RuntimeException

August 19, 2026

RuntimeException

Kind: Class

Source: `packages/core/errors/exceptions/runtime.exception.ts`

Part of: Core

`RuntimeException` represents an error that occurs during application execution and can be propagated through the core error-handling flow. It provides a `what()` method for retrieving a human-readable description of the failure. Use it when an operation cannot continue because of an unexpected runtime condition.

Extends: `Error`

Methods

Method	Signature	Returns
`what`	`what()`	`void`

Diagram

```
graph LR
  A[Application operation] --> B{Runtime failure?}
  B -- Yes --> C[RuntimeException]
  C --> D[what()]
  D --> E[Readable error description]
  B -- No --> F[Continue execution]
```

Usage

```
import { RuntimeException } from '@your-package/core/errors/exceptions/runtime.exception';

function loadConfiguration(configPath?: string): void {
  if (!configPath) {
```

```
    throw new RuntimeException('Configuration path is required.');
  }

  // Continue loading the configuration...
}

try {
  loadConfiguration();
} catch (error) {
  if (error instanceof RuntimeException) {
    console.error(error.what());
  } else {
    throw error;
  }
}
```

AI Coding Instructions

- Throw `RuntimeException` for runtime conditions that prevent an operation from completing normally.
- Provide actionable, human-readable error messages when constructing the exception.
- Use `instanceof RuntimeException` when handling this specific error type.
- Prefer `what()` when consuming the exception description to follow the established exception API.
- Avoid using this exception for expected control flow; validate inputs and return normal results where appropriate.

How it works

`RuntimeException` is an exported subclass of the built-in `Error` class. `packages/core/errors/exceptions/runtime.exception.ts:1` It is also re-exported through the core exceptions barrel. `packages/core/errors/exceptions/index.ts:2`

- Its constructor accepts an optional `message` argument, defaulting to an empty string, and passes that value to `Error`; consequently, the inherited `message` is initialized from that argument. `packages/core/errors/exceptions/runtime.exception.ts:2-4`
- `what()` returns the instance's current inherited `message` property. `packages/core/errors/exceptions/runtime.exception.ts:6-8`
- The class has no code-visible input validation, custom error fields, logging, or other side effects beyond calling the `Error` constructor.

`packages/core/errors/exceptions/runtime.exception.ts:1-8`

In this repository, it is both a base class for more specific exceptions—including `CircularDependencyException`, `UnknownDependenciesException`, and `InvalidClassException`—and a directly thrown error type. `packages/core/errors/exceptions/circular-dependency.exception.ts:1-3` `packages/core/errors/exceptions/unknown-dependencies.exception.ts:1-6` `packages/core/errors/exceptions/invalid-class.exception.ts:1-4`

Direct call sites throw a message-less `RuntimeException` when an injector cannot find a token in its collection, when a module's provider map lacks its metatype, or when middleware resolution cannot find an instance wrapper. `packages/core/injector/injector.ts:171-174` `packages/core/injector/module.ts:141-144` `packages/core/middleware/middleware-module.ts:203-206`

Used by

11 references from 11 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (11)

- `RpcDecoratorMetadata` — `packages/microservices/errors/invalid-grpc-message-decorator.exception.ts`:3
- `InvalidGrpcPackageDefinitionMissingPackageDefinitionException` — `packages/microservices/errors/invalid-grpc-package-definition-missing-package-definition.exception.ts`:3
- `InvalidGrpcPackageDefinitionMutexException` — `packages/microservices/errors/invalid-grpc-package-definition-mutex.exception.ts`:3
- `InvalidGrpcPackageException` — `packages/microservices/errors/invalid-grpc-package.exception.ts`:6
- `InvalidGrpcServiceException` — `packages/microservices/errors/invalid-grpc-service.exception.ts`:6
- `InvalidKafkaClientTopicException` — `packages/microservices/errors/invalid-kafka-client-topic.exception.ts`:6
- `InvalidMessageException` — `packages/microservices/errors/invalid-message.exception.ts`:6
- `InvalidProtoDefinitionException` — `packages/microservices/errors/invalid-proto-definition.exception.ts`:6

…and 3 more.