MRG Software Training
W1- Version Control

Fall 2023- Tues 9/12/23

Presentation Slides- Manuel Roglan

Additional Resources

Git Cheat Sheet
The Odin Project- Introduction to Git
What is Git? Explained in 2 Minutes!
Pro Git Book
Collaborating with pull requests

A Scenario

Bob begins to create a website to see what users think the answer to how riddle is.

The Code

```html
<!DOCTYPE html>

<html>

    <head>
        <title>
            Bob's Great Website!
        </title>
    </head>

    <body>
        <h1>
            Welcome to my Website!
        </h1>
        <div style="margin-top:10px;">
            <h4>
                Please answer my very important question:
                what have I got in my pocket?
            </h4>
        </div>
        <div style="display:flex;gap:10px;">
            <button id="handses">
                hanses
            </button>
            <button id="knife">
                knife
            </button>
            <button id="nothing">
                nothing
            </button>
        </div>
        <div style="margin-top:10x;">
            <span>Handses count: </span><span id="handses-count"></span>
            <span>, Knife count: </span><span id="knife-count" /></span>
            <span>, Nothing count: </span><span id="nothing-count"></span>
        </div>
    </body>

    <script>
        let handsesCount = 0
        let knifeCount = 0
        let nothingCount = 0

        document.querySelector("#handses").addEventListener("click", () => {
            document.querySelector("#handses-count").innerHTML = ++handsesCount
        })
        document.querySelector("#Knife").addEventListener("click", () => {
            document.querySelector("#knife-count").innerHTML = ++knifeCount
        })
        document.querySelector("#nothing").addEventListener("click", () => {
            document.querySelector("#nothing-count").innerHTML = ++handsesCount;
        })
    </script>

</html>
```

MRG

It Works! ... right?

Welcome to my Website!

Please answer my very important question: what have I got in my pocket?

hanses knife nothing

Handses count: 5 , Knife count: 1 , Nothing count: 6

MRG

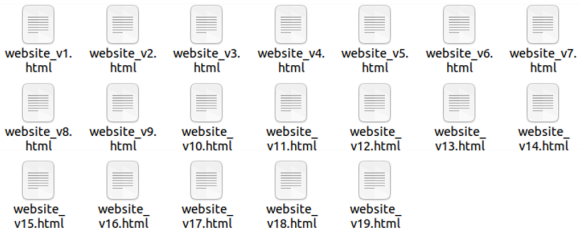
Issues

Oh No!

It looks like something is wrong. Bob's users are getting angry and he doesn't know what the bug is. Fortunately, Bob's friend recommended he save a copy of the previous website before pushing the new change.

MRG

Amazing!



MRG

Without version control, we'll only have one copy of the project in its current state, unless we constantly make duplicate copies when saving changes, which is difficult to remember and keep track of.

Bob's Team Grows!

Bob's friend Alice has decided to help Bob with his website. Already, Alice has fixed Bob's previous bug and is willing to keep growing the site. Now, features can get added twice as fast!

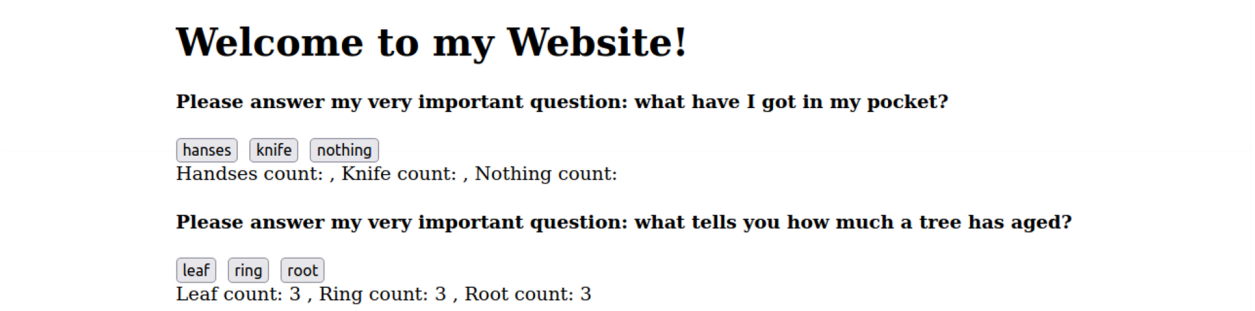
Bob will be working on adding another answer to his riddle, while Alice creates an entirely new one!

