# Що таке GitFlow?

# Що таке GitFlow?

**GitFlow** це розгалужена модель для Git, створена Вінсентом Дріссеном. Він привернув багато уваги, оскільки дуже добре підходить для співпраці та масштабування команди розробників.

# Ключові переваги GitFlow

- **Паралельний розвиток**

Однією з чудових переваг GitFlow є те, що він робить паралельну розробку дуже легкою, ізолюючи нову розробку від завершеної роботи. Нова розробка (наприклад, функції та ненадзвичайні виправлення помилок) виконується в гілках функцій і об’єднується в основну частину коду лише тоді, коли розробник (и) задоволений, що код готовий до випуску.

- **Співпраця**

Гілки функцій також спрощують співпрацю двох або більше розробників над однією функцією, оскільки кожна гілка функції є пісочницею, де єдині зміни – це зміни, необхідні для роботи нової функції. Завдяки цьому дуже легко бачити та стежити за тим, що робить кожен співавтор.

# Ключові переваги GitFlow

- **Звільніть сценічну зону**

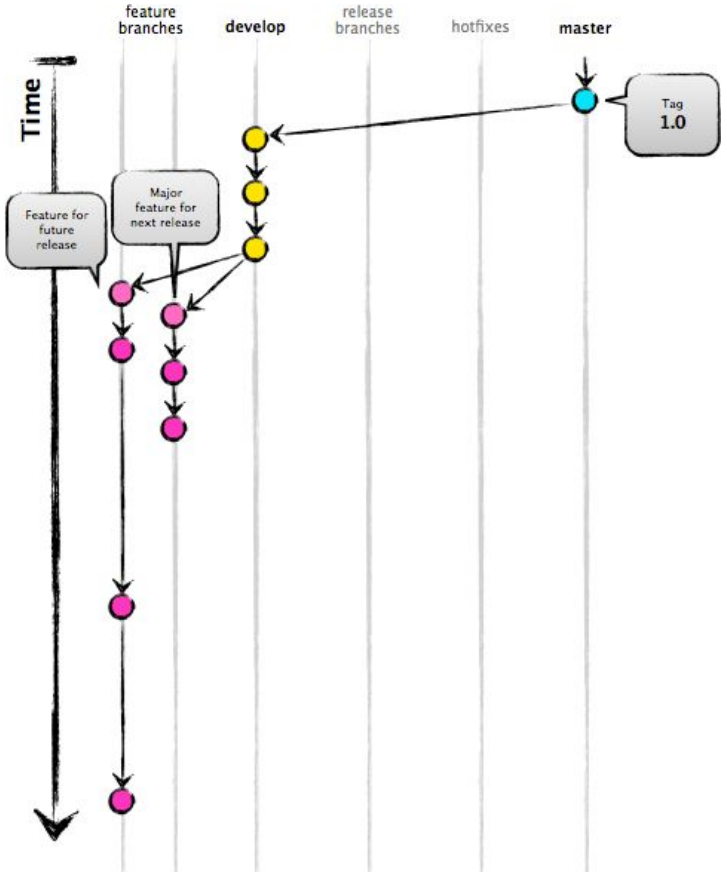
Коли нова розробка завершена, її знову об’єднують у гілку розробки, яка є місцем для всіх завершених функцій, які ще не випущено. Отже, коли наступний випуск відгалужується від розробки, він автоматично міститиме всі нові матеріали, які були завершені.

- **Підтримка екстрених виправлень**

GitFlow підтримує гілки виправлень — гілки, створені з випуску з тегами. Ви можете використовувати їх, щоб внести екстрені зміни, знаючи, що виправлення міститиме лише ваше екстрене виправлення. Немає ризику, що ви випадково об’єднаєтеся з новою розробкою одночасно.

