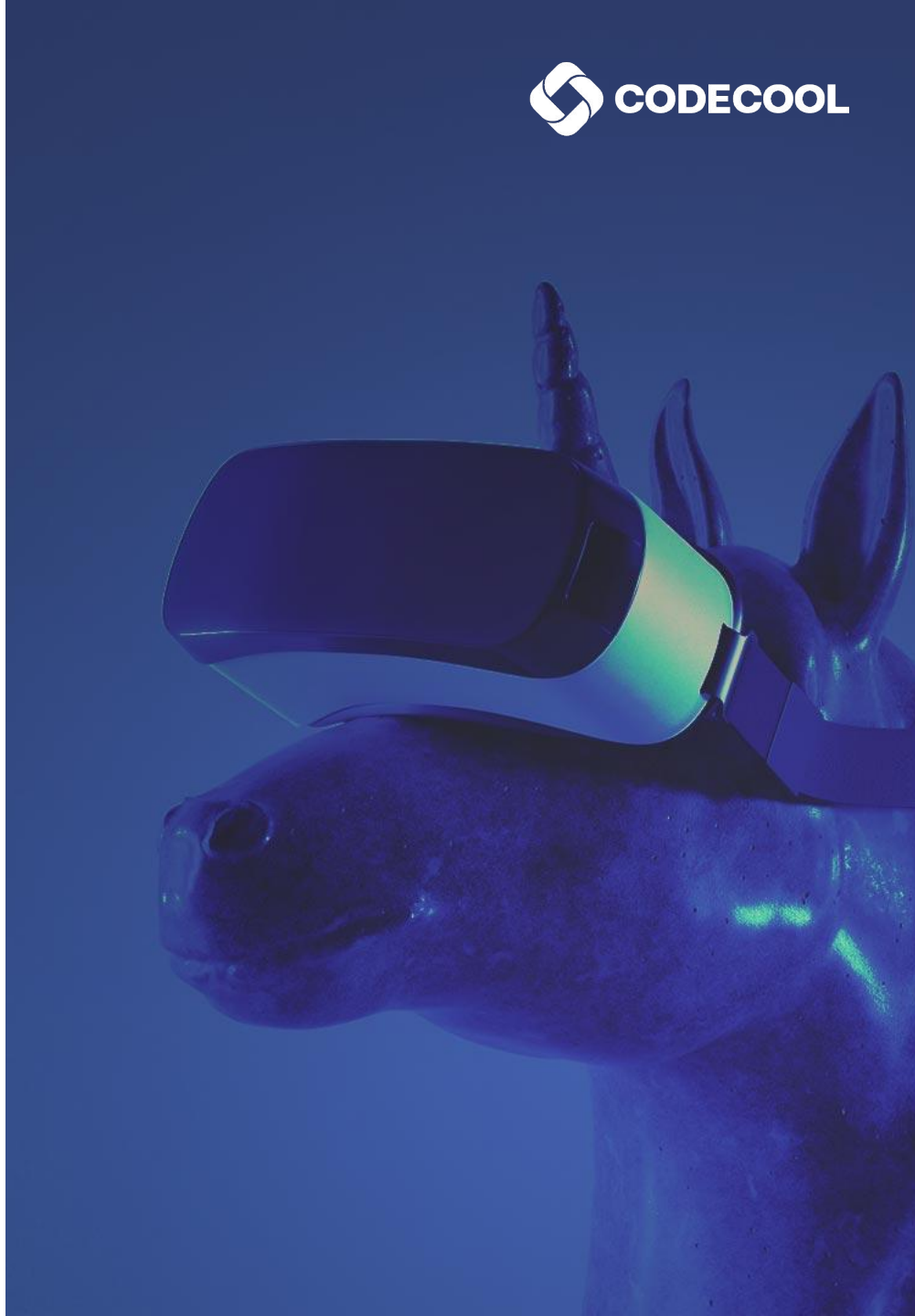
CODECOOL

Dependency Injection in Java

Based on Spring documentation

- https://docs.spring.io/spring-framework/docs/current/reference/html



Inversion of Control

- feature by which an object defines its dependencies without creating them. This object delegates the job of constructing and instantiating such dependencies to an IoC container, the Spring lightweight container.
- https://medium.com/javarevisited/spring-beans-in-depth-a6d8b31db8a1

What is bean?

- a Spring bean is an object that form the backbone of your application and that is managed by the Spring IoC container
- A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container

CODECOOL

Stereotype Annotations

- @Component
- @Controller
- @RestController
- @Service
- @Repository

CODECOOL

Similarities

- First point worth highlighting again is that **with respect to scan-auto-detection and dependency injection for BeanDefinition** all these annotations (viz., @Component, @Service, @Repository, @Controller) are the same.
- **We can use one in place of another and can still get our way around.**

@Component

- This is a general-purpose stereotype annotation indicating that the class is a spring component.
- **<context:component-scan>** only scans **@Component** and does not look for **@Controller**, **@Service** and **@Repository** in general. They are scanned because they themselves are annotated with **@Component**.
- @Controller, @Service and @Repository are special types of @Component annotation.

