一、项目简介

- 项目名称
  - 中文: 通用工具包
  - 简写: GM-TOOLS / GT
- 项目背景

> 随着技术部门日益扩招, 成员个性化代码情况显著, 为了统一部门的技术要求和编码规范
>
> 急需通过现有技术手段约束新老成员使用指定技术方案和编码规范以保证代码质量

- 项目说明
  - 通过利用MAVEN的三大特性: 继承、聚合、版本控制, 及其各项便利控制GT的各依赖的版本号
  - 通过借助现有的主流技术解决方案, 设计各种类型项目均可使用的独立便捷的工具体系
  - 截止目前该项目已将各类工具分类封装到相应模块, 每个模块可单独引用, 相对独立

二、版本纪要

2.1 最新版本

2.1.1 版本简介

- 最新版本: 4.0.0
- 继承示例: 一级控制系统、二级开发项目、三级引用模块
  - 系统继承的好处: 可由技术负责人 **统一** 升级调整 **所有项目** 版本
  - 项目继承的好处: 可由项目负责人 **选择** 升级调整 **自身项目** 版本
  - 模块继承的弊端: 可由项开发人员 **自由** 升级调整 **自身模块** 版本

```
<parent>
    <groupId>cn.hll.tools</groupId>
    <artifactId>tools-parent</artifactId>
    <version>${latest.version}</version>
</parent>
```

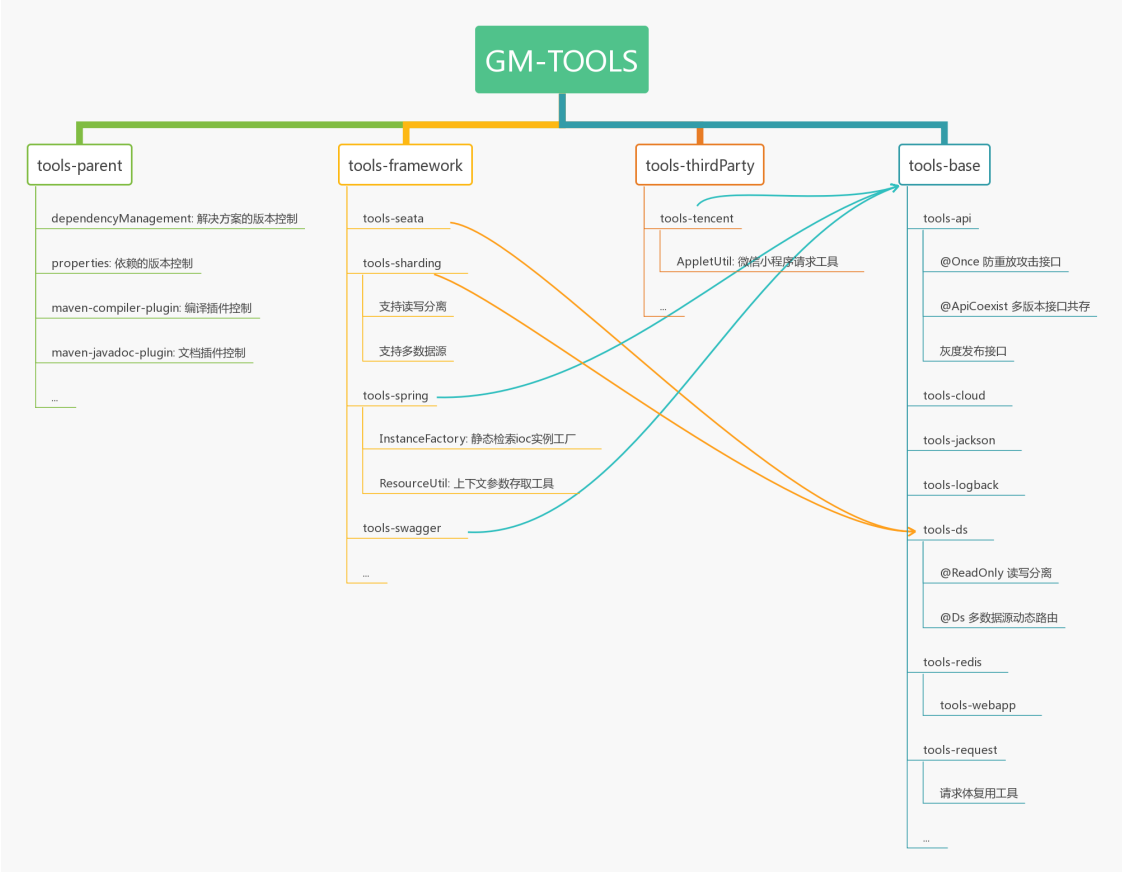
- 引用示例: 三级引用模块

> 复杂的旧系统可以直接加版本号引用工具包, 否则请选用继承, 新系统建议系统继承

```
<dependency>
    <groupId>cn.hll.tools</groupId>
    <artifactId>tools-{module}</artifactId>
    <!-- 未继承请添加以下版本号 -->
    <!-- <version>${latest.version}</version> -->
</dependency>
```

