GeneratorRunnerService

September 4, 2026

GeneratorRunnerService

Kind: Service

Source: `atloria-monorepo/apps/sdk-worker/src/app/sdk-gen/generator-runner.service.ts`

Invokes the PINNED openapi-generator jar: `java -jar <jar> generate -i ⋯ -g ⋯ -o ⋯`. The argv itself is the shared, unit-tested contract (buildGeneratorArgs in parser-core); this service only owns process plumbing. `runProcess` is an instance property so tests stub it and assert the exact argv without a JRE.

`GeneratorRunnerService` runs the pinned OpenAPI Generator JAR for SDK generation. It delegates command construction to the shared, unit-tested `buildGeneratorArgs` contract in `parser-core`, while owning Java process execution, binary/JAR resolution, and timeout handling within the SDK worker.

Methods

Method	Signature	Returns	Description
`javaBin`	`javaBin()`	`string`	
`jarPath`	`jarPath()`	`string`	
`timeoutMs`	`timeoutMs()`	`number`	
`generate`	`generate(options: {		

```
language: SdkGeneratedLanguage;
specPath: string;
outDir: string;
packageBase: string;
```

}) | Promise<{ argv: string[] }>` | Generate one language into outDir. |

Dependencies

- `ConfigService`

Where it refuses work

- `GeneratorRunnerService` stops the work with `GeneratorFailedError` when `result.code !== 0`.

Diagram

```
sequenceDiagram
  participant Worker as SDK Worker
  participant Runner as GeneratorRunnerService
  participant Args as parser-core<br/>buildGeneratorArgs
  participant Java as java -jar
  participant Generator as OpenAPI Generator JAR

  Worker->>Runner: generate(options)
  Runner->>Args: buildGeneratorArgs(options)
  Args-->>Runner: generator argv
  Runner->>Runner: javaBin(), jarPath(), timeoutMs()
  Runner->>Java: runProcess(java, ["-jar", jar, ...argv])
  Java->>Generator: generate -i ... -g ... -o ...
  Generator-->>Java: generated SDK files
  Java-->>Runner: process result
  Runner-->>Worker: { argv }
```

Usage

```
import { GeneratorRunnerService } from './generator-runner.service';

const generatorRunner = new GeneratorRunnerService();

// The exact generator arguments are built by parser-core.
const result = await generatorRunner.generate({
  inputSpecPath: '/workspace/openapi.yaml',
  generatorName: 'typescript-fetch',
  outputPath: '/workspace/generated-sdk',
});

console.log('OpenAPI Generator invoked with:', result.argv);
```

AI Coding Instructions

- Keep OpenAPI Generator argument construction in `buildGeneratorArgs` from `parser-core`; do not duplicate or reorder arguments in this service.

- Use `javaBin()`, `jarPath()`, and `timeoutMs()` rather than hardcoding executable paths, JAR locations, or process limits.
- Preserve `runProcess` as an instance property so tests can stub process execution without requiring a local JRE or generator JAR.
- When changing generation behavior, update the shared argv contract and its unit tests before modifying process plumbing here.
- Ensure `generate()` returns the exact argv used for execution so callers and tests can inspect the invocation.

Relationships

- DEPENDS_ON → `configservice`

Referenced By

- `SdkGenModule` (MODULE_PROVIDES)