@Controller

- The @Controller annotation indicates that a particular class serves the role of a controller. The @Controller annotation acts as a stereotype for the annotated class, indicating its role.
- We can use **@RequestMapping** on/in only those methods whose classes are annotated with @Controller and it will NOT work with @Component, @Service, @Repository etc...

CODECOOL

@Service

- Apart from the fact that it's used to indicate, that it's holding the business logic, there’s nothing else noticeable in this annotation

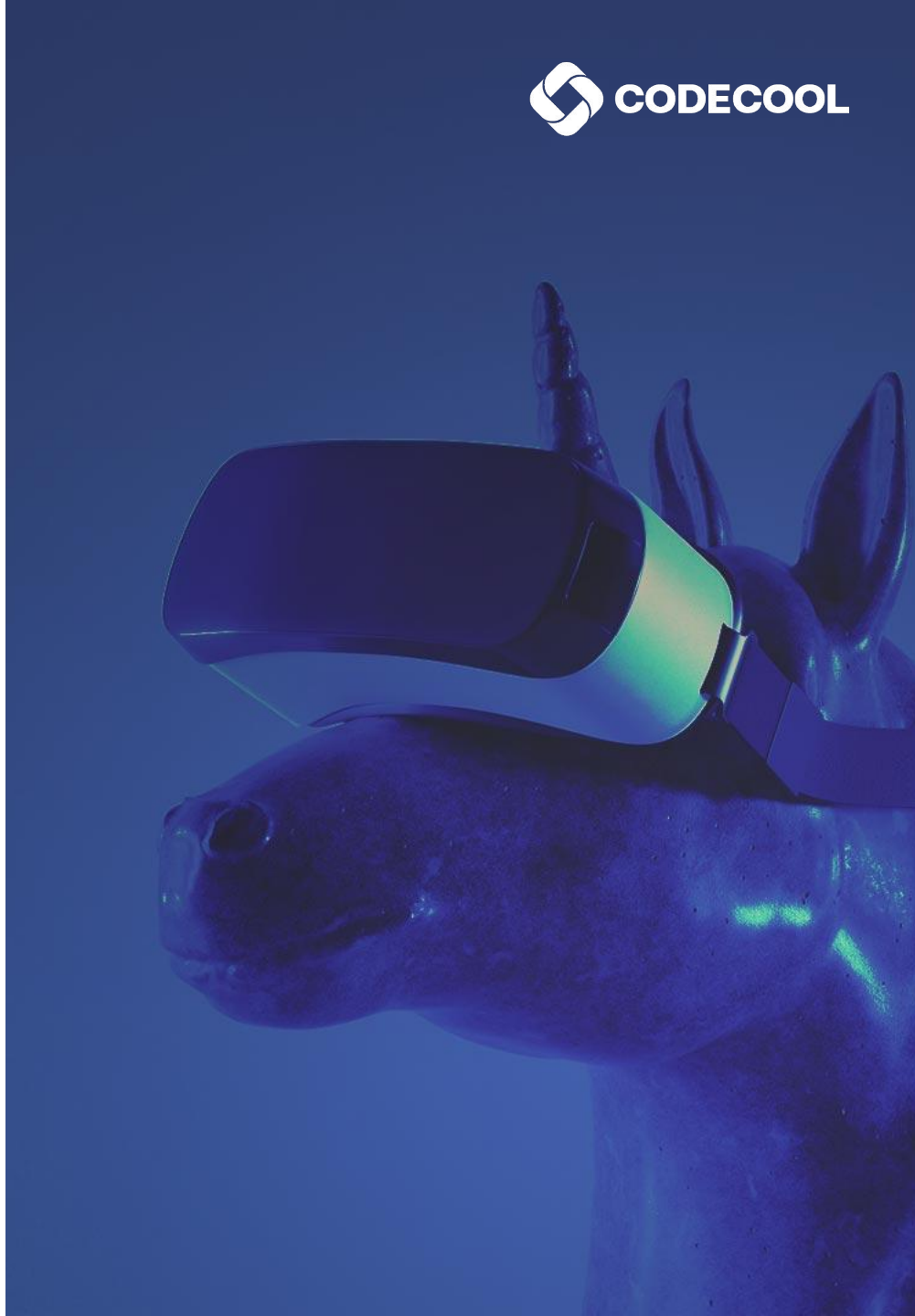
CODECOOL

@Repository

@Repository's job is to catch platform specific exceptions and re-throw them as one of Spring's unified unchecked exception.

Note for future self

- Spring may add special functionalities for @Service, @Controller and @Repository based on their layering conventions. Hence, it's always a good idea to respect the convention and use it in line with layers.



CODECOOL

Bean Scopes

Basic Scopes:

- Singleton
- Prototype

Web-aware scopes:

- Request
- Session
- Application
- Websocket


CODECOOL