изменений в компьютерные файлы.

Unified diff format: @@ -1,s +1,s @@ optional section heading

```

13
14 diff --git a/README.md b/README.md
15 index afadff2..bde857e 100644
16 --- a/README.md
17 +++ b/README.md
18 @@ -40,10 +40,10 @@ __Цель данной работы__ - реализовать набор пр
19     содержащий простейшую реализацию класса матриц. Предполагается, что он не
20     редактируется при реализации фильтров.
21     - Модуль `filters`(`./include/filters.hpp`, `./src/filters_opencv.cpp`,
22 -   `./src/filters_fabrics.cpp`), содержащий объявление абстрактного класса
23 +   `./src/filters_factory.cpp`), содержащий объявление абстрактного класса
24     фильтров (`filters.hpp`) и его наследника, который реализует перечисленные
25     фильтры средствами библиотеки OpenCV (`filters_opencv.cpp`), а также метод
26 -   создания конкретной реализации класса фильтров (`filters_fabrics.cpp`).
27 +   создания конкретной реализации класса фильтров (`filters_factory.cpp`).
28     - Тесты для класса матриц и фильтров (`matrix_test.cpp`, `filters_test.cpp`).
29     - Пример использования фильтра (`matrix_sample.cpp`).
30
31 @@ -489,11 +489,11 @@ __Примечание:__ генератор проекта должен сов
32     значение, соответствующее вашей реализации фильтра. Назовите его
33     согласно вашей фамилии `YOUR_NAME`. Указанное перечисление используется
34     при прогоне одних и тех же тестов на всех реализациях класса фильтров.
35 -   1. В файле `filters_fabrics.cpp` объявите функцию
36 +   1. В файле `filters_factory.cpp` объявите функцию
37     `Filters* createFiltersYourName()`. Данная функция будет использована
38     при создании объекта класса с вашей реализации фильтров.
39     1. В функции `Filters* createFilters(FILTERS_IMPLEMENTATIONS impl)` (файл
40 -   `filters_fabrics.cpp`) необходимо добавить еще одну ветку у оператора-
41 +   `filters_factory.cpp`) необходимо добавить еще одну ветку у оператора-
42     переключателя `switch`, по которой будет проходить исполнение программы,
43     если создан объект класса фильтров `YOUR_NAME`.
44     1. В файл `filters_YOUR_NAME.cpp` необходимо поместить реализацию функции

```

Отображение на GitHub

8	README.md	
✳	@@ -40,10 +40,10 @@ __Цель данной работы__ - реализовать набор пр	
40	40	содержащий простейшую реализацию класса матриц. Предполагается, что он не
41	41	редактируется при реализации фильтров.
42	42	- Модуль <code>`filters`</code> (<code>`./include/filters.hpp`</code> , <code>`./src/filters_opencv.cpp`</code> ,
43	-	<code>`./src/filters_fabrics.cpp`</code>), содержащий объявление абстрактного класса
43	+	<code>`./src/filters_factory.cpp`</code>), содержащий объявление абстрактного класса
44	44	фильтров (<code>`filters.hpp`</code>) и его наследника, который реализует перечисленные
45	45	фильтры средствами библиотеки OpenCV (<code>`filters_opencv.cpp`</code>), а также метод
46	-	создания конкретной реализации класса фильтров (<code>`filters_fabrics.cpp`</code>).
46	+	создания конкретной реализации класса фильтров (<code>`filters_factory.cpp`</code>).
47	47	- Тесты для класса матриц и фильтров (<code>`matrix_test.cpp`</code> , <code>`filters_test.cpp`</code>).
48	48	- Пример использования фильтра (<code>`matrix_sample.cpp`</code>).
49	49	
✳	@@ -489,11 +489,11 @@ __Примечание:__ генератор проекта должен сов	
489	489	значение, соответствующее вашей реализации фильтра. Назовите его
490	490	согласно вашей фамилии <code>`YOUR_NAME`</code> . Указанное перечисление используется
491	491	при прогоне одних и тех же тестов на всех реализациях класса фильтров.
492	-	1. В файле <code>`filters_fabrics.cpp`</code> объявите функцию
492	+	1. В файле <code>`filters_factory.cpp`</code> объявите функцию
493	493	<code>`Filters* createFiltersYourName()`</code> . Данная функция будет использована
494	494	при создании объекта класса с вашей реализации фильтров.
495	495	1. В функции <code>`Filters* createFilters(FILTERS_IMPLEMENTATIONS impl)`</code> (файл
496	-	<code>`filters_fabrics.cpp`</code>) необходимо добавить еще одну ветку у оператора-
496	+	<code>`filters_factory.cpp`</code>) необходимо добавить еще одну ветку у оператора-
497	497	переключателя <code>`switch`</code> , по которой будет проходить исполнение программы,
498	498	если создан объект класса фильтров <code>`YOUR_NAME`</code> .
499	499	1. В файл <code>`filters_YOUR_NAME.cpp`</code> необходимо поместить реализацию функции
✳		

Отображение в командной строке

```
~/Work/summer-school-2015/practice1-devtools (master*)> git show dc2ca9d95c
commit dc2ca9d95cbd5586e9e5ef0fe1ce7db91ea7d3d1
Author: Daniil Osokin <daniil.osokin@itseez.com>
Date: Sun Aug 16 14:35:44 2015 +0300

    Switched to factory

diff --git a/README.md b/README.md
index afadff2..bde857e 100644
--- a/README.md
+++ b/README.md
@@ -40,10 +40,10 @@ __Цель данной работы__ - реализовать набор пр
    содержащий простейшую реализацию класса матриц. Предполагается, что он не
    редактируется при реализации фильтров.
-    Модуль `filters` ( `./include/filters.hpp`, `./src/filters_opencv.cpp`,
+    `./src/filters_fabrics.cpp`), содержащий объявление абстрактного класса
+    `./src/filters_factory.cpp`), содержащий объявление абстрактного класса
    фильтров (`filters.hpp`) и его наследника, который реализует перечисленные
    фильтры средствами библиотеки OpenCV (`filters_opencv.cpp`), а также метод
-    создания конкретной реализации класса фильтров (`filters_fabrics.cpp`).
+    создания конкретной реализации класса фильтров (`filters_factory.cpp`).
-    Тесты для класса матриц и фильтров (`matrix_test.cpp`, `filters_test.cpp`).
-    Пример использования фильтра (`matrix_sample.cpp`).

@@ -489,11 +489,11 @@ __Примечание:__ генератор проекта должен сов
    значение, соответствующее вашей реализации фильтра. Назовите его
    согласно вашей фамилии `YOUR_NAME`. Указанное перечисление используется
    при прогоне одних и тех же тестов на всех реализациях класса фильтров.
-    1. В файле `filters_fabrics.cpp` объявите функцию
+    1. В файле `filters_factory.cpp` объявите функцию
    `Filters* createFiltersYourName()`. Данная функция будет использована
    при создании объекта класса с вашей реализации фильтров.
    1. В функции `Filters* createFilters(FILTERS_IMPLEMENTATIONS impl)` (файл
-    `filters_fabrics.cpp`) необходимо добавить еще одну ветку у оператора-
+    `filters_factory.cpp`) необходимо добавить еще одну ветку у оператора-
    переключателя `switch`, по которой будет проходить исполнение программы,
    если создан объект класса фильтров `YOUR_NAME`.
    1. В файл `filters_YOUR_NAME.cpp` необходимо поместить реализацию функции
```

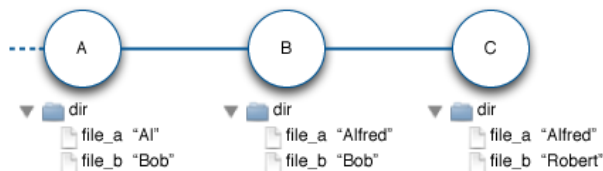
Терминология

Патчи

- Патч — это простой текстовый файл, его можно наложить при помощи инструментов (`patch`).
- Один патч может содержать изменения сразу нескольких файлов в разных директориях.
- Люди могут обмениваться изменениями, посылая друг другу патчи.
- Патч — это **атомарное изменение** проекта!

