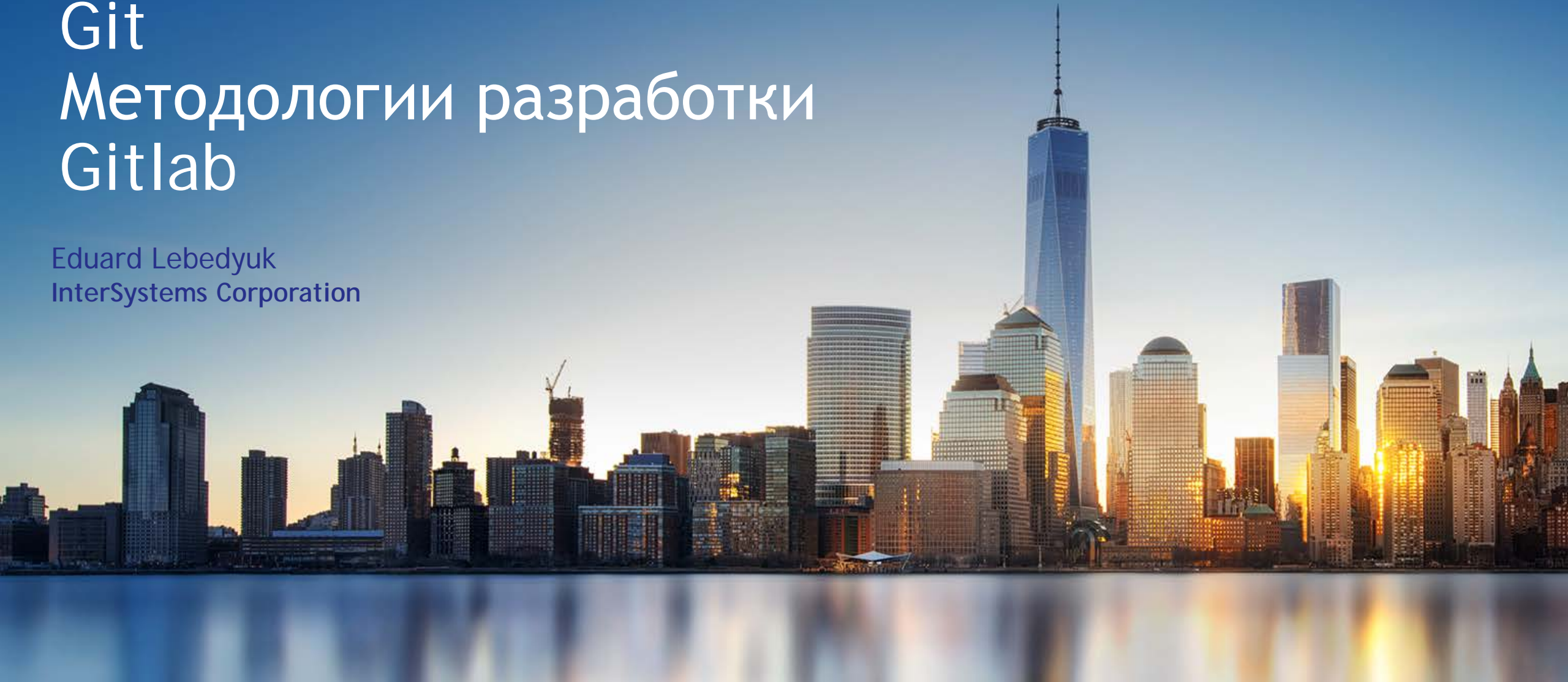


# Git Методологии разработки Gitlab

Eduard Lebedyuk  
InterSystems Corporation



# План

- Infrastructure as Code
- Git
- Методологии разработки
  - GitHub flow
  - GitLab flow
  - Git flow
- GitLab
- Демо



# Git

---

<https://git-scm.com/book/ru/v1/>

<https://habrahabr.ru/post/268951/>

# Введение

- Git — система контроля и управления версиями
- Цели создания:
  - Скорость
  - Простота
  - Нелинейность
  - Распределённость



# Repository (Репозиторий)

Repository — место, где хранятся и поддерживаются какие-либо данные

- “Физически” — папка в ОС
- Хранит файлы и папки
- Хранит историю их изменения

**Локальный репозиторий** — репозиторий, расположенный на локальном компьютере

**Удалённый репозиторий** — репозиторий, находящийся на удалённом сервере



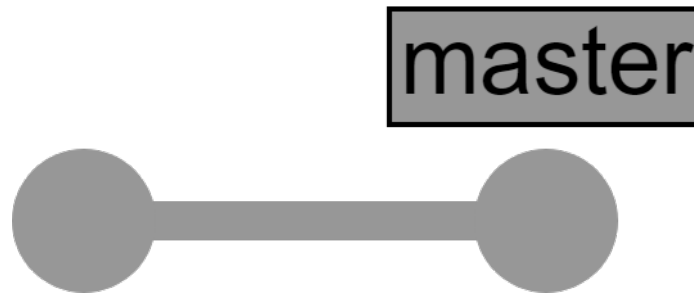
# Commit (Коммит)

