Spring Boot Actuator Endpoints

# 1. Base Actuator Endpoint: /actuator

* **URL:** http://localhost:8761/actuator
* **Purpose:** Provides a hypermedia-driven index of all available Actuator endpoints.
* **Key JSON properties:** { “\_links”: { “self”: { “href”: “http://localhost:8761/actuator”, “templated”: false } } }
* \_links → HAL property listing hypermedia links.
* self → points to the current resource.
* href → URL for the resource.
* templated: false → not a URI template, fixed URL.

# 2. /actuator/beans

* **Purpose:** Shows all beans registered in the Spring ApplicationContext.
* **Each bean entry contains:**
  + Bean name (ID in context)
  + Bean type (class)
  + Scope (singleton, prototype, etc.)
  + Dependencies (other beans it depends on)

## Sources of Beans

1. **Your code:** Classes annotated with @Component, @Service, @Repository, @Controller, or @Bean.
2. **Spring Boot Starters:** Auto-configurations create beans automatically.
3. **Third-party libraries:** If they include Spring configurations.

## Example Mapping: Dependency → Beans

| Dependency | Example Beans Created |
| --- | --- |
| spring-boot-starter-web | DispatcherServlet, Controllers, HandlerMapping, MessageConverters |
| spring-boot-starter-data-jpa | EntityManagerFactory, JpaRepositories, TransactionManager |
| spring-boot-starter-security | SecurityFilterChain, AuthenticationManager, PasswordEncoder |
| mysql-connector-java | DataSource (if DB properties set) |
| spring-boot-starter-actuator | Actuator endpoints like /actuator/beans |
| commons-lang3 | ❌ No beans (only utility classes) |

## Why /actuator/beans is Useful

* **Debugging:** Check loaded beans, avoid duplicates.
* **Dependency check:** Confirm starters created expected beans.
* **Troubleshooting:** Verify the presence of a specific bean.

**Key takeaway:** /actuator/beans = live snapshot of ApplicationContext beans.

# 3. /actuator/caches

* **Purpose:** Shows all caches registered in your application.
* **Supported cache providers:** Redis, Caffeine, EhCache, Hazelcast, Simple in-memory (default)

## Example with Spring Cache

@Service public class UserService { @Cacheable(“users”) public User getUser(Long id) { return userRepository.findById(id).orElseThrow(); } }

## Sample Actuator Response with Redis

{ “cacheManagers”: { “cacheManager”: { “caches”: { “users”: { “target”: “RedisCache”, “name”: “users” }, “products”: { “target”: “RedisCache”, “name”: “products” } } } } }

## Clearing a Specific Cache

DELETE http://localhost:8080/actuator/caches/users - Only clears the users cache.

**Key takeaway:** /actuator/caches = list of all cache “containers” and their engines (Redis, EhCache, etc.).

# 4. /actuator/health

* **Purpose:** Shows the health status of your application.
* **Default response:** { “status”: “UP” }

## What “UP” Means in Enterprise Apps

* JVM is running.
* **Infrastructure checks:** DB connectivity, cache availability, message brokers, file storage, disk space, external APIs.
* **Security checks:** Keystore/certificates valid, token provider reachable.
* **Business-critical checks:** Transaction processing, job queues, fraud detection engine.

## Example Enterprise Response

{ “status”: “UP”, “components”: { “db”: { “status”: “UP”, “details”: { “database”: “PostgreSQL”, “validationQuery”: “SELECT 1” } }, “diskSpace”: { “status”: “UP”, “details”: { “free”: 123456789, “threshold”: 10485760, “path”: “/” } }, “redis”: { “status”: “UP”, “details”: { “version”: “7.0.12” } }, “kafka”: { “status”: “UP”, “details”: { “brokers”: “3 online” } }, “s3”: { “status”: “UP”, “details”: { “bucket”: “bank-docs”, “region”: “ap-south-1” } }, “paymentGateway”: { “status”: “UP”, “details”: { “latencyMs”: 120 } }, “authServer”: { “status”: “UP”, “details”: { “issuer”: “https://keycloak.company.com” } } } }

## How /actuator/health Works

1. Spring Boot registers HealthIndicators for each component.
2. Each HealthIndicator returns UP or DOWN.
3. Spring aggregates statuses into a single response.

## Enterprise Usage

* Kubernetes / Docker liveness & readiness probes.
* Load balancers remove instances marked DOWN.
* Monitoring tools alert DevOps/SRE on component failures.

**Key takeaway:** /actuator/health = comprehensive health check combining infrastructure, security, and business-critical flows.

# 5. Summary of Actuator Key Endpoints

| Endpoint | Purpose |
| --- | --- |
| /actuator | Base endpoint listing all actuator links |
| /actuator/beans | Lists all beans loaded in ApplicationContext |
| /actuator/caches | Shows all caches and their providers |
| /actuator/health | Checks health of application and dependencies |