2.1.2 功能说明

- 模块总览



- 模块简介

tools-parent

> 航母包: 用于版本控制

- 解决方案版本管理
- 统一依赖的版本号
- 统一编译插件配置
- 统一文档插件配置

tools-base

> 基础包: 用于纯工具类的封装

- **异常类**
  - SkillException: 技术异常
    - 特点是: 包含异常码便于排查问题
  - AgreedException: 约定异常
    - 异常码: 开发者自定义
    - 特点是: 该类异常的信息将在前端直接提示给用户
  - UserExperienceException: 用户体验
    - 特点是: 继承了AgreedException其异常信息用于展示给用户
  - AssertException: 断言失败
    - 特点是: 代码简化
    - 使用前: 集成全局异常处理

    - DataException: 数据异常
        - 特点是: 数据库的数据效验类的异常 (如: 数据不存在, 数据已存在, 数据缺失, 数据不合法)
    - RemoteInvokeException: 远程调用异常
        - 异常码: 被调用方异常码 或 开发者自定义异常码
        - 特点是: 继承远程被调用方的响应码和提示信息
- **实体类**
    - HttpResult: 响应对象
        - 用于封装Http调用后的响应结果
    - JsonLog: 日志对象
        - 用于封装日志包输出结果
    - JsonResult: 响应对象
        - 用于统一前后端分离场景的响应数据格式
    - PageRequest: 分页请求对象
    - PageResponse: 分页响应对象
    - Login: 登陆用户信息对象
        - 用于统一前后端分离场景存储的用户信息
    - ~~Xcode~~: 异常码枚举类(未来将采用XCode但现在使用Xcode)
    - XCode: 响应码枚举类
    - XState: 状态码枚举类
- **工具类**
    - BoolUtil: 布尔工具, 任何需要判断的逻辑均须用到
        - 远程调用OK判断
        - 全部是真是假判断
        - 是否全部空字符判断
        - 全部非空是空判断
        - 大于等于小于判断
    - MathUtil: 数学公式工具, 简单的数学公式运算
        - 支持 `+` `-` `*` `÷`
        - 支持 `(` `)`
        - 支持 `>` `>=` `==` `<` `<=` `!=`
        - 支持 `?` `:`
    - MailUtils: 邮件工具, 发送各种类型的邮件
        - 支持文本、多图片、多附件
        - 支持多回执、多接收、多抄送、多密送

> 图文混合方式时, 文本内容需要使用HTML配合, 在image标签src属性中使用 `c i d : index` 代替图片, 其中index是图片文件数组的下标值

```
String[] images = {"c:\表情.png", "c:\头像.jpg", "d:\全身照.jpeg"};
String content = "这是个表情: <image src='cid:0'/>, 再来个头像: <image
src='cid:1'/><br/>这是全身照: <image src='cid:2'/>";
```

```
// 默认163服务器, 有其他服务器重载函数
MailUtils.Mail mail = MailUtils.build(username, password);
// mial.send(...);
```

- AesUtil: AES对称算法加密解密工具
  - 支持生成密钥和还原密钥
  - 默认采用ECB分组加/解密方式
- DesUtil: DES对称算法加密解密工具
- RsaUtil: Rsa非对称算法加密解密工具
- AssertUtil: 断言工具, 必须符合要求的业务逻辑需要用到
  - 布尔工具类有的, 这里都有
- BeanUtil: 对象工具类
  - 对象的参数值复制
  - 对象的类型转换
- ClassUtil: 字节码工具
  - 获取对象中包含的泛型类型
- CollectionUtil: 集合工具
  - 将Map根据key排序
- CoordsUtil: 坐标工具
  - 根据坐标获取指定举例的坐标范围
  - 根据两个坐标计算其直径距离
- ExceptionUtil: 异常工具
  - 优雅抛出异常 (比如: return 一个异常有没有见过呢?嘿嘿..)
  - 压制代码异常 (明确知道代码有可能异常,但不希望它抛出来)
- FileUtil: 文件工具
  - 文件上传工具集
- HttpUtil: Http协议工具
  - 发起各种请求
  - 解析各种响应
- IdUtil: 主键生成工具, 单机理论唯一
- ImgUtil: 图片工具
  - File与Base64互转
- DesensitizationUtil: 脱敏工具
- 身份证号码脱敏
- 银行卡号码脱敏
- 手机号码脱敏
- 真实姓名脱敏
- 邮件地址脱敏
- JsonUtil: 格式转换工具, 注意Long类型会转换成String类型
  - json转对象/Map/List
  - 对象转json
- 对象转其他类型对象
  - json格式化
- LocalDateTimeUtil: 新时间转换工具
- Date与LocalDateTimeUtil互转
- Md5Util: 不可逆加密算法
- Sha256Util: 不可逆加密算法

