# 一、功能说明

该工具包的作用是允许后端发布相同业务功能不同版本的接口, 并使其兼容发布同时提供服务

**背景**

- 产品终端多样化进入常态, 但各终端版本更新迭代的频率不一致
- 因此后端需要兼容多个版本的业务请求, 提供多版本的业务接口

# 二、工具使用

- 添加依赖: pom.xml

```
<dependency>
    <groupId>cn.gmlee</groupId>
    <artifactId>tools-api</artifactId>
    <version>3.0.0</version>
</dependency>
```

- 启用工具

> 在启动类上或其他配置类上扫描包: cn.gmlee.tools

```
@SpringBootApplication
@ComponentScan("cn.gmlee.tools")
public class XxxApp {
    public static void main(String[] args) {
        SpringApplication.run(XxxApp.class, args);
    }
}
```

- 发布版本资源

1. 无版资源:

```
@PostMapping("gray")
@ApiPrint("无版测试")
@ApiOperation(value = "无版测试")
public JsonResult version(
    @RequestHeader(value = "version", required = false) String version,
    @CookieValue(value = "GRAY_COOKIE", required = false) String
grayToken
) {
    System.out.println("进入无版..");
    System.out.println("================= version: " + version);
    System.out.println("================= grayToken: " + grayToken);
    return JsonResult.OK.newly("进入无版..");
}
```

2. 旧版资源:

```
@PostMapping("gray")
@ApiOperation(value = "1.0.0测试")
@ApiPrint("1.0.0测试")
@ApiCoexist("1.0.0")
public JsonResult version100(
    @RequestHeader(value = "version", required = false) String version,
    @CookieValue(value = "GRAY_COOKIE", required = false) String
grayToken
) {
    System.out.println("进入1.0.0..");
    System.out.println("================= version: " + version);
    System.out.println("================= grayToken: " + grayToken);
    return JsonResult.OK.newly("进入1.0.0..");
}
```

3. 新版资源: @ApiCoexist("1.0.1")

```
@PostMapping("gray")
@ApiOperation(value = "1.0.1测试")
@ApiPrint("1.0.1测试")
@ApiCoexist("1.0.1")
public JsonResult version101(
    @RequestHeader(value = "version", required = false) String version,
    @CookieValue(value = "GRAY_COOKIE", required = false) String
grayToken
) {
    System.out.println("进入1.0.1..");
    System.out.println("================= version: " + version);
    System.out.println("================= grayToken: " + grayToken);
    return JsonResult.OK.newly("进入1.0.1..");
}
```

- 注解说明: @ApiCoexist
  - 默认值: 1.0.0

# 三、工具原理

**逻辑**

- 重写RequestMappingHandlerMapping封装资源
- 所有带有@ApiCoexist注解的资源将作为自动匹配条件之一

  > 此功能将是多版本共存的核心支持

- 请求时根据请求头中的`version`参数作为条件匹配资源
- 选择符合条件的资源给予访问, 没有合适的version资源则放弃version条件

#### 代码

- 重写getMatchingCondition(..)条件
- 返回请求头中的version值与资源注解@ApiCoexist的值一致的匹配规则
- 匹配到响应版本的资源并访问

#### 数据

- 没有产生数据

## 四、使用示例

- 访问无版:

  > 不携带version或者version资源不存在

  ```
  {
    "code": 200,
    "msg": "进入无版..",
    "data": null,
    "responseTime": "1606895651136"
  }
  ```

- 访问1.0.0

  > 携带请求头: version=1.0.0

  ```
  {
    "code": 200,
    "msg": "进入1.0.0..",
    "data": null,
    "responseTime": "1606895715217"
  }
  ```

- 访问1.0.1

  > 携带请求头: version=1.0.1

  ```
  {
    "code": 200,
    "msg": "进入1.0.1..",
    "data": null,
    "responseTime": "1606895757925"
  }
  ```