Cal.diy capabilities

Use this index to choose the Cal.diy operation family that owns your task and to interpret its AIP contract before invocation. The reviewed catalogue contains 81 unique capabilities. Each id begins with cap:caldiy: and maps to one fixed pro

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/cal-diy/capabilities/README.md

The source-derived capability snapshot contains stable catalogue markers for tooling and review. The family pages own inputs, results, side effects, and task guidance; the JSON snapshot does not replace the AIP schemas or the code that constructs each capability.

Catalogue identity

Property	Value
Reviewed AIP source	`97be86e9efedf07ecf1783b03800f683f107fb04`
Pinned Cal.diy source	`f00434927386c9ecdcbd7e6c5f82d22044a245bc`
Connector profile	`aip.connector.cal_diy.v1`
Capability prefix	`cap:cal_diy:`
Operations	81
Reads	36
Mutations	45
Approval required	47

A different connector artifact or provider revision requires a new snapshot and review. Do not infer support from a matching capability id alone.

Choose a family

Family	Page	Total	Reads	Mutations	Low	Medium	High	Approval
Profile	`profile.md`	2	1	1	1	1	0	1
Event types and private links	`event-types-and-private-links.md`	15	5	10	5	6	4	10
Slots and reservations	`slots-and-reservations.md`	5	2	3	2	2	1	3
Bookings	`bookings.md`	22	10	12	8	4	10	14
Calendars and free/busy	`calendars-and-free-busy.md`	21	11	10	11	6	4	10
Conferencing	`conferencing.md`	5	2	3	2	2	1	3
Schedules	`schedules.md`	6	3	3	3	2	1	3
User webhooks	`webhooks.md`	5	2	3	2	2	1	3
Total		**81**	**36**	**45**	**34**	**25**	**22**	**47**

Booking attendee and artifact operations are documented separately in `booking-attendees-and-artifacts.md`, but they remain part of the 22-operation booking family. The family table counts each operation once.

Read a capability id

The suffix after `cap:cal_diy:` names the resource and operation:

```
TEXT
cap:cal_diy:booking.create
■             ■       ■■ operation
■             ■■■■■■■■■ resource family
■■■■■■■■■■■■■■■■■■■■■■■■ connector namespace
```

Nested resources add stable segments:

```
TEXT
cap:cal_diy:event_type.private_link.create
cap:cal_diy:calendar.connection.event.update
cap:cal_diy:booking.conferencing_session.list
```

Do not construct an undocumented suffix from a provider route. Discovery and the source-derived snapshot are the operation inventory.

Each advertised capability has bindings for `aip.native.http.v1` and `aip.connector.cal_diy.v1`. Binding metadata identifies the Cal.diy system, connector id, operation suffix, provider method, fixed path template, and selected API version.

HTTP method and operation class

Provider method	Count	AIP class
`GET`	36	Read
`POST`	22	Mutation
`PATCH`	10	Mutation
`PUT`	1	Mutation
`DELETE`	12	Mutation

The connector classifies a capability as a read only when its provider method is `GET`. Every other method receives the mutation contract, even when a provider operation sounds reversible or administrative.

Risk legend

Risk is a declared input to policy, not authorization by itself.

Risk	Count	Cal.diy classification rule	Required decision
Low	34	Read other than recording or transcript access	Authenticate, authorize, and apply data policy
Medium	25	Mutation that is neither destructive nor one of the named high-impact booking transitions	Require approval and an idempotency key
High	22	Destructive operation, recording or transcript read, booking reschedule, confirmation, absence update, or reassignment	Apply the stricter tenant policy and inspect exact impact

`booking.create` is medium risk in the reviewed source but still requires approval, external-account idempotency, and a plan before commit. Never use a risk label to skip the other contract fields.

Approval legend

Approval is required for:

- all 45 mutations; and
- `cap:cal_diy:booking.recording.list` and

`cap:cal_diy:booking.transcript.list` because they may expose restricted meeting data.

The advertised approval policy uses the tenant-policy selector, a 900,000 ms validity window, and three evidence requirements:

- reason;
- input snapshot;
- policy decision.

The capability does not grant delegated approval authority. An approval must bind the actual authenticated actor, tenant, capability, and reviewed input. Expiry or input change requires a new decision.

Idempotency and retry legend

Contract	Reads	Mutations
Idempotency key	Optional	Required
Key scope	Action	External account
Collision behavior	Revalidate input hash	Revalidate input hash
Connector retry support	Yes	No
Retry safety	Safe	Unsafe
Published key TTL	None	None

For a mutation, reuse the same key only with the same capability, account, and canonical input. A different input is an idempotency collision.

The connector records a durable claim before provider dispatch. A completed matching claim can return its stored result. An in-flight claim remains in progress. A dispatch that may have crossed the provider boundary without a

trusted outcome becomes uncertain and must not be retried automatically.