- NullUtil: 空异常规避工具
  - 空字符串转换为 ""
  - 空对象使用默认值
  - 空对象使用反射新对象
  - 空对象抛出提示性异常
- SignUtil: 接口鉴权工具, 前后端接口调用安全需要用到
- StreamUtil: 流工具
  - 输入输出流互转
  - 复用不可复用流
- TimeUtil: 时间格式化工具, 只有你想不到, 没有它做不到
- BeanUtil: 对象处理工具
  - 对象互换
  - 对象克隆/复制
- UrlUtil: URL地址工具
  - url编码、解码
- WebUtil: 请求工具
  - 字符集处理
  - 请求头工具集
  - ip, port, url, cookie工具集
  - 参数处理工具
- JdbcUtil: 数据库操作工具
  - 使用用户密码直接操作数据库
  - 使用数据源获取连接方式直接操作数据库
  - 使用自定义获取连接方式直接操作数据库

tools-logback

> 日志包: 用于日志输出的统一配置

- 日志输出配置: logback-dev.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<included>
    <!-- 以下字段支持自定义 -->
    <!--    <property name="CUSTOM_LOG_LEVEL" value="DEBUG"/>-->
    <!--    <property name="CUSTOM_SYS_PATH" value="/data/logs/"/>-->
    <!--    <property name="CUSTOM_SYS_NAME" value="tools-logback"/>-->
    <!--    <property name="CUSTOM_BUG_PACKAGE" value="cn.huolala"/>-->
    <!--    <property name="CUSTOM_SQL_PACKAGE" value="cn.huolala"/>-->
</included>
```

- 日志持久化: **方式1**
  - 用法: 使用tools日志工厂获取MysqlLogger实例

```
// import cn.hll.tools.tools.logback.db.LoggerFactory;
private MysqlLogger logger =
LoggerFactory.getLogger(AliPayServiceImpl.class, Log.class);
// ....
logger.print("支付宝H5请求: {}", JsonUtil.format(aliPayH5Request));
AlipayTradeWapPayResponse response = AliPayKit.execute(aliPayConfig,
aliPayH5Request);
logger.print("支付宝H5响应: {}", JsonUtil.format(response));
```

- 效果: 打印Logback日志的同时还会将日志插入到数据库表: `tools_log`

```
id  msg throwable_msg   create_at
6   ... xxx             2021-03-08 16:23:44
7   ... xxx             2021-03-08 16:23:44
8   ... xxx             2021-03-08 16:23:44
9   ... xxx             2021-03-08 16:23:44
10  ... xxx             2021-03-08 16:23:44
```

- 重写: 需要按自己的表、格式、内容保存到数据库
  1. 实现cn.hll.tools.tools.logback.db.Log实体

     > 该实体相当于保存到数据库的DTO

  2. 获取LoggerFactory.getLogger(AliPayServiceImpl.class, { **DTO** }.class);

     > 在获取MysqlLogger实例时使用新创建的Log实现类

  3. 在resources/mapper下创建{ DtoMapper.xml }

     > 该文档用于持久化DTO实体的SQL映射

     - 命名空间namespace: 必须为DTO类路径
     - 至少包含insert: 一定要有statementId为insert的语句

     ```
     <?xml version="1.0" encoding="UTF-8" ?>
     <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
     "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
     <!-- Log: 默认实体 -->
     <mapper namespace="cn.hll.tools.tools.logback.db.Log">

         <!-- insert: 默认句柄 -->
         <insert id="insert" useGeneratedKeys="true" keyProperty="id">
             insert into `tools_log` (`msg`, `throwable_msg`)
             values (#{msg}, #{throwableMsg})
         </insert>

     </mapper>
     ```

- 禁用: @DisabledInsertLogger: 禁止将日志保存到数据库中

```
@DisabledInsertLogger
@SpringBootApplication
public class App {

    public static void main(String[] args) {
        SpringApplication.run(App.class, args);
    }
}
```

- 日志持久化: **方式2**
  - 用法: 在方法上使用@ApiPrint注解
  - 效果: [2020-12-03 18:05:40.538] [http-nio-9740-exec-1] [INFO] c.huolala.common.logback.config.aop.ApiPrintAspect -

```
{
    "请求地址":"http://localhost:9740/tx/gray",
    "请求接口":"灰度测试 - 1.0",
    "请求参数":" version=1.0.0-GRAY grayCookie=[

