

# hypermesh后处理

基础概念：层级结构

层级作用的范围

后处理二次开发的基本思路

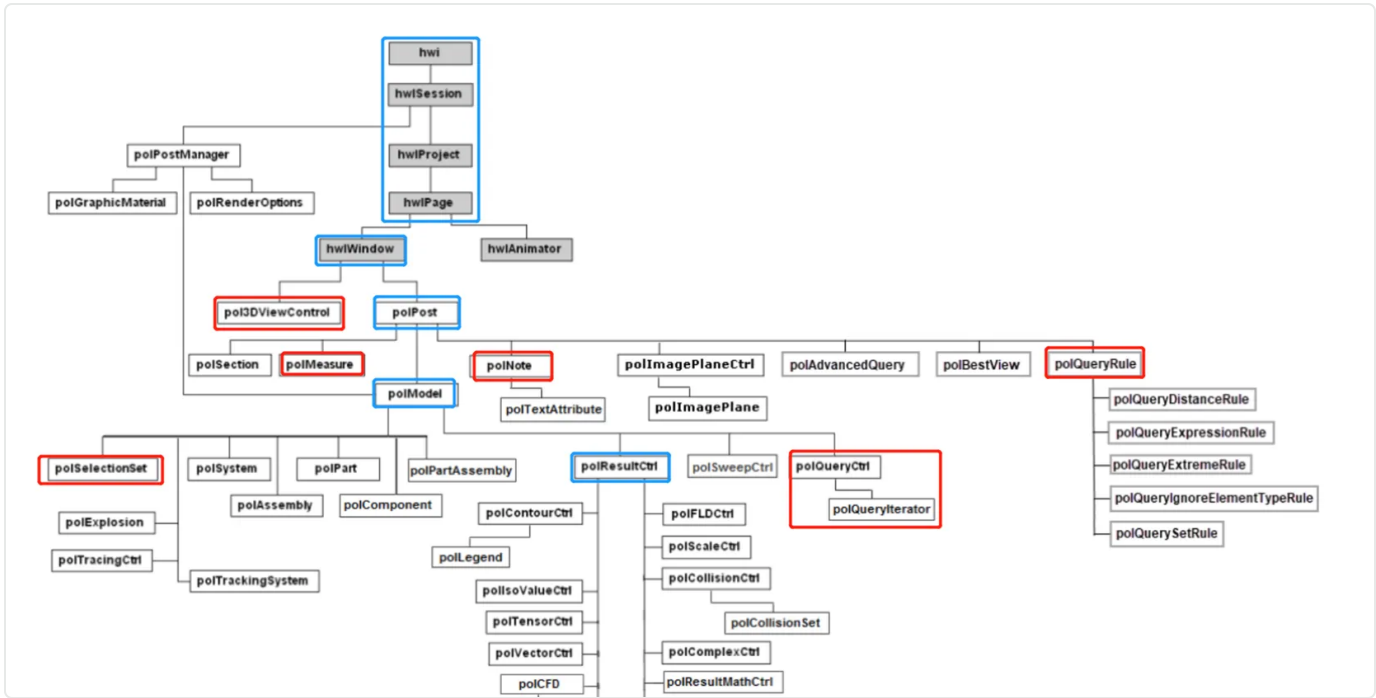
HWC介绍

在新版本后处理二次开发的流程思路

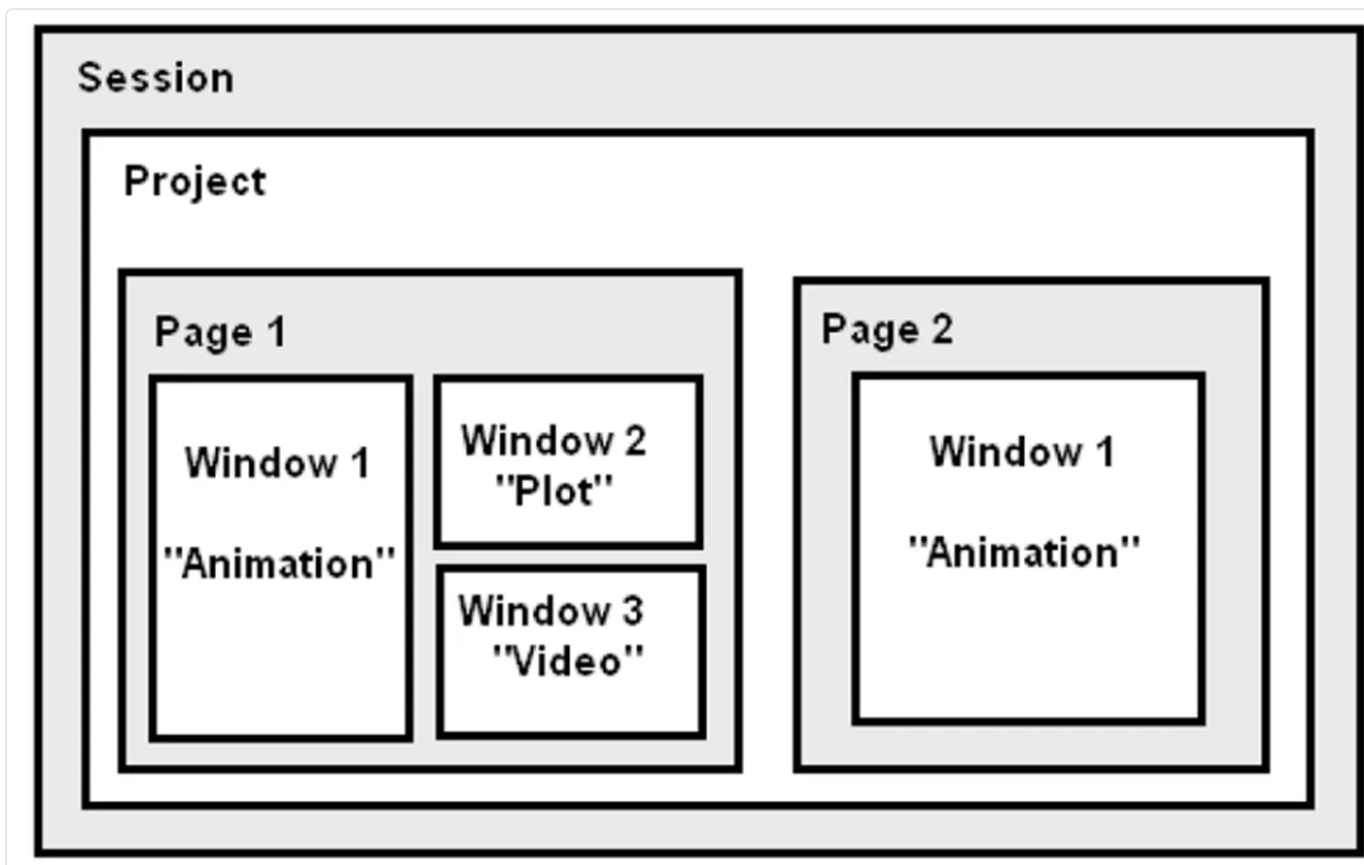
在HV中读取数据

HG中添加曲线

## 基础概念：层级结构



## 层级作用的范围



## 后处理二次开发的基本思路

1. 查找你的操作或者查询所在的对象层级；
2. 从hwi开始着级任命你的各个层级的对象；
3. 当任命到你需要操作或查询的命令所在的对象时， 调用操作或者查询命令，完成动作；
4. 动作都完成后，一定要**关闭所有任命的对象**；

例1: 新增界面

```
1 hwi OpenStack
2     hwi GetSessionHandle sess
3     sess GetProjectHandle proj
4     proj AddPage
5 hwi CloseStack
```