Патчи и СКВ

- СКВ — это своего рода БД патчей, ее называют **репозиторием**.
- Патчи, помещенные в СКВ называются **commit**.
- Последовательности **commit** называются **changeset**.



История изменений

ddc4a1d — Readme
bug fixes.

- README.md

f9e76e6 — Remove
dummy
implementation

- include/filters.hpp
- samples/matrix_sample.cpp
- src/filters_dummy.cpp
- src/filters_fabrics.cpp
- test/filters_test.cpp

























aa1611b — Switch
from strings to enums

- include/filters.hpp
- samples/matrix_sample.cpp
- src/filters_fabrics.cpp
- test/filters_test.cpp

8e8b21b — Add some
error checking

- include/filters.hpp
- src/filters_fabrics.cpp

Commits on Aug 11, 2015

	Readme bug fixes. valentina-kustikova authored 24 days ago		ddc4a1d	
	Remove dummy implementation kirill-kornyakov authored 24 days ago		f9e76e6	
	Switch from strings to enums kirill-kornyakov authored 24 days ago		aa1611b	
	Add some error checking kirill-kornyakov authored 24 days ago		8e8b21b	
	Run polymorphic tests kirill-kornyakov authored 24 days ago		a2ab7d7	
	Description bug fixes. valentina-kustikova authored 24 days ago		6591fb4	
	Description bug fixes. valentina-kustikova authored 24 days ago		b30fba0	
	Description bug fixes. valentina-kustikova authored 24 days ago		71db5e6	

Последовательность патчей — это полная история проекта.

```
- test/filters_test.cpp
```

Визуализация истории изменений

OpenCV 2.4.0



- <http://www.youtube.com/watch?v=ToD91PYaQOU>

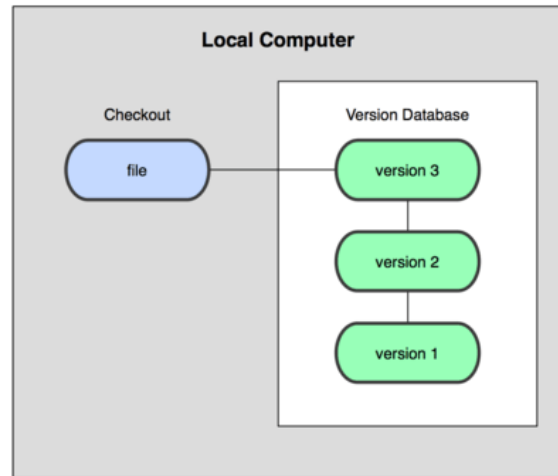
- Сделано при помощи [gource](#)

Три поколения СКВ

Generation Networking		Operations	Concurrency	Examples
First	None	One file at a time	Locks	RCS, SCCS
Second	Centralized	Multi-file	Merge before commit	CVS, Subversion, SourceSafe, Team Foundation Server
Third	Distributed	Changesets	Commit before merge	Git, Mercurial, Bazaar

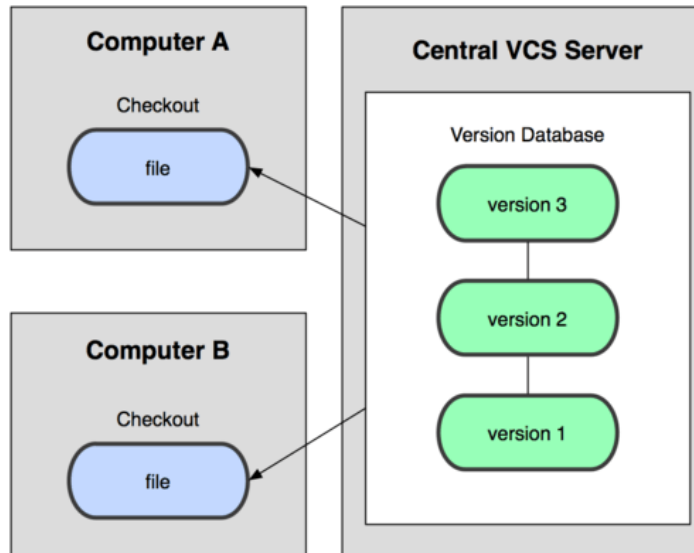
Eric Sink ["A History of Version Control"](#)

Три поколения СКВ: Локальные



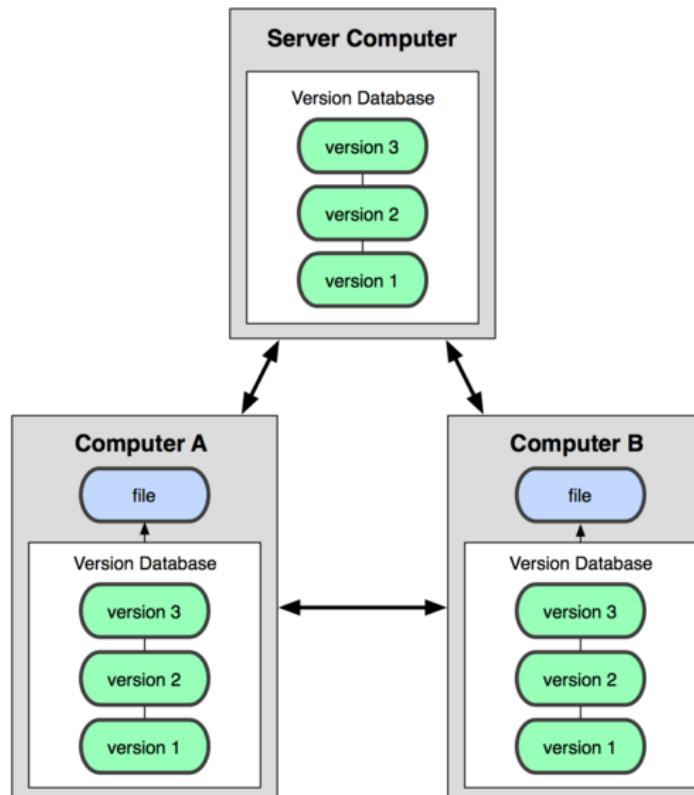
Примеры: RCS, SCCS

Три поколения СКВ: Централизованные



Примеры: Subversion, CVS

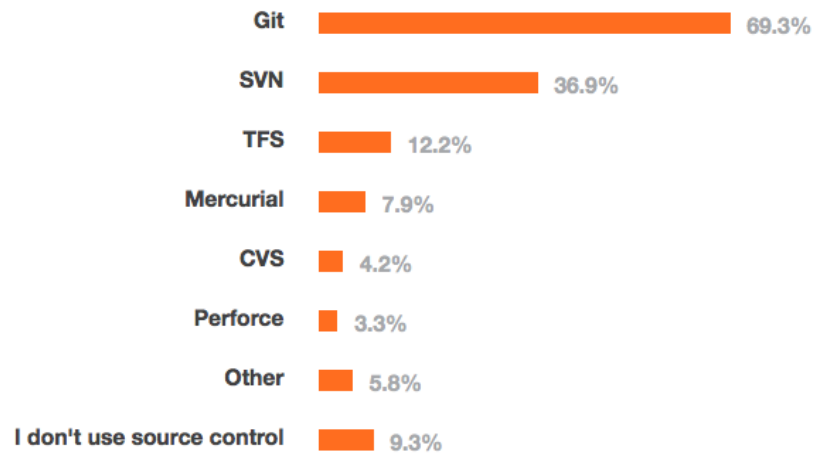
Три поколения СКВ: Распределенные



- Примеры: Git, Mercurial
- Фактически стали стандартом де-факто
- Сильные стороны:
 - Допускают локальную работу (коммиты без наличия интернет)

