Spring Boot Microservices

Beginner to Guru

API Versioning

API Versioning

- Versioning your APIs is considered a best practice
- Example "/api/v1/beer" - "v1" is the API version
- API Versioning allows you to evolve the API without breaking existing API consumers
- Typical lifespan:
  - v1 - first release
  - v2 - second release, notify consumers v1 version is deprecated
  - v3 - remove v1 (optional), notify consumers v2 is deprecated

Semantic Versioning 2.0.0

- See website - https://semver.org
- Version - MAJOR.MINOR.PATCH
  - **MAJOR** - version for major incompatible API changes - aka breaking changes
  - **MINOR** - new functionality - backwards compatible changes
  - **PATCH** - backwards compatible bug fixes
- API URLs typically only use MAJOR versions
  - Can optionally use MINOR and PATCH
  - /v1 or /v1.1

Non-Breaking Changes

- Non-Breaking changes may be performed under **MINOR** or **PATCH** versions
- Examples:
  - New optional parameter
  - New response fields
  - New service (endpoint)
  - Bug fixes - behavior change, NOT change to API itself

Breaking Changes

- Breaking Changes should be done under a **MAJOR** version
- Examples:
  - New required parameter
  - Removal of existing parameter
  - Removal of response value
  - Parameter name change or type
  - Deprecation of a service

SPRING FRAMEWORK

GURU