- Зафиксированное состояние репозитория
- Diff (отличие) от какого-либо другого коммита который называется родительским
- Родителей бывает:
  - 0 – у первого коммита
  - 1 – как правило бывает именно так
  - 2 – для слияния изменений
  - >2 – для слияния изменений (но так делать не надо)



# Branch (Ветка)

- Указатель на коммит
- Всегда можно посмотреть на его историю до первого коммита



# Дерево коммитов

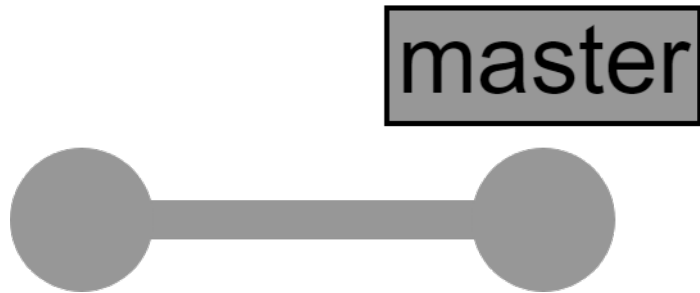
---

Простой вариант



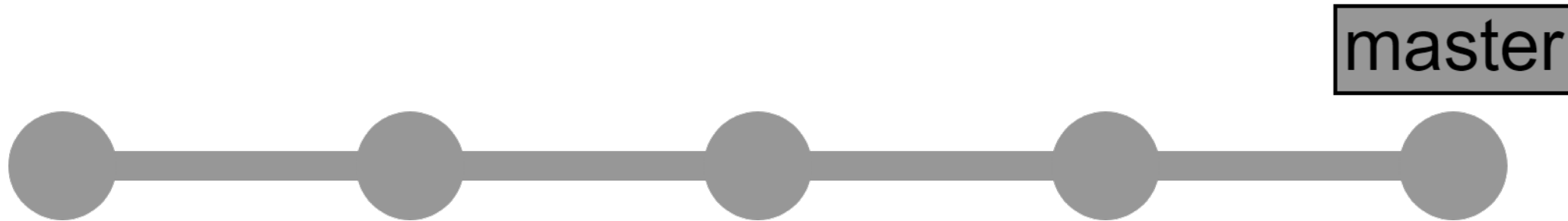
# Дерево коммитов

Самый простой вариант



# Дерево коммитов

Самый простой вариант



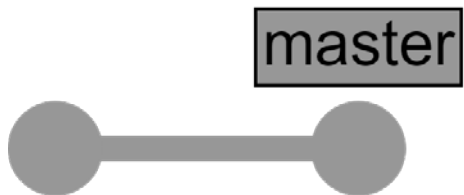
