

First sync of the message journal

A device that adds an account with a SylkServer journal has no history of it. The journal holds years of it, tens of thousands of entries. The first sync brings it in without freezing the client, flooding the user with badges and notifications, or sending receipts for old messages to the people who wrote them.

The rules here are the same for every client. The last section says where Blink Qt implements them.

The journal is a snapshot

When a device asks for the journal, every message in it already has its final state, and removed messages are not in it any more. A first sync therefore only inserts messages. The order does not matter, and the events in the journal have nothing left to act on:

Entry	On a first sync
text, file transfer, location, metadata, call record	stored
message/imdn (delivery and display receipts)	skipped: the message carries its state
application/sylk-message-remove, conversation remove, messages bulk delete	skipped: what they removed is not in the journal
application/sylk-conversation-read	not applied as an event; used for the read points (below)
text/pgp-public-key	skipped: keys are asked from the server at the end

The state of a message is the recipient's answer to its sender: an outgoing message carries its final state (`delivered`, `displayed`, `error`). An incoming message has no such state; whether the user read it is told by the user's own devices, in the read markers. The read points below turn those into the read state of incoming messages.

Read points

A conversation's **read point** is the later of:

- the time of its latest `application/sylk-conversation-read` marker (read on another device), and
- the time of the user's latest outgoing message in it (who answered had read).

An incoming message at or before its conversation's read point is stored read; a newer one is unread. Read points are made from all the pages a phase downloads before any of them is applied, so every message is stored with its final read state once, and no badge counts a message that a later page would mark read. A conversation read on another device while the first sync is still going moves its read point on.

Sequence

T0 is the moment the first sync starts. The device records it for ever (`first-sync.date`).

1. The recent days

1. Download the journal `since` T0 – 30 days, all of it.
2. Make the read points from those pages.
3. Apply the pages newest first, as a snapshot: inserts only, read state from the read points. Nothing is shown live: no notification, no sound, no receipt, no badge counted up. The contact list and an open conversation are refreshed once per conversation per batch.
4. Finish the first sync: unread badges are counted once, contacts the history mentions are filed, the Messages group is promoted.

The client is usable now. Normal sync continues from the head of the journal, and everything after T0 is handled live, as on any device.

2. The older history, in the background

1. Download the journal from five years ago up to T0 – 30 days. This uses the network only. The position is saved per page, so a quit or a network error resumes where it stopped.
2. Merge the read points of all these pages.
3. Apply the pages newest first, with the same snapshot rules, paced so that the user does not notice (below). Each page is deleted once applied.
4. When all of it is in: file the contacts it mentions, count the badges again, and ask the server for the public keys of the contacts that have none. This is the only key lookup of a first sync.

Receipts

A device never sends a delivery or display receipt for a message older than T0 – 30 days. That history is an archive; its senders were answered by the devices that were there. A message within the 30 days that the user reads or receives on the new device is confirmed as on any device. Importing does not send receipts by itself; they go out when the user reads a conversation, or when a message is delivered live.

Notifications and badges

- No desktop notification for a message older than the start of the client.
- During the recent phase, badges are not counted. They are set once when it ends.
- During the older phase, messages are stored with their final read state, so badges do not change unless an old message was genuinely never read.

Pace of the older history

The user chooses the speed by the progress bar, as for the background decryption. The choice is kept between runs.

Speed	Entries per batch	Pause after a batch	Works
Fast	250	0.05 s	all the time
Normal	50	2 × the time the batch took	a third of the time
Gentle	20	6 × the time the batch took	a seventh of the time
Pause	—	until resumed	—

The recent phase runs at full speed: the first sync is finished only when it is done.

Logging

What the journal brings is logged per entry at debug level only. The Activity log keeps the summaries (sync start and end, pages, first sync finished with its unread count, older history complete) and the errors.

Blink Qt

All in `blink/messages.py` (MessageManager) unless said otherwise.

Recent days	<code>journal_recent_days = 30</code>
Older history from	<code>journal_since_years = 5</code>
Snapshot rules	<code>journal_snapshot_skipped;</code> <code>_apply_snapshot_entries</code>
Recent phase	<code>_fetch_server_history</code> (download), <code>_apply_snapshot_recent</code>
Older phase	<code>_apply_snapshot_backlog</code> (thread <code>journal-backlog, nice 10</code>), <code>_download_snapshot_older</code> , <code>_run_snapshot_backlog</code>
Read points	<code>_snapshot_read_points</code> , <code>_merge_read_points</code> ; at insert in <code>_journal_store</code> ; <code>MessageHistory.apply_read_points</code> (<code>blink/history.py</code>)
Receipts	<code>_before_first_sync</code> , called by <code>send_imdn_message</code>
Notifications	<code>MessageNotifier._from_the_past</code> (<code>blink/notifications.py</code>)
Badges	<code>MessageHistory.unread_counts</code> , <code>settle_first_sync_read</code> (<code>blink/history.py</code>)
Pace	<code>journal_import_speeds</code> , <code>set_journal_import_speed</code> ; the button in <code>blink/mainwindow.py</code>
GUI updates	<code>MessageHistory.flush_batched_stores</code> (<code>blink/history.py</code>)

Files in `~/./blink/journal/<account>/:`

File	Meaning
<code>first-sync.marker</code>	the recent phase is running (its start time)
<code>first-sync.date</code>	T0, kept for ever (receipts)
<code>snapshot/*.json</code>	the pages of the recent days
<code>snapshot/complete</code>	the recent days are downloaded
<code>snapshot/read-points</code>	<code>{conversation: read point}</code>
<code>snapshot/backfill</code>	the older phase: <code>until</code> , <code>cursor</code> , <code>fetch</code> , <code>download</code> , <code>points</code> , <code>done</code> , <code>total</code> , <code>recent</code>
<code>snapshot/older/*.json</code>	the pages of the older history

`~/./blink/journal_import_speed` holds the chosen speed.

The journal log (`~/./blink/logs/journal.txt`) shows the sequence: `sync` start, `download` page, `snapshot recent` start, `read points` made, `snapshot` page applied, `read points`, `snapshot recent` end, `settle first_sync`, `backfill` start, `backfill` download page, `read points` made `older=yes`, `snapshot` page applied, `snapshot` complete.