Documentation Language: Swift

JobsPostgres / PostgresJobQueue

Class

PostgresJobQueue

Postgres Job queue implementation

```
final class PostgresJobQueue
```

Overview

The Postgres driver uses the database migration service `DatabaseMigrations` to create its database tables. Before the server is running you should run the migrations to build your table.

```
let migrations = PostgresMigrations()
let jobqueue = await JobQueue(
    PostgresQueue(
        client: postgresClient,
        migrations: postgresMigrations,
        configuration: configuration,
        logger: logger
   ),
   numWorkers: numWorkers,
   logger: logger
)
var app = Application(...)
app.beforeServerStarts {
    try await migrations.apply(client: postgresClient, logger: logger, dryRu
}
```

Topics

Structures

`struct Configuration`

Queue configuration

`struct JobOptions`

Options for job pushed to queue

`struct JobPriority`

Job priority from lowest to highest

Initializers

`init(client: PostgresClient, migrations: DatabaseMigrations, configuration: Configuration, logger: Logger) async`

Initialize a PostgresJobQueue

Instance Properties

`let client: PostgresClient`

Postgres client used by Job queue

`let configuration: Configuration`

Job queue configuration

`let logger: Logger`

Logger used by queue

Instance Methods

`func cancel(jobID: JobID) async throws`

Cancel job

```
func cleanup(failedJobs: JobCleanup, processingJobs: JobCleanup,
pendingJobs: JobCleanup) async throws
```

Cleanup job queues

```
func failed(jobID: JobID, error: Error) async throws
```

This is called to say job has failed to run and should be put aside

```
func finished(jobID: JobID) async throws
```

This is called to say job has finished processing and it can be deleted

```
func getMetadata(String) async throws -> ByteBuffer?
```

```
func onInit() async throws
```

```
func pause(jobID: UUID) async throws
```

Pause job

```
func push<Parameters>(JobRequest<Parameters>, options: JobOptions)
async throws -> JobID
```

Push Job onto queue

```
func registerJob<Parameters>(JobDefinition<Parameters>)
```

Register job

```
func resume(jobID: JobID) async throws
```

Resume job

```
func retry<Parameters>(JobID, jobRequest: JobRequest<Parameters>,
options: JobRetryOptions) async throws
```

Retry an existing Job

```
func setMetadata(key: String, value: ByteBuffer) async throws
```

```
func shutdownGracefully() async
```

shutdown queue once all active jobs have been processed

```
func stop() async
```

stop serving jobs

Type Aliases

```
typealias JobID
```

Enumerations

```
enum JobCleanup
```

what to do with failed/processing jobs from last time queue was handled

```
enum PostgresQueueError
```

Errors thrown by PostgresJobQueue

Default Implementations

- AsyncSequence Implementations
- JobQueueDriver Implementations

Relationships

Conforms To

```
CancellableJobQueue
JobQueueDriver
ResumableJobQueue
Swift.Sendable
_Concurrency.AsyncSequence
```