# Дерево коммитов

---

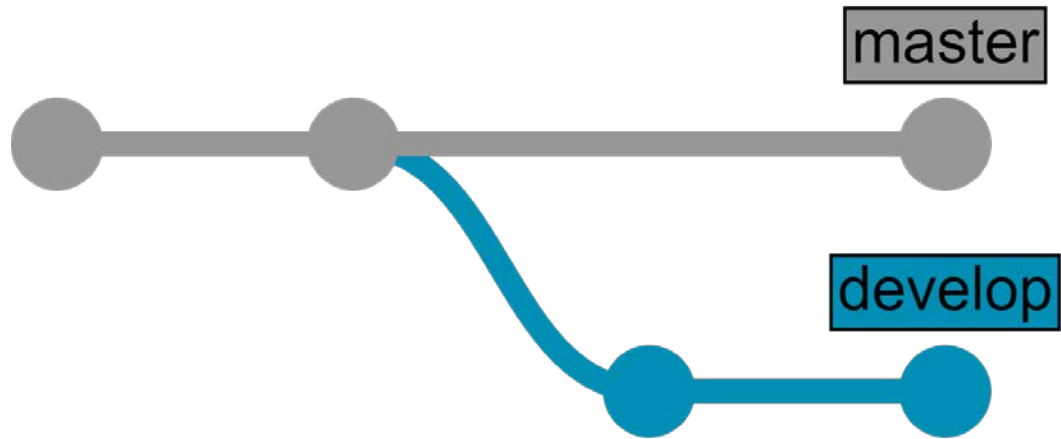
Сложнее

# Дерево коммитов

Сложнее



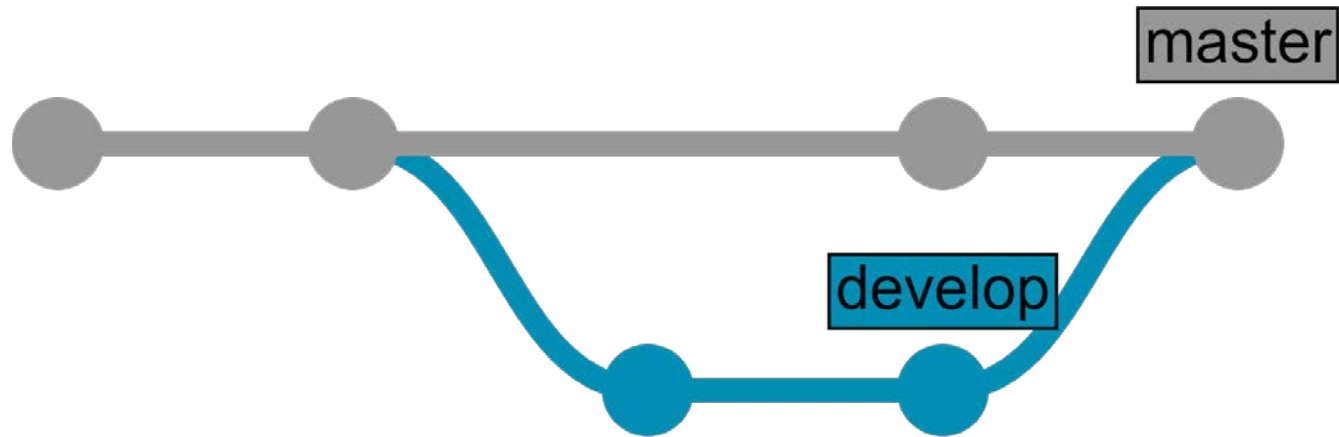
# Дерево коммитов



Pull request (Merge request) - запрос на слияние двух веток



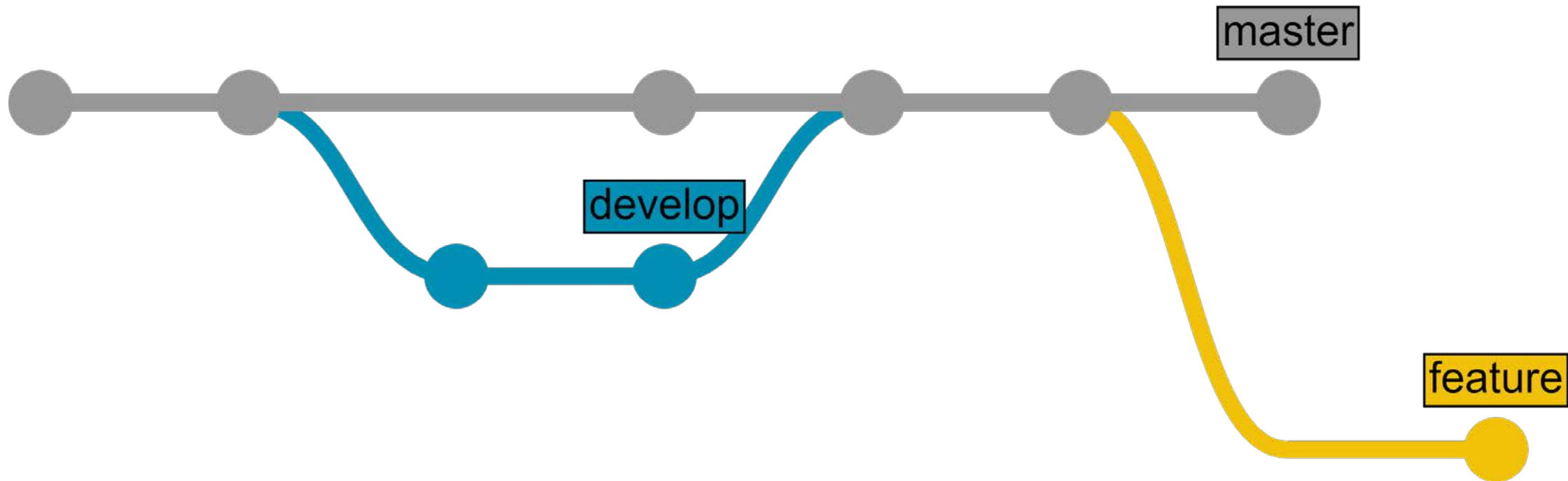
# Дерево коммитов



Результат успешного запроса на слияние — коммит с двумя родителями



# Дерево коммитов



# Git 101

- Git — система контроля и управления версиями
- Repository — место, где хранятся и поддерживаются какие-либо данные
- Commit — зафиксированное состояние репозитория
- Branch — указатель на коммит
- Pull request (Merge request) - запрос на слияние двух состояний репозитория