\"http://localhost:9740/tx/gray\",\"http://localhost:9740/swagger-
resources/configuration/ui\",\"http://localhost:9740/swagger-
resources/configuration/security\",\"http://localhost:9740/swagger-
resources\",\"http://localhost:9740/v2/api-
docs\",\"http://localhost:9740/tx/come\"
    ]",
    "响应参数":{
        "code":200,
        "msg":"进入灰度..",
        "responseTime":"1606989940532"
    },
    "响应时间":"2020-12-03 18:05:40",
    "总耗时":"2",
    "代码位
置":"93:cn.huolala.nacos.server.pos.order.controller.TxController.gray"
}
```

  - **Redis输出**: 将日志输出到Redis

```
<appender name="REDIS"
class="cn.huolala.common.logback.appender.RedisAppender">
    <source>logback-redis</source>
    <host>127.0.0.1</host>
    <port>6300</port>
    <password>root</password>
    <database>3</database>
    <key>logstash</key>
    <timeout>3600</timeout>
    <type>${log.systemName}</type>
    <tags>${log.moduleName}</tags>
</appender>
```

```
<logger name="xxx">
    <appender-ref ref="REDIS"/>
</logger>
```

- **Mysql输出**: 将日志输出到Mysql

```
<appender name="MYSQL" class="ch.qos.logback.classic.db.DBAppender">
    <connectionSource
class="ch.qos.logback.core.db.DataSourceConnectionSource">
        <dataSource class="com.alibaba.druid.pool.DruidDataSource">
            <driverClassName>com.mysql.cj.jdbc.Driver</driverClassName>
            <url>jdbc:mysql://127.0.0.1:3306/xx?
serverTimezone=Asia/Shanghai</url>
            <username>root</username>
            <password>root</password>
        </dataSource>
    </connectionSource>
</appender>
```

```
<logger name="xxx">
    <appender-ref ref="MYSQL"/>
</logger>
```

tools-api

> 接口包: 用于强化后端接口

- 接口签名: 不依赖第三方组件实现的接口签名方案

  > 好处: 只需如下所示的简单配置即可生效
  >
  > 缺点: 采用了配置化存储密钥方案, 不可动态更新应用密钥, 仅用于后端口接口签名不支持

  ```
  common.api.sign.expireSecond=10
  common.api.sign.appKeys={'应用编号1':'密钥1','应用编号2':'密钥2'}
  ```

- 防止攻击: 在需要防攻击的接口方法上加@Once注解即可

  > 详见: 接口防重放机制工具包使用简要说明.md

- 接口共存: 如果需要发布相同地址不同版本的接口只需要在方法上加@ApiCoexist注解即可

  > 当然还有不加注解也能实现的办法, 但不建议, 因为那略微复杂一丢丢

  > 可以排除加载: @SpringBootApplication(exclude = ApiCoexistAutoConfiguration.class)

  > 详见: 多版本接口共存工具包使用简要说明.md

- 灰度发布: 如果需要在正式环境释放部分用户进行访问 (支持全链路灰度)

  1. 配置灰度策略
  2. 在接口方法上加@ApiCoexist标明灰度版本号

  > 可以排除灰度: @SpringBootApplication(exclude = ApiGrayFilterAutoConfiguration.class)

> 可以排除全链路灰度: @SpringBootApplication(exclude = ApiCoexistFeignLineGrayAutoConfiguration.class)
>
> 详见: 多版本灰度发布工具包使用简要说明.md

tools-cloud

> 远程包: 用于远程调用工具的管理

- ApiFallback: 熔断回调通用对象
- RemoteUtil: 远程调用结果断言工具类

tools-jackson

> Jackson包: 用于jackson相关工具包的封装

- 全局WEB响应处理配置
- 主要包含以下功能模块: Long转String响应、时间转换处理

```
// 注册时间类型转换模块
JacksonAssist.registerTimeModule(objectMapper);
// 注册其他类型转换模块
JacksonAssist.registerConvertModule(objectMapper);
// 注册异常禁止抛出模块
JacksonAssist.registerDisableModule(objectMapper);
// 注册特定忽略场景模块
JacksonAssist.registerIgnoreModule(objectMapper);
// 注册开启特殊功能模块
JacksonAssist.registerEnableModule(objectMapper);
```

tools-ds

> Ds包: 用于Mysql持久化功能的集成

- 读写分离

  > 详见: 数据源读写分离工具包使用简要说明.md

- 多数据源动态路由
- Druid连接池及其监控集成

tools-mybatis

> Mybatis包: 用于集成Mybatis持久化框架

- 集成myabtis
- 集成myabtis-plus
- 内置代码生成工具

tools-redis

> Redis包: 用于Redis解决方案的集成

- RedisUtil: Redis聚合工具包, 包含Client, ID, LOCK
- RedisClient: Redis客户端操作工具
- RedisId: Redis分布式主键生成方案
- RedisLock: Redis分布式锁解决方案

tools-webapp

> Web包: 用于Web应用通用解决方案的集成

- 统一跨域处理
- 静态资源映射

  > 映射后, 只需要访问: `/disk/**` 即可访问以下所有资源

  - 项目类路径下的文件可以直接访问
  - 资源目录下的文件可以直接访问
- AuthController: 统一认证工具, **该工具必须配合TokenGenerator使用**

  > 用法: 将需要登陆的接口所在的控制器继承 AuthController

