

Лекция 8: Работа в команде

Юрий Литвинов
y.litvinov@spbu.ru

23.04.2024

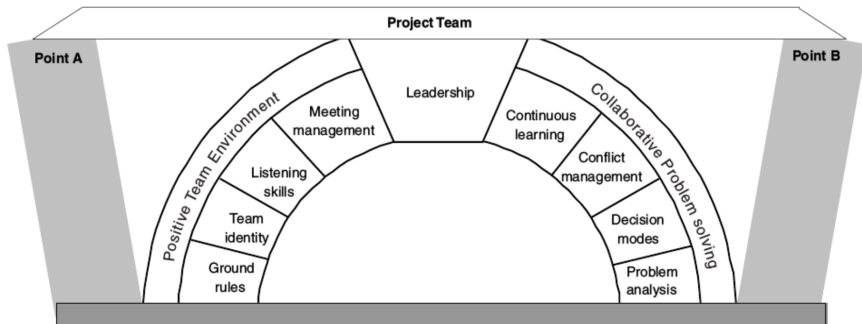
Что такое команда?

- ▶ Люди должны кооперироваться, чтобы выполнять свои задачи
- ▶ Продукт или сервис в целом не равен сумме отдельных его частей

Высокоэффективная команда — это сложно!

- ▶ Разный уровень работы сообща
- ▶ Разный профессиональный уровень
- ▶ Разный уровень культуры
- ▶ Разные темпераменты
- ▶ Разные подходы к решению задач
- ▶ Личные отношения
- ▶ ...

Составляющие успешной команды



Позитивная экосистема команды

- ▶ Набор базовых правил
- ▶ Сплочённость команды
- ▶ Умение слушать
- ▶ Умение проводить совещания

Базовые правила

- ▶ Позволяют структуризовать команду
- ▶ Определяют внутреннюю культуру
- ▶ Формируют ожидания членов команды

Пример списка правил

- ▶ Соблюдение конфиденциальности
- ▶ Поощрение обучения
- ▶ Взаимоуважение
- ▶ Дисциплина обязательств
- ▶ Проведение совещаний
 - ▶ Активное слушание
 - ▶ Ориентация на результат
 - ▶ Пунктуальность
 - ▶ Минимум отвлечений
 - ▶ Подготовленность

Сплочённость команды

- ▶ Понимание целей проекта членами команды
 - ▶ Понимание потребностей заказчика/пользователей
- ▶ Ощущение поддержки со стороны руководства
- ▶ Понимание сильных сторон и особенностей друг друга
- ▶ И не стоит забывать:
 - ▶ Собеседования
 - ▶ Регулярные встречи “вживую”
 - ▶ Тимбилдинг
 - ▶ Печеньки!

Умение слушать

- ▶ Активное слушание
 - ▶ Отсутствие отвлекающих факторов
 - ▶ Непредвзятость
 - ▶ Стил ь подачи материала
 - ▶ Фокусировка на процессе
- ▶ Отсутствие преждевременных суждений

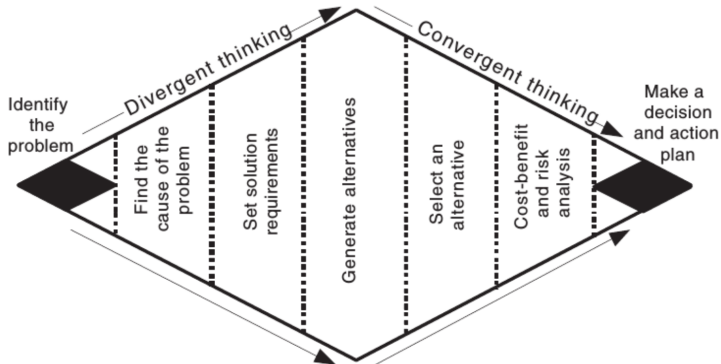
Умение проводить совещания

- ▶ До
 - ▶ Приглашение
 - ▶ Повестка
- ▶ Во время
 - ▶ Пунктуальность
 - ▶ Следование повестке
 - ▶ Наличие секретаря
 - ▶ Участие всех заинтересованных
- ▶ В конце
 - ▶ Резюме
 - ▶ Пунктуальность!!!
- ▶ После
 - ▶ Протокол (minutes)

