SdkQueue

September 4, 2026

SdkQueue

Kind: Service

Source: `atloria-monorepo/apps/api/src/sdk/sdk-queue.service.ts`

The 'sdk-generate' queue registration (C2). Present in EVERY api pod for ENQUEUE only — processing lives in apps/sdk-worker (the only place with a JRE + the pinned openapi-generator jar). Mirrors DocsRepoQueue: never blocks boot; an unavailable queue just means enqueue returns false and the next spec upsert retries.

jobIds are deterministic per (project, language, specHash) so a burst of identical spec upserts collapses to one pending job. With deterministic ids BOTH removeOnComplete AND removeOnFail must be `true`: a retained completed/failed job with the same id makes BullMQ silently ignore a later add(), wedging regeneration (same trap documented on docs-repo).

`SdkQueue` registers the `sdk-generate` BullMQ queue in every API pod so application code can enqueue SDK generation work without requiring local generator tooling. Actual job processing runs exclusively in `apps/sdk-worker`, where the JRE and pinned OpenAPI Generator JAR are available. Queue initialization never blocks API startup; when unavailable, enqueue operations return `false` so a later spec upsert can retry.

Methods

Method	Signature	Returns	Description
`onModuleInit`	`onModuleInit()`	`void`	
`enqueueGenerate`	`enqueueGenerate(job: SdkGenerateJob)`	`Promise<boolean>`	Enqueue one language × spec-hash generation.
`onModuleDestroy`	`onModuleDestroy()`	`Promise<void>`	

Dependencies

- `ConfigService`

Where it refuses work

- `SdkQueue` stops the work with an early return when `!this.queue`.

When something fails

- `SdkQueue` handles failure in 2 places: it logs it and continues in 1, and turns it into a return value in 1.

Diagram

```
sequenceDiagram
  participant API as API Pod
  participant Queue as SdkQueue
  participant Redis as BullMQ / Redis
  participant Worker as apps/sdk-worker
  participant Generator as OpenAPI Generator

  API->>Queue: enqueueGenerate(project, language, specHash)
  Queue->>Queue: Build deterministic job ID
  Queue->>Redis: Add sdk-generate job
  alt Queue available
    Redis-->>Queue: Job accepted
    Queue-->>API: true
    Worker->>Redis: Consume job
    Worker->>Generator: Generate SDK from OpenAPI spec
    Generator-->>Worker: Generated SDK
    Worker->>Redis: Complete and remove job
  else Queue unavailable
    Redis-->>Queue: Connection/add failure
    Queue-->>API: false
    Note over API: Next spec upsert retries enqueueing
  end
```

Usage

```
import { Injectable } from '@nestjs/common';
import { SdkQueue } from './sdk-queue.service';

@Injectable()
export class SpecUpsertService {
  constructor(private readonly sdkQueue: SdkQueue) {}
```

```
  async handleSpecUpsert(
    projectId: string,
    language: string,
    specHash: string,
  ): Promise<void> {
    const enqueued = await this.sdkQueue.enqueueGenerate(
      projectId,
      language,
      specHash,
    );

    if (!enqueued) {
      // Do not fail the spec upsert solely because Redis is unavailable.
      // A subsequent upsert for this spec can retry the enqueue operation.
      console.warn('SDK generation job was not enqueued');
    }
  }
}
```

AI Coding Instructions

- Keep SDK generation processing out of the API application; API pods only enqueue jobs, while `apps/sdk-worker` consumes them.
- Preserve deterministic job IDs based on project, language, and spec hash so duplicate spec upserts collapse into one pending job.
- Keep both `removeOnComplete` and `removeOnFail` enabled for deterministic IDs; retained jobs can cause BullMQ to silently ignore later `add()` calls with the same ID.
- Treat queue initialization and enqueue failures as non-fatal: return `false` rather than blocking boot or failing the spec upsert workflow.
- Ensure queue resources are closed during `onModuleDestroy()` to avoid leaked Redis/BullMQ connections during shutdown.

Relationships

- DEPENDS_ON → `configservice`

Referenced By

- `SdkArtifactsService` (DEPENDS_ON)
- `SdkModule` (MODULE_PROVIDES)