changelog.md for @api.global/typedrequest

2026-03-03 - 3.3.0 - feat(typedrouter)

add middleware support to TypedRouter and export middleware type

- Introduces TMiddlewareFunction and TypedRouter.addMiddleware() to run pre-handler middleware (throw TypedResponseError to reject).
- Adds getTypedHandlerAndRouter() to determine owning router so middleware runs on the correct router.
- Middleware errors are converted into response errors, encoded for network, and outgoing hooks are called; request short-circuits if middleware rejects.
- TypedRouter is now generic (TReqConstraint) and several internal references updated to TypedRouter for compatibility.
- Exports TMiddlewareFunction from index.ts so consumers may reference middleware types.
- Documentation (readme) updated with middleware usage and guidance.
- Tests updated: browser test replaced/renamed to chromium, server tests updated to use TypedServer. Data handling in test adjusted to support Buffer/Uint8Array and serialized Buffer shapes.
- package.json: devDependencies and some deps bumped; build script simplified (removed legacy flags).

2026-03-01 - 3.2.7 - fix(virtualstream)

reconstitute JSON-serialized binary data in VirtualStream; update docs, build config, and dependency bumps

- Add VirtualStream.reconstituteBinaryData to restore Buffer/Uint8Array shapes lost to JSON serialization and use it when handling chunk data and response chunkData.
- Remove .gitlab-ci.yml and add a tsbundle bundle config in npmextra.json; update package.json build script to invoke tsbundle without the previous extra argument.
- Bump multiple devDependencies and dependencies to newer patch/minor versions.
- Significantly expand README with usage examples for TypedRequest, TypedHandler, TypedRouter, TypedTarget, VirtualStream, error handling, hooks, and architecture overview.
- Minor README/legal wording clarifications.

2026-02-11 - 3.2.6 - fix(deps)

upgrade @push.rocks/webrequest to ^4.0.1 and adapt TypedRequest to new API

- Bump dependency @push.rocks/webrequest from ^3.0.37 to ^4.0.1 and update code to match new client/API
- ts/classes.typedrequest.ts: replace plugins.webrequest.WebRequest with plugins.webrequest.WebrequestClient and change postJson call to pass options ({ cacheStrategy: 'cache-first' }) instead of a boolean cache flag
- npmextra.json: reorganize config keys (gitzone -> @git.zone/cli, tsdoc -> @git.zone/tsdoc), add release registries and accessLevel, and add @ship.zone/szci entry

2025-12-04 - 4.0.0 - BREAKING CHANGE(typedrouter)

Introduce options object for TypedRouter.routeAndAddResponse (localRequest, skipHooks); add defaultRouteOptions and make hook calls respect skipHooks; bump package version to 3.2.3

- Changed TypedRouter.routeAndAddResponse signature to accept an options object ({ localRequest?: boolean; skipHooks?: boolean }) instead of a boolean second argument — this is a breaking API change.
- Added TypedRouter.defaultRouteOptions with defaults { localRequest: false, skipHooks: false }.
- Routing now respects options.localRequest and options.skipHooks; hook calls (onIncomingRequest, onOutgoingResponse, onIncomingResponse) are skipped when skipHooks is true to avoid hook recursion or duplicate handling (useful for broadcast-received messages).
- Bumped package.json version to 3.2.3.

2025-12-04 - 3.2.2 - fix(typedrequest)

Add skipHooks flag to TypedRequest to optionally suppress global hooks for internal requests

- Introduce public skipHooks boolean on TypedRequest (default false) with documentation comment explaining it should be used for internal/logging requests to prevent infinite loops.
- Guard calls to global hooks (onOutgoingRequest and onIncomingResponse) in TypedRequest.fire() so hooks are not invoked when skipHooks is true.

2025-12-04 - 3.2.1 - fix(typedrouter)

Use globalThis-backed globalHooks for TypedRouter to enable cross-bundle sharing; fix merging and clearing of global hooks.

- Replace static globalHooks field with getter/setter that stores hooks on globalThis so hooks are shared across bundles.
- Fix setGlobalHooks to merge new hooks with existing ones (avoiding accidental overwrite).
- Update clearGlobalHooks to clear the globalThis storage used for hooks.

2025-12-04 - 3.2.0 - feat(typedrouter)

Add request/response hooks and monitoring to TypedRouter; emit hooks from TypedRequest; improve VirtualStream encoding/decoding; re-export hook types

- Introduce ITypedRequestLogEntry and ITypedRouterHooks interfaces to represent structured traffic log entries and hook callbacks.
- Add static globalHooks on TypedRouter with helper APIs TypedRouter.setGlobalHooks and TypedRouter.clearGlobalHooks for global traffic monitoring.

- Add instance-level hooks on TypedRouter (setHooks) and a unified callHook() that invokes both global and instance hooks safely (errors are caught and logged).
- TypedRequest now emits onOutgoingRequest before sending and onIncomingResponse after receiving, including timestamps, duration and payload/error details.
- TypedRouter now emits lifecycle hooks while routing: onIncomingRequest when a request arrives, onOutgoingResponse for responses (both success and handler-missing error cases), and onIncomingResponse when responses arrive to be fulfilled.
- VirtualStream.encodePayloadForNetwork and decodePayloadFromNetwork were enhanced to recurse into arrays and nested objects (preserving special built-ins) to correctly handle embedded virtual streams.
- Re-export ITypedRequestLogEntry and ITypedRouterHooks from the package index for external consumption.

