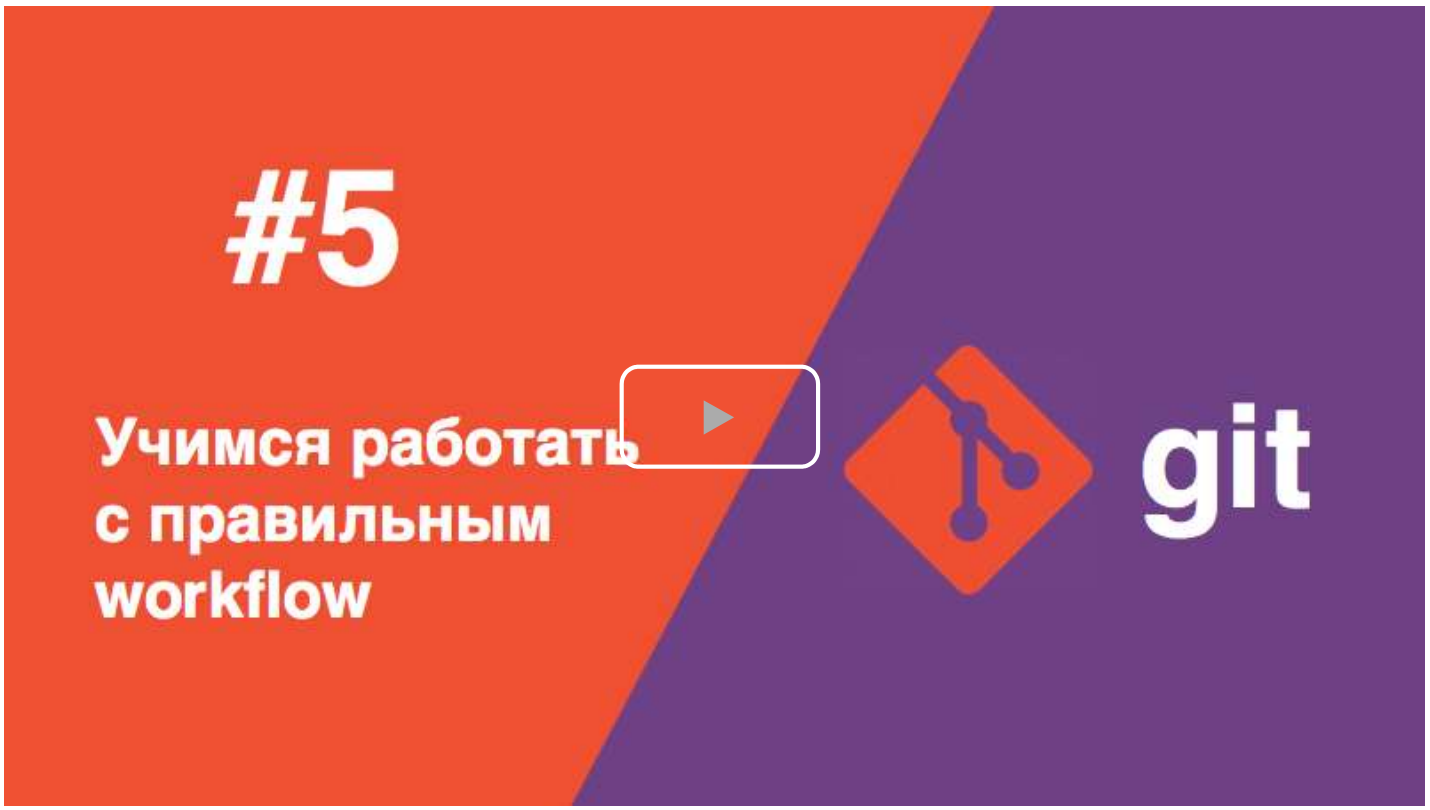
# #5 Git - учимся работать с правильным workflow

Git

Git для начинающих




Предыдущий урок Следующий урок

В этом уроке мы разберем правильный workflow с git. Научимся создавать релизы, фичи и хотфиксы.