# Методологии разработки

---

# Методологии разработки

- Серия подходов к разработке программного обеспечения
- На основе Git реализовано множество методологий разработки
- Рассмотрим
  - GitHub flow
  - GitLab flow
  - Git flow



# GitHub flow

---

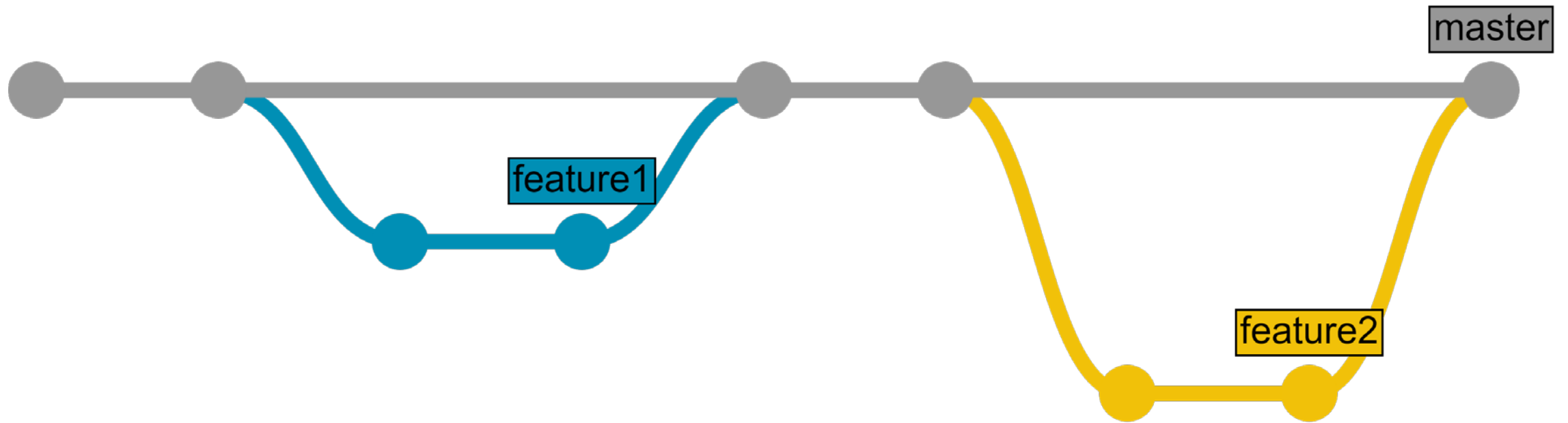
<https://habrahabr.ru/post/189046/>

# GitHub flow

- Содержимое ветви master всегда работоспособно
- Начиная работу над чем-то новым, ответвляйте от ветви master новую ветвь, имя которой соответствует её предназначению
- Когда вам понадобится отзыв, или помощь, или когда вы сочтёте ветвь готовой к слиянию, отправьте запрос на слияние (pull request)
- После того, как кто-то другой просмотрел и одобрил фичу, вы можете слить вашу ветвь в ветвь master.
- После того, как ветвь master пополнилась новым кодом, вы можете немедленно внедрить его в промышленное окружение