Совместное решение задач

- ▶ Анализ задач
- ▶ Варианты принятия решений
- ▶ Разрешение конфликтов
- ▶ Непрерывное обучение

Анализ задач



Варианты принятия решений

- ▶ Консенсус
- ▶ Голосование
- ▶ Делегирование
- ▶ Автократия

Разрешение конфликтов

- ▶ Конфликты — это нормально!
 - ▶ Если без фанатизма
- ▶ Варианты решения
 - ▶ Предотвращение
 - ▶ Уход от конфликта
 - ▶ Сглаживание конфликта
 - ▶ Насаждение силой
 - ▶ Компромисс
 - ▶ Конфронтация конфликта
 - ▶ Признание конфликта
 - ▶ Определение контекста и целей
 - ▶ Определение и выбор альтернатив
 - ▶ План Б

Непрерывное обучение

- ▶ Улучшение культуры команды
- ▶ Вовлечение членов команды в процессы проекта
- ▶ Открытость и честность
- ▶ Регулярная переоценка и обратная связь
- ▶ Технический рост
- ▶ Пересмотр целей и планов проекта

Особенности формирования команды

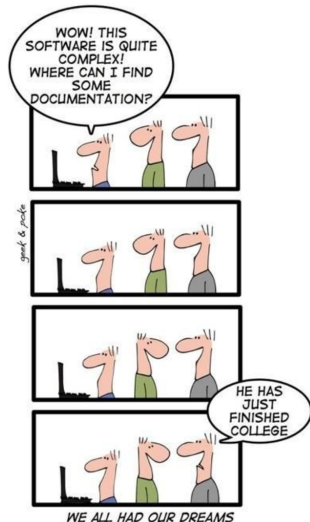
- ▶ Подбор подходящих людей
 - ▶ Суперзвёзды, простые смертные, новички
 - ▶ Различные темпераменты
 - ▶ Различные роли в команде
 - ▶ Личные характеристики
 - ▶ Регулировка численности команды
 - ▶ Различные специальности и компетенции
 - ▶ Не забыть про QA
- ▶ Подходящее помещение
 - ▶ Кабинеты, кубики, опенспейс
- ▶ Поддержание командного духа

Важные личные характеристики

- ▶ Внутренняя/внешняя управляемость
- ▶ Высокая/низкая мотивация
- ▶ Умение быть точным
- ▶ Исполнительность
- ▶ Терпимость к неопределенности
- ▶ Эгоизм
- ▶ Степень увлеченности
- ▶ Склонность к риску
- ▶ Самооценка
- ▶ Личные отношения в коллективе

Командная разработка ПО

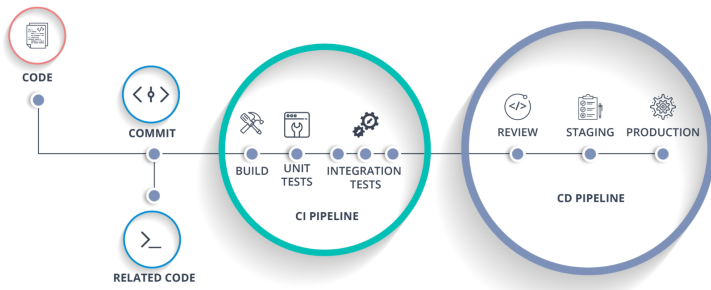
- ▶ Стиль оформления
- ▶ Тесты
- ▶ Документирование
- ▶ Версии библиотек и инструментов
 - ▶ Третьесторонние библиотеки
- ▶ Ревью
 - ▶ Design review
 - ▶ code review



