TDS-GA2-Q3-Notes (by moon)

Answer `4986021`

How to Use Git to Find Who or What Created a Problem

Start Simple

1. Check What Changed Recently

```
git log --oneline
```

This shows recent commits in short form.

Look at:

- The latest commit
- Commit messages
- Suspicious changes

If the issue started today, it is probably in one of the latest commits.

2. See What a Specific Commit Changed

After finding a commit hash:

```
git show <commit_hash>
```

This shows:

- What files were changed
- What lines were added or removed
- Who made the commit

This helps you understand what exactly caused the issue.

3. Check Who Modified a Specific Line

If the problem is in a specific file:

```
git blame filename
```

This shows:

- Which commit changed each line
- Who wrote that line
- When it was written

This is very useful when one particular line is causing an error.

4. Check Changes to One File Only

```
git log -- filename
```

This shows the history of that file only.

Useful when you know the issue is inside one file.

Case Study: Finding When a Timeout Value Was Introduced

Problem

A production issue occurred because a timeout value was changed in `config.json`.
We need to:

1. Find the commit where `"timeout"` was changed to `30`
2. Identify the parent commit of that commit
3. Submit the 7-character short hash of the parent commit

Important Concepts

1. Git Commit History

Git stores history as a chain of commits.

Example:

A → B → C → D

If commit C changed the timeout to 30,
then commit B is its parent.

The parent commit is simply the commit that came immediately before it.

---

2. git log

`git log` shows commit history.

To search inside a specific file:

```
git log -- config.json
```

To search for a specific value:

```
git log -S "30" -- config.json
```

- `-S "30"` searches for commits where the value 30 was added or removed
- `-- config.json` limits the search to that file

---

3. git show

After finding a commit hash, inspect it:

```
git show <commit_hash>
```

Look for something like:

- "timeout": 120

- "timeout": 30

This confirms that this commit introduced the new timeout value.

---

4. Parent Commit

Every commit (except the first one) has a parent.

If your commit hash is:

```
abc1234
```

The parent commit is:

```
abc1234^
```

The `^` symbol means "parent of this commit".

---

5. Get the 7-Character Short Hash

To get the short hash of the parent commit:

```
git rev-parse --short abc1234^
```

This outputs something like:

```
f7e3a21
```

That short hash is the final answer.

Step-by-Step Procedure

Step 1: Search for timeout value 30

```
git log -S '"timeout": 30' -- config.json
```

If needed, try:

```
git log -S 30 -- config.json
```

Step 2: Confirm the correct commit

```
git show <commit_hash>
```

Make sure the diff clearly shows the timeout changing to 30.

Step 3: Get the parent commit

```
git rev-parse --short <commit_hash>^
```

The output is the answer.

Key Understanding

- A commit represents a snapshot of changes.
- The parent commit is the version just before the change happened.
- The `^` symbol moves one step backward in history.
- `-S` helps find when a specific value was introduced.

Final Takeaway

When investigating production issues:

1. Search for the exact value introduced.
2. Confirm the commit using `git show`.
3. Move to the parent commit to see the previous stable state.
4. Use the short hash when required.