  ```
  @Validated
  @RestController
  @RequestMapping("user")
  @Api(tags = {"用户接口"})
  public class UserController extends AuthController {...}
  ```

  > 好处: 前端只需传token请求头, 后端可以直接在控制器获取到用户信息

- SimpleAuthController: 统一认证工具, **适用新老系统**

  > 用法: 将需要登陆的接口所在的控制器继承 SimpleAuthController

  ```
  @Validated
  @RestController
  @RequestMapping("user")
  @Api(tags = {"用户接口"})
  public class UserController extends SimpleAuthController {...}
  ```

  > 好处: 可以匹配任何令牌发放的形式, 不限制token存储的对象类型 (包括token前缀)

- ParameterController: 线程安全的请求响应对象存取

  > 用法: 将需要请求响应对象的接口所在的控制器继承 ParameterController

- SignController: 接口鉴权控制器

  > 用法: 将需要鉴权接口所在的控制器继承 SignController
  >
  > 优势: 采用了Redis存储密钥方案, 可以动态更新应用密钥, 提高密钥安全性, 可用于前端签名

```
common.webapp.sign.appKeyPrefix=REDIS:SIGN:APP_KEY_
common.webapp.sign.noncePrefix=REDIS:SIGN:IDEM_NONCE_
common.webapp.sign.expireSecond=10
common.webapp.sign.appKeys={'应用编号1':'密钥1','应用编号2':'密钥2'}
```

- TokenGenerator: 令牌发放工具类

  > 在任何加入Ioc对象的位置继承 TokenGenerator类

```
@Service
public class LoginServiceImpl extends TokenGenerator<UserInfo,
PermissionInfo, SettingInfo, DataInfo, ConfigInfo> implements LoginService {
    ...
    // 该方法采用Redis保存用户信息
    super.generate(login, EXPIRE_SECOND);
}
```

tools-request

> 请求包: 用于封装通用的Request改造工具

- 请求体复用: 用于多次读取请求体数据

  > 一般工具中使用

  - 只读支持的枚举类型数据: form、data, json

tools-seata

> Seata包: 用于分布式事务Seata解决方案的集成

- 分布式事务自动装配: 在需要分布式事务的方法上加: @GlobalTransactional注解即可

tools-shardingJdbc

> Sharding包: 用于分库分表ShardingJdbc解决方案的集成

- 分库分表自动装配: 只需要添加分库分表配置即可

```
# 多数据源配置 (支持读写分离)
spring.shardingsphere.datasource.names=master0,master0slave0
spring.shardingsphere.datasource.master0.type=com.alibaba.druid.pool.DruidDa
taSource
spring.shardingsphere.datasource.master0.driver-class-
name=com.mysql.cj.jdbc.Driver
spring.shardingsphere.datasource.master0.url=jdbc:mysql://127.0.0.1:3306/ord
er?useUnicode=true&characterEncoding=UTF-8&serverTimezone=Asia/Shanghai
spring.shardingsphere.datasource.master0.username=root
spring.shardingsphere.datasource.master0.password=root

spring.shardingsphere.datasource.master0slave0.type=com.alibaba.druid.pool.D
ruidDataSource
spring.shardingsphere.datasource.master0slave0.driver-class-
name=com.mysql.cj.jdbc.Driver
```

```
spring.shardingsphere.datasource.master0slave0.url=jdbc:mysql://127.0.0.1:33
06/order?useUnicode=true&characterEncoding=UTF-
8&serverTimezone=Asia/Shanghai
spring.shardingsphere.datasource.master0slave0.username=root
spring.shardingsphere.datasource.master0slave0.password=root

# 打印SQL
spring.shardingsphere.props.sql.show=true

# 分库分表主表策略配置 (是否分库分表由是否配置策略决定)
spring.shardingsphere.sharding.tables.t_pos_order_midway.actual-data-
nodes=ds$->{0..0}.t_pos_order_midway$->{1..2}
spring.shardingsphere.sharding.tables.t_pos_order_midway.table-
strategy.inline.sharding-column=order_id
spring.shardingsphere.sharding.tables.t_pos_order_midway.table-
strategy.inline.algorithm-expression=t_pos_order_midway$->
{order_id.toBigInteger() % 2 + 1}

# 分库分表主表策略配置 (是否分库分表由是否配置策略决定)
spring.shardingsphere.sharding.tables.t_pos_order_log.actual-data-nodes=ds$-
>{0..0}.t_pos_order_log$->{1..12}
spring.shardingsphere.sharding.tables.t_pos_order_log.table-
strategy.inline.sharding-column=order_id
spring.shardingsphere.sharding.tables.t_pos_order_log.table-
strategy.inline.algorithm-expression=t_pos_order_log$->
{order_id.toBigInteger() % 12 + 1}

