

How to Preserve an Existing Field Value in Your Iterable Pipeline

Last Modified on 10/01/2026 10:12 am EDT

What You'll Learn in This Article:

- Why an Iterable pipeline overwrites profile fields on every submission, and when that's a problem.
- How to look up the subscriber's existing Iterable profile before you update it.
- How to order your mappings so an existing value is kept, and a new value is only written when the field is empty.

By default, every time a subscriber submits a Campaign, your Iterable pipeline sends the mapped fields to Iterable and overwrites whatever was already on that user's profile. That's usually what you want, but not always. Some fields should only be set once, such as the date someone first signed up, their original signup source, or the first Campaign they converted on.

This article walks through an example pipeline that sets a **signUpDate** field in Iterable to today's date the first time a subscriber comes through, and leaves the original date untouched every time after that. You can use the same pattern for any field you want to preserve.

How It Works

The pipeline has four tasks. The key idea is in task 3: you map the new value first, then map the existing value from Iterable to the same field **below** it. Mappings run top to bottom, so when the subscriber already has a value in Iterable, the second mapping replaces the new one and the original value is sent back unchanged. When they don't have a value yet, there's nothing for the second mapping to copy, so the new value is kept.

#	Task	Task type	What it does
1	Get Iterable Profile	Iterable API (Get User)	Looks up the subscriber in Iterable by email and returns their current profile, including any existing signUpDate.
2	Set Current Date	Render Template	Creates a todayDate value containing the current date and time.
3	Map Iterable Profile	Map Data	Builds the profile to send. Maps todayDate to signUpDate, then maps the existing signUpDate from task 1 to the same field so it wins when present.
4	Update Profile	Iterable API (Update User)	Sends the mapped profile from task 3 to Iterable. This is a subtask of task 3.

