Cache Locations Contract v1.0 - Media Audit Tool(TM) / Asset Intelligence Engine(TM) - 2026-07-12

Relationship to the Analyzer Fact Contract and Inventory Hints Contract

This is a separate, sibling contract. The Analyzer Fact Contract v1.0 and the Inventory Hints Contract v1.0 are both frozen and unaffected. Facts describe references to specific assets. Inventory hints describe scan scope and an Analyzer's own reserved interface furniture. Cache locations describe neither -- they describe where an Analyzer's own extension writes generated or derivative files, for the sole purpose of clearing them before a reliable front-end visual check. These are different concerns and do not belong in the same contract.

Why this exists

The recommended workflow after trashing unreferenced assets is a front-end visual check before permanently purging anything. That check is meaningless if the front end is still pulling stale cached copies of images that are about to be deleted, or failing to show ones that are still needed. Media Audit Tool(TM) already knows about Joomla's own file-based system cache, and about a fixed set of conventionally-named folders it finds heuristically while scanning (images/thumbs, images/cache, images/resized, and similar). Neither of those covers a case like YOOtheme Pro's own template cache, which lives entirely outside /images and follows no naming convention Media Audit Tool(TM) could guess at on its own. This hook lets an Analyzer declare exactly where its own extension's cache lives, so Clear Image Caches can find and offer to clear it.

The hook

onMediaaudittoolCollectCacheLocations()
Returns:

```
[
   'items' => [
      [
         'path'       => 'templates/yootheme/cache',
         'label'      => 'YOOtheme Pro template cache',
         'confidence' => 'known_safe',
         'severity'   => 'low',
      ],
   ],
]
```

path -- required. A directory path relative to the Joomla site root. Declares a location the Analyzer's own extension writes cached or derivative files to.

label -- optional. A human-readable name shown in the Clear Image Caches list in place of the raw path. Defaults to the raw path if omitted.

confidence -- optional, defaults to 'known_safe' for a declared item. Set to 'heuristic' if the Analyzer cannot confirm its own cache location automatically rebuilds after being cleared.

severity -- optional, defaults to 'high'. Answers a different question than confidence: not "are we sure this regenerates," but "if we are wrong, is the damage reversible and harmless, or does it leave something visibly broken." Set to 'low' if clearing this location without regeneration is, at worst, a temporary performance cost (a slower first page load while something rebuilds). Leave as 'high', or set it explicitly, if clearing this location without regeneration can leave missing or broken content visible to site visitors.

Why two separate fields instead of one

confidence and severity answer different questions and an Analyzer's belief about one says nothing about the other. A location can be confidently safe (known_safe) and still be high severity if that confidence ever turns out to be mistaken -- worth flagging as a "we are sure, but here is what is riding on being right" case. A location can also be unconfirmed (heuristic) but low severity, if the worst realistic outcome of clearing it is nothing worse than a slower rebuild. Collapsing these into a single field would lose exactly the distinction that motivated this contract: the difference between "probably fine" and "if this goes wrong, a customer sees a broken product image on your storefront."

This is not theoretical. Testing surfaced a real case: images/thumbs on a live site held webp thumbnails a J2Commerce category listing depended on. It was treated as safe to clear under the old heuristic-folder logic, with no severity signal at all. Nothing regenerated those files afterward -- not on page refresh, and not even through a fresh image re-upload via J2Commerce's own uploader -- until the missing folder was manually recreated on the server. That incident is exactly what severity is meant to have flagged in advance: heuristic confidence and high severity together, warranting the strongest possible warning before clearing.

Deletion is always non-destructive to the directory itself

Regardless of the declared confidence or severity, Media Audit Tool(TM) only ever empties the contents of a declared cache location. It never removes the directory itself, and it preserves Joomla's own directory-listing protection files (index.html, index.php, .htaccess) inside it. This was also learned the hard way: an earlier version of the cache-clearing code removed the folder along with its contents, and at least one extension did not recreate the missing directory on its own, leaving every file in that location permanently broken rather than simply cleared. An Analyzer implementing this hook does not need to account for this -- it is handled centrally and applies uniformly to every declared location.

Relationship to Joomla's own system cache

This hook is only for cache locations owned by the declaring Analyzer's own extension. Joomla's own file-based system cache (cache/ and administrator/cache/) is handled separately and directly by Media Audit Tool(TM); an Analyzer should not declare it here.

Aggregation

Like Included Directories and Inventory Hints, this hook is aggregated across every enabled Analyzer plugin through Media Audit Pro(TM)'s onMediaaudittoolAggregateCacheLocations, which delegates to each other enabled Analyzer's own onMediaaudittoolCollectCacheLocations and merges the results. An Analyzer plugin only needs to implement the direct hook; aggregation is handled centrally, the same as the other two hooks in this family.

Rollout note

Implement this hook for any Analyzer whose extension writes generated or derivative files to a location Media Audit Tool(TM) would not otherwise find -- outside /images entirely (as with YOOtheme's template cache), or inside /images but under a name Media Audit Tool(TM)'s naming-convention scan does not recognize. Not required for extensions with no cache of their own.

Future considerations (out of scope for v1.0)

Declared cache locations are not currently attributed to the Analyzer that declared them in the stored result -- unlike the Fact Contract and the Inventory Hints Contract, which both carry an explicit analyzer field per record. The Inventory Hints Contract's transparency requirement ("which Analyzer declared each one must be visible somewhere in the admin UI") does not yet have an equivalent here. A future revision could add an analyzer field to each declared item and surface that attribution in the Clear Image Caches list, the same way Inventory Hints does for reserved paths. Not needed for v1.0 -- noted here so it isn't lost.