Ревью кода

- ▶ Цель — сделать код лучше
 - ▶ Но не в ущерб проекту
- ▶ Просмотры кода
- ▶ Pre- и post-commit review
- ▶ Раунды инспекций
- ▶ Доносим изменение кода правильно
 - ▶ Работоспособность, размер, стиль, описание
- ▶ Доносим своё мнение правильно
 - ▶ Скорость, благожелательность,
 - ▶ Декларативность, приоритеты замечаний
- ▶ Мир костылей vs Перфекционизм

Работа с системой контроля версий



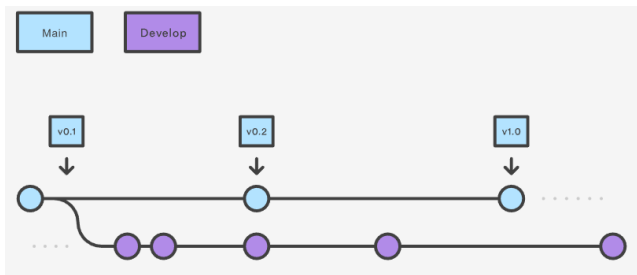
Модели ветвления

- ▶ Разработка в главной ветке
- ▶ Gitflow
- ▶ GitHub flow
- ▶ GitLab flow
- ▶ OneFlow

Разработка в главной ветке

- ▶ Избежание “merging hell”
- ▶ Инкрементальная разработка
- ▶ Частые коммиты (хотя бы раз в день)
 - ▶ Не каждый коммит может попасть в релиз
- ▶ Соккрытие неготовой функциональности
- ▶ Допускаются релизные или другие самостоятельные ветки

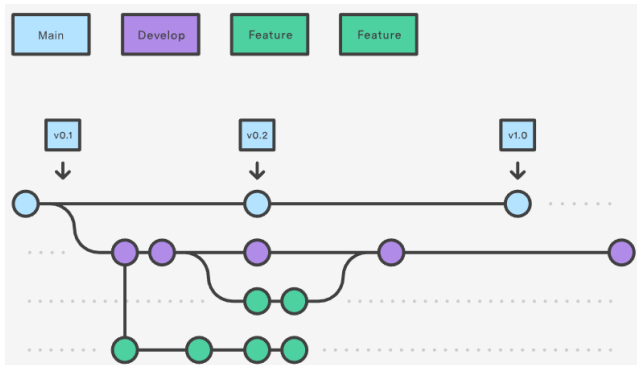
Gitflow, main и develop



© <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

- ▶ master — история релизов
- ▶ develop — история разработки

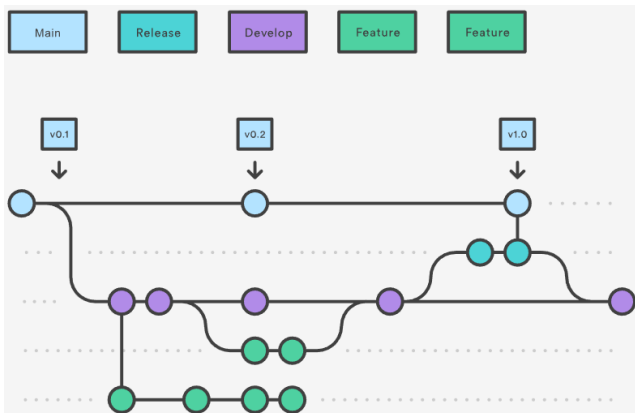
Gitflow, feature-ветки



© <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

- ▶ feature-ветки — отводятся от develop
- ▶ Для разработки функциональности и правки багов