2025-12-03 - 3.1.11 - fix(virtualstream)

Expose transport localData to handlers via TypedTools; improve VirtualStream payload encode/decode to preserve built-ins and handle nested arrays/objects

- TypedHandler: pass typedRequest.localData into a TypedTools instance so handlers can access transport-layer context (e.g. websocket peer).
- TypedTools: add a public localData property to store transport-specific context available to handlers.
- VirtualStream.decodePayloadFromNetwork: preserve built-in objects (Set, Map, Date, RegExp, Error, Promise or thenable) to avoid incorrect transformation.
- VirtualStream.encodePayloadForNetwork / decodePayloadFromNetwork: added proper recursion for arrays and objects to correctly handle nested payloads and virtual streams, with path tracking to support deduplication logic.

2024-10-16 - 3.1.10 - fix(VirtualStream)

Fix stream closure logic in `writeToWebstream` method

- Added `writer.releaseLock()` call before closing WritableStream when `closingBit` is received in `writeToWebstream` method.

2024-10-16 - 3.1.9 - fix(VirtualStream)

Ensure writable streams are correctly closed asynchronously to prevent potential sync issues.

- Updated VirtualStream to use 'await' when closing writable streams, ensuring proper asynchronous handling.

2024-10-16 - 3.1.8 - fix(VirtualStream)

Fix stream closing behavior to correctly handle closing bits

- Introduced a 'closingBit' constant to properly signal the end of stream data.
- Updated the 'readFromWebstream' function to send a closing bit upon completion if 'closeAfterReading' is true.
- Modified the 'close' method to optionally send a closing bit when terminating the stream.

2024-10-16 - 3.1.7 - fix(VirtualStream)

Fix issue in VirtualStream to handle null values during data writing.

- Ensured writableStream closes gracefully when null values are encountered.
- Added a null check before writing data to the writableStream to prevent errors.

2024-10-16 - 3.1.6 - fix(VirtualStream)

Fix backpressure handling in VirtualStream workOnQueue method

- Resolved an issue in the workOnQueue method of VirtualStream where concurrent execution was not properly managed.
- Introduced a workingDeferred promise to ensure proper queue handling and resolve potential race conditions.

2024-10-16 - 3.1.5 - fix(virtualstream)

Add console log for debugging backpressure feedback loop

- Inserted a console log message to provide insight when waiting due to backpressure in the workOnQueue method.

2024-10-16 - 3.1.4 - fix(VirtualStream)

Corrected the logic for backpressure handling in response

- Fixed backpressure flag assignment in the response handling logic of VirtualStream.
- Ensured correct negation logic for checking receive backpressure status.

2024-10-14 - 3.1.3 - fix(VirtualStream)

Fix keepAlive flag handling in VirtualStream and added stream closure in tests

- Ensure that the keepAlive status is correctly maintained in the keepAlive trigger method.
- Added closure of VirtualStreams in the test suite for proper resource cleanup.

2024-10-14 - 3.1.2 - fix(core)

Fix incorrect backpressure logic in VirtualStream class

- Corrected the logic for determining backpressure status by checking the available space in the receiveBackpressuredArray.
- Introduced a looping mechanism to wait when the other side is backpressured before sending more data.

2024-10-14 - 3.1.1 - fix(virtualstream)

Fix handling of virtual streams for proper shutdown

- Ensured that writeToWebstream method checks for remaining items in receiveBackpressuredArray before closing.
- Corrected package.json dependency for @push.rocks/tapbundle.
- Updated @types/node to version 22.7.5.

2024-10-11 - 3.1.0 - feat(virtualstream)

Enhance VirtualStream with optional closure when reading from webstream

- Added an optional parameter `closeAfterReading` to the `readFromWebstream` method.
- The stream will close automatically after reading if `closeAfterReading` is set to true.

2024-10-11 - 3.0.33 - fix(test)

Increase delay duration before stopping the server in test suite.

- Adjusted the delay time from 1000 ms to 10000 ms before stopping the server to ensure tests complete smoothly.

2024-09-06 - 3.0.32 - fix(virtualstream)

Fix keep-alive loop handling and test cleanup

- Prevent unnecessary keep-alive loop from starting on the responding side
- Add logging for keep-alive loop initiation in VirtualStream
- Temporarily comment out stream close and tap forceful stop in test to avoid abrupt termination

2024-09-06 - 3.0.31 - fix(core)

Updated dependencies and added close method to VirtualStream

- Updated dependencies in package.json for better compatibility
- Added close method to VirtualStream class in ts/classes.virtualstream.ts for more graceful stream termination

2024-05-31 - 3.0.28 - Error Handling

Enhancement to error handling mechanisms.

- Logs now include the method to which an error was given.

2023-08-04 - 3.0.0 - Core

Introduced a breaking change.

- Major update to core functionalities.

Revision #4
Created 2026-03-28 12:50:11 UTC by foss.global Team
Updated 2026-03-29 16:48:03 UTC by foss.global Team