Reference Manual

Backup Pro for Magento 2.x — Extmag_Backup

2026-05-24

1 Package layout

Module	Composer package	Role
`Extmag_Backup`	`extmag/backuppro`	Thin registration wrapper. Pulls the core module.
`Extmag_BackupCore`	`extmag/backup-core`	Schema, services, controllers, UI components.

Both modules must be enabled. The wrapper carries no controllers or services of its own — every reference below points at namespaces under `Extmag\BackupCore`.

2 Service contracts (PHP API)

The module ships PHP-only service contracts. There is no `webapi.xml`, so the APIs below are not exposed over REST / SOAP — consume them via Magento DI from your own modules or scripts.

2.1 `Extmag\BackupCore\Api\BackupRepositoryInterface`

CRUD for Backup Set entities.

```
public function save(Backup $backup): Backup;
public function getById($id): Backup;
public function delete(Backup $backup): bool;
public function deleteById($id): bool;
public function getList(SearchCriteriaInterface $searchCriteria):
        SearchResultsInterface;
```

2.2 `Extmag\BackupCore\Api\StorageRepositoryInterface`

CRUD for Storage Set entities.

```
public function save(Storage $storage): Storage;
public function getById($id): Storage;
public function delete(Storage $storage): bool;
public function deleteById($id): bool;
public function getList(SearchCriteriaInterface $searchCriteria):
        SearchResultsInterface;
```

2.3 `Extmag\BackupCore\Api\Data\StorageInterface`

Storage Set entity accessors. Field constants double as `extmag_storage` column names.

```
public function getStorageId(): ?int;
public function getTitle(): ?string;
public function getStorageType(): ?string;   // google_drive | one_drive |
        dropbox | s3
public function getIsEnabled(): ?int;
public function getClientId(): ?string;
public function getClientSecret(): ?string;  // auto-decrypted on read
public function getAuthCode(): ?string;
public function getRefreshToken(): ?string;  // auto-decrypted on read
public function getAccessToken(): ?string;   // auto-decrypted on read
public function getOauthErrorNotifyMess(): ?string;
public function getCreatedAt(): ?string;
public function getUpdatedAt(): ?string;
```

Each getter has a matching `setX(...)` setter. Writes to `client_secret`, `oauth_refresh_token`, `oauth_access_token` are transparently encrypted by `Extmag\BackupCore\Model\Storage::beforeSave()`; reads are transparently decrypted by `afterLoad()` and a per-read fallback in the getter, so persisted encrypted values remain readable even before load events have fired.

2.4 `Extmag\BackupCore\Model\Backup`

Backup Set entity. The Set model exposes typed helpers for the JSON columns:

```
const TYPE_DB = 'db';
const TYPE_FILES = 'files';

public function getBackupTypesArray(): array;       // ['db', 'files']
public function setBackupTypesArray(array $types);  // joins with ','

public function getExcludeDirsArray(): array;       // JSON-decoded
public function setExcludeDirsArray(array $dirs);

public function getExcludeDBTablesArray(): array;   // JSON-decoded

public function getStoragesConfigArray(): array;    // [{storage_id,
        keep_copies, path, enabled, ...}, ...]
public function setStoragesConfigArray(array $config);

public function getLocalStoragesConfigArray(): array; // only rows with
        storage_id === 'local'
```

`extmag_storage_log` rows reference the Set by `backup_id`; the FK constraint uses `ON DELETE SET NULL` so deleting a Set preserves its audit history.

3 System configuration reference

`Stores → Configuration → Extmag → Backup Pro` (`section_id = extmag_backup`).

3.1 General (`extmag_backup/general/...`)

Field	Default	Source model
`active`	`1`	`Magento\Config\Model\Config\Source\Yesno`

The cron driver short-circuits on `active = 0`; no Backup Set fires while the master switch is off.

4 ACL resources

Granted under `Magento_Backend::admin`. Assign each resource individually to limit the blast radius of a compromised admin account.

