OneSignal user identity and event specification kit

A contract on identifiers, events, and error handling that development and marketing agree on

A practical IXC resource for adoption decisions and delivery

Published 2026.09 · English edition · PDF + Excel worksheets

Using this resource

Defines the user relationships and event fields a campaign needs and sets implementation criteria for handling duplicates, delays, and consent changes.

Backend and app developers, data owners, CRM marketers, and QA owners

Contents

01 Separate users from message recipients 3

02 Define login, logout, and account deletion as separate operations 4

03 Define each event at the moment its business action is committed 5

04 Completed example: internal events to API fields 6

05 Decide the outcome for missing, duplicate, and out-of-order data 7

06 Event contract and change approval worksheet 8

How to complete and reuse it

Read the criteria in this PDF, then fill the blank fields with your actual conditions and anything still unconfirmed. The Excel file for the same practice area has a worksheet for each resource. Keep the worked examples separate from real customer cases and confirmed quotes.

With attribution, you may quote and adapt this material for internal review and your own work. The latest files are at www.ixc.co.kr/en/assets.

Separate users from message recipients

A member in your business system, a OneSignal User, and a channel Subscription are not the same identifier. Settle this relationship first so that messages meant for another person do not get mixed in during account switching or across multiple devices.

Item	Role	Completed example
User	The user unit in OneSignal	Relationship across multiple channels and devices
External ID	Stable member ID set by your service	usr_demo_01
Subscription	Recipient unit, such as a device or email	Kept separate from the member ID
Business correlation key	Identifies the workspace and invitation the work belongs to	ws_demo_01 + inv_demo_01

The completed example IXC-DEMO-01 guides members of a fictional B2B collaboration service who have accepted a team invitation toward completing a shared task. It links user_id to the External ID and keeps workspace_id and invitation_id as business relationships. Using workspace_id, to which several members belong, as the External ID would group the wrong people together. Also check whether a value that can change, such as an email address, serves as the member's only internal key.

> **Design example and expected outcomes**
>
> The example identifiers are synthetic values. Event delivery and message receipt tests on a real OneSignal account have not been performed. The field tables and sample inputs are drafts for writing an implementation specification, not approved API integration code or a record of completed tests.

Define login, logout, and account deletion as separate operations

A user logging in is not consent to receive messages. For the member relationship, the device's subscription status, and marketing consent, decide who updates each and how failures are handled.

OneSignal.login identifies the user by External ID and links the Subscription. logout clears the current identity context and reverts to an anonymous state. Deleting data and unsubscribing are separate operations. In this example, the logged-in member ID is taken from the server's authentication result, and during account switching the previous member's events are not sent under the new member ID. Delayed jobs also preserve the member ID that applied when the event occurred.

Account lifecycle agreement

Item to define	Evidence checked	Your entry	Owner and status
When is the member ID linked?	Login and session restore flows		
How are notifications blocked during account switching?	Users and Subscriptions before and after linking		
Who applies unsubscribes and consent changes?	Channel status and consent history		
How is resending blocked after account deletion?	Deletion handling and queued deliveries		

Also check for re-creation after deletion

On account deletion, assign owners for handling the member source record, the OneSignal deletion scope, queued deliveries, and reprocessing records. If you stop at confirming that the delete API succeeded, a remaining integration job can create the user again. Also run a separate test that excludes users who withdrew consent from follow-up sends.

Define each event at the moment its business action is committed

Distinguish a tap or click on screen from the business action being saved. Retries and performance measurement keep the same meaning only when there is a producer that emits the success event and a source record.

Event	Producer and timing	Required business relationships
invitation_accepted	Invitation server: acceptance committed	Member, workspace, invitation
shared_task_completed	Task server: shared task completion committed	Same member, workspace, invitation

Both events require the string fields user_id, workspace_id, and invitation_id. This design has the server record the relationship to the invitation when a shared task is created. If you cannot confirm who performed a task and which invitation it followed, do not count it as a completion for that campaign. The name expresses the meaning of the action, and workspace- or campaign-specific values go in properties. Treat user properties as current state and events as facts tied to a point in time.

Common field	Format	What the specification defines
event_id	Unique UUID string	Created when the business action occurs; unchanged on retry
occurred_at	ISO 8601 UTC timestamp	When the business action was committed
received_at	ISO 8601 UTC timestamp	When the relay server received it
schema_version	Integer	Field meanings and change history

Completed example: internal events to API fields

Map the internal names of business events to the external API fields. A resend of the same event keeps the same identifier, and a different event gets a new one.

