STORAGE SCRIPT INTERFACE

View online: https://www.construct.net/en/make-games/manuals/construct-3/scripting/scripting-reference/interfaces/istorage

The `IStorage` interface provides access to storage for the project. It essentially wraps a simple key-value storage engine based on IndexedDB. This means any data that can be stored in IndexedDB can be stored with these methods, such as numbers, strings, Blobs, etc. It is typically accessed by the IRuntime `storage` property.

This interface accesses the same storage as the Local Storage plugin. Therefore an item stored from the event sheet can also be read from script, and vice versa. Note however that Construct expressions can only be strings or numbers, so if a script stores a different type it cannot be used in the event sheet.

As with the Local Storage plugin, the storage is unique to the specific project. It is not shared with any other projects or other website storage, even on the same origin.

Examples

See the Local storage - script example for a demonstration of using these storage APIs to track a high score.

Storage APIs

async getItem(key)

Read an item from storage. Returns a promise that resolves to the value of the item if it exists in storage, else `null` if the item does not exist in storage.

> *If an error occurs when reading from storage, this resolves with `null` instead of throwing an exception.*

async setItem(key, value)

Write an item to storage. Returns a promise that resolves when the write has completed.

> *If the write fails - most commonly due to using up all available storage space - the promise will reject. To ensure this does not crash the game, ensure calls are in a `try...catch` block.*

async removeItem(key)

Delete an item from storage. Returns a promise that resolves when the removal has completed.

async clear()

Delete all items from storage. Returns a promise that resolves when the clear has completed.

async keys()

Retrieve a list of all keys in storage. Returns a promise that resolves to an array of key names.