# Storage Manager Overview

## Brief Overview

The WalkMe Player stores data for maintaining WalkMe's player state, monitoring compilation of Walk-Thru playback, and collecting Analytics. To this end, WalkMe uses first-party cookies, third-party cookies (local storage), and first-party local storage. No personal data is saved and no information about user usage of the website is monitored.

Cookies set by WalkMe are preceded by a "wm-" in their name. WalkMe uses different cookies depending on your WalkMe configuration; for example, we can track user information for completing Walk-Thrus using Cookies or local storage.

Looking to create your own cookies with WalkMe? Check out this article on WalkMe Data.

WalkMe stores data on the user's browser for a number of reasons, including:

1. Smart Walk-Thru state (whether the user is in the middle of a Smart Walk-Thru)
2. Autoplay status (whether an item has already played)
3. Goal completion
4. WalkMe data

Note

The User extension does not require use of 3rd party cookies

## Definitions

1. **Storage method**: This is how the storage is saved on the browser, whether first-party (same domain) or third-party (cross-domain)
2. **Storage type**: This where the storage is saved on the browser (local storage vs. cookies)
3. **Browser storage vs. server storage**: Whether the data is stored only on the end user's current browser, or also gets sent to WalkMe's servers (more on this below)

# What is Server Storage?

Server storage syncs the user state between different browsers through the WalkMe player server.

The following user state is stored in the server storage and synced to the different browsers:

1. Autoplay once evaluations
2. Goal Completion
3. TeachMe course completion
4. WalkMe data

Please note

Keep in mind the following regarding server storage:

- It requires the user ID to be defined and does not work with the WalkMe generated user ID
- It requires the user ID to be found in the client on runtime
- It requires the user ID be less then 60 characters
- Data is synced periodically every 24 hours (can be configured by R&D), which means that if you already have a state in the client, and the state changed in another browser, it will take a couple of hours for it to be updated

# Extension Storage

Extension storage is intended to increase the range of WalkMe storage options, and eventually become the default storage setting for implementations with an extension. WalkMe-related data will be securely stored within the extension settings and doesn’t require adjustments to browser security settings.

Extension storage is compatible with Chrome, Firefox, Edge Chromium, Safari.

## Storage API

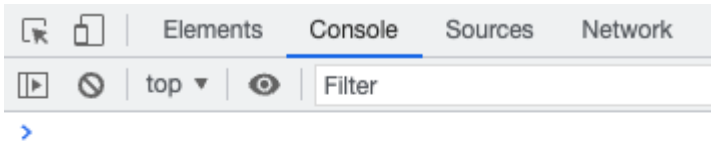
The storage API is a new browser-based API that provides a straightforward and unified interface for managing different types of storage. Use it to manage data more efficiently and selectively, without possibly losing important information or having to log back in.

It replaces the need for clearing browser cache and cookies for testing or troubleshooting WalkMe content that relies on browser-level data. Clearing cache and cookies has been a common way to reset the WalkMe-related data on the local level, but it also deletes all stored data and requires the

user to log back into the website.

Additionally, the storage API is a useful part of WalkMe's gradual rollout of extension storage. With extension storage, clearing browser cache and cookies is not an option. Instead, use the storage API when needing to clear WalkMe data.

The API is available in the browser developer tools console. To access the console, right click on the web page and select **inspect.** Then, click on the console tab.




Here is the list of available commands:

### WalkMeAPI.storage.getAll()

- Returns all WalkMe data stored locally in the current browser
- Use this command to see all pre-existing data and the data you set with any of the below keys


### WalkMeAPI.storage.removeAll()

- Removes all WalkMe data stored locally in current browser
- Default WalkMe data keys will be reset after a page refresh

### WalkMeAPI.storage.setItem()

- Use to create a custom data key for testing purposes
- Accepts two mandatory fields and one optional:

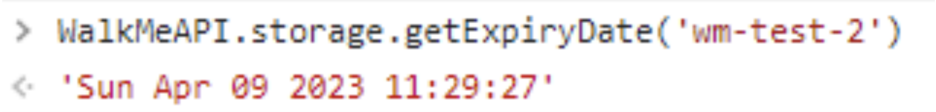
- **Key**: String name of the key you want to add to the storage. We recommend adding “wm-” by default as a prefix to ensure the key is unique.
- **Value**: The value of the key you want to add; can be a string or number
- **(optional) Expiry time in seconds**: Amount of seconds you’d like the key to exist in your local storage. If not specified, a default value of two years will be assigned.

- Examples:
    - WalkMeAPI.storage.setItem(‘test-1’,123)  sets a key with the name ‘wm-test-1’ and a value of 123 for two years. If you set this key and run WalkMeAPI.storage.getAll() you will get the below result:

- WalkMeAPI.storage.setItem(‘test-2’,’Test value’, 360)  sets a key with the name ‘wm-test-2’ and a value of ‘Test value’ for 360 seconds.


### WalkMeAPI.storage.getExpiryDate()

- Returns the date and time when the data key will expire
- Accepts the key name
- Example:
    - WalkMeAPI.storage.getExpiryDate(‘wm-test-2’)



Please reach out to your WalkMe contact or contact Support if you have any questions or need assistance with your storage settings.