Java Framework

- 1 Spring技术发展
  - Quantum Mechanics
  - Theory of Relativity
  - Cornerstones of Modern Physics
  - Predict existence of antimatter
- 2 Spring框架设计
  - Mozart of Mathematics
  - 框架
    - 定义：框架是一组类库或者工具，在特定领域根据一定的规则组合成的开放性的框架
  - 特性
    - Simple and Profound
    - 支撑性-扩展性
    - 聚合性-约束性
  - mais
    - Spring7个大模块
      - Core
        - Bean
        - Context
        - AOP(核心)
      - Testing
        - Mock
        - TestContext
      - DataAccess
        - Tx
        - JDBC
        - ORM
      - Spring MVC/WebFlux
        - Web
      - Integration
        - remoting
        - JMS
        - WS
        - EIP :Camel
      - Language
        - Kotlin
        - Groovy
    - Bean/AOP
      - Web
        - MVC
        - WebFlux
      - Context
        - Testing
        - Core
        - Batch
        - Security
      - JDBC
        - Integration
        - Data
      - Tx
        - ORM
      - 项目层次
        - Presentation layer-展示
        - Business Layer
        - Services Layer
        - Persistence Layer-持久化
        - Parabases Layer
- 3 AOP详解
  - AOP
    - 面向切面编程
    - 管理对象生命周期与对象装配
    - 为实现管理与装配，加一个中间代理层（字节码增强），来实现所有对象的托管
  - IOC
    - 控制反转/依赖注入（Dependency Injection）
    - 对象装配思路的改进
  - 代理种类
    - 接口类型
      - 默认使用JdkProxy
        - com.sun.proxy.$proxy
      - proxyTargetClass
        - EnhanceBySpringCGLIB
    - 非接口类型
      - 默认使用CGLIB
        - EnhanceBySpringCGLIB
  - 如何实现AOP
    - java代码的方式
    - XML注解
    - 引用AOP
      - 按类型引用
        - Autowired
      - 按名字引用
        - Resource
- 4 Spring Bean
  - 类字节码CGLIB工具
    - CGLIB
      - ASM
    - AspectJ
    - JavaProxy
    - Instrumentation(操作字节码)
      - Javassist
    - 字节码操作集合
      - 字节码与反射的区别
      - 字节码类似于基因操作与拍CT的区别
      - ByteBuddy作为字节码增强的新工具提供了更友好的API接口
  - spring Bean 核心原理
    - EnoironmentCapable
    - Bean Factory
      - ListableFactory
      - HierarchiBeanFactory
    - MessageSource
    - ApplicatoionEventPublisher
    - ResourcePatternResolver
    - 如何取到应用上下文
      - getBean
      - Autowired
      - resource
  - Spring Bean生命周期
    - Bean加载过程
    - 构造函数
    - 依赖注入
    - BeanNameAware
    - BeanFactoryAware
    - ApplicationContextAware
    - BeanPostProcessor前置方法
    - InitializatingBean
    - 自定义init方法
    - BenPostProcessor后置方法
    - 使用
    - pisposableBean
    - 自定义destory方法
    - AOP
  - classLoader装载过程
    - 容器启动
    - 1.实例化(instantuation：实例化Bean
    - 2.属性赋值：设置对象属性
    - 3.检查Aware的相关接口并设置相关依赖
    - 前置处理BeanPostProcessor
    - 是否实现initializingBean接口
    - 是否配置自定义的init-method 接口
    - BeanPosterProcessor 后置处理
    - 注册Destruction相关回调接口
    - 使用中
    - 是否实现DisposableBean接口
    - 是否配置自定义的destort-method接口
    - 结束
    - 加载
    - 销毁
  - Application
    - 后置处理器
    - 监听器
    - Bean工厂
      - 转化为Bean
- 5 JDBC
  - 定义了数据库连接池接口
    - DriverManager
    - Connection
    - Statement(逻辑)
    - resultset
    - DataSourcePool
  - 常见的数据库连接池
    - C3P0
    - DBCP
      - Apache CommonPool
    - Druid
    - Hikuri
  - 增强
    - 封装JDBC后做了改进，JDBC是抽象出来的针对各种数据链接
    - 加上XA
      - XA connection
    - 连接池获取
      - PooledConnection
    - MySQL驱动JDBC接口connection
- 6 JPA
  - Java PersistenceAPI
    - EntityManager
- 7 Spring JDBC 与ORM
  - Spring管理事务
    - 引入
    - 管理
    - 逻辑图
      - TransactionDefinition
        - getPropagationBehavior()
          - 事务传播性
          - @Transcation(Propagation=Propagation.REQUIRED)
        - getIsolationLevel()
          - 事务隔离级别
          - @Transcation(isolation=Isolaation.UNCOMMITED)
          - 读未提交（脏读，不可重复读）
        - getTimeout()
          - 事务超时性
          - @transactional(timeout=30)
        - isReadOnly()
          - @TranscationreadOnly=True)
          - 只读：该属性用于设置当前事务是否为只读事务，true表示只读，false表示可读写，默认false
        - 回滚
          - 单一异常类
