.gitignore Cheat Sheet

Contents

What is `.gitignore`?

A `.gitignore` file specifies intentionally untracked files that Git should ignore. This helps prevent sensitive, temporary, or unnecessary files from being committed to the repository.

Basic Syntax

- **Comment lines**: Use `#` at the start of the line.
  Example:

  ```
  # This is a comment
  ```

- **Blank lines**: Ignored and used for readability.

Ignore Patterns

1. **Exact match**:
   To ignore a specific file:

   ```
   file.txt
   ```

2. **Wildcard (`*`)**:
   Matches zero or more characters.

```
*.log    # Ignore all .log files
temp*    # Ignore files starting with "temp"
```

3. **Directories**:

   To ignore a directory and its contents, add a trailing `/`.

   ```
   /cache/ # Ignore "cache" directory
   ```

4. **Negation (`!`)**:
   To unignore files or directories.

   ```
   !important.log
   ```

5. **Patterns with paths**:
   Specify relative paths for more precise ignores.

   ```
   /config/settings.json  # Ignore only in the root
   **/debug.log           # Ignore in all subdirectories
   ```

Clearing Git Ignore Cache

Why Clear the Git Ignore Cache?

If you modify the `.gitignore` file to ignore files that are already tracked by Git, those files will remain in the repository until you remove them from tracking.

Steps to Clear the Cache

1. **Remove the files from tracking (without deleting them):**

   ```
   git rm -r --cached .
   ```

2. **Add the modified `.gitignore` to the staging area:**

   ```
   git add .gitignore
   ```

3. **Commit the changes:**

   ```
   git commit -m "Updated .gitignore and cleared cache"
   ```

Now Git will respect the `.gitignore` rules and stop tracking the ignored files.

Examples of Common Patterns

Operating System Files

- macOS:

  ```
  .DS_Store
  .AppleDouble
  .LSOverride
  ```

- Windows:

  ```
  Thumbs.db
  ehthumbs.db
  Desktop.ini
  ```

- Linux:

  ```
  *~
  .nfs*
  ```

Programming Languages

Python

```
*.pyc
*.pyo
__pycache__/
.env
.venv/
```

Node.js

```
node_modules/
npm-debug.log*
yarn-debug.log*
yarn-error.log*
.env
```

Java

~~~
*.class
*.jar
*.war
*.ear
*.iml
```shell

#### C/C++

```shell
*.o
*.out
*.so
*.a
*.d
~~~

Ruby

```
*.gem
.log
tmp/
```

Go

```
*.exe
*.test
*.out
```

Rust

```
*.rs.bk
target/
Cargo.lock
```

Frameworks and Tools

Git

```
.git/
```

IDEs

- VSCode:

  ```
  .vscode/
  ```

- IntelliJ IDEA:

  ```
  .idea/
  *.iml
  ```

- Eclipse:

  ```
  .classpath
  .project
  .settings/
  bin/
  ```

Logs and Temporary Files

```
*.log
*.tmp
*.swp
*.bak
*.old
*.~lock.*
```

Ignoring Files in Specific Directories

- Ignore all `.log` files in the `/logs` directory:

  ```
  logs/*.log
  ```

- Ignore everything in the `/temp` directory except `.keep`:

  ```
  temp/
  !temp/.keep
  ```

Useful Tips

1. **Check ignored files**:

   View which files are being ignored:

```
git status --ignored
```

2. **Exclude `.gitignore` itself**:

   Add this to `.gitignore`:

   ```
   .gitignore
   ```

3. **Use global `.gitignore`**:

   Configure a global ignore file for your user:

   ```
   git config --global core.excludesfile ~/.gitignore_global
   ```

   Example content of `~/.gitignore_global`:

   ```
   *.bak
   *.swp
   ```