# 默认分库配置: 无分库策略的库不分库
spring.shardingsphere.sharding.default-database-strategy.none=Timi°
# 默认分表配置: 无分表策略的表不分表
spring.shardingsphere.sharding.default-table-strategy.none=Timi°

# 默认数据源: 不分库的数据从这个库操作, 支持外库
spring.shardingsphere.sharding.default-data-source-name=ds0

# 读写分离配置
spring.shardingsphere.sharding.master-slave-rules.ds0.master-data-source-
name=master0
spring.shardingsphere.sharding.master-slave-rules.ds0.slave-data-source-
names=master0slave0
```

tools-spring

> Spring包: 用于提供Spring环境下通用的工具

- InstanceFactory: 通过它可以获得ioc管理的类的实例, 在任何地方
- ResourceUtil: 资源文件加载工具类, 通过它可以获取所有Spring上下文中的配置

> 并且可在运行时, 修改参数值, 使其它使用该参数的地方动态变更, 这其中的美妙, 好好想想吧, 嘿嘿..

tools-swagger

> 文档包: 用于提供后端接口的在线文档

- swagger自动装配: 只需要在接口类上使用 @Api即可生成在线文档

  > 可以排除加载: @SpringBootApplication(exclude = SwaggerAutoConfiguration.class)

tools-alibaba

- AliPayKit: 支付宝支付接口集成工具
  - H5支付
  - Native支付
  - App支付

tools-tencent

> 腾讯包: 用于集成腾讯平台的开放接口

- WxUtil: 微信基础接口集成工具
  - 接口凭证生成器
  - 用户敏感信息解密
- AppletUtil: 微信小程序开发接口集成工具
  - 小程序二维码生成工具
- MsgUtil: 消息推送接口集成工具
- RssUtil: 订阅号集成工具
- ServiceUtil: 服务号集成工具
- WxPayKit: 微信支付工具
  - JSAPI支付
  - Native支付
  - H5支付
  - App支付
- V3ApiKit: v3接口规则工具
  - 支付通知敏感信息解密

2.2 版本记录

v-4.0.0

- 全部模块更换groupId以及包名
- 全部模块进行依赖优化, tools-base不再包含任何依赖传递
- 登陆授权工具增加公司SSO适配优化

v-3.1.1

tools-base

- 增加LoginException异常类
- 增加ExceptionUtil对异常形参的支持
- 增加SnUtil工具
  - 支持自定义或默认前缀加密数值 (固定32位)
  - 支持自定义或默认前缀日期编码 (固定20位)
- 增加XoaUtil对称加密工具
  - 支持byte/int/long/string加密
- 增加SerializableUtil序列化工具
  - 支持将任意对象序列化成字符串或字节数组
  - 支持将字符串或字节数组反序列化成任意对象
- 增加JvmUtil虚拟机工具
  - 获取方法调用堆栈信息
- 增加validation枚举校验器
  - 支持数组枚举
  - 支持枚举类枚举
- 增加JdbcUtil对连接自动关闭形参的支持
- 增加BigDecimalUtil工具
  - 计算浮点型的乘除运算
  - 计算金额元转分 及其 反转的运算
- 增加NumUtil工具
  - 对数字进行补全操作
- 增加Int枚举类
  - 枚举了0-10的int数值
- 增加BoolUtil是否对象的判断工具
- 增加ThreadUtil线程池异步操作工具
  - 简单的异步操作
  - 带返回值处理的异步操作
- 增加Int枚举工具
  - 支持0~10的数字枚举
- 增加BoolUtil对象识别支持
- 增加BigDecimalUtil浮点型计算工具
  - 支持元 <-> 分 互转
  - 支持`+` `-` `*` `/`运算
- 增加ExceptionUtil功能
  - 支持沙箱操作
- 增加ThreadUtil线程池工具

tools-kv

> 致力于KV数据库的伪装方案实施

- 增加FakeRedis实现 (基于Mysql)
    - 支持get/set/getset/setNx/setEx
    - 支持自动清理key

tools-mail

> 致力于邮件发送服务的方案实施

- 增加全局切面异常告警
- 增加@SendMail方法环绕通知

tools-overstep

> 致力于越权解决方案的实施

- 支持Form表单提交解码
- 支持jackson全局无感编解码
- 支持fastJson全局无感编解码

v-3.1.0

tools-base

- 增加JdbcUtil数据源的支持
- 增加JdbcUtil其他获取连接方式的支持
- 增加RsaUtil非对称加密解密工具

tools-datalog

- 增加tools-datalog: 数据变更日志工具包

tools-request

- 升级请求/响应体复用方案: 采用spring官方缓存方案
- 增加Model响应数据脱敏工具: 采用aop方案

v-3.0.4

tools-base

- 增加AsyncHttpUtil工具
- 增加RouteHttpUtil工具