Resource ID	Grants
`Extmag_BackupCore::config`	`Stores → Configuration → Backup Pro`.
`Extmag_BackupCore::main`	Top-level *Extmag Backups* container.
`Extmag_BackupCore::backups`	Read Backup Sets grid + form.
`Extmag_BackupCore::backups_save`	Save / create Backup Sets.
`Extmag_BackupCore::backups_delete`	Delete Backup Sets.
`Extmag_BackupCore::storages`	Read Storage Sets grid + form.
`Extmag_BackupCore::storages_save`	Save / create Storage Sets.
`Extmag_BackupCore::storages_delete`	Delete Storage Sets.
`Extmag_BackupCore::logs`	Read Backup Logs grid.

5 Cron jobs

Job	Schedule	Purpose
`extmag_backup_create`	`1 * * * *`	Per-hour pass over enabled Backup Sets.
`extmag_backup_log_cleanup`	`0 3 * * *`	Daily prune. Deletes `extmag_backup_log` rows older than 10 days, 500 at a time.

Both jobs live in the Magento `default` cron group. The driver is gated by `extmag_backup/general/active = 1`; the cleanup job has no gate.

6 Database schema

6.1 `extmag_backup` — Backup Sets

Column	Type	Notes
`backup_id`	int unsigned, AI	Primary key.
`title`	varchar(255), NOT NULL	Unique. `EXTMAG_BACKUP_TITLE_UNIQUE`.
`is_enabled`	smallint unsigned, default 1	Per-Set master switch.

Column	Type	Notes
`backup_types`	varchar(255)	CSV — `db,files`.
`exclude_dirs`	text	JSON array of first-level directory names.
`exclude_db_tables`	text	JSON array of table names (exact match).
`frequency`	varchar(50), NOT NULL	One of `1`, `2`, `3`, `4`, `5`, `12`, `18`, `24`, `36`, `48` (hours).
`time_of_day`	varchar(10)	`HH,MM` — used when `frequency` is `24` or `48`.
`folder_name`	varchar(80)	Unique. `EXTMAG_BACKUP_FOLDER_NAME_UNIQUE`. Folder under `var/backups/extmag/` and remote root name.
`storages_config`	text	JSON. One object per destination row — see below.
`created_at`	timestamp	`DEFAULT CURRENT_TIMESTAMP`.
`updated_at`	timestamp	`ON UPDATE CURRENT_TIMESTAMP`.
`last_time`	timestamp	Driver-managed. Advanced to `now + frequency hours` after each pass.

Indexes: `is_enabled`.

`storages_config` JSON shape:

```
[
  {"storage_id":"local","enabled":"1","keep_copies":"5","path":""},
  {"storage_id":"1","enabled":"1","keep_copies":"30","path":"backups/my-store"},
  {"storage_id":"2","enabled":"1","keep_copies":"30","path":"my-
        bucket/magento/"}
]
```

`storage_id = "local"` is the special token for the on-disk destination. Any other value is a foreign reference into `extmag_storage.storage_id`.

6.2 `extmag_storage` — Storage Sets

Column	Type	Notes
`storage_id`	int unsigned, AI	Primary key.
`title`	varchar(255), NOT NULL	Unique. `EXTMAG_STORAGE_TITLE_UNIQUE`.
`storage_type`	varchar(50), NOT NULL	`google_drive` / `one_drive` / `dropbox` / `s3`.

Column	Type	Notes
`is_enabled`	smallint unsigned, default 1	Per-Storage master switch.
`client_id`	varchar(255)	OAuth client id / App Key / AWS access key id.
`client_secret`	text	Encrypted. OAuth client secret / App Secret / AWS secret access key.
`auth_code`	text	OAuth authorization code, or AWS region (S3).
`oauth_refresh_token`	text	Encrypted. Stored after first OAuth exchange.
`oauth_access_token`	text	Encrypted. Rotated on expiry.
`oauth_error_notify_mess`	text	Populated when token refresh fails; admin inbox surface uses it.
`created_at`	timestamp	`DEFAULT CURRENT_TIMESTAMP`.
`updated_at`	timestamp	`ON UPDATE CURRENT_TIMESTAMP`.

Indexes: `is_enabled`, `storage_type`.

`client_secret`, `oauth_refresh_token`, `oauth_access_token` are encrypted with `EncryptorInterface` (`0:3:` / `0:2:` Magento prefix). Cross-environment restores require the same Magento encryption key, otherwise re-link the cloud account.

