TTL Support for Key-Value Buckets

Metadata	Value
Date	2025-04-09
Author	@ripienaar
Status	Implemented
Tags	jetstream, client, kv, refinement, 2.11
Updates	ADR-8

Context

Since NATS Server 2.11 we support Per-Message TTLs, we wish to expose some KV specific features built on this feature.

- Improve Watchers by notifying of Max Age deleted messages
- Improve Purge so that old subjects can be permanently removed, removing the need for costly compacts, while still supporting Watchers
- Creating keys with a custom life time

In KV we call these Limit Markers.

Configuration

Configuration would get a single extra property in a language idiomatic version of `Limit Markers` that will set `allow_msg_ttl` to `true` and `subject_delete_marker_ttl` to the supplied duration.

This duration value must larger than or equal to 1 second.

This should only be set on a server with API level 1 or newer. At the moment the only way this is exposed is via the `$JS.API.INFO` API call, clients should check this when this feature is requested.

The configuration item can be enabled for buckets that have it disabled but should not support disabling it as today the Server would handle old TTLs correctly should it again be enabled later.

Status

The `Status` interface would get a new property that report on the configured setting:

```
type Status interface {
    // LimitMarkerTTL is how long the bucket keeps markers when keys are removed
by the TTL setting, 0 meaning markers are not supported
    LimitMarkerTTL() time.Duration

    //....
}
```

API Changes

Storing Values

Only the `Create()` function should support accepting a TTL and should error when a TTL is passed with the bucket not supporting this feature - though the server will also error.

Clients can implement this as a varags version of `Create()`, a configuration option for `Create()` or other idiomatic manner the language supports.

We cannot support this on `Put()` since that might mean older revisions could come back from the dead once the TTL expires.

Purging Keys

If the bucket supports Marker TTLs the `Purge()` function can accept a TTL, this should then pass `KV-Operation: PURGE`, `Nats-Rollup: sub` and `Nats-TTL: 1h`.

Clients can implement this as a varags version of `Purge()`, a configuration option for `Purge()` or other idiomatic manner the language supports.

Retrieving Values

When the bucket supports Limit Marker TTLs the clients will receive messages with a header `Nats-Marker-Reason` with these possible values and behaviors:

Value	Behavior
`MaxAge`	Treat as `PURGE`
`Purge`	Treat as `PURGE`
`Remove`	Treat as `DEL`

Watchers should be updated to handle these values also.