- 支持StreamUtil对象转字节流
- 增加JdbcUtil工具
- 增加ExcelUtil工具

v-3.0.3

tools-base

- 增加Http预检工具
- 增加ClassUtil对象反射生成Map工具
- 增加SignUtil验签实体类签名支持
- 增加AesUtil对称加密工具
- 增加DesUtil对称加密工具

tools-mybatis

- 拆分tools-mybatis工具

tools-tencent

- 增加wx-pay支付结果查询工具
- 增加公众号Oauth2.0认证工具

tools-alibaba

- 增加ali-pay支付结果查询工具

v-3.0.2

tools-base

- 增加Xml处理工具: xml转obj、obj转xml以及对应xml文件的相互转换
- 增加Zxing二维码生成/解析/合成工具

tools-logback

- 增加MysqlLogger工具

tools-mysql

- 增加IBatisDao传统映射持久化操作工具
- 增加BatchMapper批量持久化操作工具 (mp)
- 增加随机动态数据源支持并去除@ReadyOnly

tools-es

- 增加es集成工具包

tools-tencent

- 增加微信支付工具: H5PayKit、NativePayKit、JsPayKit、V3ApiKit等

tools-alibaba

- 增加阿里巴巴第三方工具包
- 增加支付宝支付工具: AliPayKit等

v-3.0.1

tools-base

- 去除Xcode采用XCode
- 增加BoolUtil、AssertUtil中包含和不包含检车工具

tools-logback

- 极简化微服务日志配置
- 增加控制台日志输出全彩支持
- 增加邮件通知工具

tools-redis

- 增加redis自动装配
- 增加jedis连接池支持

tools-webapp

- 新增登陆令牌保存权限信息

common-mysql

- 增加mybatisPlus全局配置的原生支持
- 调整mapperScan路径为: **.dao.mapper.**

tools-tencent

- 增加WxUtil: 微信通用工具
- 调整AppletUtil: 微信小程序工具
- 增加RssUtil: 微信公众号订阅号工具
- 增加ServiceUtil: 微信公众号服务号工具
- 增加MsgUtil: 微信消息推送工具 (包含公众号和小程序的消息)

v-3.0.0

- 由于 [cn.hll、cn.huolala] 包名和加载路径与项目代码有冲突, 因此改名GM-TOOLS

tools-webapp

- 新增登陆令牌保存权限信息

common-mysql

- 增加mybatisPlus全局配置的原生支持

v-2.0.7

tools-api

- 增加接口签名工具 (配置化方案)

tools-webapp

- 优化接口签名工具 (**Redis方案**)

v-2.0.6

tools-logback

- 增加logback公共配置抽取

tools-api

- 增加全链路灰度发布支持

tools-redis

- 增加redis客户端过期时间操作工具

tools-base

- 增加DesensitizationUtil脱敏工具
- 增加ExceptionUtil压制代码异常工具

tools-webapp

- 增加SimpleAuthController/SimpleTokenGenerator, 用于自定义token存储对象
- 增加common.webapp.tokenPrefix注入, 用于自定义redis的Token前缀

v-2.0.6.Final

tools-api

- 定制SpringBoot 1.5.2版本兼容的防止重复提交方案

v-2.0.5

tools-webapp

- 优化跨域过滤器自定义配置项

v-2.0.4

tools-base

- 增加数学公式计算工具
- 增加邮件发送工具类, 添加集合判空工具
- 增加Url编码解码工具

tools-api

- 优化灰度过滤器支持多容器实现

tools-mysql

- 优化druid默认前缀的配置读取

tools-swagger

- 独立在线文档集成工具包: tools-framework/tools-swagger

tools-webapp

- 增加Spring参数绑定异常的统一处理

tools-tencent

- 增加小程序客服消息推送工具

v-2.0.3

tools-base

- 增加登陆对象扩展用户信息、用户设置、用户数据、系统配置
- 优化Null异常规避工具的提示信息

v-2.0.2

tools-mysql

- 增加多数据源读写分离支持
- 增加多数据源多从随机支持

v-2.0.1

tools-api

- 增加灰度发布工具
- 增加固用灰度支持
- 增加多策略混合灰度发布支持

v-2.0.0

tools-api

- 修复多版本接口共存被覆盖问题

tools-mysql

- 增加多数据源动态路由支持
- 升级Mysql工具包自动装配
- 增加指定数据源操作支持

tools-webapp

- 升级Swagger自动装配

tools-shardingJdbc

- 解决与mysql不兼容的问题

v-1.2.5

tools-mysql

- 解决mapper扫描包含apache内部文件问题

tools-shardingJdbc

- 增加标准日期范围搜索分片算法工具

v-1.2.4

tools-base

- 增加LocalDateTime工具
- 增加数据异常工具
- 增加接口数据效验工具
- 增加分页请求和响应工具

v-1.2.3

tools-parent