The absence of a contract TTL does not authorize uncontrolled key reuse or deletion of durable state. Retention and reconciliation remain deployment policy.

Execution legend

All 81 capabilities advertise:

Field	Value
Synchronous execution	Supported
Connector-managed async execution	Not supported
Streaming	Not supported
Cancellation	Not supported
Expected completion	Synchronous
Provider timeout	30,000 ms
Expected latency hint	2,000 ms for reads; 5,000 ms for mutations
Maximum queue-delay hint	5,000 ms
Availability target	Not declared

These are capability-contract hints and bounds, not a deployment SLO or qualification result.

Transaction legend

Every mutation declares four modes:

```
TEXT
dry_run | plan | commit | reconcile
```

Only three operations require a validated plan before commit:

- `cap:cal_diy:booking.create`;
- `cap:cal_diy:booking.reschedule`;
- `cap:cal_diy:booking.cancel`.

Their dry run has downstream-validation fidelity. Other mutations declare policy-and-schema fidelity; a successful dry run does not promise that Cal.diy accepted a provider-side preview.

Reconcile interprets durable dispatch evidence. It does not issue a speculative repeat. An unresolved provider outcome requires operator evidence.

Side-effect legend

Every capability declares external network access plus `read` or `write`. Additional markers are additive:

Marker	Operations
`delete`	15 destructive operations
`send_message`	Booking create, reschedule, cancel, confirm, decline, attendee add or remove, and guest add
`identity`	Profile update

The `send_message` marker means the provider operation may update invitations, connected calendars, or customer-visible notifications. It does not guarantee that a particular notification was delivered.

Compensation legend

Eleven create or connect operations name a best-effort compensating capability:

Original	Compensation
`event_type.create`	`event_type.delete`
`event_type.private_link.create`	`event_type.private_link.delete`
`event_type.webhook.create`	`event_type.webhook.delete`
`slot.reservation.create`	`slot.reservation.delete`
`booking.create`	`booking.cancel`
`calendar.connection.event.create`	`calendar.connection.event.delete`
`calendar.event.create`	`calendar.event.delete`
`calendar.selected.add`	`calendar.selected.delete`
`conferencing.connect`	`conferencing.disconnect`
`schedule.create`	`schedule.delete`
`webhook.create`	`webhook.delete`

The declared compensation window is 86,400,000 ms and compensation itself requires approval. It is a new provider operation, not an atomic rollback. Reads require no compensation. Mutations without a named inverse declare that rollback is not supported.

Data legend

The connector declares one of three data sensitivity levels:

- `slot.list` is public;
- profile, private-link, webhook, booking, calendar, recording, and transcript

operations are restricted;

- remaining event-type, slot-reservation, conferencing, and schedule

operations are confidential.

Restricted operations and every event-type operation are marked as containing PII and requiring redaction. This contract is a minimum handling signal. Input or provider output may require stricter classification under deployment policy.

The connector recursively redacts known secret fields from provider results, but data classification and authorization must occur before output is returned to another system or model.

Credential legend

Every operation names the logical issuer `cal_diy_deployment` and an exact scope of the form:

```
TEXT
cal_diy:<operation-suffix>
```

For example, `cap:cal_diy:booking.create` names `cal_diy:booking.create`.

The standalone host's manifest does not require a per-action credential handle because that binary uses one startup-loaded provider credential. This does not make the provider credential optional. The separate connector-library tenant routing composition marks handles as required and validates issuer, tenant, expiry, and operation scope.

OAuth refresh is not allowed through the capability contract.

Use the machine snapshot

`assets/capabilities.json` is a source-derived review artifact with:

- exact source and upstream revisions;
- family, method, risk, approval, destructive, plan, and compensation counts;
- one ordered record per operation;
- stable capability id, provider method and API version;
- risk, approval reason class, side effects, idempotency, execution, data,

credential scope, and compensation markers.

It intentionally omits the full input and output schemas and provider path templates owned by their generated or family references. Treat the source revision as part of the asset identity. A catalogue change without a matching snapshot update is documentation drift and must fail review.

Choose the next page

- Use `profile.md` for authenticated profile reads and updates.
- Use `event-types-and-private-links.md` for event definitions, access links,

and event-scoped webhooks.

- Use `slots-and-reservations.md` before creating a booking.
- Use `bookings.md` for booking state and lifecycle transitions.
- Use `booking-attendees-and-artifacts.md` for people, references, recordings,

transcripts, and conferencing-session results.

- Use `calendars-and-free-busy.md` for connected calendars, events, and

conflict selection.

- Use `conferencing.md`, `schedules.md`, or `webhooks.md` for those provider

resources.

Related documentation

- Cal.diy connector overview
- Cal.diy configuration
- Capabilities and contracts
- Approvals and policy
- Transactions and compensation