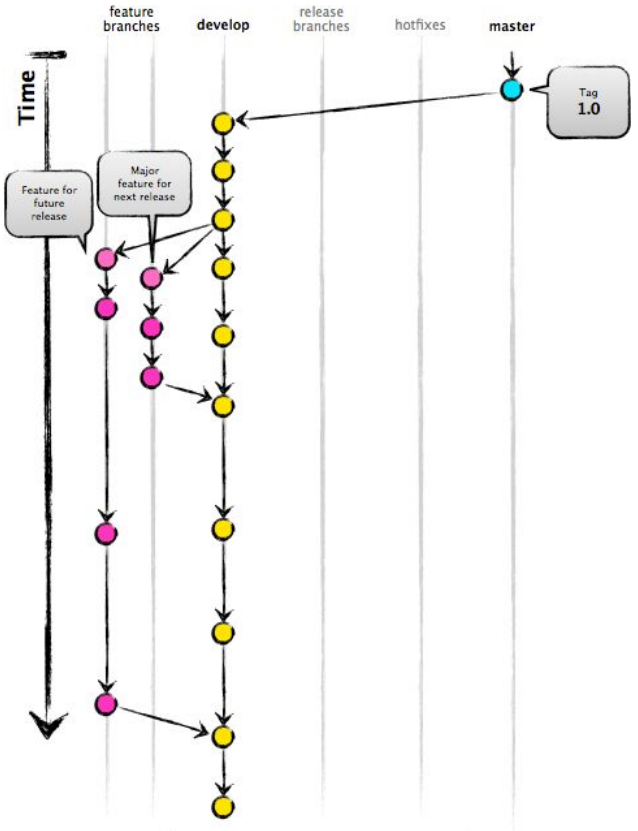
# Як працює GitFlow

Нові розробки (нові функції, ненадзвичайні виправлення помилок) вбудовані в гілках функцій:



# Як працює GitFlow

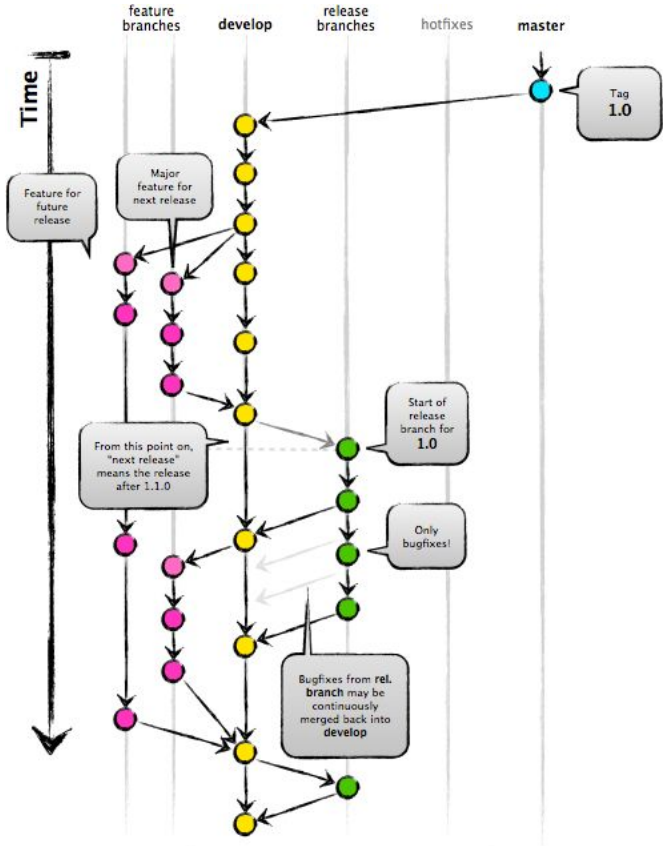
Гілки функцій відгалужуються від гілки розробки, а готові функції та виправлення об’єднуються назад у гілку розробки, коли вони готові до випуску:



# Як працює GitFlow

Коли настає час випуску, у розробці створюється гілка випуску:

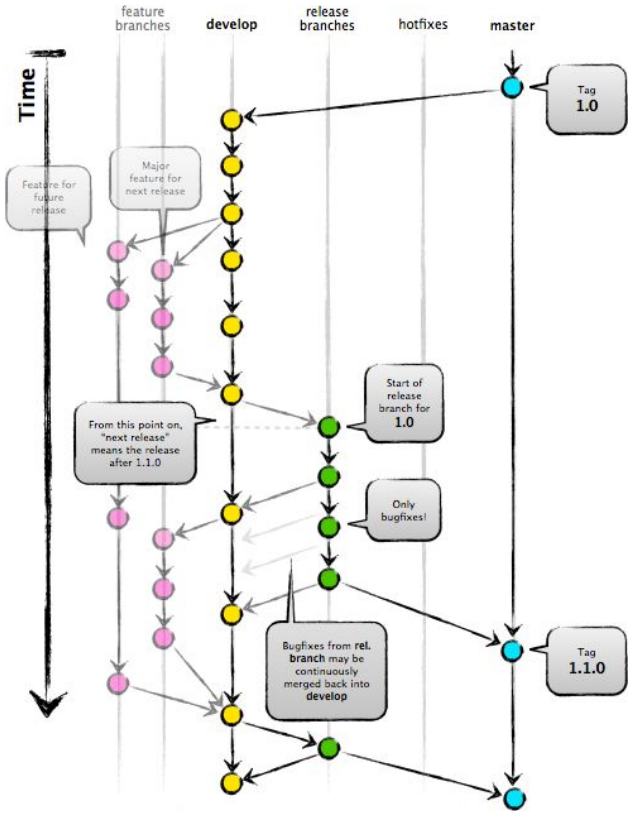
Код у гілці випуску розгортається у відповідному тестовому середовищі, тестується, і будь-які проблеми виправляються безпосередньо в гілці випуску. Цей цикл розгортання -> тестування -> виправлення -> повторне розгортання -> повторне тестування триває, доки ви не переконаєтеся, що випуск достатньо хороший для випуску клієнтам.



# Як працює GitFlow

Після завершення випуску гілка випуску об’єднується з головною та розробною, щоб гарантувати, що будь-які зміни, внесені у гілку випуску, не будуть випадково втрачені новою розробкою.

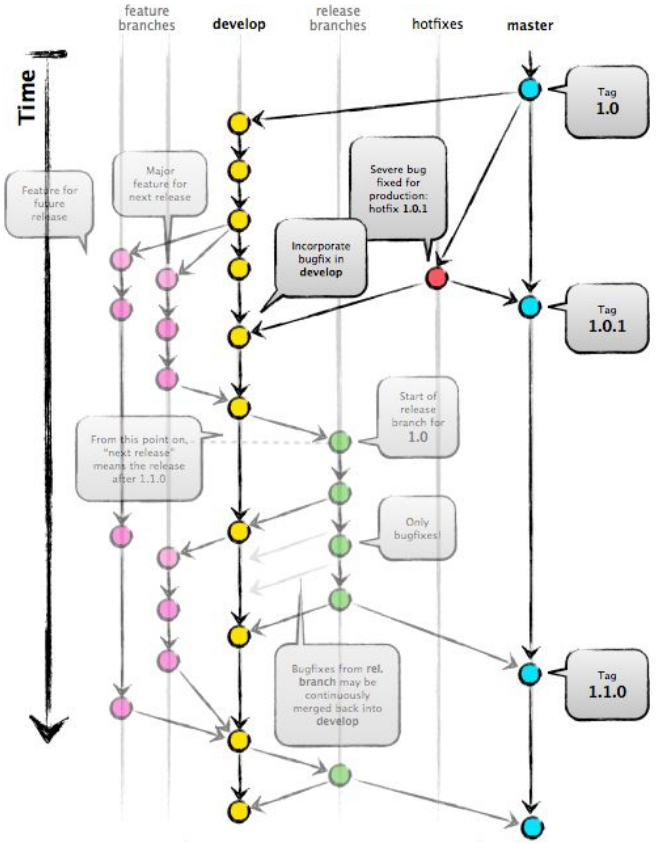
Головна гілка відстежує лише випущений код. Єдиними комітами для освоєння є злиття з гілок випуску та гілок виправлень.