- Упрощают слияние (а значит параллельную разработку)
- Дают максимальную свободу по организации рабочего процесса (workflow)

Популярные СКВ



16,694 responses

Stack Overflow Developer Survey 2015

■ Использование в ИТ-проектах:

- *Фундаментальный инструмент разработки*
- *Также используется для: файлы конфигурации, документация, тестовые данные и пр.*



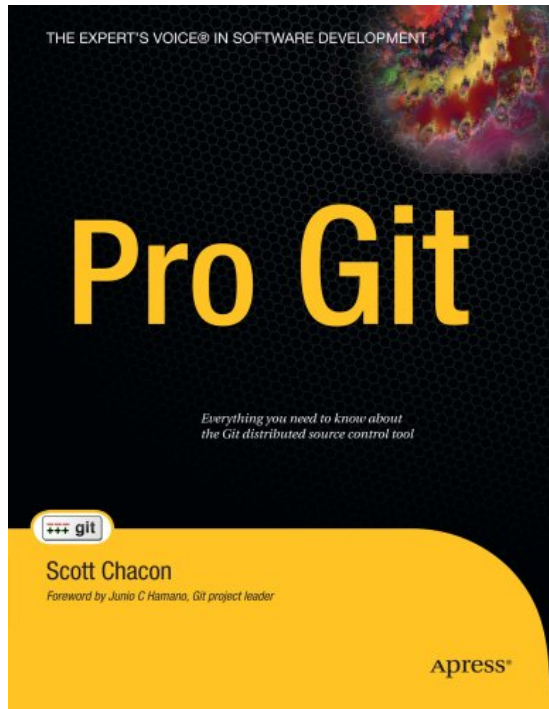
Git



git

- Разработан Линусом Торвальдсом для работы над ядром Linux в 2005 году.
- В настоящее время поддерживается Джуньо Хамано, сотрудником Google.
- Не очень прост в освоении, однако очень быстрый и функциональный.
- Имеет наиболее "сильное" сообщество, инструментальную поддержку.
- Огромное количество информации в интернет: инструкции, уроки, статьи
- Официальный сайт проекта: <http://www.git-scm.org>.

Pro Git

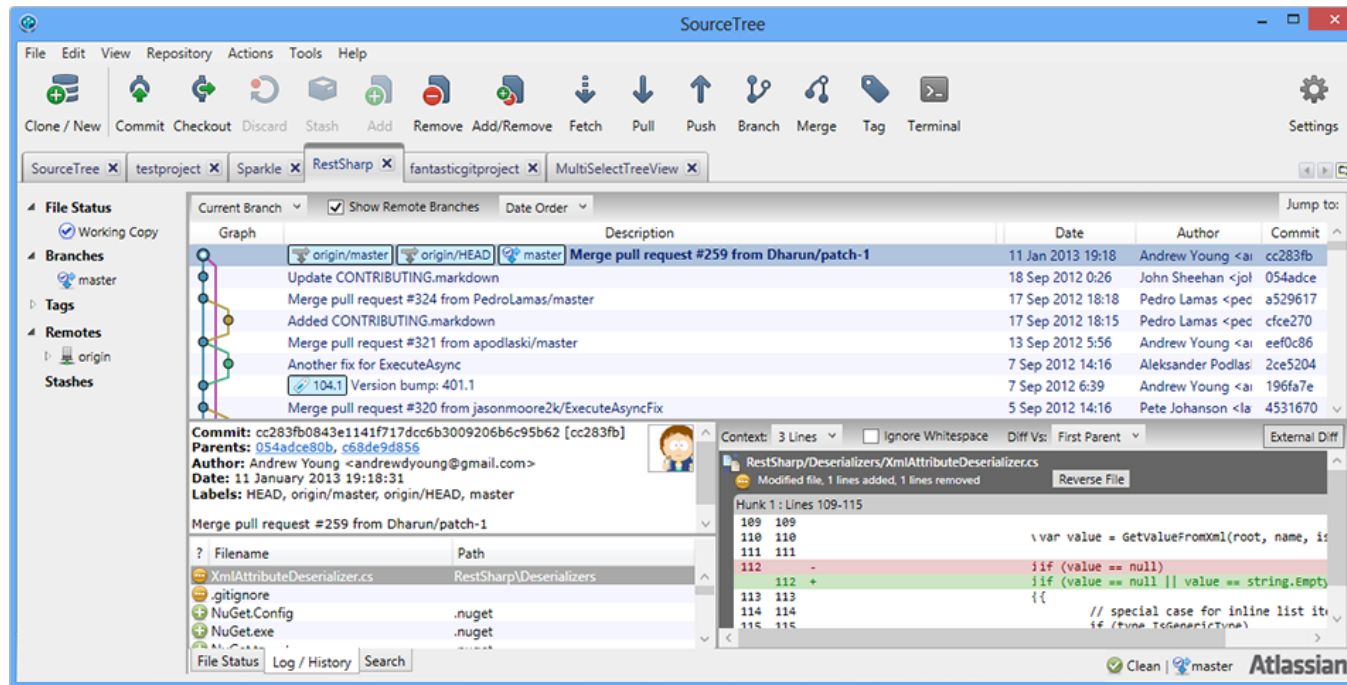


- Лучшая книга про Git
- Доступна бесплатно
- Переведена на [русский язык](#)
- Единственный способ по-настоящему понять Git — это узнать как он работает
- Нужно прочесть хотя бы первые 100 страниц

Как сказал Евклид египетскому царю Птолемею:

«Царской дороги в геометрии нет!»

Atlassian SourceTree



Tortoise Git

E:\dev\libs-nosync\1\Cinder - Log Messages - TortoiseGit

dev From: 21/04/2010 To: 12/03/2013 Filter by Subject, Messages, Paths, Authors, Emails, SHA-1, Refname, Bug-IDs Author Email

Graph	Actions	Message	Author	Date
		Working dir changes		
		dev origin/dev Merge branch 'appRewrite' into dev	Andrew Bell	12/03/2013 00:43:58
		Fixing reference to App in QuickTime.cpp	Andrew Bell	12/03/2013 00:42:46
		Changing VC11 foundation template to use v110	Andrew Bell	12/03/2013 00:30:00
		Upgrading VC11 to default to v110 compiler	Andrew Bell	12/03/2013 00:23:57
		Fixing missing sstream	Andrew Bell	11/03/2013 23:04:13
		Adding 1.53 VC10 and VC11 binaries	Andrew Bell	11/03/2013 23:03:46
		Upgrading MSW Boost binaries to 1.53	Andrew Bell	11/03/2013 23:02:43
		QuickTime.cpp change to build MSW qtime against Boost 1.53	Andrew Bell	11/03/2013 22:18:44
		Fixing Cinder-Boost submodule path	Andrew Bell	11/03/2013 20:57:23
		Adding boost to .gitmodules	Andrew Bell	11/03/2013 20:51:26
		Upgrading Cinder to Boost 1.53, submodule, Mac OS/iOS binaries	Andrew Bell	11/03/2013 20:43:57
		Finalized Mac screensaver refactor merge	Andrew Bell	10/03/2013 04:53:13
		Merge pull request #308 from calebjohnston/Vector-Color-additions	Andrew Bell	07/03/2013 17:28:21

