Тема 4

Git и GitHub: как команда работает с кодом



История изменений, ветки и ревью: как в `main` попадает только проверенный код

План лекции

Сегодня

1. Как Git хранит историю
2. Индекс, коммиты, удалённый репозиторий
3. Ветки, слияние, конфликты
4. GitHub Flow, Pull Request и ревью
5. Релизы и секреты



После лекции вы сможете

- Работать с ветками и разрешать конфликты
- Вести задачу через issue и Pull Request
- Делать ревью кода по делу
- Защитить `main` и выпустить релиз

Зачем контроль версий

История

Кто, когда и зачем поменял строку

Откат

Вернуться к рабочей версии за минуту

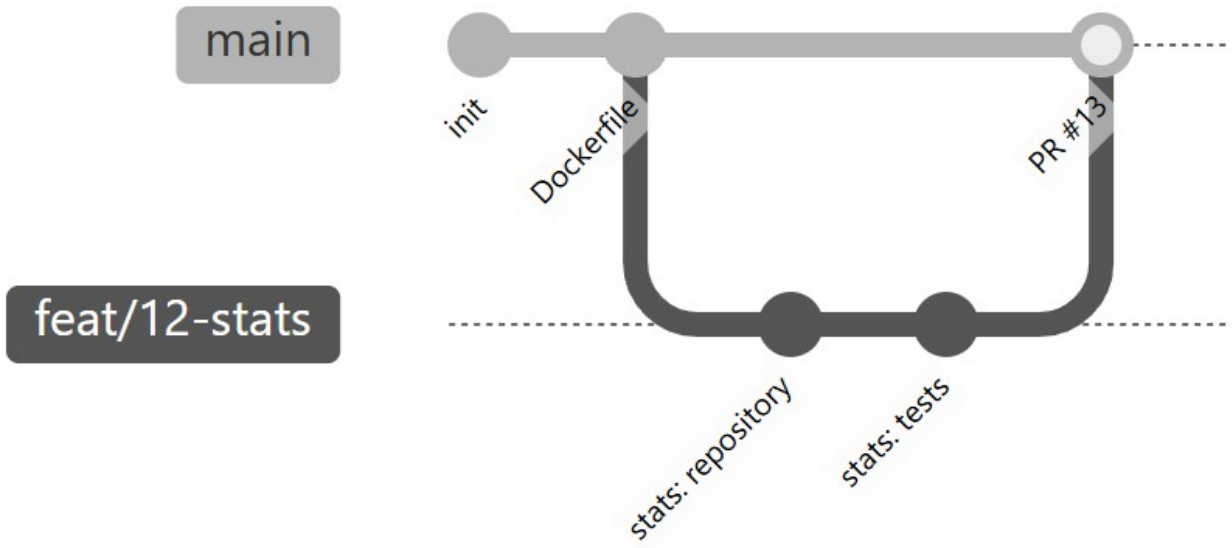
Команда

Работать параллельно и не затирать друг друга

Автоматизация

Каждый push запускает тесты и деплой (тема 5)

Коммиты и ветки



- Коммит — снимок всех файлов + автор, дата, сообщение, родитель
- Хэш — уникальный id коммита: `a1b2c3d`
- Ветка — подвижный указатель на коммит
- `HEAD` — где вы сейчас

Три области Git

Рабочая папка

Правите файлы. `git status`, `git diff`

Индекс

`git add` — «это пойдёт в коммит»

Репозиторий

`git commit` — новый снимок в истории

`.gitignore`: `bin/`, `obj/`, `.vs/`, `.env` — пароль в коммите остаётся в истории навсегда

Удалённый репозиторий: origin

Команда	Что делает
`git clone <url>`	Скачать репозиторий целиком
`git fetch`	Узнать, что нового в origin, не трогая свою работу
`git pull`	`fetch` + влить изменения в текущую ветку
`git push`	Отправить свои коммиты в origin

У каждого разработчика полная копия. Общая — на GitHub

Слияние и конфликты


```
<<<<<<< HEAD
# Сервис задач
=======
# Task Service API
>>>>>>> origin/main
```