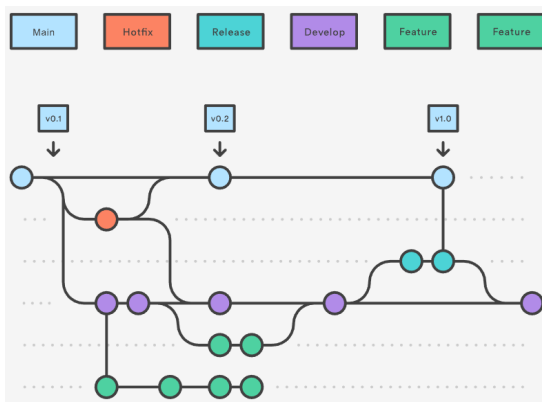
Gitflow, релизные ветки



© <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

- ▶ Релизные ветки — для исправления багов перед релизом
- ▶ Чтобы “заморозить” код

Gitflow, хотфиксы



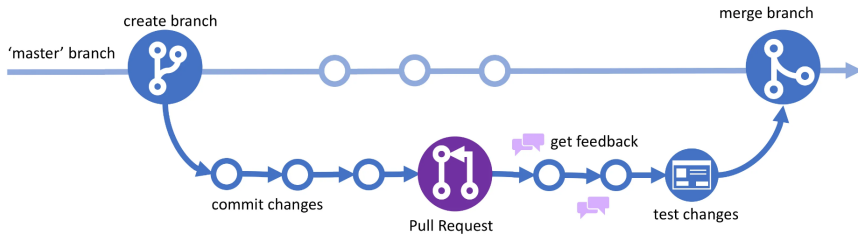
© <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

- ▶ Хотфикс-ветки — отводятся от main
- ▶ Для правки багов в уже выпущенном коде

Gitflow

- ▶ Хорошо подходит для поддержки больших проектов
- ▶ Сложное слияние изменений, не очень актуальный код в develop
- ▶ Трудночитаемая история
- ▶ Затрудняет CI/CD

GitHub flow



Copyright © 2018 Build Azure LLC

<http://buildazure.com>

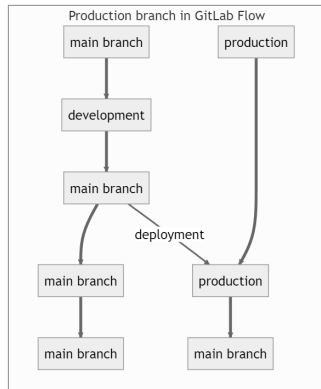
- ▶ Без develop
- ▶ Пуллреквесты для слияния
- ▶ Вариант с fork-репозиториями

Gitflow

- ▶ Легко сломать главную ветку
- ▶ Проще
- ▶ Проще CI/CD

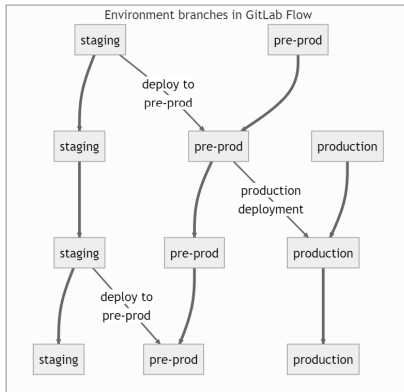
GitLab flow

- ▶ Упор на развёртывание
- ▶ Коммиты в главную ветку автоматически развёртываются в тестовом окружении
- ▶ Развёртывание в production — ручную



© https://docs.gitlab.com/ee/topics/gitlab_flow.html

GitLab flow, ветки окружения



© https://docs.gitlab.com/ee/topics/gitlab_flow.html

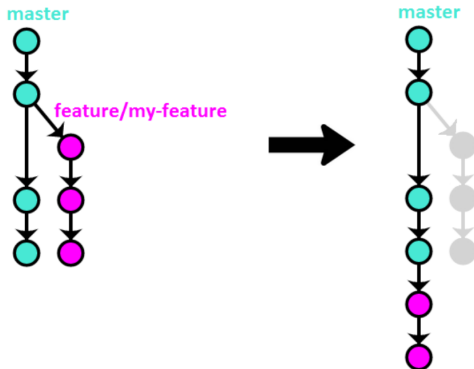
- ▶ Pre-prod и staging — два окружения для тестирования и развёртывания
- ▶ Могут быть релизные ветки