# GitHub flow



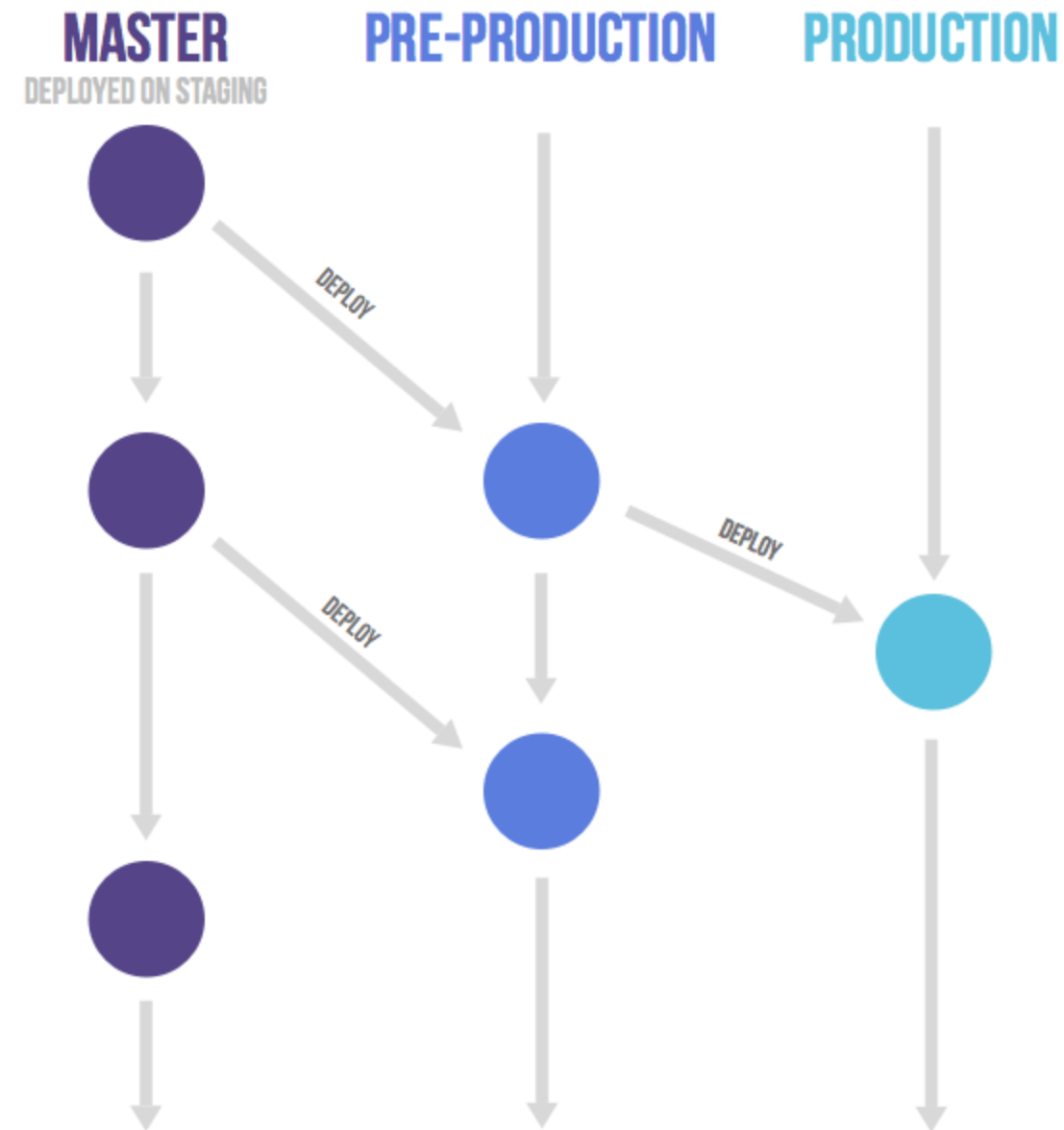
# GitLab flow

---

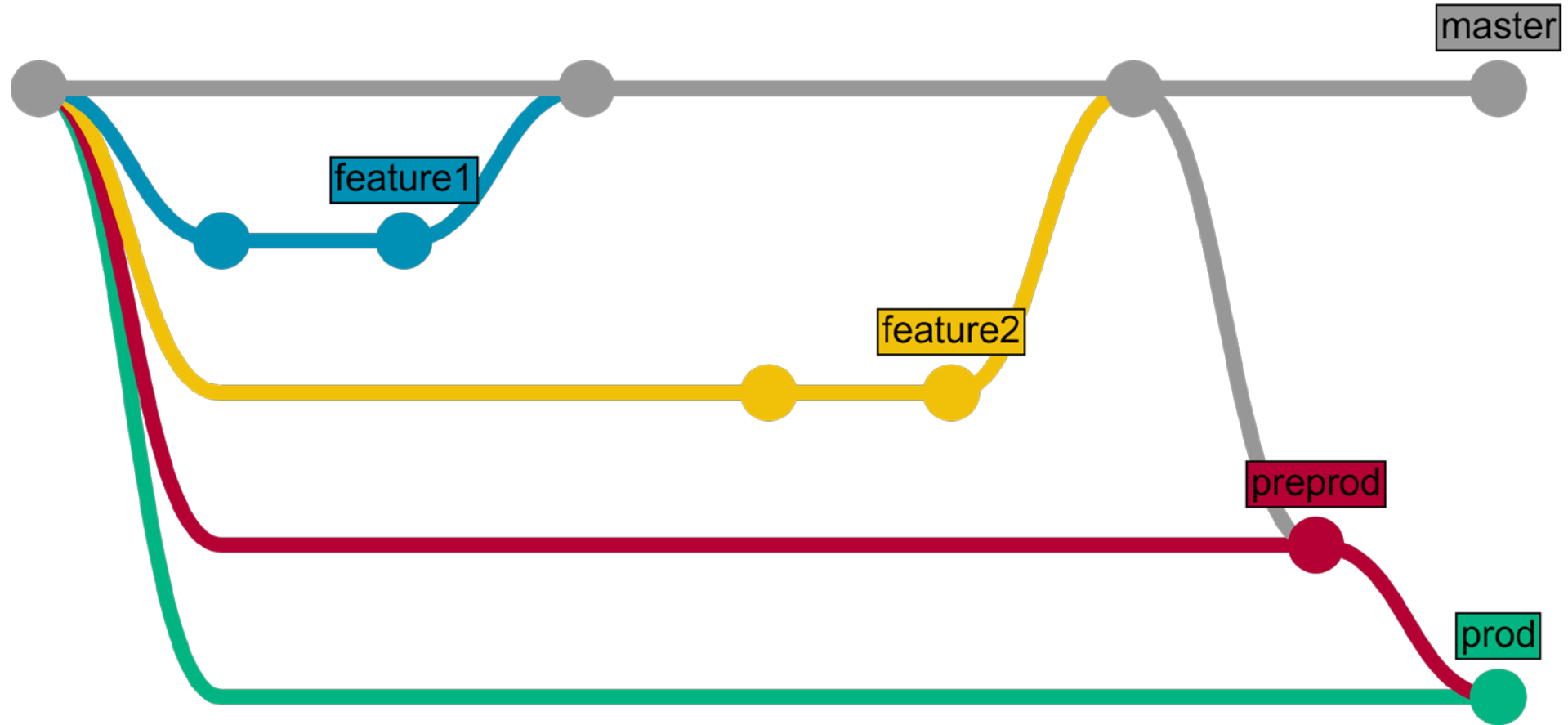
[https://docs.gitlab.com/ce/workflow/gitlab\\_flow.html](https://docs.gitlab.com/ce/workflow/gitlab_flow.html)