            - @Transcational(rollbackFor=RuntimeException.class)
          - 多个异常类
            - @Transcational(rollbackFor=RuntimeException.class,Exception.calss)
      - PlaatformTransactionManager
        - getTransaction()
        - Commit()
        - roll(back()
      - TransactionStatus
        - hasSavepoint()
        - isRollibackOnly()
        - isCompleted()
    - 经验
      - 本地事务
      - 多数据源
      - 数据库连接池配置
      - ORM 的复杂SQL 级联查询
      - ORM辅助工具和插件
- 8 Spring技术发展
- 9 XML配置
  - Bean
    - 检查XML是否正确
      - spring.schemas
    - 从DOM节点parse对象
      - spring.handlers
    - schema Location
      - 自定义标签
  - 流程
    - Floating Topic
    - Floating Topic
    - 原理
      - 根据Bean的字段结构生成XSD
      - 根据Bean的字段结构加载XML
    - 工具
      - XML与Bean的互相转换
        - XStream
      - DOM
        - DOM4j
      - SAT
        - SAT
    - 演化历史
      - XML配置，注解注入
        - 1.0/2.0
        - XML@AutoWire
      - 半自动注解配置
        - 2.5
        - @Service
      - Java config 配置
        - 3.0
        - @bean/@Configuration
      - 4.0/Spring Boot,全自动注解
        - 4.0
        - @Condition/@AutoCOnfigurationX
- 10 Spring Message
  - 作用
    - http/RPC
      - 同步转异步
    - 削峰填谷
      - 流量控制
  - 模式
    - 类JDBC模式
    - Queue模式
      - 单个消息单个消费者使用
    - Topic模式
      - 生产订阅，一条消息多个消费者
- 11 spring与spring boot
  - spring boot 简化的原因点
    - spring本身技术体系的成熟与完善，第三方组件的成熟集成
    - 去web容器化
    - 基于MAVEN的POM 的生态体系
    - 避免大量maven版本导入和冲突
  - 什么是spring boot
    - 易于创建spring生产级别的应用，最少配置，对spring及其他第三方库采取限制性措施，易于使用
  - spring boot特性
    - 创建独立运行的spring应用
    - 去web化，内置tomcat/jetty/Undertow，无需部署war包
    - 提供限制性的starter依赖简化配置
    - 必要时自动化配置spring和其他第三方组件
    - 提供生产级的production-read特性，例如指标度量，健康检查，外部配置
    - 完全代码生产和XML配置
- 12 spring boot 核心原理
  - 为什么约定大于配置
    - Maven的目录结构：默认使用resource存放配置文件，打包格式为jar
    - 默认的配置文件：application.properties 或者 application.yaml
    - 默认通过application.profiles.active 属性来决定运行环境时的配置文件
    - EnableAutoConfiguration默认对于依赖的starter进行自动装载
    - spring-boot-starter-web 默认 包含spring-mvc相关依赖以及内置web内容，使得构建web应用更简单
  - 自动化配置
    - 作用
      - 简化配置核心
    - 模块
      - Configuration
      - EnableXX
      - Condition
    - 原理
      - @SpringBootApplication
        - 入口
      - Spring.factories
        - META-INF下的配置文件
        - @configuration
          - 配置
      - @Configuration/@import
        - 使用
    - 条件化自动配置
      - 某个Bean存在
        - @ConditionOnBean
      - 某个类存在
        - @ConditionOnClass
      - 某个Bean存在
        - @ConditionOnMissingBean
      - 某个属性存在
        - @ConditionOnProperty
      - 某个资源存在
        - @ConditionOnResource
      - 某种类型的Bean存在
        - @ConditionOnSingleCondition
      - 某个应用存在
        - @ConditionOnWebApplication
  - spring-boot-starter
    - 脚手架核心
      - 整合各种第三方类库
      - 协同工具
    - 模块
      - spring.provider
      - spring.factories
      - additional-spring-configuration-metadata.json
      - 自定义的configuration类
- 13 ORM
  - Hiberbate
    - HQL
    - Criteria
    - NativeSQL
  - Mybatis
    - 特点
      - 需要映射文件mapper.xml 定义map规则和SQL
      - 需要定义mapper/DAO，基于XML规则操作数据库
    - 接口层
      - 接口类型
        - 数据增加接口
        - 数据删除接口
        - 数据查询接口
        - 数据修改接口
      - 接口调用方式
        - 基于StatementID
        - 基于Mapper接口
    - 流程图
      - 数据处理层
        - 参数处理
          - 参数映射配置
          - 参数映射解析
          - 参数类型解析
          - ParameterHander
        - SQL解析
          - SQL语句配置
          - SQL语句解析
          - SQL类型解析
          - SqlSource
        - SQL执行
          - SimpleExecutor
          - BatchExecutor
          - ReuseExecutor
          - Executor
        - 结果处理和映射
          - 结果参数映射配置
          - 结果类型转换
          - ResultSetHandler
      - 框架支撑层
        - SQL语句配置方式
          - XML
          - 注解
        - 事务连接管理
        - 连接池管理
        - 缓存机制
      - 引导层
        - XML配置
        - JAVA API配置