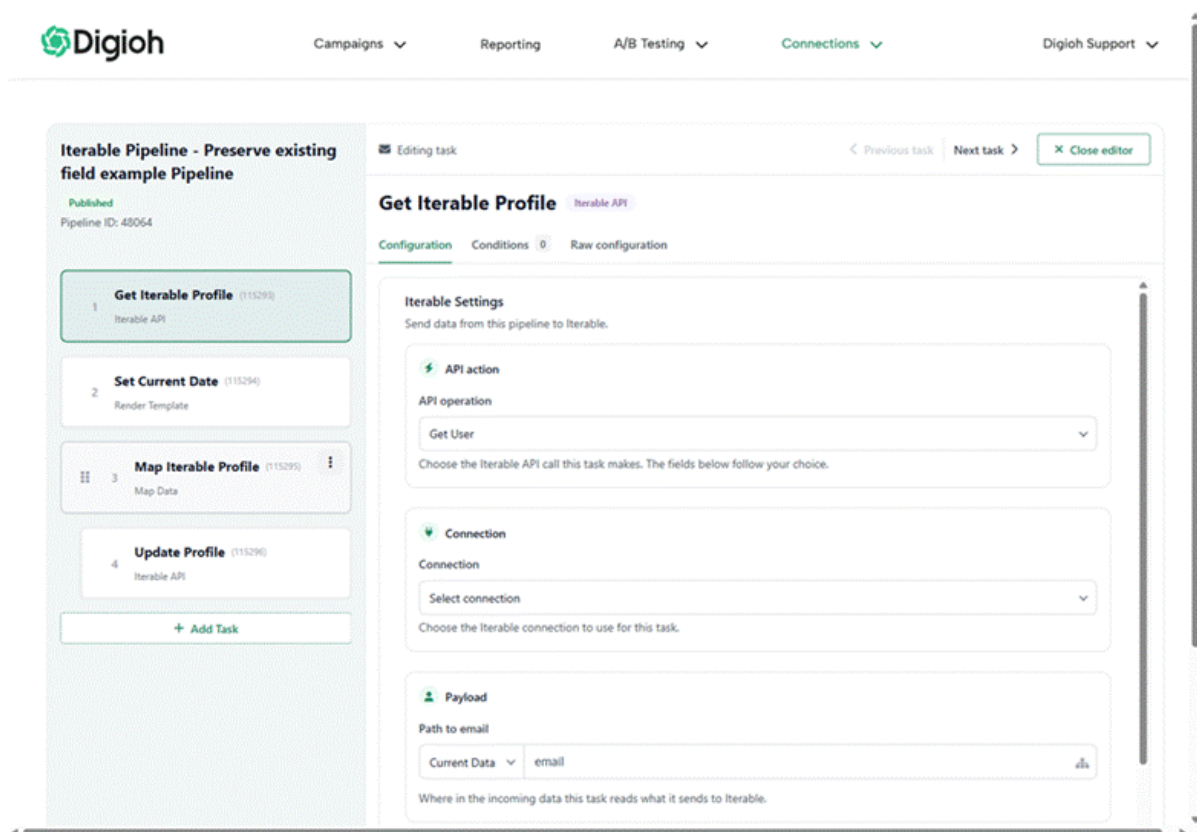
Before You Begin

- You'll need an Iterable connection set up in Digioh. If you haven't connected Iterable yet, do that first under **Connections > Connections**.
- Decide which Iterable field you want to preserve, and confirm its exact name in Iterable (for example, signUpDate). Field names are case-sensitive.
- You can build these tasks into your existing Iterable pipeline, or create a new one under **Connections > Pipelines** and click **Create Pipeline**.

Step 1: Get the Subscriber's Iterable Profile

The first task looks up the subscriber in Iterable so the pipeline knows what's already on their profile.

1. In your pipeline, click **Add Task** and choose the **Iterable API** task. Name it something clear, like **Get Iterable Profile**.
2. Under **API action**, set **API operation** to **Get User**.
3. Under **Connection**, choose your Iterable connection.
4. Under **Payload**, set **Path to email** to **Current Data** and email. This tells the task to look up the user by the email address that came in with the submission.
5. Click **Save**.



This task doesn't change anything in Iterable. It only reads the profile so the later tasks can use

it.

Step 2: Set the Current Date

Next, create the value you want to write when the field is empty. In this example, that's today's date.

1. Click **Add Task** and choose **Render Template**. Name it **Set Current Date**.
2. Leave **Input data** set to **Current Data**.
3. Set **Template language** to **Liquid** and **Output format** to **JSON**.
4. In the **Template editor**, enter the following:

```
{  
  
  "todayDate" : "{{ date.now | date.to_string '%Y-%m-%d %H:%M:%S' }}"  
  
}
```

5. Click **Save**.

The screenshot shows the Digioh web interface. At the top, there's a navigation bar with 'Digioh' logo and links for 'Campaigns', 'Reporting', 'A/B Testing', 'Connections', and 'Digioh Support'. The main content area is titled 'Iterable Pipeline - Preserve existing field example Pipeline' with a 'Published' status and 'Pipeline ID: 48064'. On the left, a task list shows four steps: '1 Get Iterable Profile (115293)', '2 Set Current Date (115294)', '3 Map Iterable Profile (115295)', and '4 Update Profile (115296)'. The 'Set Current Date' task is selected and expanded on the right. It shows the 'Configuration' tab with 'Input data' set to 'Current Data', 'Template language' set to 'Liquid', and 'Output format' set to 'JSON'. Below this is the 'Template editor' with a code editor containing the Liquid template:

```
{  
  "todayDate" : "{{ date.now | date.to_string '%Y-%m-%d %H:%M:%S' }}"  
}
```

. An 'Auto format' button is next to the code editor. A note at the bottom right of the template editor states: 'No run of this pipeline has data for that input, so there is nothing to show.'



If your Campaign already collects the value you want to write, such as a hidden field, you can skip this task and map that form field directly in Step 3 instead.