# GitLab flow

- GitHub flow + Environment branches (ветки окружений)
- Environment branch — ветка содержимое которой развёрнуто на сервере (или в контейнере)
  - Автоматически (при каждом новом коммите)
  - Полуавтоматически (по нажатию кнопки)
  - Вручную
- Разработка ведется в feature branches, аналогично GitHub Flow



# GitLab flow





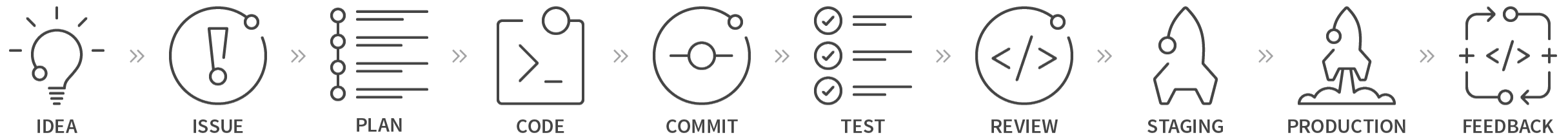
# GitLab

---

<https://about.gitlab.com/2016/10/25/gitlab-workflow-an-overview/>

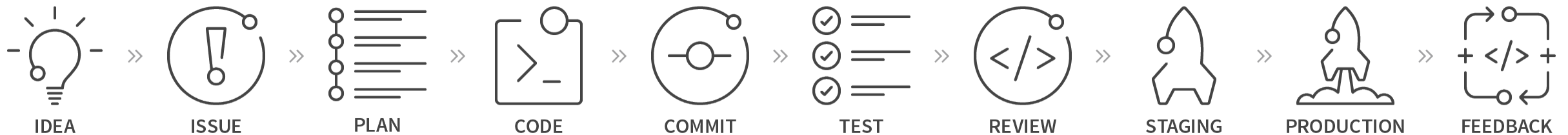
# GitLab Workflow

- Методология создания программного обеспечения, от идеи до продукта



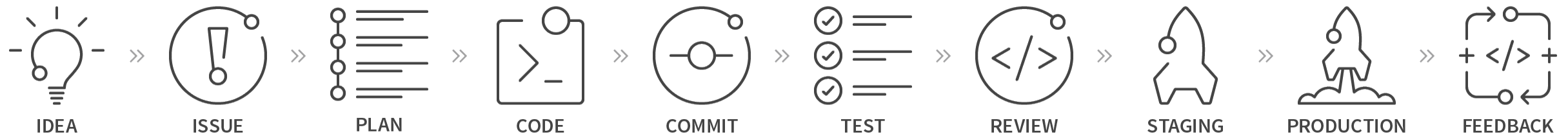
# Issue (Задача)

- Баг или новая функциональность
- Идёт обсуждение
- Коммиты связаны с Issue
- Для каждого Issue существует отдельная ветка