**Понравилось?**
**Поделитесь с друзьями!**

Facebook

Twitter

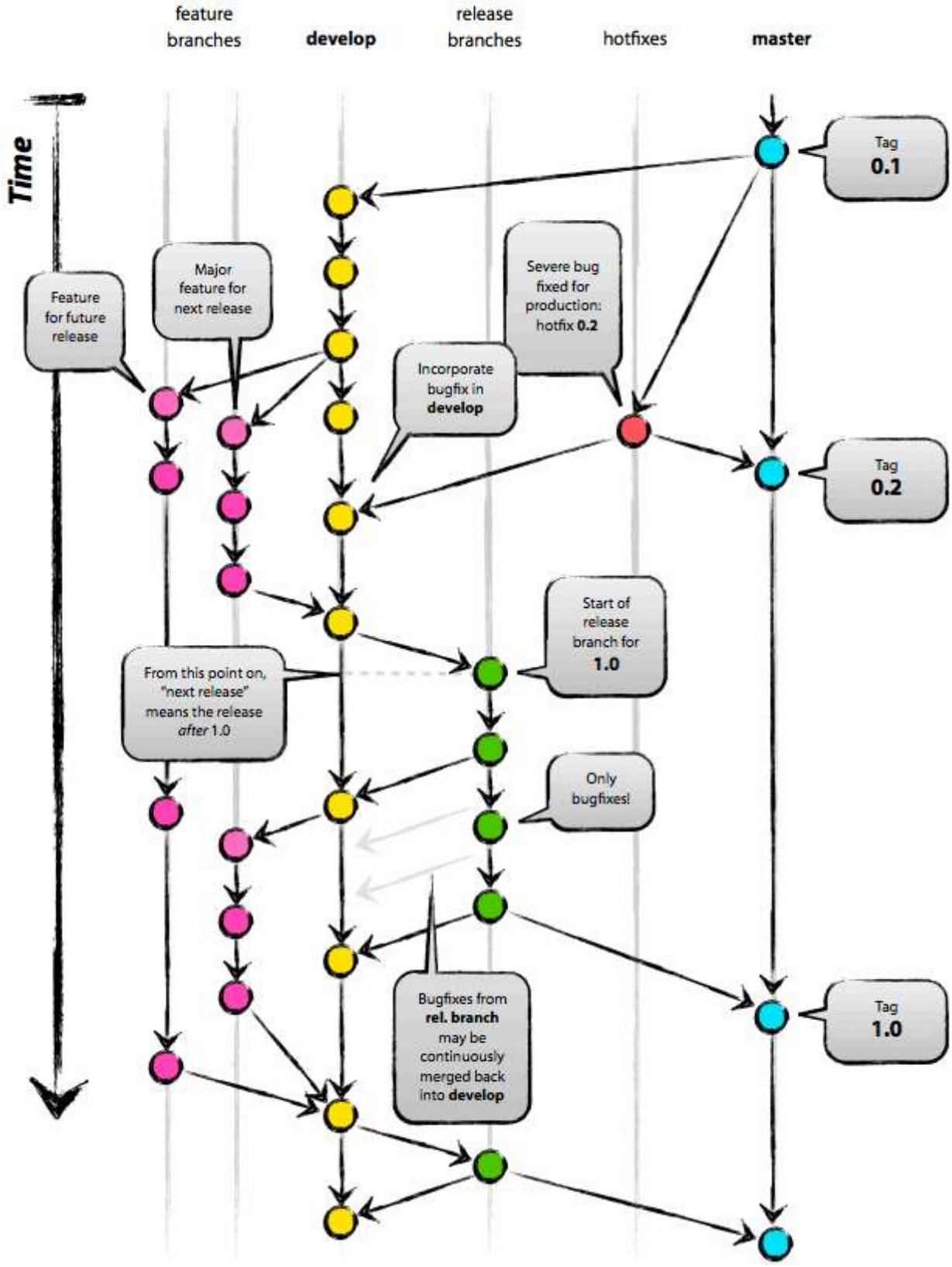
Вконтакте

Google+

Комментарии Текст видео

Всем привет. Мы продолжаем изучать гит и сегодня у нас тема workflow в гит. Мы с вами разберем как несколько человек обычно работают в проекте и как это выглядит, чтобы никто ничего не поломал.

На картинке выглядит это приблизительно вот так.