SHA-1: 746dcefbeddd8a7aa8f339465e2e1c415dada728

* Merge branch 'appRewrite' into dev

Path	Extension	Status	L..	L..
Diff with parent 1				
.gitignore	.gitignore	Modified	2	2
.gitmodules	.gitmo...	Added	9	0
README.md	.md	Modified	4	6
blocks/Box2D/cinderblock.png	.png	Added	-	-
blocks/Box2D/cinderblock.xml	.xml	Added	27	0
blocks/Box2D/src/Box2D/Box2D.h	.h	Added	67	0
blocks/Box2D/src/Box2D/Collision/Shapes/b2Chai...	.cpp	Added	1..	0
blocks/Box2D/src/Box2D/Collision/Shapes/b2Chai...	.h	Added	1..	0
blocks/Box2D/src/Box2D/Collision/Shapes/b2Circle...	.cpp	Added	1..	0
blocks/Box2D/src/Box2D/Collision/Shapes/b2Circle...	.h	Added	91	0
blocks/Box2D/src/Box2D/Collision/Shapes/b2Edge...	.cpp	Added	1..	0
blocks/Box2D/src/Box2D/Collision/Shapes/b2Edge...	.h	Added	74	0

Showing 1847 revision(s), from revision 1fdb3ba to revision 746dcef - 1 revision(s) selected

☒ Hide Unrelated Changed Paths
 ☐ Show Whole Project

☐ All Branches
 ☐ Follow renames

☐ First Parent
 ☒ Show tags

Refresh

Statistics

Help

OK

GitHub Desktop

The screenshot displays the GitHub Desktop application interface. On the left, a sidebar lists repositories: Akavache, atom, GitPad, libgit2, libgit2sharp, octokit.net (selected), ReactiveUI, SeeGit, Windows, windows.github.com, tutorial, and sweet-gifs. The main area is divided into three panes. The top pane shows the 'History' tab with a list of commits, including 'Merge pull request #481 from octokit...' and 'cache enum label'. The bottom-left pane shows the 'Merge pull request #481 from octokit/sort-is-actually-case-sensitive' by Phil Haack, with a commit hash 'aad94fe'. The bottom-right pane displays a code diff for 'Octokit.Tests.Integration\Clients\SearchClientTests.cs', showing changes to the 'SearchForOpenIssues' method. The diff highlights a new method signature and implementation. The bottom-right pane also shows a code diff for 'Octokit\Helpers\EnumExtensions.cs', showing changes to the 'using' statements.

master ▾ Sync +20 ⚙

Filter repositories

GitHub

- Akavache
- atom
- GitPad
- libgit2
- libgit2sharp
- octokit.net
- ReactiveUI
- SeeGit
- Windows
- windows.github.com

Enterprise

- tutorial

Other

- sweet-gifs

History

- Merge pull request #481 from octoki... 29 days ago by Phil Haack
- cache enum label 29 days ago by Brendan Forster
- test fields 29 days ago by Brendan Forster
- removed redundant attributes from i... 29 days ago by Brendan Forster
- if the attribute cannot be found, Lowe... 29 days ago by Brendan Forster
- bugfix - the State value when searchi... 1 month ago by Brendan Forster
- Merge pull request #479 from octoki... 1 month ago by Phil Haack
- More jacked up tests 1 month ago by Paul Betts
- Fix up stubs 1 month ago by Paul Betts

Merge pull request #481 from octokit/sort-is-actually-case-sensitive

Phil Haack aad94fe GitHub Revert

searching for issues hits a case-sensitive field

Octokit.Tests.Integration\Clients\SearchClientTests.cs 12

```
... @@ -54,4 +54,16 @@ public class SearchClientTests
54 54
55 55     Assert.NotEmpty(repos.Items);
56 56 }
57 +
58 + [Fact]
59 + public async Task SearchForOpenIssues()
60 + {
61 +     var request = new SearchIssuesRequest("phone");
62 +     request.Repo = "caliburn-micro/caliburn.micro";
63 +     request.State = ItemState.Open;
64 +
65 +     var issues = await _githubClient.Search.SearchIssues(request);
66 +
67 +     Assert.NotEmpty(issues.Items);
68 + }
57 69 }
58 70
```

Octokit\Helpers\EnumExtensions.cs 9

```
... @@ -1,4 +1,5 @@
1 1 using System;
2 + using System.Diagnostics.CodeAnalysis;
2 3 using System.Linq;
3 4 using System.Reflection;
```

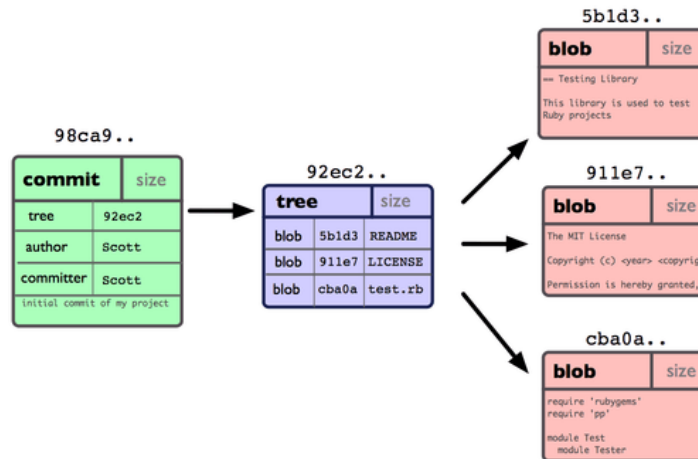
Command Line Interface!

```
-> git help
usage: git [--version] [--help] [-C <path>] [-c name=value]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p|--paginate|--no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        <command> [<args>]

The most commonly used git commands are:
add          Add file contents to the index
bisect       Find by binary search the change that introduced a bug
branch       List, create, or delete branches
checkout     Checkout a branch or paths to the working tree
clone        Clone a repository into a new directory
commit       Record changes to the repository
diff         Show changes between commits, commit and working tree, etc
fetch        Download objects and refs from another repository
grep         Print lines matching a pattern
init         Create an empty Git repository or reinitialize an existing one
log          Show commit logs
merge        Join two or more development histories together
mv           Move or rename a file, a directory, or a symlink
pull         Fetch from and integrate with another repository or a local branch
push         Update remote refs along with associated objects
rebase       Forward-port local commits to the updated upstream head
reset        Reset current HEAD to the specified state
rm           Remove files from the working tree and from the index
show         Show various types of objects
status       Show the working tree status
tag          Create, list, delete or verify a tag object signed with GPG

'git help -a' and 'git help -g' lists available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
```

Git objects



- По сути это *внутреннее* представление патча
- Пользователю приходится работать только с коммитами (слава богу!)