- 增加GT各工具包独立版本号支持

tools-jackson

- 优化jackson配置

tools-mysql

- 增加mybatis系列映射文件自定义路径扫描

tools-shardingJdbc

- 增加标准日期分片键算法工具
- 增加分片键算法对Date和LocalDateTime的兼容支持

v-1.2.2

tools-parent

- 升级mybatis-plus指最新版本

tools-mysql

- 修改数据源默认配置
- 增加Mybatis读写分离支持
- 增加MybatisPlus读写分离支持

tools-seata

- 修改数据源默认配置
- 增加Mybatis读写分离支持
- 增加MybatisPlus读写分离支持

v-1.2.1

tools-jackson

- 优化WebMvc共存问题

tools-webapp

- 优化WebMvc共存问题

v-1.2.0

tools-api

- 增加防重放攻击工具

tools-webapp

- 修复WebMvc共存问题

v-1.1.11

tools-shardingJdbc

- 独立分库分表工具包: tools-framework/tools-shardingJdbc

tools-request

- 独立请求改造工具包: tools-request
- 迁入请求体复用工具

v-1.1.10

tools-redis

- 修复RedisId和RedisClient&RedisLock不能共用问题

v-1.1.9

tools-base

- 迁入接口签名工具
- 增加静态资源映射工具
- 增加静态资源自定义映射功能

v-1.1.8

tools-parent

- 升级spring-cloud-alibaba到最新版本
- 升级druid连接池至最新版本

v-1.1.7

tools-base

- 增加远程调用工具类
- 增加远程调用异常工具

tools-jackson

- 独立jackson工具包: tools-jackson

v-1.1.6

tools-mysql

- 解决多数据源懒加载问题
- 优化多数据源内存泄露问题

tools-seata

- 独立分布式事务工具包: tools-framework/tools-seata
- 增加分布式事务自动装配

v-1.1.5

tools-api

- 增加多版本接口共存工具

tools-logback

- 修复日志输出到redis问题

tools-base

- 升级坐标计算工具

tools-tencent

- 优化小程序请求异常提醒

v-1.1.4

tools-redis

- 独立redis工具包: tools-redis

tools-mysql

- 独立mysql工具包: tools-mysql

tools-api

- 独立api工具包: tools-api

v-1.1.3

tools-tencent

- 独立腾讯工具包: tools-thirdParty/tools-tencent
- 增加小程序接口凭证生成工具
- 增加小程序二维码生成工具
- 增加小程序用户敏感信息解密工具

v-1.1.2

tools-webapp

- 修复请求体复用不支持文件上传的问题
- 增加Http请求处理工具
- 增加坐标计算工具

v-1.1.1

tools-parent

- 增加通用编译插件rt.jar、jce.jar支持

v-1.1.0

tools-base

- 增加Null异常规避工具
- 增加断言工具
- 增加布尔值判断工具

v-1.0.9

tools-base

- 增加统一登录实体

tools-webapp

- 优化登陆用户信息解析工具

v-1.0.8

tools-webapp

- 增加登陆令牌发放工具
- 增加令牌用户数据解析工具
- 增加Swagger环境自动装配方案
- 解决Redis 连接重置问题

v-1.0.7

tools-parent

- 增加GM-TOOLS版本控制

tools-webapp

- 增加404异常转为HTTP响应码 `200` 响应
- 增加Swagger支持自定义访问前缀

v-1.0.6

~~**tools-database**~~

- 优化RedisClient泛型存取方法

v-1.0.5

tools-base

- 增加Xcode响应码枚举工具
- 增加技术类异常工具

~~**tools-database**~~

- 增加请求/响应日志自动输出工具
- 增加Druid连接池监控工具

v-1.0.4

tools-parent

- 独立航母工具包: tools-parent
- 优化微服务版本控制方案
- 导入spring-boot-dependencies版本控制
- 导入spring-cloud-dependencies版本控制
- 导入spring-cloud-alibaba-dependencies版本控制

tools-base

- 独立基础工具包: tools-base
- 增加SpringMvc文件上传工具
- 增加MD5工具
- 增加签名工具
- 增加Json交互响应工具

tools-logback

- 独立日志工具包: tools-logback
- 增加spring-profile装载微服务日志配置文件

tools-cloud

- 独立云服务工具包: tools-cloud
- 增加熔断工具

~~tools-database~~

- 独立数据库工具包: 导入~~tools-database~~ (新版更名)
- 增加mysql读写分离工具
- 增加Redis缓存工具
- 增加Redis分布式主键工具

tools-webapp

- ~~增加幂等算法~~
- 增加登陆认证工具
- 增加全局统一异常处理工具