Step 3: Map the New Value, Then the Existing Value

This is where the preserve logic happens. In the Map Data task, you'll add two mappings that both point to the same Iterable field, in a specific order.

1. Click **Add Task** and choose **Map Data**. Name it **Map Iterable Profile**.
2. Add your standard mappings first, such as email → email, firstName → dataFields.firstName, and lastName → dataFields.lastName.
3. Click **Add Mapping** to add the **new value**:

Source: Data field → **Set Current Date** → todayDate

Destination: dataFields.signUpDate

Transform: Date

4. Click **Add Mapping** again, directly below it, to add the **existing value**:

Source: Data field → **Get Iterable Profile** → dataFields.signUpDate

Destination: dataFields.signUpDate

Transform: Date

5. Turn on **Remove unmapped fields** so only the fields you mapped are sent to Iterable.
6. Click **Save**.

Iterable Pipeline - Preserve existing field example Pipeline

Published
Pipeline ID: 48064

1 **Get Iterable Profile** (115293)
Iterable API

2 **Set Current Date** (115294)
Render Template

3 **Map Iterable Profile** (115295)
Map Data

4 **Update Profile** (115296)
Iterable API

+ Add Task

Editing task

< Previous task

Next task >

Close editor

Map Iterable Profile

Map Data

Configuration

Conditions 0

Raw configuration

Mappings (5)

Map incoming fields or fixed values to the output data.

+ Add Mapping

Source	Destination	Fallback value	Transform
firstName	→ dataFields.firstName	-	None
lastName	→ dataFields.lastName	-	None
email	→ email	-	None
Set Current Date .todayDate	→ dataFields.signUp	-	Date (10/01/2026)
Get Iterable Profile .dataFields	→ dataFields.signUp	-	Date (10/01/2026)

+ Add Mapping

☒ Remove unmapped fields

Only the fields mapped above carry on to the next task. Everything else the item arrived with is dropped. Leave it off to add the mapped fields and keep the rest.



The order of these two mappings is what makes this work. The existing-value mapping must be **below** the new-value mapping. If you reverse them, the current date will overwrite the original value every time. You can drag mappings by the handle on the left to reorder them.

Here's what happens on each submission:

Subscriber in Iterable	Mapping 4 (todayDate)	Mapping 5 (existing signUpDate)	Sent to Iterable
New, or no signUpDate yet	Sets signUpDate to today	Nothing to copy, so it's skipped	Today's date
Already has a signUpDate	Sets signUpDate to today	Replaces it with the existing value	The original date, unchanged

Step 4: Update the Profile in Iterable

The last task sends the mapped profile to Iterable.

1. On the **Map Iterable Profile** task, open the task menu (⋮) and click **Add Subtask**. Choose the **Iterable API** task and name it **Update Profile**.
2. Set **API operation** to **Update User**.
3. Under **Connection**, choose your Iterable connection.

4. Under **Payload**, set **Path to user** to **Map Iterable Profile**. This sends the output of the Map Data task, not the original submission.
5. Leave **Merge nested objects** on, so the fields you send are merged into the user's existing data fields instead of replacing them all.
6. Click **Save**.

Step 5: Publish, and Connect to Your Campaign

1. Once you're completed with updating your pipeline, click **Publish**. Changes don't go live until the pipeline is published.
2. If the pipeline isn't already connected to your Campaign, add it from the Campaign's **Connections** settings.

Using This for Other Fields

The same pattern works for any field you only want to set once. Swap out the pieces like this:

- **The new value** can come from a Render Template task (like the date above), a form field, or a fixed value.
- **The existing value** always comes from the Get Iterable Profile task, using the field's path on the profile, such as `dataFields.originalSource`.
- Both mappings point to the same destination, with the existing-value mapping last.

You can preserve more than one field in the same pipeline. Just add a new-value and existing-value pair for each one.
