What is the difference between local storage and session storage?

Local storage is destroyed once the user closes the browser while session storage has no expiration date

Local storage keeps store the user information data with expiration date while session storage will delete data when browser windows closed by a web user

No differences

***Session storage is working per window or tab while local storage needed to be deleted via JS or manually.***

Define float property of CSS?

Float same like positing property except it has two directions left and right

***By float property, the image can be moved to the right or the left along with the text to be wrapped around*** i

Float same like positing property except it has one direction Elements before this property is applied,

it do change their properties

Name the property for controlling the image position in the background

background-position

What is the CSS Box model and what are its elements?

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content edges: **the content edge, padding edge, border edge, and margin edge**

***Explain the difference between visibility: hidden and display: none***

Both occupy space and not hide elements

**Visibility hidden occupies space and affects the layout of the document while display none not occupy space**

Display none occupies space and affects the layout of the document while visibility hidden not occupy space

Both hide elements and not occupy space

**How to update a component every second?**

**Using setInterval to trigger the change**

Other answer

React doesn’t allow continuous change for component

Using setstate in class component

**Observables help you manage . . . . . . . . data**

**Asynchronous**

**Which of the following component of HTTP request indicates HTTP methods such as GET, POST, DELETE, PUT, etc.?**

**Verb − Indicate HTTP methods such as GET, POST, DELETE, PUT etc.**

**.....is a collection of open protocols and standards used for exchanging data between applications or systems**

**A web service**

**Which mechanism is applied to use a design pattern in an OO system?**

1. **Inheritance)**
2. **Composition)**
3. **All of the mentioned)**
4. **None of the mentioned**

**. Which of the following is a design pattern?**

a) Behavioral

b) Structural

c) Abstract Factory

d) All of the mentioned

**If you want to minimize development costs by reusing methods? Which design pattern would you choose?**

You want to minimize development cost by reusing methods? Which design pattern would you choose?  
a) Adapter Pattern  
b) Singleton Pattern  
c) Delegation pattern  
d) Immutable Pattern

**Which design pattern defines one-to-many dependency among objects?**

Which design pattern defines one-to-many dependency among objects?  
a) Singleton pattern  
b) Facade Pattern  
c) Observer pattern  
d) Factory method pattern

**What is a merge conflict in Git?**

Conflicts generally arise when two people have changed the same lines in a file, or if one developer deleted a file while another developer was modifying it. In these cases, Git cannot automatically determine what is correct. Conflicts only affect the developer conducting the merge, the rest of the team is unaware of the conflict. Git will mark the file as being conflicted and halt the merging process. It is then the developers' responsibility to resolve the conflict.

**What is Git stash?**

Git stash is a built-in command with the distributed Version control tool in [Git](https://www.techtarget.com/searchitoperations/definition/Git) that locally stores all the most recent changes in a workspace and resets the state of the workspace to the prior commit state.

A user can retrieve all files put into the stash with the git stash pop and git stash apply commands. Git stash acts as a mechanism to locally version files without those versions being seen by other developers who share the same git repository.

**Git stash vs. commit**

The git commit and git stash commands are similar in that both take a snapshot of modified files [in the git working tree](https://www.theserverside.com/video/Understand-the-Git-working-tree-status-command-for-easy-DVCS) and store that snapshot for future reference. The key differences between the two are as follows:

* A commit is part of the public git history; a stash is stored locally.
* A commit creates a new save point on a branch; a stash reverts to a previous save point.
* A new commit leaves files in the working tree unchanged; a stash resets files in the working tree to the previous commit point.
* A commit is a public record of file changes; a stash is local.

**Git stash vs. reset**

The git stash and the git reset hard commands are similar, as both commands will revert all files in the working directory back to the state at the point of the previous commit. Differences between the two include:

* A reset creates a new commit point in the branch history; stash does not.
* A reset can jump back to any prior commit; a stash can only reset the files in the workspace to the point of the previous commit.
* A hard reset will discard all changes; a stash saves a snapshot of all locally modified files.

**Git stash vs. stage**

The git stash and git stage commands can be confused because of their similar names, but the two commands are different. The git stage command simply adds files to the git index. This allows those files to be part of a filesystem snapshot when a git commit occurs. This is a different construct as compared to git stash

**What does the following command do? git diff-tree –r {commit hash}**

Given the commit hash, this will list all the files that were changed or added in that commit. The -r flag makes the command list individual files, rather than collapsing them into root directory names only.

**What does the following command do? git diff-tree –r {commit hash}**

**Shows a DAG of changes for a particular branch and hash. Get a list of files that has been changed in a particular commit.**

**Get a list of files that has been added by another user to a branch.**

**Get a list of files that begin with the letter r.**

**Which of these Git commands does the following? - Incorporates all the new commits in the master branch - Rewrites the project history by creating brand new commits for each commit in the original branch**

### The Rebase Option

**Which Git command enables you to pick up commits from a branch within a repository and apply it to another branch?**

### git cherry-pick

**Which hash function does Git use for commits?**

SHA-1 hash function

**What file is needed in a project root to track the commits in Git?**

.git/logs/refs/heads contains files for each branch that have all the commit hashes. That’s what tracks the history of your project.

The hash goes across each file belonging to that version of your repository.

.git/index contains pointers to all sorts of things inside all your objects, which gives you finer detail of diffs, but it is not human readable.

You can use various git tools if you need to diff things to see changes.

2

**What are Hooks in Git?**

Git hooks are scripts that run automatically every time a particular event occurs in a Git repository. They let you customize Git’s internal behavior and trigger customizable actions at key points in the development life cycle.

Common use cases for Git hooks include encouraging a commit policy, altering the project environment depending on the state of the repository, and implementing continuous integration workflows. But, since scripts are infinitely customizable, you can use Git hooks to automate or optimize virtually any aspect of your development workflow.

In this article, we’ll start with a conceptual overview of how Git hooks work. Then, we’ll survey some of the most popular hooks for use in both local and server-side repositories.