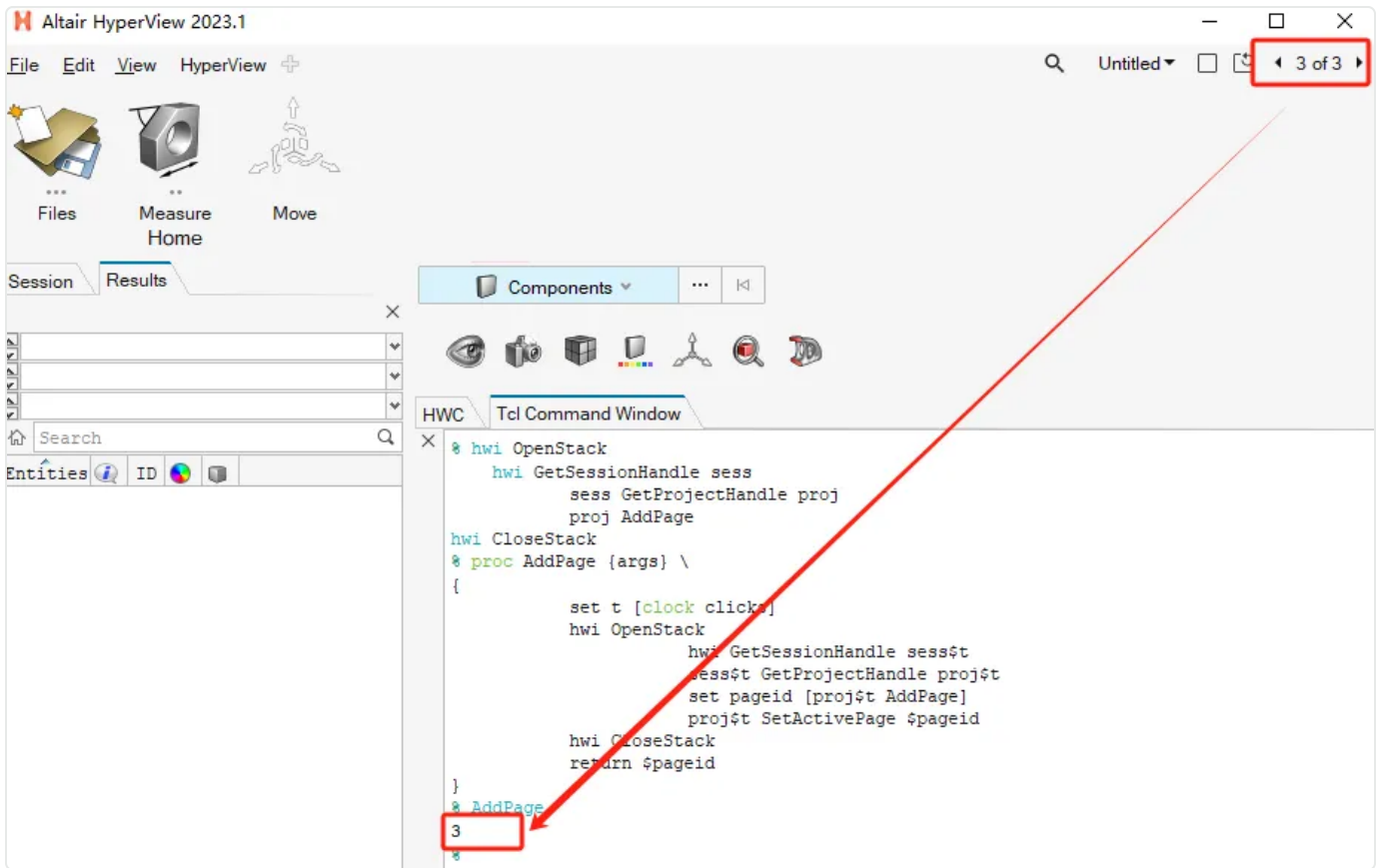
## 1. 创建一个时间变量

```
1 set t [clock clicks]
```