This method also doesn't scale at all, especially with multiple people working on the same project.

Alice's Code

```
<html>

    <head>
        <title>
            Bob's Great Website!
        </title>
    </head>

    <body>
        <h1>
            Welcome to my Website!
        </h1>
        <div style="margin-top:10px;">
            <h4>
                Please answer my very important question:
                what have I got in my pocket?
            </h4>
        </div>
        <div style="display:flex;gap:10px;">
            <button id="handses">
                hanses
            </button>
            <button id="knife">
                knife
            </button>
            <button id="nothing">
                nothing
            </button>
        </div>
        <div style="margin-top:10x;">
            <span>Handses count: </span><span id="handses-count"></span>
            <span>, Knife count: </span><span id="knife-count" /></span>
            <span>, Nothing count: </span><span id="nothing-count"></span>
        </div>
        <div style="margin-top:10px;">
            <h4>
                Please answer my very important question:
                what tells you how much a tree has aged?
            </h4>
        </div>
        <div style="display:flex;gap:10px;">
            <button id="leaf">
                leaf
            </button>
            <button id="ring">
                ring
            </button>
            <button id="root">
                root
            </button>
        </div>
        <div style="margin-top:10x;">
            <span>Leaf count: </span><span id="leaf-count"></span>
            <span>, Ring count: </span><span id="ring-count" /></span>
            <span>, Root count: </span><span id="root-count"></span>
        </div>
    </body>

    <script>
        let handsesCount = 0
        let knifeCount = 0
        let nothingCount = 0

        document.querySelector("#handses").addEventListener("click", () => {
            document.querySelector("#handses-count").innerHTML = ++handsesCount
        })
        document.querySelector("#knife").addEventListener("click", () => {
            document.querySelector("#knife-count").innerHTML = ++knifeCount
        })
        document.querySelector("#nothing").addEventListener("click", () => {
            document.querySelector("#nothing-count").innerHTML = ++nothingCount;
        })


        let aliceCounters = {leaf: 0, ring: 0, root: 0}

        document.querySelector("#leaf").addEventListener("click", () => {
            document.querySelector("#leaf-count").innerHTML = ++aliceCounters.leaf;
        })
        document.querySelector("#ring").addEventListener("click", () => {
            document.querySelector("#ring-count").innerHTML = ++aliceCounters.ring;
        })
        document.querySelector("#root").addEventListener("click", () => {
            document.querySelector("#root-count").innerHTML = ++aliceCounters.root;
        })

    </script>
</html>
```

Alice's Code



MRG

Off to Production!

Now, all Bob needs to do is copy Alice's emailed code and paste it into his file. Then, he should be good to go!

MRG

We have to constantly toss around the latest code via email or other mechanisms and then manually merge the changes.