Показать содержимое коммита:

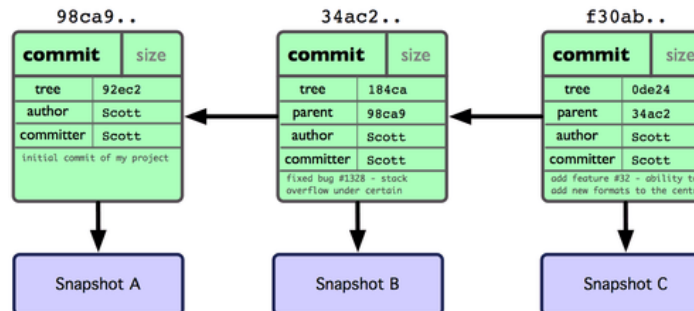
```
$ git show --raw dc2ca9d95c

commit dc2ca9d95cbd5586e9e5ef0fe1ce7db91ea7d3d1
Author: Daniil Osokin <daniil.osokin@itseez.com>
Date:   Sun Aug 16 14:35:44 2015 +0300

    Switched to factory

:100644 100644 afadff2... bde857e... M README.md
:100644 000000 c977bf3... 0000000... D src/filters_fabrics.cpp
:000000 100644 0000000... c977bf3... A src/filters_factory.cpp
```


Git commits



Вывести историю изменений:

```
$ git log

commit aaa321be9191da60ad52c2bc41bd749ed546b409
Merge: 98fce98 3c1d15a
Author: Valentina <valentina-kustikova@users.noreply.github.com>
Date: Thu Aug 13 10:14:47 2015 +0300

    Merge pull request #11 from valentina-kustikova/master

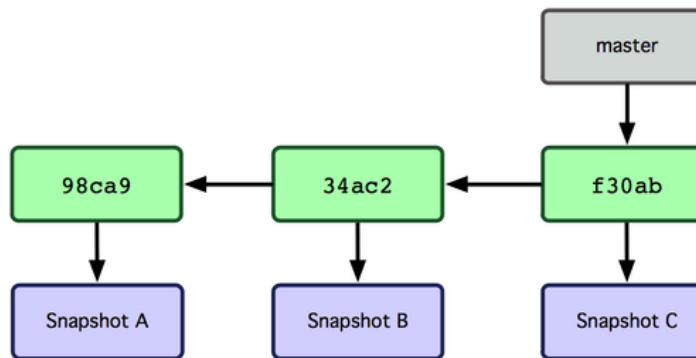
    Practice description (bug fixes).

commit 3c1d15a1bf366864593f2320fa9a0e6cf3586f52
Author: valentina-kustikova <valentina.kustikova@gmail.com>
Date: Thu Aug 13 10:08:59 2015 +0300

    Practice description (bug fixes).
```


Понятие ветки (branch)

- Ветка в Git'e — это просто **указатель** на один из коммитов.
- Есть соглашение, что имя **master** используется для ветки, указывающей на последнее актуальное состояние проекта.



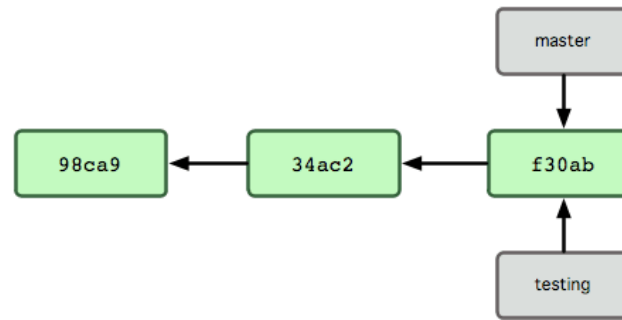
Вывести список существующих веток:

```
$ git branch
* master
```

Git branch

Создать новую ветку с именем `testing` (указатель на коммит!):

```
$ git branch testing
```

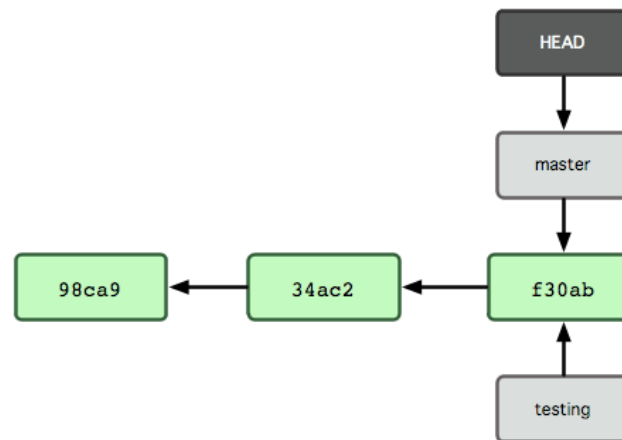


Текущий список веток:

```
$ git branch
* master
  testing
```

HEAD

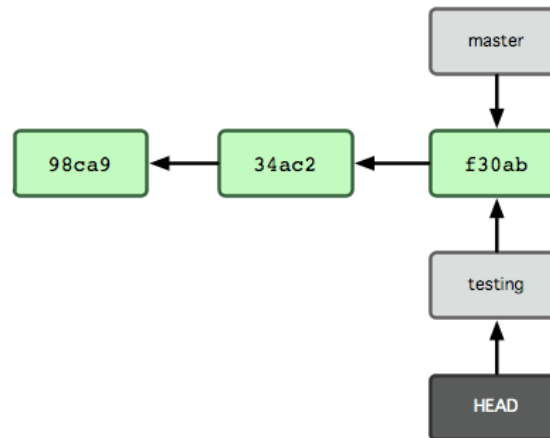
- HEAD — специальный указатель, ссылающийся на локальную ветку, на которой вы находитесь.
- Это просто алиас для текущей ветки, введенный для удобства.



Git checkout

Извлечь состояние репозитория, соответствующее ветке `testing`:

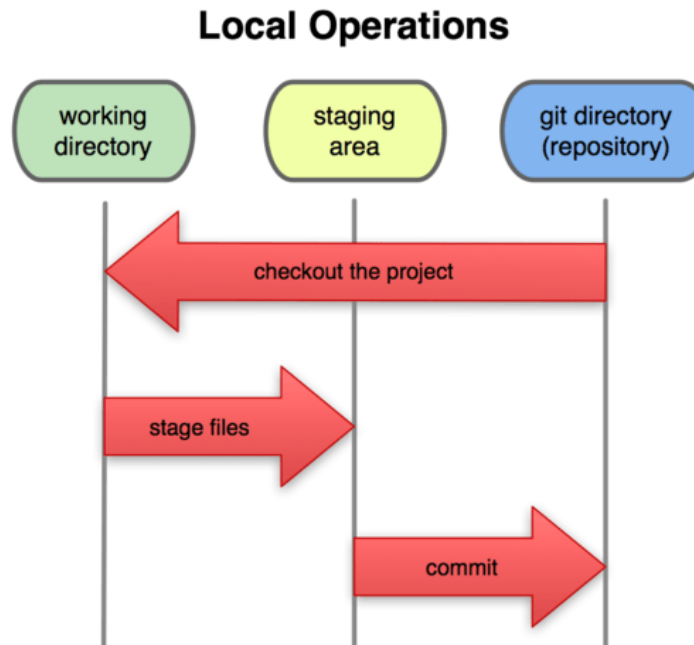
```
$ git checkout testing
```



Вывести список существующих веток:

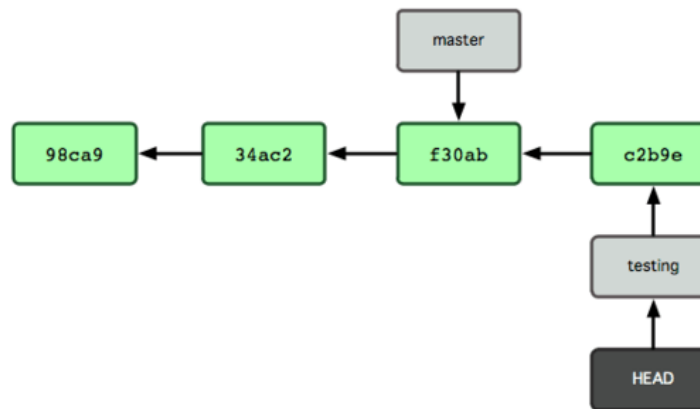
```
$ git branch
master
* testing
```

Три состояния файлов



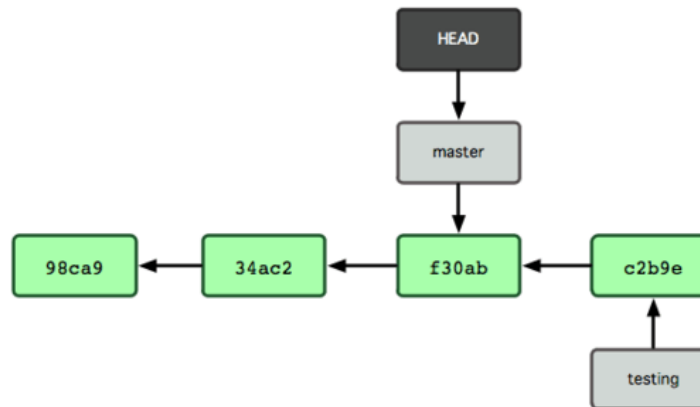
Git commit

```
$ vim README.md  
$ git add README.md  
$ git commit -m 'Made a change'
```



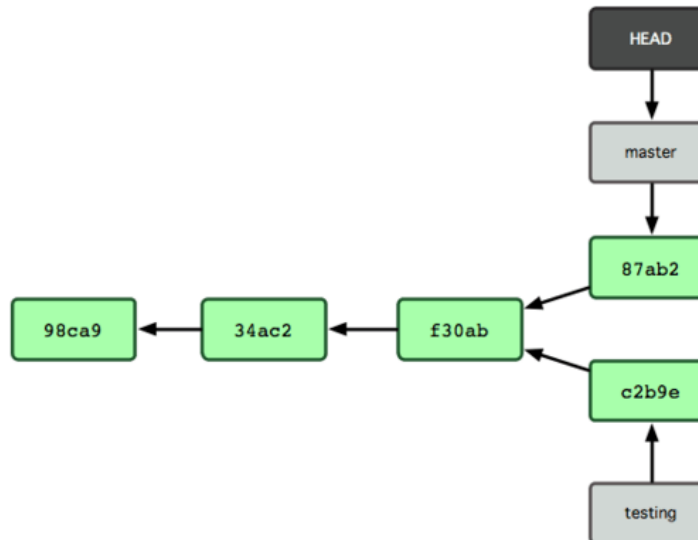
Go back to master

```
$ git checkout master
```

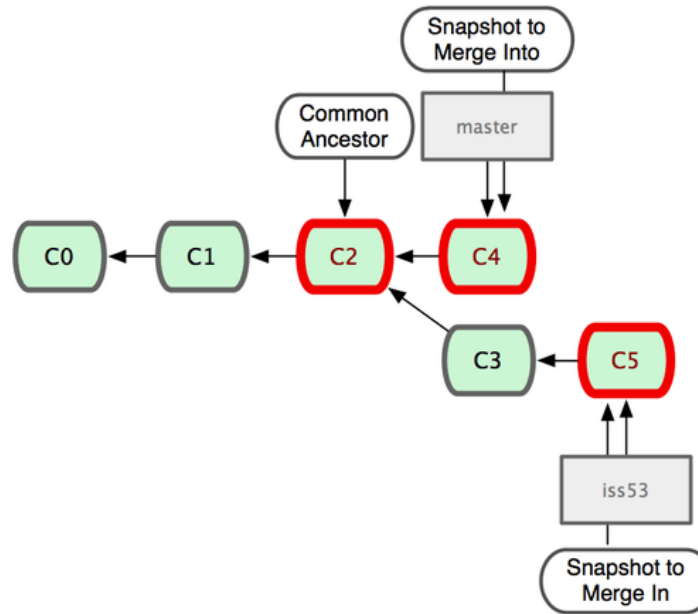


Make a commit to master

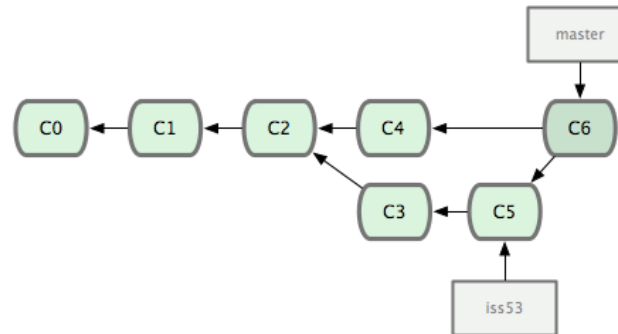
```
$ vim main.cpp  
$ git add main.cpp  
$ git commit -m 'Made other changes'  
  
# Или можно сделать так  
$ git status  
$ git commit -a -m 'Made other changes'
```



Merging

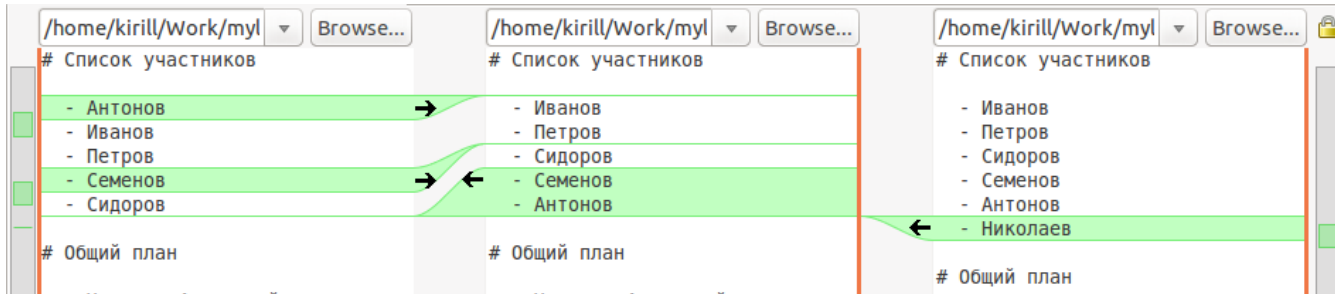


Merging



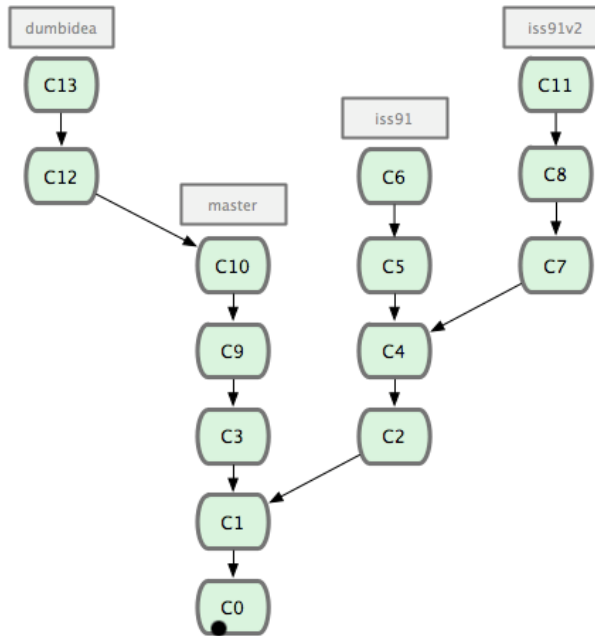
- C6 — это так называемый *merge commit*
- Он основан не на каком-то патче, он указывает на состояние проекта, в котором наложены патчи обеих ветвей (*master* и *testing*).