У нас есть несколько бренчей с кооторыми мы работаем. Вот справа у нас есть master branch. Это ветка которую выливают в продакшен. Она самая стабильная. Так же у нас есть ветка develop. Это основная ветка разработки. Идея этого флоу такая:

Программист должен реализовать новую фичу в проекте - например авторизацию

Он находится на develop ветке и создает новую ветку с нее под эту фичу. Т.е. он ответвляется от ветки develop и создает фиче ветку. Он никогда не делает изменения напрямую на develop ветке. Он всегда ветвится с нее, чтобы сделать что-то новое.

Когда фича сделана и ее проверили на качество кода другие программисты, а потом ее проверили тестировщики - она сливается обратно в develop

В определенный момент, когда программисты сделали несколько новых фич, и менеджер хочет вылить это в продакшен создается release ветка. Релиз ветка создается с ветки develop и потом тестируется отдельно.

Если в релизе все баги пофикшены, релиз вливается в мастер

Хотфикс это быстрый фикш, который, например, когда все поломалось на мастере и который нужно сделать ну вот прям сейчас. Для этого программист создает новую ветку с мастера, фиксит там баг, ветку тестируют и вливает одновременно в develop и master так как изменения должны быть на обоих ветках.

Звучит сложно, поэтому давайте пробовать работать по этому флоу.

Сейчас у нас есть только master ветка. Считаем что это наш стабильный релиз на данный момент. Давайте создадим develop ветку с master ветки.

Раньше мы использовали код для создания ветки и перехода на нее

```
git branch -b name
git checkout name
```

Но в гите существует короткая форма записи, которая сразу создает ветку и переходит на нее

```
git checkout -b develop
```

Этим самым мы создали develop ветку и перешли на него. Давайте запушим ее в репозиторий

```
git push
```

Теперь давайте создадим фичу для для авторизации

```
git checkout -b feature/implementing-auth
```

Как вы видите я назвал ветку начиная с слова feature. Это хорошая практика, чтобы из названия сразу было видно где фича, где баг, а где релиз.

Теперь давайте создадим файл auth.js и добавим туда функцию авторизации

```
function authenticate (login, password) {
  if (login === 'login' && password === 'password') {
    return 'You were logged in';
  } else {
    return 'Login is incorrect';
  }
}
```

Теперь считаем, что фичу мы закончили. Поэтому давайте закоммитим ветку

```
git add .
git commit -m "Added authentication"
git push
```

И мы залили нашу ветку на гитхаб.

Теперь давайте сольем нашу фичу обратно в develop. Для этого переходим на ветку develop.

```
git checkout develop
```

и мерджим нашу фичу в develop.

```
git merge feature/implementing-auth
```

Мы видим что ветка обновилась и в git log мы увидим наш коммит Implemented auth. Теперь пушим develop опять в репу, а ветку на которой мы делали фичу впринципе можно удалять.

```
git push
```

Теперь представим, что наш менеджер хочет вылить то, что мы сделали в продакшен. Мы должны создать релиз ветку, протестировать что все работает, и слить в мастер.

Обычно все релизы именуются с правильной версионностью, но об этом мы поговорим в другом уроке.

```
git checkout -b release/1.0
```

Этим самым мы создаем релиз с версией 1.0 и переходим на эту ветку.

Так как мы никаких изменений делать не будем, то мы можем сливать эту ветку сразу в master.

Поэтому переходим на мастер и мерджим релиз в мастер

```
git checkout master
git merge release/1.0
git push
```

Вы спросить почему такой сложный флоу.

> Он проверен многолетней практикой работы многих команд и компаний и он отлично работает
>
> Когда вы создаете фичи, вы отделяете новый функционал от основной ветки и этим самым упрощаете его проверку и вы можете в любой момент его померджить либо сделать это позже.
>
> Создание релизов уберегает от вливания непроверенного кода в мастер, потому что если вы будете вливать в мастер из девелопа, то кто-то может успеть влить новый коммит в develop, вы смерджите это в мастер и все отломается.
>
> Ну и в целом флоу дает гибкость не поломать продакшен при новых изменениях и процес разработки становится более приятным.

Видео
Контакты
RSS