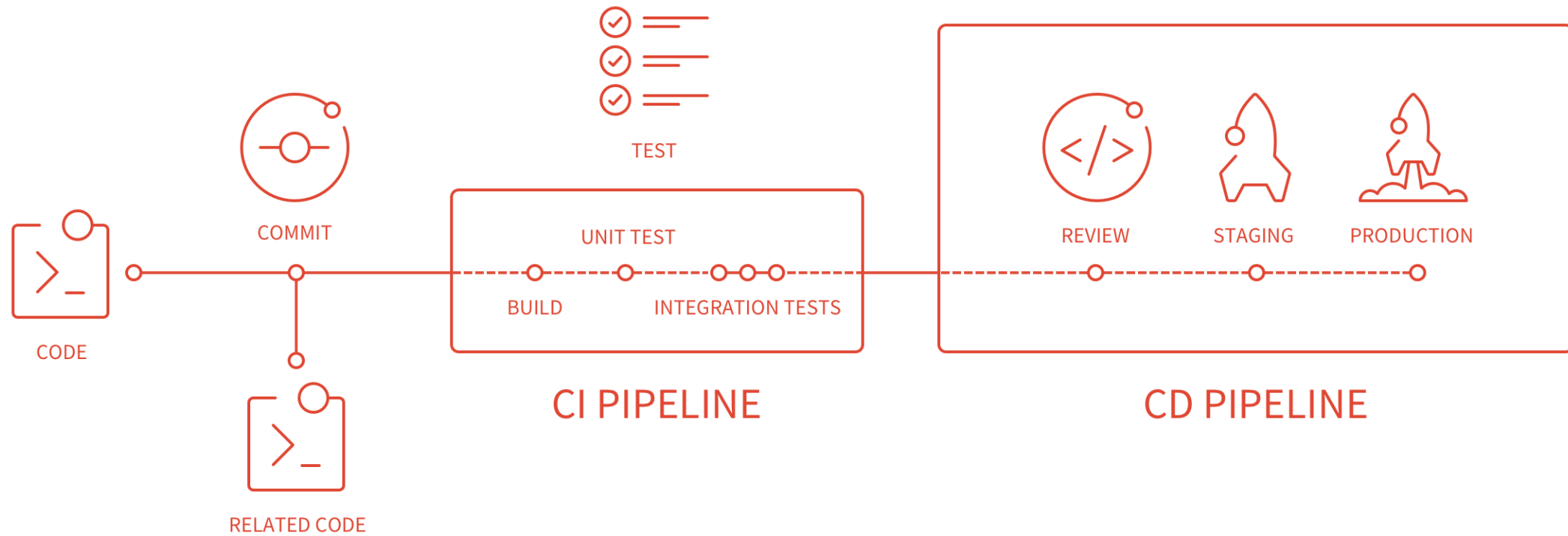
# Планирование

- Issue Board: Issue объединены в канбан доски
- Milestones: Issue объединены в спринты
- Для Issue настраивается:
  - Срок выполнения
  - Ответственный
  - Затраченное время



# Continuous Integration (Непрерывная интеграция)

## Continuous Delivery (Непрерывная доставка)



# Continuous Integration (Непрерывная интеграция)

## Continuous Delivery (Непрерывная доставка)

- Набор практик жизненного цикла программного обеспечения для:
  - Автоматической сборки
  - Автоматического тестирования
  - Автоматической доставки до окружений
- В GitLab — конфигурация CI/CD в формате YAML
  - Набор последовательных этапов (stages)
  - Каждый этап состоит из действий (scripts) которые выполняются параллельно



# Script (действие)

- Что делать
- Условие запуска
  - Вручную / автоматически
  - Только если предыдущие этапы окончились успешно / автоматически
  - Только в случае коммита в определённых ветках / для любого коммита
- В каком окружении (Environment) выполнить
- Что сохранить после выполнения (Artifacts)



# Runners (исполнители)

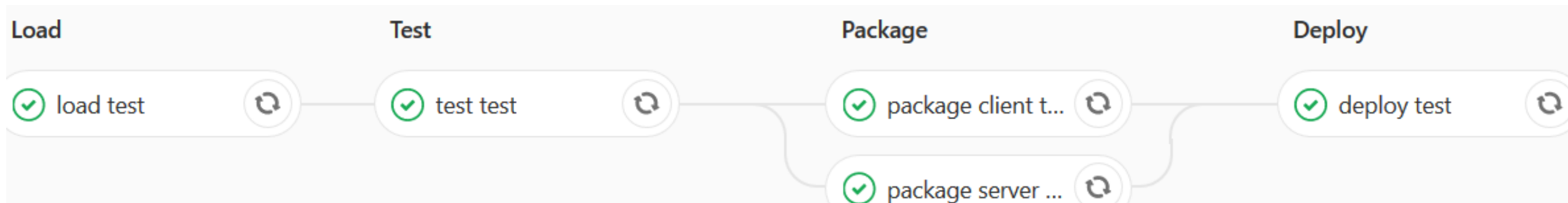
- Репозиторий связан с runners — службами которые выполняют действия
- Runner запускается в Environment (Окружении)
- Окружением может быть
  - Сервер
  - Контейнер
  - Локальный ПК

