# Як працює GitFlow

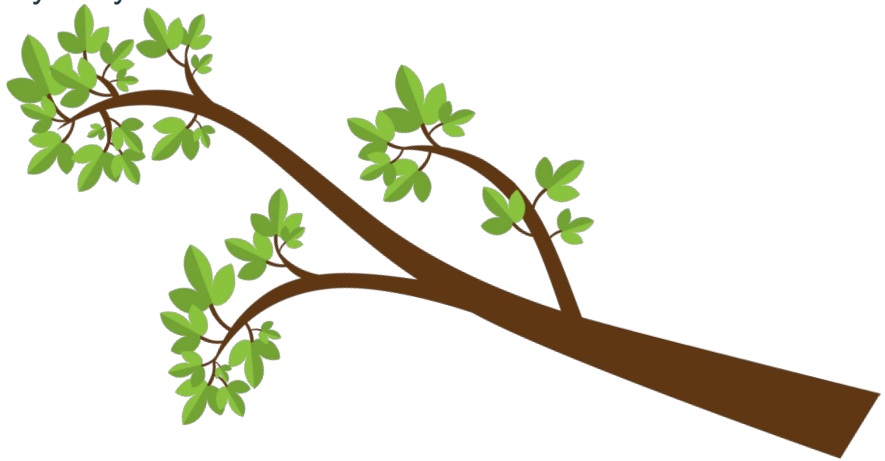
*Гілки виправлень* використовуються для створення екстрених виправлень:

Вони розгалужуються безпосередньо від випуску з тегами в головній гілці, а після завершення об’єднуються назад у головну та розробну гілку, щоб гарантувати, що виправлення випадково не буде втрачено під час наступного регулярного випуску.