6.3 `extmag_backup_log` — Audit log

Column	Type	Notes
`entity_id`	int unsigned, AI	Primary key.
`backup_id`	int unsigned, nullable	FK → `extmag_backup.backup_id`, `ON DELETE SET NULL`.
`backup_set_name`	varchar(255)	Snapshot of `extmag_backup.title` at run time.
`overall_status`	varchar(32), default `pending`	`success` / `failed` / `partial` / `pending`.
`error_message`	text	Top-level error if the pass failed before the storage phase.
`started_at`	datetime	When the runner picked up the Set.
`finished_at`	datetime	When the runner finalised the log row.
`duration_sec`	int unsigned	Wall-clock seconds.

Column	Type	Notes
`files_json`	text	JSON array of archive filenames produced by the pass.
`storages_json`	text	JSON array of per-storage status objects — see below.
`created_at`	timestamp	`DEFAULT CURRENT_TIMESTAMP`.

Indexes: `backup_id`.

`storages_json` per-row shape:

```
{"storage_id":"1","title":"Google
        Disk","type":"google_drive","status":"success","message":""}
```

`overall_status = success` requires every row to have `status = "success"`. Any failure flips the overall status to `failed`.

7 Source models

Class	Purpose
`Extmag\BackupCore\Model\Source\StorageType`	Storage type dropdown — Google Drive / OneDrive / Dropbox / Amazon S3.
`Extmag\BackupCore\Model\Source\Frequency`	Backup frequency dropdown — 1, 2, 3, 4, 5, 12, 18, 24, 36, 48 hours.
`Extmag\BackupCore\Model\Source\ExcludeDir`	Multiselect of first-level directories under the Magento root, alphabetised.
`Extmag\BackupCore\Model\Source\Backup\Type`	Backup types multiselect — Database, Files.
`Extmag\BackupCore\Model\Source\Storage\Options`	Storage Sets dropdown for the Backup Set *Storages Configuration* row.

`ExcludeDBTable` is sourced dynamically via the UI component — every table in the configured database is selectable.

8 Backup pass workflow

`Extmag\BackupCore\Service\Backup::create(Extmag\BackupCore\Model\Backup $backupModel)`.

1. Short-circuit if the Set is disabled (`is_enabled = 0`).
2. Create the log row with `overall_status = pending`, `started_at = now`.
3. If `TYPE_FILES` is selected:
    - `var/backups/extmag/<folder>/<timestamp>/files/` is created.
    - The archiver streams the Magento root into one or more `files.tgz` archives, honouring *Exclude Directories*. Archives over 4 GB split into multiple parts automatically.
4. If `TYPE_DB` is selected:
    - `var/backups/extmag/<folder>/<timestamp>/db/db.sql.gz` is produced by Magento's native backup factory, honouring *Exclude DB Tables*.
5. The runner walks `<timestamp>/` and collects the file list into `files_json`.
6. `deleteOldBackups()` enforces the local row's *Copies to Keep* by deleting older `<timestamp>/` folders.
7. `processBackupToStorage()` iterates every enabled storage row:
    - Loads the Storage Set by `storage_id`.
    - Dispatches to the matching helper (`Dropbox`, `GoogleDrive`, `MicrosoftOneDrive`, `AwsS3`).
    - Each helper uploads every file under `<timestamp>/`, then sweeps older remote folders beyond `keep_copies`.
    - Per-row status / message recorded for `storages_json`.
8. The log row is finalised — `overall_status`, `finished_at`, `duration_sec`, `files_json`, `storages_json`.

A single uncaught exception in steps 3–7 sets `overall_status = failed` and records `error_message`. Storage-row failures keep the pass going across the remaining rows but flip `overall_status` to `failed`.

9 Filesystem layout

```
var/backups/extmag/
└── <folder_name>/
    ├── 2026-05-24_03-15-22/
    │   ├── db/
    │   │   └── db.sql.gz
    │   └── files/
    │       └── files.tgz           # or files.tgz.001, files.tgz.002, …
    ├── 2026-05-24_04-15-21/
    │   └── …
    └── 2026-05-25_03-15-23/
        └── …
```

Cloud destinations mirror the same `<timestamp>/` folders under the configured Path. `keep_copies` is enforced separately for local and each cloud row.