GitLab flow

- ▶ Продвинутый CI/CD в нескольких окружениях
- ▶ Не так удобно поддерживать несколько веток

OneFlow

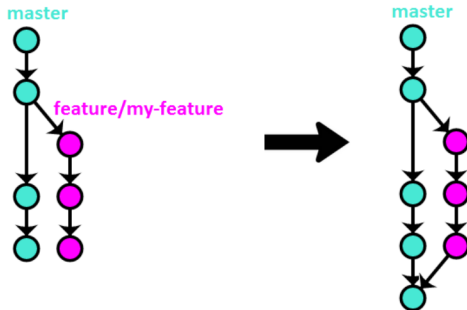
- ▶ Главная ветка, feature-ветки
- ▶ Три способа слияния
- ▶ Первый — rebase



© <https://www.endoflineblog.com/oneflow-a-git-branching-model-and-workflow>

OneFlow

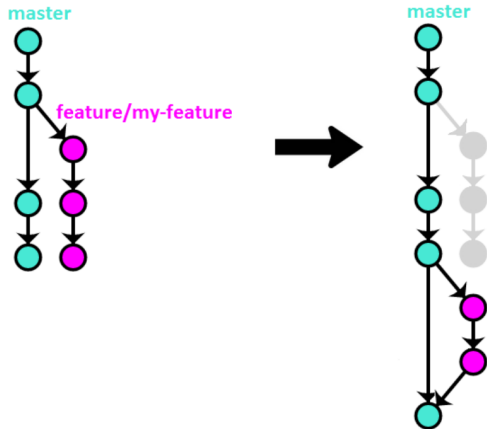
► Второй — merge



© <https://www.endoflineblog.com/oneflow-a-git-branching-model-and-workflow>

OneFlow

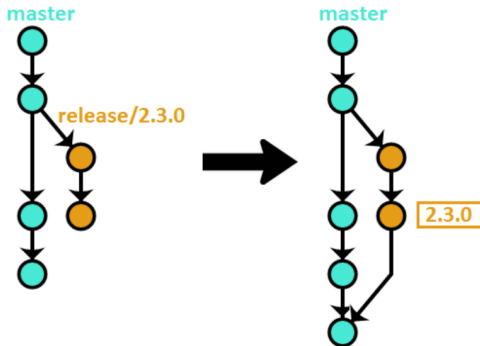
- ▶ Третий — rebase и merge



© <https://www.endoflineblog.com/oneflow-a-git-branching-model-and-workflow>

OneFlow, релизные ветки

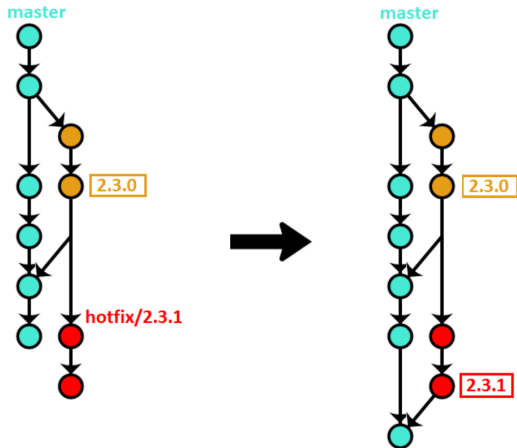
- ▶ Релизные ветки — от основной
- ▶ После релиза — тэг и слияние в основную



© <https://www.endoflineblog.com/oneflow-a-git-branching-model-and-workflow>

OneFlow, хотфиксы

- ▶ Хотфиксы — от релизной ветки
- ▶ После релиза — тэг и слияние в основную



© <https://www.endoflineblog.com/oneflow-a-git-branching-model-and-workflow>

OneFlow

- ▶ Прямолинейная история коммитов
- ▶ Нетривиальный CI/CD
- ▶ Сложно поддерживать несколько версий