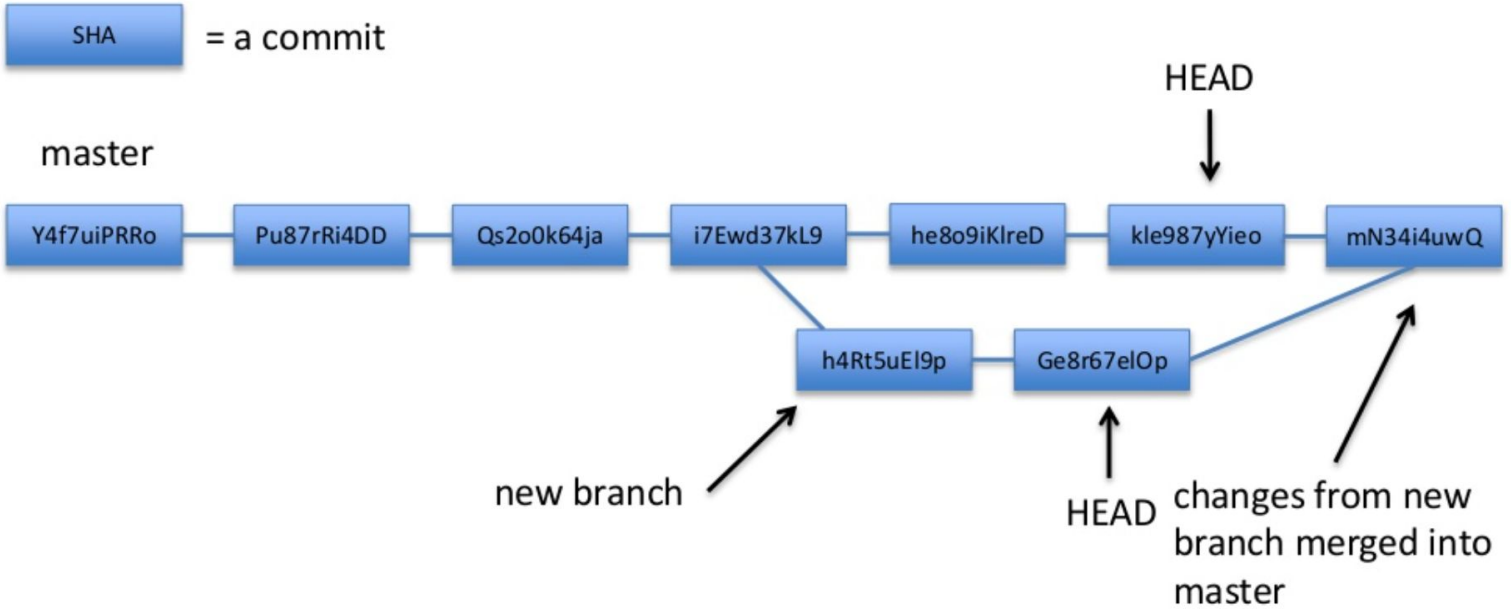


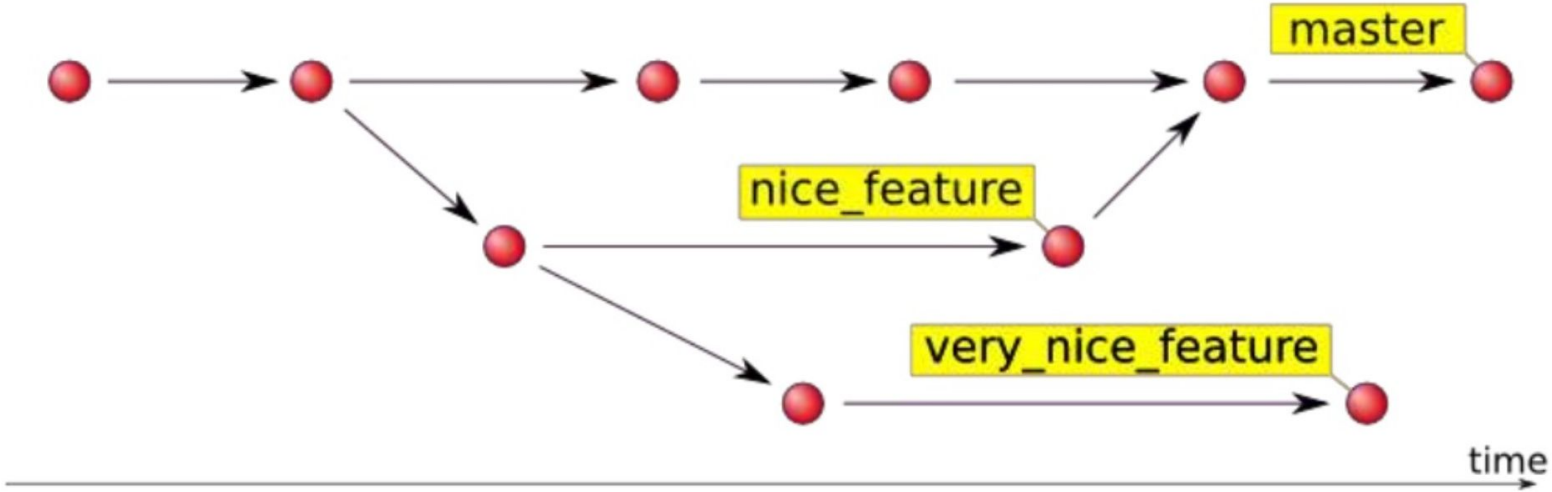
# Розгалуження

- дозволяє спробувати нові ідеї
- Якщо ідея не працює, викиньте гілку. Не потрібно скасовувати багато змін до головної гілки
- Якщо це спрацює, об’єднайте ідеї в головну гілку.
- Є лише один робочий каталог



# Приклад розгалуження та злиття




master
nice_feature
very_nice_feature
time

# В якій я гілці?

**git branch**

```
[dolanmi]$ git branch
* master
```

# Як створити нову гілку?

**git branch** new_branch_name

```
[dolanmi]$ git branch
* master
  new_feature
```

Примітка: на цьому етапі обидва HEAD гілки вказують на той самий комміт (комміт master)

# Як мені перейти на нову гілку?

**git checkout** new_branch_name

```
[dolanmi]$ git checkout new_feature
Switched to branch 'new_feature'
[dolanmi]$ git branch
  master
* new_feature
```

На цьому етапі можна перемикатися між гілками, робити коміти тощо в будь-якій гілці, при цьому обидві залишаються окремо одна від одної.

Примітка: щоб перейти до іншої гілки, ваш поточний робочий каталог має бути чистим (без конфліктів, що призводять до втрати даних).

# Порівняння гілок

**git diff** first_branch..second_branch

```
[dolanmi]$ git diff master..new_feature
diff --git a/file1.txt b/file1.txt
index 5626abf..1684a0f 100644
--- a/file1.txt
+++ b/file1.txt
@@ -1 +1 @@
-one
+new information
```

# Як об'єднати гілку?

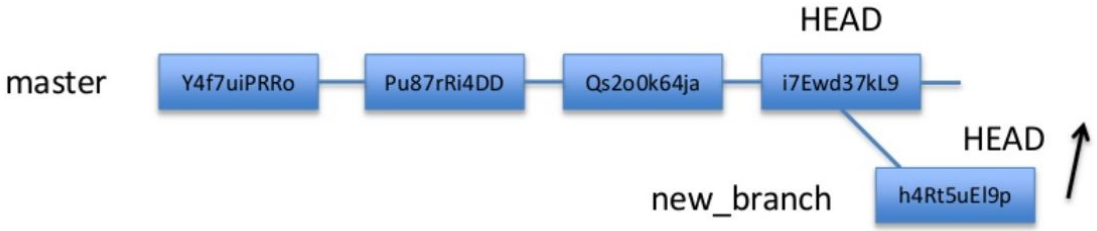
З гілки, у яку ви хочете об’єднати іншу гілку….