10 OAuth flow internals

The redirect target route is registered both in `etc/frontend/routes.xml` and `etc/adminhtml/routes.xml`:

- `extmag_backup` — `/extmag_backup/...`
- `extmag_backupcore` — `/extmag_backupcore/...` (admin grids).

The OAuth redirect URI to register with each provider:

```
https://<your-domain>/extmag_backup/token/show/
```

`Extmag\BackupCore\Controller\Token\Show` renders the resulting authorization code on screen via `Extmag\BackupCore\Block\Token\Show`. The operator copies the code into the Storage Set *Authentication Code* field and saves; the corresponding helper (`Extmag\BackupCore\Helper\GoogleDrive`, `Extmag\BackupCore\Helper\MicrosoftOneDrive`, or `Extmag\BackupCore\Helper\Dropbox`) exchanges it for a refresh token on the next authenticated call.

11 AWS S3 path format

The `Path` field on a Backup Set Cloud Storage row pointing at an S3 Storage Set must be `bucket-name/optional/prefix/`. The `Extmag\BackupCore\Helper\AwsS3::extractBucketAndPrefix()` split assumes the bucket

is the first slash-separated segment; an empty bucket logs `[AWS S3] Upload failed: bucket is empty.` and the row's status is recorded as `failed`.

S3 region resolution:

- `Storage::auth_code` — primary region source.
- Default `us-east-1` if empty.

S3 retention: `Extmag\BackupCore\Helper\AwsS3::clearOldBackups()` lists immediate folders under `cloudRootPath`, sorts lexicographically (the `<timestamp>` format `YYYY-MM-DD_HH-mm-ss` is sort-friendly), and deletes recursively those beyond `keep_copies`.

12 Admin notifications

The module registers three system-message classes under `Extmag\BackupCore\Notification\System\Message\` for Dropbox, Google Drive and OneDrive. They post into the admin inbox when the corresponding Storage Set's `oauth_error_notify_mess` column is populated — typically after a failed token refresh. Re-linking the account from the Storage Set edit form clears the message on the next successful authenticated call.

13 Performance and scale notes

- The backup pass is single-threaded per Set. Multiple Sets fire serially inside one cron tick — long-running Sets can delay later ones; consider staggering with different `frequency` values.
- The native Magento backup factory shells out to `mysqldump` for the database dump. Long catalogs can hold a table lock for tens of seconds; pair with replica-side backups for very large stores.
- Cleanup deletes at most 500 log rows per cron tick to avoid table-level locks. A backlog of years' worth of rows shrinks at 500/day until caught up.
- Upload throughput is bounded by outbound bandwidth from the Magento server to the cloud provider. The driver streams files sequentially per storage row — multi-cloud rows do not parallelise.

14 Compatibility notes

Concern	Notes
Adobe Commerce vs OS	Same code path. No staging-aware joins required — the module owns its own three tables.
Multi-website	The backup pass is global. No per-website / per-store knobs.
Multi-database	Single primary connection only. Split-DB / sales / quote checkouts dump from the `default` connection.
Encryption key rotation	Re-encrypted on next `save()` of each Storage Set. To force, edit and save every Storage Set after rotation.

15 Known limitations

- **No automated restore.** Restoration is left to the operator using native tooling (`mysql`, `tar`). See the *User Guide* for the standard recipe.
- **Frequency granularity is hourly.** The minimum interval is 1 h; sub-hour schedules are not supported by design — the cron job fires hourly.
- **Storage row ordering is the array order.** Upload happens row by row, top to bottom. Put the local row first if you want the on-disk copy to survive a cloud-upload outage.
- **No mid-pass progress.** The log row goes from `pending` straight to `success` / `failed` on completion. There is no incremental percentage surface during the pass.
- **`Time of Day` is server timezone.** It uses Magento's `TimezoneInterface::date()`, which honours the store-scope timezone. Cross-timezone deployments should align the configured Time of Day with the storefront timezone they care about.
- **OAuth tokens are bound to the source environment.** Restoring a database dump into a different environment requires the same Magento encryption key or a re-link of each cloud account.