Final Code

```html
<html>

    <head>
        <title>
            Bob's Great Website!
        </title>
    </head>

    <body>
        <h1>
            Welcome to my Website!
        </h1>
        <div style="margin-top:10px;">
            <h4>
                Please answer my very important question:
                what have I got in my pocket?
            </h4>
        </div>
        <div style="display:flex;gap:10px;">
            <button id="handses">
                hanses
            </button>
            <button id="knife">
                knife
            </button>
            <button id="nothing">
                nothing
            </button>
            <button id="ring">
                ring
            </button>
        </div>
        <div style="margin-top:10x;">
            <span>Handses count: </span><span id="handses-count"></span>
            <span>, Knife count: </span><span id="knife-count" /></span>
            <span>, Nothing count: </span><span id="nothing-count"></span>
            <span>, Ring: </span><span id="ring-count"></span>
        </div>
        <div style="margin-top:10px;">
            <h4>
                Please answer my very important question:
                what tells you how much a tree has aged?
            </h4>
        </div>
        <div style="display:flex;gap:10px;">
            <button id="leaf">
                leaf
            </button>
            <button id="ring">
                ring
            </button>
            <button id="root">
                root
            </button>
        </div>
        <div style="margin-top:10x;">
            <span>Leaf count: </span><span id="leaf-count"></span>
            <span>, Ring count: </span><span id="ring-count" /></span>
            <span>, Root count: </span><span id="root-count"></span>
        </div>
    </body>

    <script>
        let handsesCount = 0
        let knifeCount = 0
        let nothingCount = 0
        let ringCount = 0

        document.querySelector("#handses").addEventListener("click", () => {
            document.querySelector("#handses-count").innerHTML = ++handsesCount
        })
        document.querySelector("#knife").addEventListener("click", () => {
            document.querySelector("#knife-count").innerHTML = ++knifeCount
        })
        document.querySelector("#nothing").addEventListener("click", () => {
            document.querySelector("#nothing-count").innerHTML = ++nothingCount;
        })
        document.querySelector("#ring").addEventListener("click", () => {
            document.querySelector("#ring-count").innerHTML = ++ringCount;
        })

        let aliceCounters = {leaf: 0, ring: 0, root: 0}

        document.querySelector("#leaf").addEventListener("click", () => {
            document.querySelector("#leaf-count").innerHTML = ++aliceCounters.leaf;
        })
        document.querySelector("#ring").addEventListener("click", () => {
            document.querySelector("#ring-count").innerHTML = ++aliceCounters.ring;
        })
        document.querySelector("#root").addEventListener("click", () => {
            document.querySelector("#root-count").innerHTML = ++aliceCounters.root;
        })
    </script>

</html>
```

MRG

Looks Good! ... right?

Welcome to my Website!

Please answer my very important question: what have I got in my pocket?

hanses knife nothing ring

Handses count: 2 , Knife count: 1 , Nothing count: 1 , Ring: 4

Please answer my very important question: what tells you how much a tree has aged?

leaf ring root

Leaf count: 1 , Ring count: , Root count: 2

MRG

Oh no

Somehow, Alice's ring button is broken. Who is to blame, Alice or Bob? Both their code was working separately, but together it failed. Now, Bob will need to revert to a previous version and try to debug again.

And it only gets worse

In this example, Bob and Alice were working on different features during the same time frame.

Consider a scenario where Alice is working on a feature that gets merged after Bob has finished three others? Bob will likely have a lot of debugging to do!

Or consider a scenario where Bob's team consists of more people. Now, Bob will need to coordinate the merging of lots more code!

What to do?

What to do?

Keeping separate files labelled by version number as Bob does is quite tedious and prone to error.

Luckily, there are sophisticated open-source version control tools we can use! We use Git.

MRG

Version Control

Version control is a system that records changes you make to files you create so you can easily track your project history and revert to previous states if needed. It also makes it easier to work collaboratively on files with others and see who made changes at different times.

Git

Git is a very popular open source version control system (VCS) that works on your local machine. It performs version control by taking snapshots of your files at different points in time, which are captured via commits, which include changes to files. Once connected to a network, Git also supports collaboration by allowing users to synchronize their local repositories with remote repositories hosted on servers (e.g. GitHub).

Git

Instead of creating files with version number (e.g. website_v3.html), we create commits.

Instead of working on a feature somewhere separately, and then copying the new code over, we create branches and merge those branches.

This can all be done from the command line.

MRG

Using Git with the Command Line

Setup & Init

git init

In a new directory, Bob uses "git init" to create a new git repository. Any changes to files made within this repository will be tracked.