Internal event field	Completed example
campaign_id · user_id	IXC-DEMO-01 · usr_demo_01
workspace_id · invitation_id	ws_demo_01 · inv_demo_01
name · schema_version	invitation_accepted · 1
event_id	0a16ce21-7e35-4f9e-a5ac-9a331944ab01
occurred_at	2026-09-10T01:00:00Z

Field sent to OneSignal	Completed example and mapping
name · external_id	name · user_id
timestamp	occurred_at
idempotency_key	Use the event_id UUID as is
properties	workspace_id, invitation_id, campaign_id, schema_version

This table designs one item in the events array. For shared_task_completed, enter the same business relationships, a new event_id, and the actual completion time. received_at stays in the relay server's observability records and does not overwrite the time the business action occurred. An HTTP 202 response from the API can still include errors for individual events, so separate user identification failures from successfully received events, item by item. Raw task content and invitation tokens are not needed for the campaign decision, so they are left out of the fields sent.

Decide the outcome for missing, duplicate, and out-of-order data

Handling every data error as a retry can repeat the same reminder. Separate input that cannot be processed, input to retry, and input already processed.

Input condition	Expected handling	Evidence to keep
Member or invitation ID missing	Quarantine; reprocess after fixing the source	Field error and source ID
Same event_id resent	Treat as a duplicate; block further entry	Record of processed events
Completion arrives before acceptance	Keep the completed state; exclude from the reminder	Occurrence and receipt times
Completion event from another workspace	Not counted as completion of the current task	Correlation key comparison
Pending events left after account deletion	Block delivery; clean up remaining jobs	Deletion and processing records

The internal processor keeps each event_id and its processing result to control duplicate execution of business actions. OneSignal's idempotency_key suppresses duplicate events with the same UUID for 4 hours on a best-effort basis, so do not treat it as a permanent safeguard against duplicates. Keep tracking the original event ID through long delays and operator reprocessing. Do not assume that supplying timestamp resolves out-of-order arrival on its own; compare against the final business state in the source.

> **Hold event delivery when the account is not confirmed**
>
> If usr_demo_01 could not be linked to a OneSignal User, do not substitute an anonymous user or a shared ID. Check the authentication and linking errors, and decide whether the same event can be reprocessed. If, by the time of recovery, the user has already finished the task or changed consent, reassess the earlier send plan as well.

Event contract and change approval worksheet

An event definition sheet is ready for handover to development only when it carries both the field list and the test evidence. For a field whose meaning changes, also agree on how existing consumers will read it.

Event specification and verification agreement

Item to define	Evidence checked	Your entry	Owner and status
Which successful save does the event represent?	Event name, source record, and producer		
Are field types, required values, and relationships correct?	Valid, missing, and other-workspace input		
Are duplicates and events that arrive out of order handled?	Processing records and the final business state		
Which consumers does a change affect?	Version, journeys, measurement, and recovery plan		

Link each event to its owning development team, the campaigns and metrics that use it, example input, expected outcome, actual result, and log location. Leave the example's actual result as not run. Before deployment, check input that mixes old and new versions, and record the change approver and when the change takes effect. When you remove a field or change its meaning, update the journey conditions and measurement queries along with it, so the same event name does not produce different results.

> **Integration scope to set with IXC**
>
> IXC's Customer Data Integration service defines the events and properties a campaign needs, and implements identifier and event delivery or reviews your development team's implementation. Based on the specification and error-handling items you have written, we agree on the integration scope, the role of measurement tools, and acceptance tests.

Publication and use

Based on the member identifiers and event definitions you use today and the conditions of your first campaign, we discuss data integration and acceptance testing.

IXC Corp.

CEO Nam Kichul | Business registration no. 109-86-56732
4F 408, Chuncheon ICT Venture Center
7 Huseok-ro 462beon-gil, Chuncheon-si, Gangwon-do, Republic of Korea
+82-33-242-0210 | info@ixc.co.kr

www.ixc.co.kr

www.ixc.co.kr/en/services/customer-data-integration

www.ixc.co.kr/en/assets/onesignal-event-specification

References

OneSignal Users

https://documentation.onesignal.com/docs/en/users

OneSignal Web SDK reference

https://documentation.onesignal.com/docs/en/web-sdk-reference

OneSignal Custom events

https://documentation.onesignal.com/docs/en/custom-events

OneSignal Create custom events

https://documentation.onesignal.com/reference/create-custom-events

OneSignal Delete users

https://documentation.onesignal.com/docs/en/delete-users

IXC OneSignal SDK, customer identity, and integration guide

https://www.ixc.co.kr/en/solutions/onesignal/integration

IXC Customer Data Integration

https://www.ixc.co.kr/en/services/customer-data-integration