Merge Conflicts



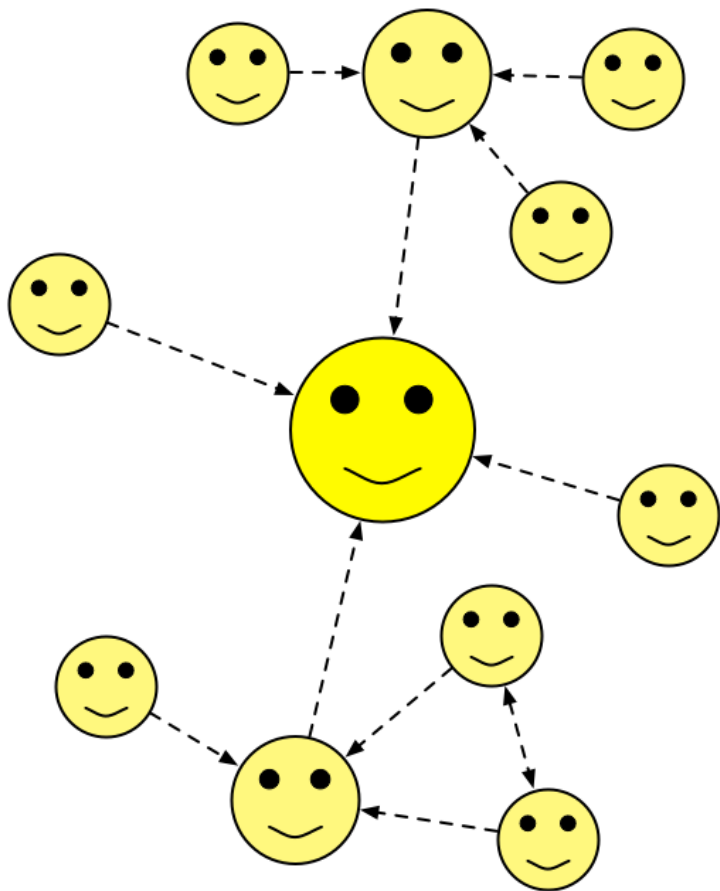
- Возникают когда несколько участников отредактировали одинаковые строки, или когда это произошло в разных ветках.
- Разрешаются человеком при помощи инструментов (`git mergetool`).
- В реальности довольно редкая ситуация, если соблюдать практики:
 - Грамотное распределение задач
 - Частые коммиты, много маленьких веток, частая интеграция

Multiple branches

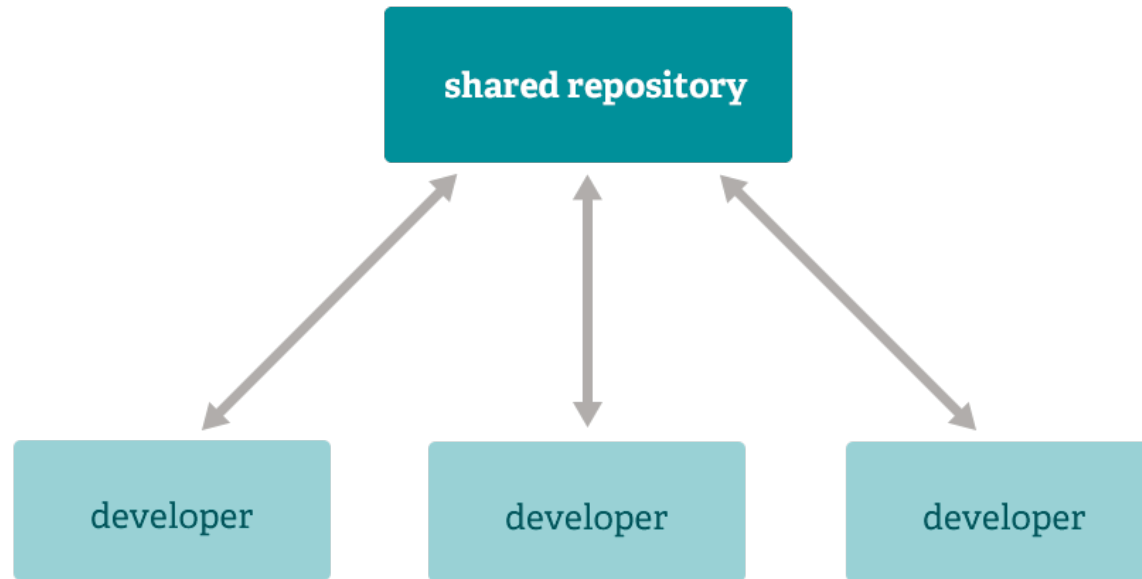


- Даже у одного разработчика может быть несколько активных веток.
- Правильно создавать отдельную ветку на каждую логически независимую задачу.
- Долгоживущие ветки — это неправильно, они быстро устаревают.