MRG

Tracking versions

Whenever Bob has finished making some changes to his website.html file, he can save a version of that file using the following commands:

git add website.html

git commit –m "relevant commit message"

git add <path>

The add command tells git which files it should stage for a commit. If you have multiple files changed in your repository, it may be useful to write separate commit messages for different groups of files. For example:

git add website1.html

git add website1.js

git commit –m "website 1 changes"

git add website2.html

git add website2.js

git commit –m "website 2 changes"

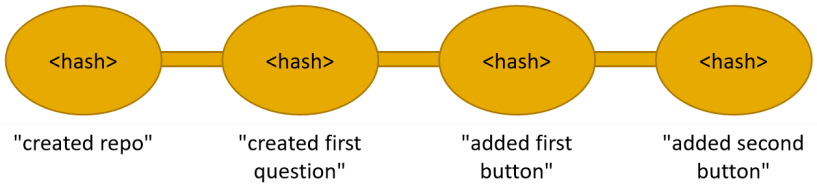
git commit

The commit command tells git to save a new version of the changes you have made. The –m flag allows you to give the version an accompanying commit message, which is useful if you need to look at previous versions.

Other Useful Git Commands

- Git status
  - Shows what files have changes made that are not added yet, and it shows what files have been added but are not yet committed.
- Git log
  - Shows a history of previous commits from the branch you are currently on

Bob's git repository



Rewrite History

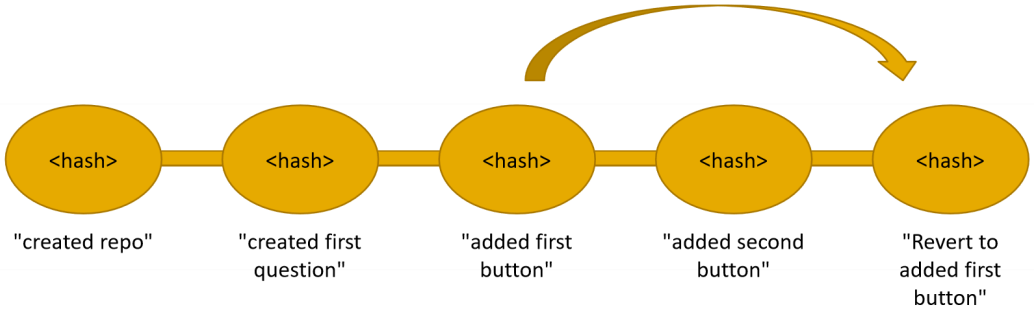
Bob is debugging

During Bob's first attempt, to change his production code to a previous version, he needed to use a separate file. With git, we don't need to do this.

Note: Normally, there are many checks that should be performed before code goes into production (unlike in Bob's case). This goes beyond using a sophisticated version control but using one does help.

Reverting to a previous commit

Bob can use "git revert HEAD" to revert to the previous version.



git revert <commit>

Git revert creates a new commit containing the code before the <commit> was made. It does not delete any previous commits. You can revert a single commit:

git revert HEAD

<commit message entered when prompted>

Or you can revert multiple commits:

git revert --no-commit HEAD~<number of commits>

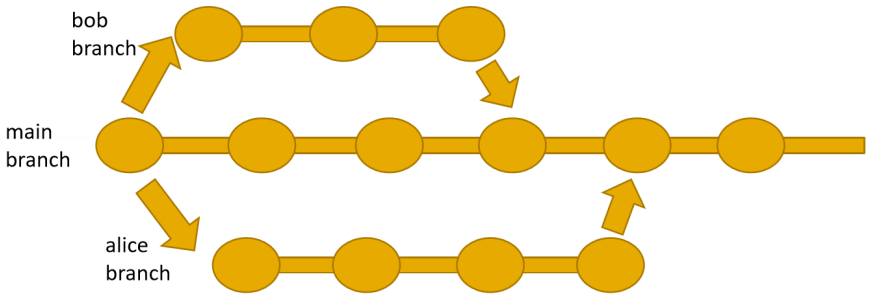
git commit –m "<message>"

Bob and Alice Work Together

Last time, Bob and Alice worked on their features independently, and Bob then manually added Alice's code to his. With git, Bob and Alice can work on separate branches and then merge these branches with the production branch when their work is done.

How do they work on the same repo from different machines? That will be answered when we talk about GitHub. For now, assume they are on the same machine.

