Описание git-flow для проекта

Содержание

Список иллюстраций

Список таблиц

В проекте используется `gitflow-avh` https://github.com/petervanderdoes/gitflow-avh.

1 Установка git-flow

- Gentoo:

```
emerge git-flow
```

2 Инициализация git-flow

Для начала использования `git-flow` проинициализируйте его внутри существующего git-репозитория:

```
git flow init
```

3 Шаги git-flow avh

3.1 Фичи

Разработка новых фич для последующих релизов. Обычно присутствует только в репозиториях разработчиков.

3.1.1 Начало новой фичи

Разработка новых фич начинается из ветки `develop`.

Для начала разработки фичи выполните:

```
git flow feature start MYFEATURE
```

Это действие создаёт новую ветку фичи, основанную на ветке «develop», и переключается на неё.

3.1.2 Завершение фичи

Окончание разработки фичи. Это действие выполняется так:

- Слияние ветки MYFEATURE в `develop`.
- Удаление ветки фичи.
- Переключение обратно на ветку `develop`.

```
git flow feature finish MYFEATURE
```

3.1.3 Публикация фичи

Если вы разрабатываете фичу совместно с коллегами, то опубликуйте фичу на удалённом сервере, чтобы её могли использовать другие пользователи.

```
git flow feature publish MYFEATURE
```

3.1.4 Получение опубликованной фичи

Получение фичи, опубликованной другим пользователем.

```
git flow feature pull origin MYFEATURE
```

Вы можете отслеживать фичу в репозитории origin с помощью команды

```
git flow feature track MYFEATURE
```

3.2 Создание релиза

Поддержка подготовки нового релиза продукта. Позволяет устранять мелкие баги и подготавливать различные метаданные для релиза.

3.2.1 Начало релиза

Для начала работы над релизом используйте команду

```
git flow release start RELEASE [BASE]
```

Она создаёт ветку релиза, ответляя от ветки `develop`.

При желании вы можете указать [BASE]-коммит в виде его хеша sha-1, чтобы начать релиз с него. Этот коммит должен принадлежать ветке `develop`.

3.2.2 Публикация релиза

Желательно сразу публиковать ветку релиза после создания, чтобы позволить другим разработчиками выполнять коммиты в ветку релиза. Это делается так же, как и при публикации фичи, с помощью команды:

```
git flow release publish RELEASE
```

3.2.3 Отслеживание опубликованного релиза

Вы также можете отслеживать удалённый релиз с помощью команды

```
git flow release track RELEASE
```

3.2.4 Завершение релиза

Завершение релиза – один из самых больших шагов в git-ветвлени. При этом происходит несколько действий:

- Ветка релиза сливается в ветку `master`.
- Релиз помечается тегом равным его имени.
- Ветка релиза сливается обратно в ветку `develop`.
- Ветка релиза удаляется.

```
git flow release finish RELEASE
```

Не забудьте отправить изменения в тегах с помощью команды

```
git push --tags
```

3.3 Исправления

Исправления нужны в том случае, когда нужно незамедлительно устранить нежелательное состояние продакшн-версии продукта. Может ответвляться от соответствующего тега на ветке «master», который отмечает выпуск продакшн-версии.

3.3.1 Начало исправления

Как и в случае с другими командами git flow, работа над исправлением начинается так:

```
git flow hotfix start VERSION [BASENAME]
```

Аргумент VERSION определяет имя нового, исправленного релиза.

При желании можно указать BASENAME-коммит, от которого произойдёт ответвление.

3.3.2 Завершение исправления

Когда исправление готово, оно сливается обратно в ветки `develop` и `master`. Кроме того, коммит в ветке `master` помечается тегом с версией исправления.

```
git flow hotfix finish VERSION
```