Распределенная работа

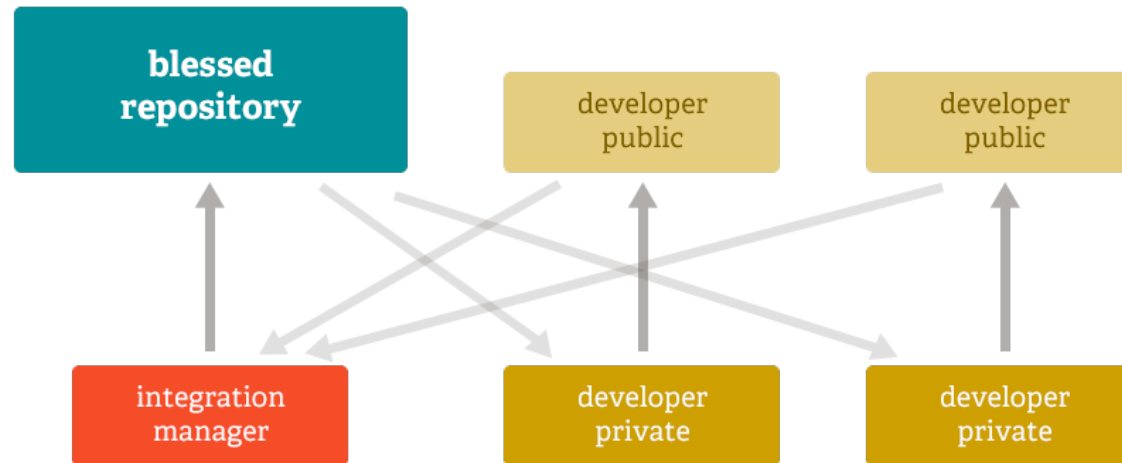


Centralized Workflow



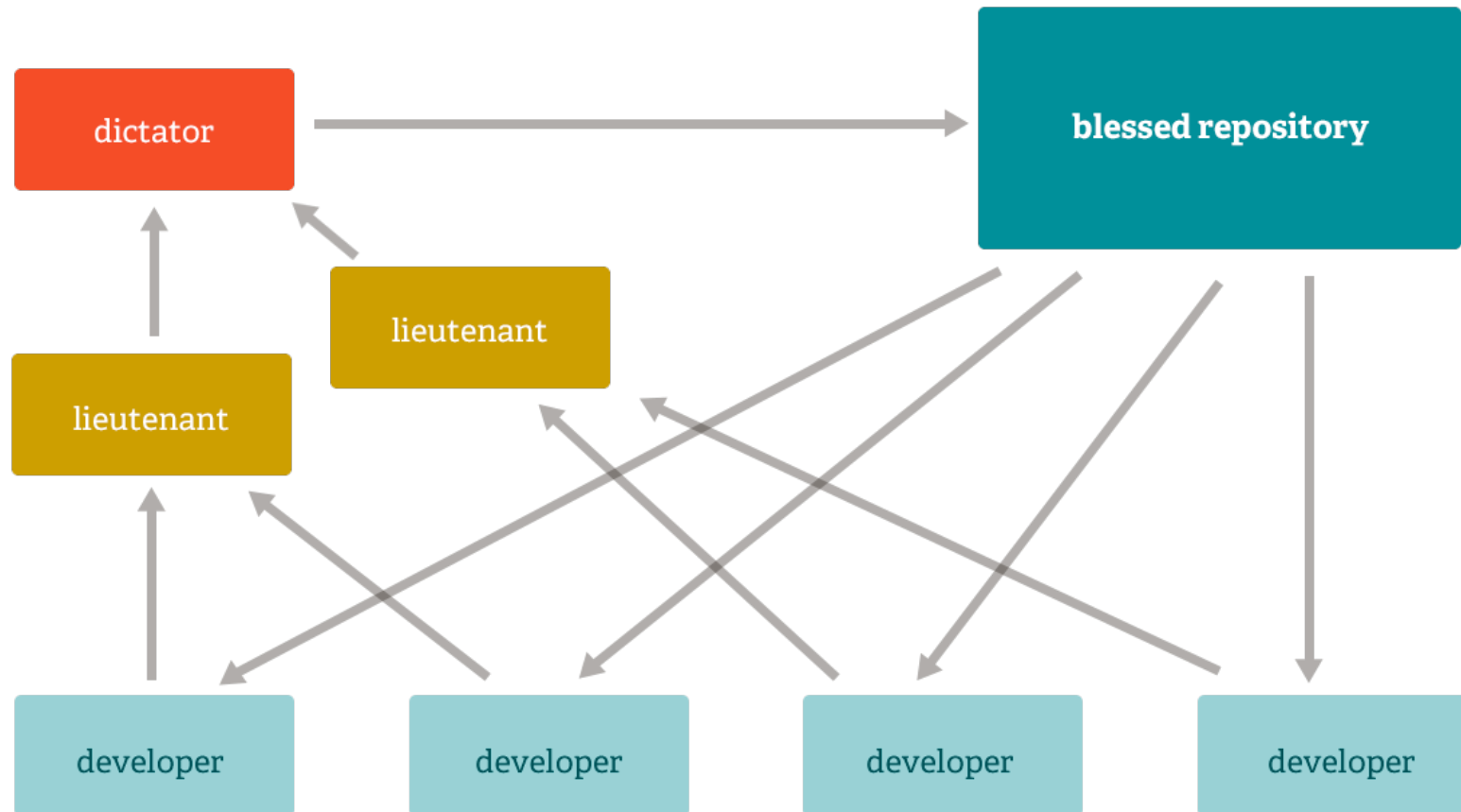
Плюсы и минусы данного подхода?

Integration Manager Workflow

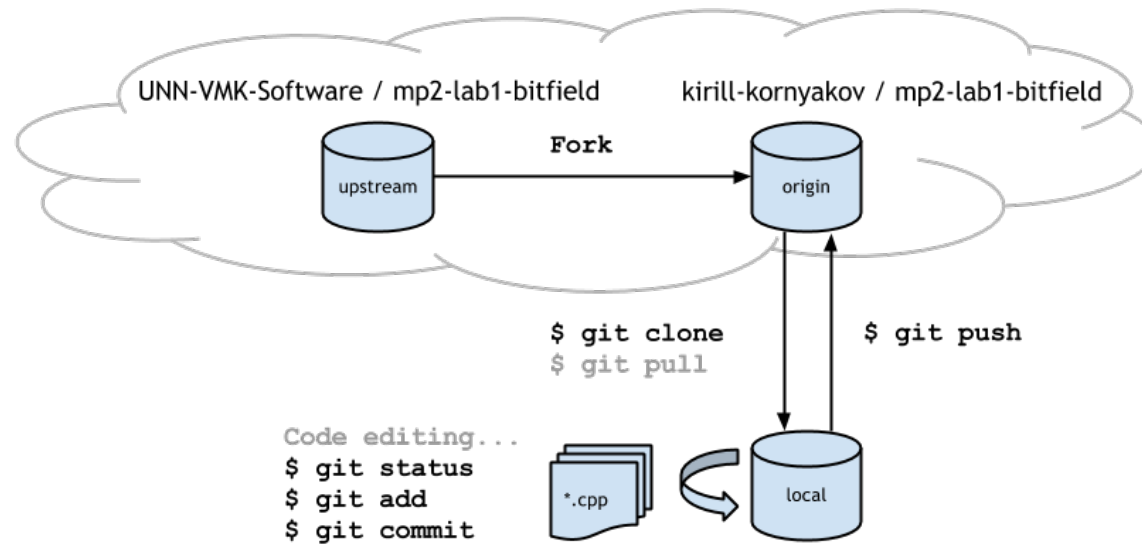


Плюсы и минусы данного подхода?

Dictator and Lieutenants Workflow



Triangular Workflow (GitHub)



```
$ cd mp2-lab1-bitfield
$ git remote -v
origin https://github.com/kirill-kornyakov/mp2-lab1-bitfield.git (fetch)
origin https://github.com/kirill-kornyakov/mp2-lab1-bitfield.git (push)
upstream https://github.com/UNN-VMK-Software/mp2-lab1-bitfield.git (fetch)
upstream https://github.com/UNN-VMK-Software/mp2-lab1-bitfield.git (push)
```

Инструкция по работе с Git / GitHub

1. Предварительные действия

1. Установка клиента Git (*git help*)
2. Конфигурация Git (*git config*)

2. Первичное получение исходных кодов

1. Регистрация на GitHub
2. Создание форка центрального репозитория
3. Клонирование форка к себе на локальную машину (*git clone*)
4. Конфигурация удаленного репозитория (*git remote -v*)
5. Просмотр истории (*git log*, *git show*)

3. Рабочий цикл (многократно)

1. Опциональное получение обновлений из удаленного сервера (*git pull*)
2. Создание новой ветки (*git branch*, *git checkout -b*)
3. Модификация файлов
4. Анализ изменений (*git status*, *git diff*)
5. Регистрация изменений (*git add*, *git commit*)
6. Отправка изменений на удаленный сервер (*git push*)

Резюме

1. Системы контроля версий — центральный инструмент разработки

- *Навигация по истории изменений*
- *Централизованный доступ*

2. Распределенные СКВ фактически стали стандартом. Их сильные стороны:

- *Допускают локальные коммиты (без наличия интернет или доступа к серверу)*
- *Упрощают слияние (а значит параллельную разработку)*
- *Дают максимальную свободу по организации рабочего процесса (workflow)*

3. Git не самая простая в освоении СКВ, однако очень функциональная, к тому же дает максимальную свободу по организации процесса разработки.

Контрольные вопросы

1. Определение СКВ.
2. Основные функции/возможности современных СКВ.

Ссылки

1. Wikipedia ["Системы контроля версий"](#).
2. [Pro Git](#) by Scott Chacon.
3. ["Mercurial tutorial"](#) by Joel Spolsky.

Спасибо!

Вопросы?

Демонстрация (если хватает времени)

- Создание новой ветки `git checkout -b`
- Добавление коммитов в нее `git commit`
- Сравнение с master: `git diff`
- Публикация на GitHub (`git push`)
- Вливание в master, удаление ветки
- `git log --graph`

Подумать

- Настройка работы с SSH-ключами