

1.1.7 Revisit Our Git Workflow

As you get into a comfortable workflow, you'll want to commit to Git often. This ensures that you don't accidentally lose any work and gives you a "save point" that you can potentially revert back to. We just finished an important milestone: outlining the high-level structure of a webpage. This is the perfect time to commit.

Create a Gitignore File

Before we commit, though, let's create another file in our project:

```
touch .gitignore
```

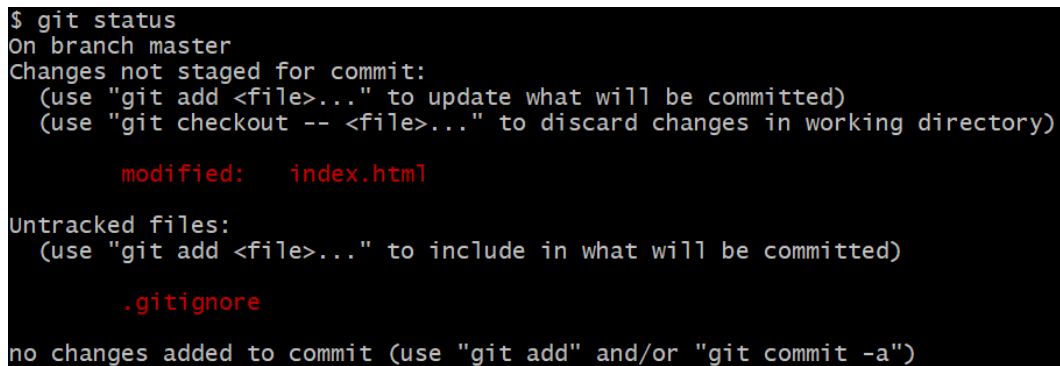
A `.gitignore` file (yes, the dot is part of it) is where we can specify any files we don't want Git to track. Developers often get into the habit of running a Git command that will automatically pick up all untracked or modified files in a project folder. Though this is convenient, sometimes unwanted files end up getting committed.

For example, on macOS, every directory has a hidden `.DS_Store` file that can create unnecessary headaches if accidentally added to Git. Mac users probably already noticed Git continually pointing out this file. Windows users, you can simulate the `.DS_Store` problem by creating your own `.DS_Store` file with the `touch .DS_Store` command (yes, go ahead and do that now!).

IMPORTANT

If you have a Windows computer, it's still a good idea to add `.DS_Store` to your `.gitignore` file, as you never know when you may work with someone that uses macOS instead!

Now that everyone has this file in their project, open the `.gitignore` file in VS Code and add the following line: `.DS_Store`. Save the file, then run `git status`, which should display information that resembles this image:

A terminal window with a black background and white text. The output of the 'git status' command is shown. It indicates the current branch is 'master'. Under 'Changes not staged for commit', it lists 'index.html' as 'modified'. Under 'Untracked files', it lists '.gitignore'. At the bottom, it says 'no changes added to commit'.

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .gitignore

no changes added to commit (use "git add" and/or "git commit -a")
```

Note that Git has now ignored the `.DS_Store` file. It also sees one modified file (`index.html` with our new semantic elements) and one new, untracked file (`.gitignore`).

Let's add both of these to staging at the same time, using ONE of these commands:

```
git add .
```

```
git add -A
```

How do you know which one to use? Here's how they differ:

- `git add .` adds any untracked or modified files in the current directory (the current directory being represented by `.`) and all subdirectories.
- `git add -A` adds any files in the entire project.

Both are useful when you need to stage multiple files at once but should be used carefully and not without a `.gitignore` file in place first!

With these changes staged, let's commit now by typing the following:

```
git commit -m "html outline and gitignore"
```