Visualizing version control

Joli Holmes

1

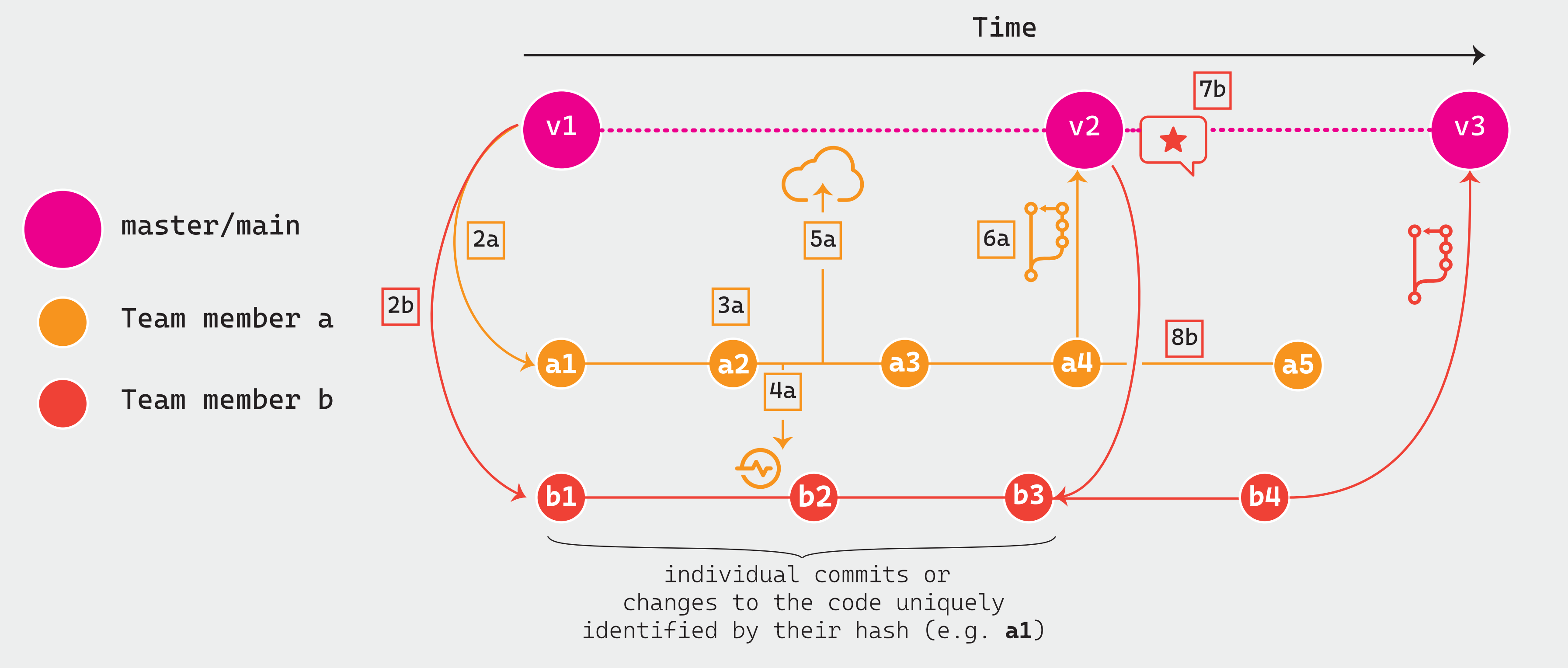
v1

```
<!DOCTYPE html>
<html>
      <head>
      </head>
      <body>
      </body>
</html>
```

2 Checkout a new branch with the **-b** command

a
```
git checkout -b a_changes
```

b
```
git checkout -b b_changes
```



individual commits or changes to the code uniquely identified by their hash (e.g. **a1**)

3 Make changes to the code and commit those changes

a2 - index.html

```
<!DOCTYPE html>
      <head>
      </head>
      <body>
            <header>
            </header>
            <footer>
            </footer>
      </body>
</html>
```

```
git add index.html
git commit -m “added header and footer tags”
```

a2 - styles.css

```
{
   margin: 0;
   padding: 0;
   font-family: Arial,
   sans-serif;
   font-size: 14;
}
```

```
git add styles.css
git commit -m “initial commit”
```

4 See what changes you made to the code since your last commit

a
```
git status
git diff index.html
```

5 Push code to the remote server

a
```
git push origin a_changes
```

6 Open a Pull Request (PR) on Github www.github.com so your team member can review the changes you wish to make to the **master** code

a Open a PR on www.github.com

7 On Github team member b reviews your code and does one of two things

1. Accepts and merges the code

2. Makes comments, asks for revisions, and suggests changes to the code

b Review code

8 Once the master branch is updated you should update your working branch

b
```
git checkout master
git pull
git checkout b_changes
git merge master
```

9 At this point the cycle repeats itself. Team members continue to create changes to the code,commit the changes, and then send pull requests to have those changes reviewed by their team members.

!

You may encounter **merge conflicts**. This happens when two team members edit the same code and try and merge their code together.

```
git checkout master
git pull
git checkout a_changes
git merge master
```

Open the code and you’ll see something like this in a file, for example **index.html**

```
<<<<<<HEAD
      <head>
            <title>
                  TITLE
            </title>
      </head>
=======
      <head>
            <title>
                  My title
            </title>
      </head>
>>>>>>> a_branch
```

Fix the conflict by keeping the parts of the code that you want. Then remove the

```
<<<<<<< HEAD
=======
>>>>>>> a_branch
```

Save the file and then run the following commands

```
git add filename (name of file where there was a conflict)
git commit -m ‘fixed conflict’
git push
```