1. **Merge**

   Объединяет ветки: fast-forward или коммит слияния

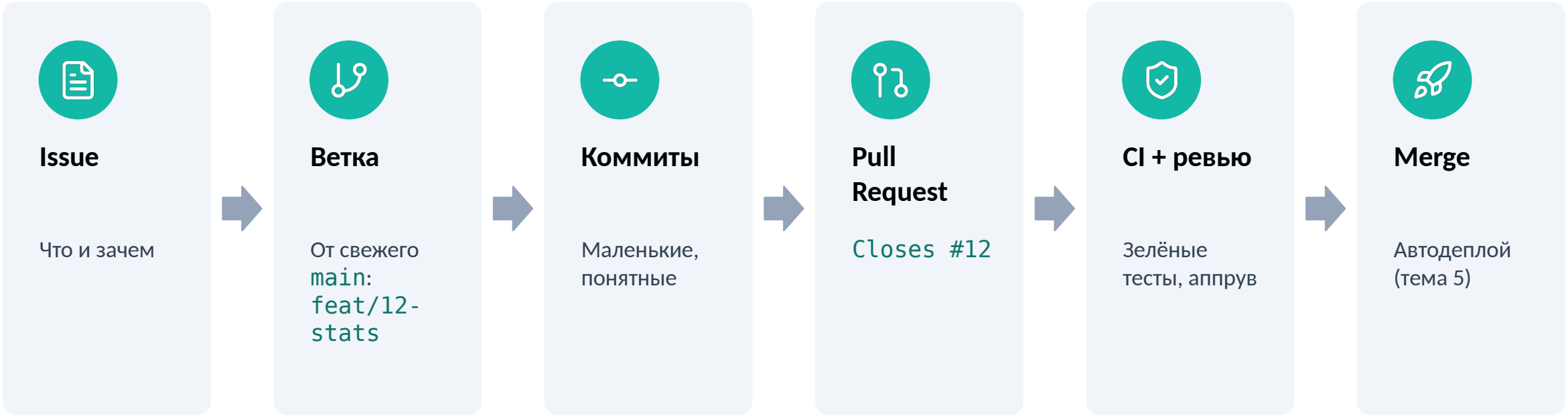
2. **Rebase**

   Переписывает ваши коммиты поверх свежего main. Общие ветки не ребейзим

3. **Конфликт**

   Одна строка изменена по-разному. Оставьте нужное, уберите маркеры, `add` + `commit`

GitHub Flow



В `main` попадает только проверенный и просмотренный код

Сообщения коммитов: Conventional Commits


```
feat: статистика задач проекта
fix: 404 вместо 500 для несуществующего
проекта
ci: публиковать образ в GHCR
docs: инструкция запуска
test: переход Todo → Done запрещён
```

1. **Префикс**

   Видно, что меняет поведение

2. **История**

   Читается как список изменений

3. **Автоматизация**

   По префиксам собирают release notes

Ревью кода: что смотреть

Что	Пример в нашем проекте
Слои	Нет SQL в контроллере, правила — в сервисе
Оба хранилища	Новый метод есть и в памяти, и в PostgreSQL
Тесты	Тест на успех и на ошибку
Безопасность	Параметры в SQL, нет паролей и токенов
Эксплуатация	Health checks, логи, настройки через окружение

Ревьюер ищет проблемы, а не опечатки

Инструменты GitHub

Issues и Projects

Задачи на доске; закрытые уезжают в Done сами

Rulesets

Запрет push в main, обязательные PR, аппрувы, CI

PR template

Чек-лист автора в каждом PR

CODEOWNERS

Ревьюеры назначаются автоматически

Dependabot

Сам присылает PR с обновлениями пакетов и образов

Releases

Тег + заметки о выпуске

Версии: SemVer

MAJOR

Ломающие изменения API: удалили поле из ответа

MINOR

Новая функциональность без поломок: добавили `/stats`

PATCH

Исправления без изменения API

Тег `v1.2.3` → релиз на GitHub → образ `1.2.3` в реестре (тема 5)



Утёк секрет — ротируйте его.
Удалить коммит — не поможет.

История уже скопирована. Выпустите новый пароль или токен, отзовите старый. Секреты для CI — в Settings → Secrets

Типичные ошибки

- Коммиты прямо в `main`
- PR на 2000 строк: его не ревьюят, а «аппрувят»
- Merge при красном CI
- Сообщения «fix», «wip», «asdf»
- Конфликт решён выбором «всё моё» без чтения
- `git add -A` не глядя — и в репозитории `.env`

Проверьте себя

1. Что такое ветка для Git и почему её создать «дёшево»?
2. Зачем индекс, если можно коммитить всё сразу?
3. Чем `git revert` лучше `git reset` для `main`?
4. Какие правила вы поставите на `main` и почему?
5. Утёк токен. Порядок действий?



Практика

Лаба 4. Репозиторий сервиса и работа через PR

Работаем в парах: автор и ревьюер

- Репозиторий на GitHub и защита `main`
- README через issue и PR с ревью
- Конфликт с партнёром и его разрешение
- `PUT /api/projects/{id}` через PR и релиз `v0.1.0`