# Pipeline (Запуск конфигурации)

- Конфигурация — все возможные действия
- Pipeline — один конкретный запуск конфигурации
- Pipeline
  - Какие действия были выполнены
  - Логи
  - Статус



# Environment (Окружение)

- Название
- Ссылка на окружение
- История выполненных pipeline
- Можно откатиться к любой точке истории

preprod	#153 by 	deploy preprod #340	 8526e43e  Merge branch 'master' into 'preprod'	about an hour ago	 Re-deploy
prod	#156 by 	deploy prod #343	 6651946d  Merge branch 'preprod' into 'prod'	about an hour ago	  Re-deploy
test	#158 by 	load test #345	 8074f95d Forgot script	about an hour ago	 Re-deploy



# Демо

---

<http://gitlab.eduard.win>

# Демо — окружения

Окружение	Ветка	Доставка	Кто может коммитить	Кто может сливать
Тестовое (~BASE)	master	Автоматически	Разработчики Владельцы	Разработчики Владельцы
Опытное (~UAT)	preprod	Автоматически	Никто	Владельцы
Промышленное (~LIVE)	prod	По нажатию кнопки	Никто	Владельцы








# Демо — план

1. Постановка задачи, разработка и автоматическое тестирование
  - a. Владелец ставит задачу
  - b. Разработчик создаёт feature-ветку и коммитит туда код решающий задачу
  - c. Разработчик сливает feature-ветку в master
  - d. Новый код автоматически доставляется в тестовое окружение и тестируется
2. Доставка в опытное окружение
  - a. Разработчик создаёт запрос на слияние кода master в preprod
  - b. Владелец одобряет слияние кода из master в preprod
  - c. Новый код автоматически доставляется в опытное окружение
3. Доставка в промышленное окружение
  - a. Разработчик (или владелец) создаёт запрос на слияние кода из preprod в prod
  - b. Владелец одобряет слияние кода preprod в prod
  - c. Владелец нажимает кнопку «Развернуть»
  - d. Новая версия приложения разворачивается в автоматическом окружении



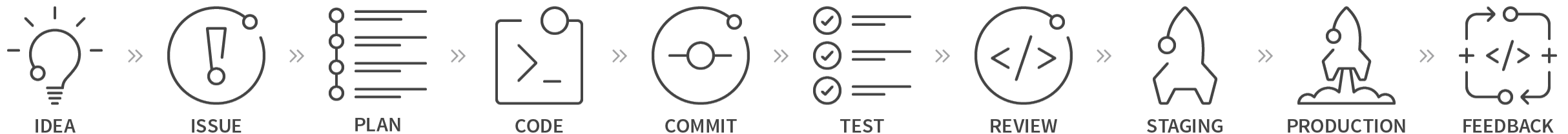
# Аналитика

Pipeline Health			
15 New Issues	24 Commits	5 Deploys	Last 30 days ▾
Stage ?	Median ?	Related Issues ?	Total Time ?
Issue	7 days	Time before an issue gets scheduled	
Plan	8 days	 Cycle Analytics: Consequatur dolor dolore voluptat... #15 · Opened 34 minutes ago by Dr. Catherine Schultz	7 days 12 hr
Code	8 days	 Cycle Analytics: Ut omnis suscipit optio animi nesciu... #14 · Opened 34 minutes ago by Administrator	7 days
Test	about 5 hours	 Cycle Analytics: Quaerat eos eius quo dolor ut et mol... #13 · Opened 34 minutes ago by Administrator	6 days 12 hr
Review	16 days	 Cycle Analytics: Quia assumenda temporibus ut omn... #12 · Opened 34 minutes ago by Jamal Hirthe	6 days
Staging	7 days	 Cycle Analytics: Veniam ex mollitia saepe nobis qui a... #11 · Opened 34 minutes ago by Curtis Franecki	5 days 12 hr
Production	about 1 month		



# Выводы

- Существует ряд методологий разработки основанных на Git
  - От простых до сложных
- GitLab позволяет организовать весь цикл разработки программного обеспечения
  - Организуйте процесс разработки и доставки так как нужно вам
  - Используйте GitLab.com или установите GitLab в своей сети



# Ссылки

- [Статья на Хабре](#)
- [Серия статей на Developer Community](#)
- [Тестовый репозиторий](#)
- [Репозиторий с кодом для GitLab](#)





The power behind what matters.



*Thank you.*

