Update collection | WeWeb Documentation

|  |  |
| --- | --- |
| URL source | <https://docs.weweb.io/workflows/actions/update-collection.html> |
| Date de scraping | 2025-04-08T15:57:06.668854+00:00 |

## Update collection ​

The Update collection action optimizes UI updates after back-end data modifications by eliminating the traditional pattern of making an API request followed by a fetch collection action to refresh the UI.

## The problem it solves ​

Previously in WeWeb, when modifying data through an API request (like deleting a user or updating a record), you had to follow a two-step process:  
Make an API request to modify back-end data Fetch the entire collection again to update the UI  
  
This approach had some drawbacks:  
Required two server requests Created unnecessary network traffic Increased server load  
Ultimately, these meant potentially unnecessary load times for the user of your app.

### Images

<https://docs.weweb.io/assets/updatecollection1.BpPR_t7z.gif>

## The solution ​

The Update collection action enables immediate UI updates by directly modifying the collection after a successful API request, eliminating the need to re-fetch the collection. This means you can:  
Immediately update your collection Update your UI instantly Use just a single API request to update your back-end data

### Images

<https://docs.weweb.io/assets/updatecollection2.TZDgti4e.gif>

## Use cases ​

Immediate user feedback Temporary data views Performance optimization Enhanced user experience Client-side data manipulation

## Update types ​

## Replace All ​

The Replace All update type overwrites the entire content of a collection with new data. In the example below, we are setting the collection to have a text value of "No collection" and using the Log action to display it in the editor's Logs tab. This will show you the updated content of the collection after it's been replaced:

### Images

<https://docs.weweb.io/assets/updatecollection3.EXvBz_BK.gif>

## Update ​

Updates specific records in a collection while preserving the rest of the collection data.

### Images

<https://docs.weweb.io/assets/updatecollection4.DOXUL6mZ.png>

## Configuration ​

Position Type  
By index : Update by position (0 = first record) By id : Update by unique identifier  
Merge  
ON : Updates only specified fields, keeps others OFF : Replaces entire record with new data  
Data  
Accepts an object: {"fieldName": "newValue"} Defines what to update in the record

## Example with Merge ON: ​

When modifying records with Merge ON , you can update specific fields while preserving other data:  
Collection : characters Position : 1 (second record) Merge : ON Data : {"status": "Dead"}

### Images

<https://docs.weweb.io/assets/updatecollection5.14uTk6fK.png>

## Example with Merge OFF: ​

With Merge OFF , the provided data completely replaces the existing record:  
Collection : characters Position : 1 (second record) Merge : OFF Data : {"status": "Dead"}

### Images

<https://docs.weweb.io/assets/updatecollection6.DuERF-tl.png>

## Insert ​

The Insert type adds a new record to your collection at a specified position (index), allowing you to temporarily add data while preserving existing records.

### Images

<https://docs.weweb.io/assets/updatecollection7.CR3Pa8Ml.png>

## Delete ​

The Delete type removes a single record from your collection based on position or ID, leaving all other records unchanged.

## Refresh filters and sort ​

Refresh Filters and Refresh Sort control whether collection settings are reapplied after updating data.  
Filters  
ON : items that no longer match your filters are automatically removed from the collection after your update OFF : items stay in the collection view until manual refresh, even if they no longer match filters  
Sort  
ON : items automatically move to their new positions when sorting values change OFF : items maintain their current positions regardless of value changes