# LTFSArchiver 1.5.2 API summary (version refers to this API doc)

## Synchronous services

For the following services the request is processed immediately and the result is returned in response to the request.

|  |  |  |  |
| --- | --- | --- | --- |
| **Service** | **Input** | **Output** | **Description** |
| Tape Management |  |  | Manage tapes and pools |
| WithdrawTape | TapeID=<id>,  PoolName=<pn>,  [Output=XML|JSON] (default value is read from configuration file) | None | Withdraw from the system the tape with specified ID and pool. In order to succeed there must exist a tape with the specified ID belonging to the specified PoolName. |
| QueryTapes | [TapeID=<id>],  [PoolName=<pn>],  [Output=XML|JSON] (default value is read from configuration file) | List of matching pools and tapes with details according to XML Schema | Query the tapes, filtering by TapeID and/or PoolName.  If a parameter is given it will filter by exact matching; otherwise no filter will be applied and all items are given. |
| Read/Write |  |  | Read and write files and folders |
| Direct Access (only read) |  |  |  |
| Queue Management |  |  |  |
| CancelTask | TaskID=<tid>,  [Output=XML|JSON] (default value is read from configuration file)] | None | Delete the request from the queue, accepted only if the current task status is *waiting* or *fallout* |
| ResubmitTask | TaskID=<tid>,  [Output=XML|JSON] (default value is read from configuration file)] | None | Resubmit a failed request, accepted only if the current task status is *fallout*  *TaskID* stays unchanged. |
| QueryTasks | [LTFSAction= WriteToLTO|RestoreFromLTO|MakeAvailableMount|MakeAvailableUnmount|AddTape],  [Status= waiting|starting|running|completed|fallout,  [PoolName=<pn>],  [TapeID=<tid>],  [FileMode=exact|like] (default=’exact’),  [FileName=<fn>],  [Date=<dt>] |[  [FromDate=<fd>],  [ToDate=<td>]],  [Output=XML|JSON] (default value is read from configuration file)] | List of matching Tasks | All parameters are optional.  If missing no filter will be applied.  Actually result is the same as that of GetStatus foreach task matching the query. |
| Status and Result services |  |  |  |
| GetStatus | TaskID=<tid>  [Output=XML|JSON] (default value is read from configuration file)] | Status the task, according to XML Schema. | Get the current status for the given task.  Returned details other than status will depend on the status itself and on the type of task. |
| GetResult | TaskID=<tid>  [Output=XML|JSON] (default value is read from configuration file)] | Full detailed result of an operation.  See table for asynchronous services | Status of the task must be either “*completed*” or “*fallout*” |
| New services |  |  |  |
| QueryDevices | [Device=<did>]  [Status=free|busy|locked]  [Output=XML|JSON] (default value is read from configuration file)] | Drives and/or Libraries | returns details according to XML Schema |
| UnlockDevice | Device=<did>  [Output=XML|JSON] (default value is read from configuration file)] | None | Unlocks the specified device |
| ManualLoadQuery | [Output=XML|JSON] (default value is read from configuration file)] | Manual Load Requests for Operator | returns details according to XML Schema |
| ManualLoadConfirm | TaskID=<tid>  [Output=XML|JSON] (default value is read from configuration file)] | None | Confirm to server that a requested manual load has been operated, in order to execute the subsequent step of the task |

Note: for the services WithdrawTape and UnlockDevice a TaskID will be also created , although not returned in the response, in order to allow the tracking of their execution by means of the service QueryTasks.

## ASynchronous services

For the following services the request is parsed immediately only for checking the request correctness and the actual execution is postponed and may take an unpredictable time to end. On successful acceptance of the request a TaskID is returned within the response.

The client application has to track the completeness of the task and its result by using the GetStatus and GetResult services. In the following Table the column “Output” refers to the content of that element when GetResult is requested for the relevant TaskID.

|  |  |  |  |
| --- | --- | --- | --- |
| **Service** | **Input** | **Output** | **Description** |
| Tape Management |  |  | Manage tapes and pools |
| AddTape | TapeID=<id>,  [PoolName=<pn>] (default value is read from configuration file),  [Format=Y|F|N] (default=’N’, meaning Yes, Force and No),  [Output=XML|JSON] (default value is read from configuration file) | Result with only exit code and string | Add the tape with the specified ID to the specified pool. If pool is not given the default one is used.  The input parameter Format allows to require the tape to be formatted (Y). The value F means force and implies to format the tape even if it is already LTFS formatted (everything will be lost).For any other value or not specified, the tape will not be formatted. |
| Read/Write |  |  | Read and write files and folders |
| WriteToLTO | FileName=<fn>,  [PoolName=<pn>] (default value is read from configuration file),  [Checksum=N|MD5|MD5\_both|SHA1|SHA1\_both|FILE] (default=’N’),  [ChecksumFile=<cf>] (expected if Checksum=’FILE’),  [Output=XML|JSON] (default value is read from configuration file) | Result with FLocats | Write file or folder with or without checksum calculation/crosscheck on the specified tape pool.  The input parameter FileName must contain the full path of the file or folder to be written. The destination is always a tape pool.  See detailed doc. for checksum stuff. |
| RestoreFromLTO | FileName=<fn>,  DestPath=<dp>  [Output=XML|JSON] (default value is read from configuration file) | Result with FLocats | Restore files or folder from LTO to a locally mounted storage area. The input parameter FileName must be the FLocat containing: the lto-ltfs prefix, the tapeID and the full path, within the tape, of the file or folder to be restored. The parameter DestPath must contain the full path of the local folder to be used as destination. |
| Direct Access (only read) |  |  |  |
| MakeAvailableMount | TapeID=<id>  [Output=XML|JSON] (default value is read from configuration file) | Result with Mount | Mounts the specified LTO tape on a folder (local to the LTFSArchiver server) having the same name as the tapeID. This folder can also be made available from remote through CIFS share. When “made available” the tape is busy and cannot be used as target of other read/write operations. |
| MakeAvailableUnmount | TapeID=<id>,  [Output=XML|JSON] (default value is read from configuration file)] | Result with only exit code and string | Unmounts a previously mounted LTO tape. |
| Queue Management |  |  |  |
| Status and Result services |  |  |  |
| New services |  |  |  |
| ListTape | TapeID=<tid>,  [Output=XML|JSON] (default value is read from configuration file)] | Result with FLocats | Performs direct inspection of the given Tape and provide a complete list of Tape content |
| Checksum | FileName=<fn>,  [Checksum=MD5|SHA1|FILE] (default=auto),  [ChecksumFile=<cf>] (expected if Checksum=’FILE’),  [Output=XML|JSON] (default value is read from configuration file)] | Result with FLocats | If Checksums are already written on Tape, this will check the fixity against those values.  Otherwise new values will be computed and written on Tape.  If Checksum=FILE the computed checksums are compared against those given in ChecksumFile  If required checksum has type different from the one already stored on Tape, then new checksums are computed and also store on tape |
| LockDevice | [Device=<did>] | Result with only exit code and string | Locks the specified device |

Note: When XML, the output must be adherent to the ad-hoc XML Schema; when JSON the output is just a conversion to JSON of the XML output.

Note2: Information on Tasks are subject to be removed after a certain time, according to the configuration policies , e.g. 100 days.