

【反射】動態執行某類別方法

```
@PostMapping(value = "/test/getApi/{className}", consumes = MediaType.ALL_VALUE)
public ResponseEntity<?> getApi(
    @PathVariable String className,
    @RequestBody HashMap<String, Object> param
) throws Exception{

    HttpStatus status = HttpStatus.OK;
    Object res = null;
    HashMap classMap = new HashMap<String, String>();
    // 對應完整class
    classMap.put("TencentStreamingAPI", "com.test.api.TencentStreamingAPI");

    String fullClassName = (String)classMap.get(className);
    String func = "";

    try {
        if(fullClassName == null) throw new Exception("no className mapping fullClassName");
        if((String)param.get("func") != null){
            func = (String)param.get("func");
        }else{
            throw new Exception("no func name");
        }
    }

    Class c = Class.forName(fullClassName);
    Object ts = c.newInstance();

    switch (className){
        case "TencentStreamingAPI":
            ts = tencentStreamingAPI;
            c = ts.getClass();
            break;
        case "TencentIMAPI":
            ts = tencentIMAPI;
            c = ts.getClass();
            break;
    }

    Method[] methods = c.getMethods();
    for (Method method : methods) {
        if (method.getName().equals(func)) {

            if (method.getParameters().length == 0) {
                Method m = c.getMethod(func, null);
                res = m.invoke(ts, null);

            } else {
                ArrayList<Class<?>> parameterTypesList = new ArrayList<Class<?>>();
                ArrayList<Object> argsList = new ArrayList<Object>();

                for(Parameter parameter : method.getParameters() ){
                    String parameterName = parameter.getName();
                    Class clazz = parameter.getType();
                    Object arg = param.get(parameterName);
                    if(arg == null) throw new Exception("no such parameter: "+ parameterName);
                    parameterTypesList.add(clazz);
                    argsList.add(arg);
                }

                Method m = c.getMethod(func, String.class);
                Method m = c.getMethod(func, parameterTypesList.toArray(new Class[parameterTypesList.size()]));
                res = m.invoke(ts, argsList.toArray(new Object[argsList.size()]));
            }
            break;
        }
    }
}
```

```

    }

    }

    } catch (ClassNotFoundException e) {
        logger.error(e.getMessage());
        throw new Exception("no such class: " + className);
    } catch (NoSuchMethodException e) {
        logger.error(e.getMessage());
        throw new Exception("no such metod:" + func);
    } catch (IllegalArgumentException e) {
        logger.error(e.getMessage());
        throw new Exception("illegal argument :" + func);
    } catch (Exception e) {

        //取得目前執行方法名稱
        String methodName = "[noGetMethodName]";
        Method method = this.getClass().getEnclosingMethod();
        if(method != null){
            methodName = method.getName();
        }else{
            StackTraceElement[] stackTrace = Thread.currentThread().getStackTrace();
            for (StackTraceElement stackTraceElement : stackTrace) {
                methodName = stackTraceElement.getMethodName();
                if (!methodName.equals("getStackTrace") &&
                    !methodName.equals("getMethodName") &&
                    !methodName.equals("main")) {
                    break;
                }
            }
        }

        logger.error(this.getClass().toString()+" -> "+
            methodName + ": " + e.getMessage());
        status = HttpStatus.FORBIDDEN;
    }

    HttpHeaders headers = new HttpHeaders();
    headers.setContentType(MediaType.APPLICATION_JSON);
    String resultStr = "{}";

    if(res instanceof JSONObject){
        resultStr = ((JSONObject) res).toString();
    }else if(res instanceof JSONArray){
        resultStr = ((JSONArray) res).toString();
    }else if(res instanceof String){
        resultStr = (String) res;
    }

    // logger.debug("func name is {}",func);
    // logger.debug("func res type is {}",res.getClass().getName());
    return new ResponseEntity<>(resultStr, headers, status);
}

```

```

public class TencentStreamingAPI {
    public JSONObject describeStreamPlayInfoList(String streamId,String starttime, String endtime){
        .....
    }
}

```

```

// 呼叫
http://localhost:8080/api/test/getApi/TencentStreamingAPI

/** json 參數
func: 執行方法名稱
其他帶入同名參數
*/
{

```

```
"func": "describeStreamPlayInfoList",
"streamId": "AABBCCDD",
"starttime": "2023-03-29 13:18:00",
"endtime": "2023-03-29 13:18:00"
}
```

🕒 修訂版本 #1
★ 由 treeman 建立於 29 🕒🕒🕒 2023 13:48:27
✍ 由 treeman 更新於 5 🕒🕒🕒🕒 2023 10:18:39