Bob and Alice Work Together



MRG

Using Branches

- git branch
  - Lists all branches in the repository and shows an * next to the branch you are currently on
- git branch <name>
  - Creates a new branch with the name specified.
- git checkout <name>
  - Moves the HEAD to the specified branch (i.e. changes the branch you are working on)
- git checkout –b <name>
  - Creates a new branch with the name specified and moves the HEAD to it

MRG

Using Branches

- git merge <name>
  - Merges the specified branch with the current branch you are on.

MRG

Branch management
https://git-scm.com/book/en/v2/Git-Branching-Branch-Management

Some Best Practices

Some Best Practices

1. Commit frequently and give your commits good messages.
2. When working on a new feature/optimization/bug-fix/anything, create a new branch.
3. Never commit directly to the main branch. More on this later.

Collaborating With GitHub

Bob and Alice Work Together

Bob and Alice use different computers. Git is a local version control, meaning it only stores version information on your machine.

To share code with each other, Bob and Alice need some online medium to upload and download code pertaining to their repo.

There are many different tools available, such as GitHub, GitBucket, and GitLab to name a few. We use GitHub.

Sharing Code on GitHub

GitHub provides a straightforward user interface for creating a new repo.

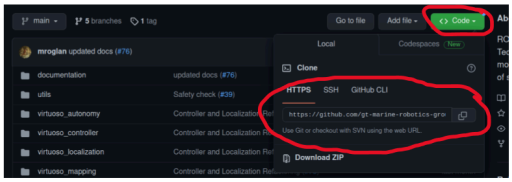
Upon creating a repo, GitHub provides us with directions on how to link our GitHub repo with our local git repo, and we will walk through them.

Sharing Code on GitHub

From your local repo, run the following command:

git remote add origin <link>

Where <link> is the link to your repository on GitHub:



MRG

Share & Update

git remote add <origin> <link>

The remote add command creates a new "remote" to some online repository. A local git repo can have multiple remote repos. The <origin> argument specifies the name of the remote to add and the <link> argument specifies the location of the remote repo.

For the rest of these slides, we assume the remote repo added is called "origin".

MRG

Pushing Changes

After working on a branch in your local git repo, you can upload or "push" your changes to the GitHub repo using "git push <origin> <name>". For example:

git push origin bob

Or:

git push origin alice

Make sure the name of the branch you are pushing to on GitHub matches the name of the branch you are on in your local repo.

MRG

Pulling Changes

If you are on a branch in your local repo and want update it with changes someone else has pushed to GitHub, you can run "git pull <origin> <name>". For example:

git pull origin bob

If you want to pull a branch from GitHub that you have not worked on in your local repo, you will want to run the following commands:

git fetch origin

git checkout <name>

MRG

Pull Requests

Pull Requests

Pull requests are essentially merges that others can look over before merging. A pull request is submitted through GitHub. You can add reviewers to your request, add comments, and make push changes to the branch while the pull request is under review.

MRG

Merging Changes

For any branch EXCEPT THE MAIN BRANCH, you can simply merge the branches in your local repository and push the changes to GitHub.

To merge with the main branch, you will need to submit a pull request. Virtuoso is configured so that any merges to the main branch must go through a pull request and be approved by the Software Lead.

MRG

Training Project: Version Control

https://github.com/gt-mrg-training/version-control

This is a practice repo for learning to use Git.

1. Clone this repo on your local machine.
- `git clone https://github.com/gt-mrg-training/version-control.git`
2. Create a new branch with your first and last name.
- `git checkout -b <first name>_<last name>`
- For example, `git checkout -b manuel_roglan`
3. Create a new text file containing your name, gt email, and any specific interests you have for the software.
- Look at `manuel_roglan.txt` for an example
4. Commit your changes.
- `git add .`
- `git commit -m "a useful message"`
5. Push your changes to GitHub.
- `git push origin <branch name>`
- For example, `git push origin manuel_roglan`
6. Submit a pull request
- Under the pull requests tab on GitHub, select your branch to merge into main.
7. Add mroglan as a reviewer.

8. Wait for a review.
9. Upon approval, Squash and Merge changes.

https://github.com/gt-mrg-training/version-control/pull/17