**git merge** branch_to_merge

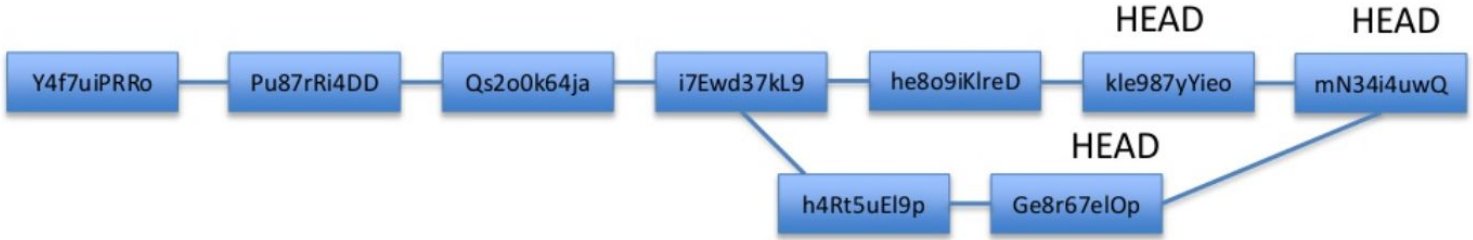
```
[dolanmi]$ git branch
* master
  new_feature
[dolanmi]$ git merge new_feature
Updating 3789cd3..1214807
Fast-forward
 file1.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
[dolanmi]$ git diff master..new_feature
[dolanmi]$
```

Примітка: під час об’єднання завжди мати чистий робочий каталог

Злиття *«швидке перемотування вперед»* відбувається, коли HEAD головної гілки видно при огляді назад

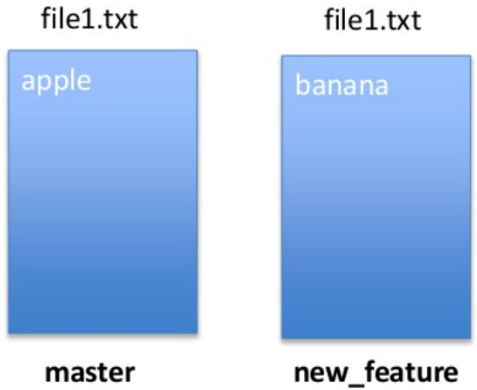



*«рекурсивне»* злиття відбувається шляхом огляду назад і комбінування предків для вирішення злиття



# Об’єднати конфлікти

Що робити, якщо є дві зміни в одному рядку в двох різних комітах?




```
[dolanmi]$ git merge new_feature
Auto-merging file1.txt
CONFLICT (content): Merge conflict in file1.txt
Automatic merge failed; fix conflicts and then commit the result.
```

# Вирішення конфліктів злиття

Git помітить конфлікт у файлах!

```
<<<<<<< HEAD
apple
=======
banana
>>>>>>> new_feature
```

**Рішення:**

1. Перервати злиття за допомогою **git merge –abort**
2. Вручну виправте конфлікт
3. Використовуйте інструмент злиття (їх багато)

# Графік історії злиття

**git log --graph --oneline --all --decorate**

```
[dolanmi]$ git log --graph --oneline --all --decorate
*   7367e1e (HEAD -> master) fix merge conflict
|\
| * b4f09a5 (new_feature) add banana
* | df043c1 add apple
|/
* 1214807 new information added
* 3789cd3 file3.txt
* 6bfebcd new dir
* 730c6bd files
* 48f1ecf c
* 60f1c1a another message yet
* d685ff9 another message
* 6e073c6 message
```

# Поради щодо зменшення болю при злитті

- зливати часто
- зберігати комміти невеликими/зосередженими
- часто вносити зміни, що відбуваються в майстер, у вашу гілку («відстеження»)

# Питання та відповіді