2. 从hwi开始着级任命你的各个层级的对象，创建的handle必须以\$t结尾

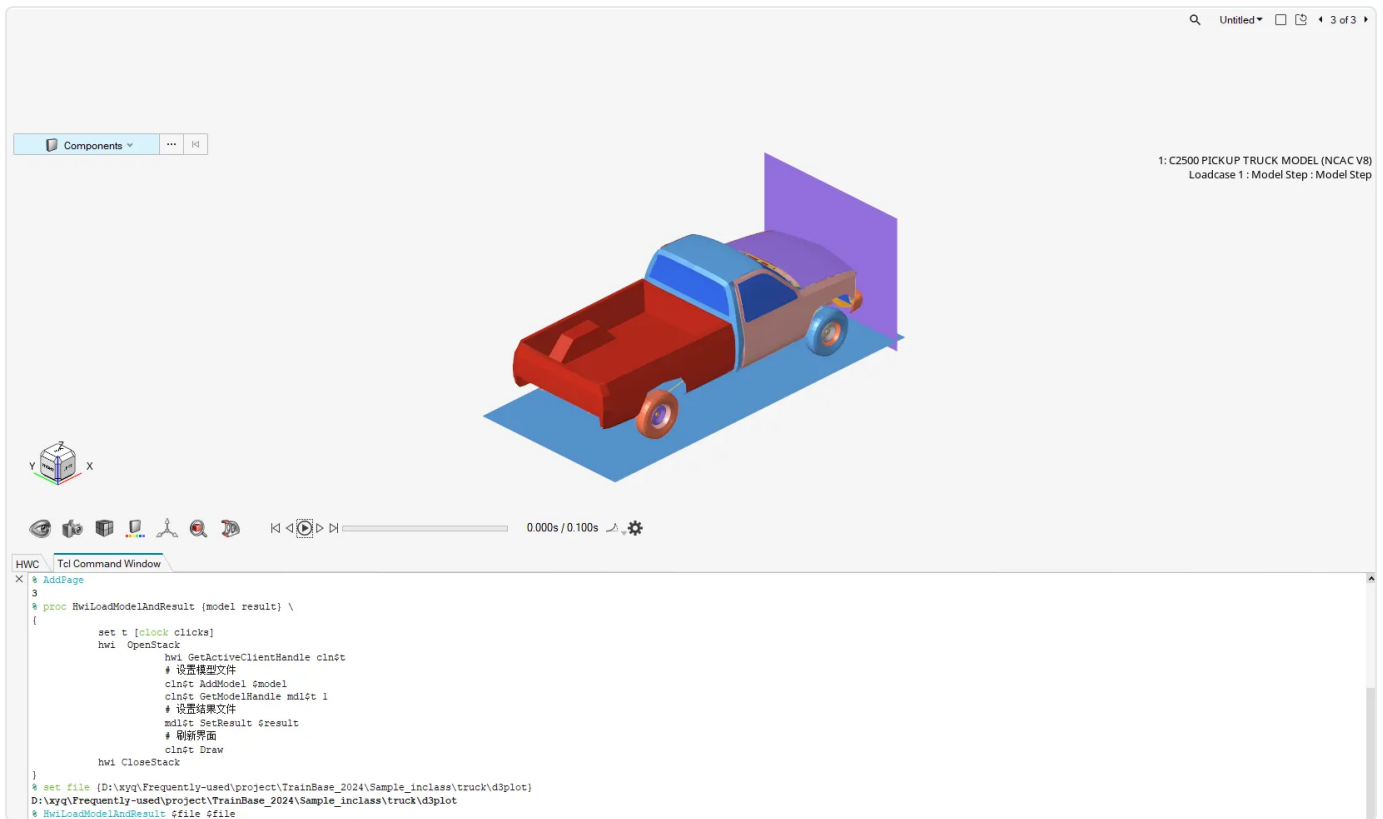
3. 测试并完成相应的功能后，将该功能封装成一个独立的过程

```
1 proc AddPage {args} \
2 {
3     set t [clock clicks]
4     hwi OpenStack
5     hwi GetSessionHandle sess$t
6     sess$t GetProjectHandle proj$t
7     set pageid [proj$t AddPage]
8     proj$t SetActivePage $pageid
9     hwi CloseStack
10    return $pageid
11 }
```



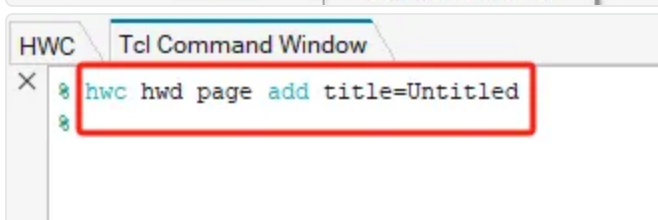
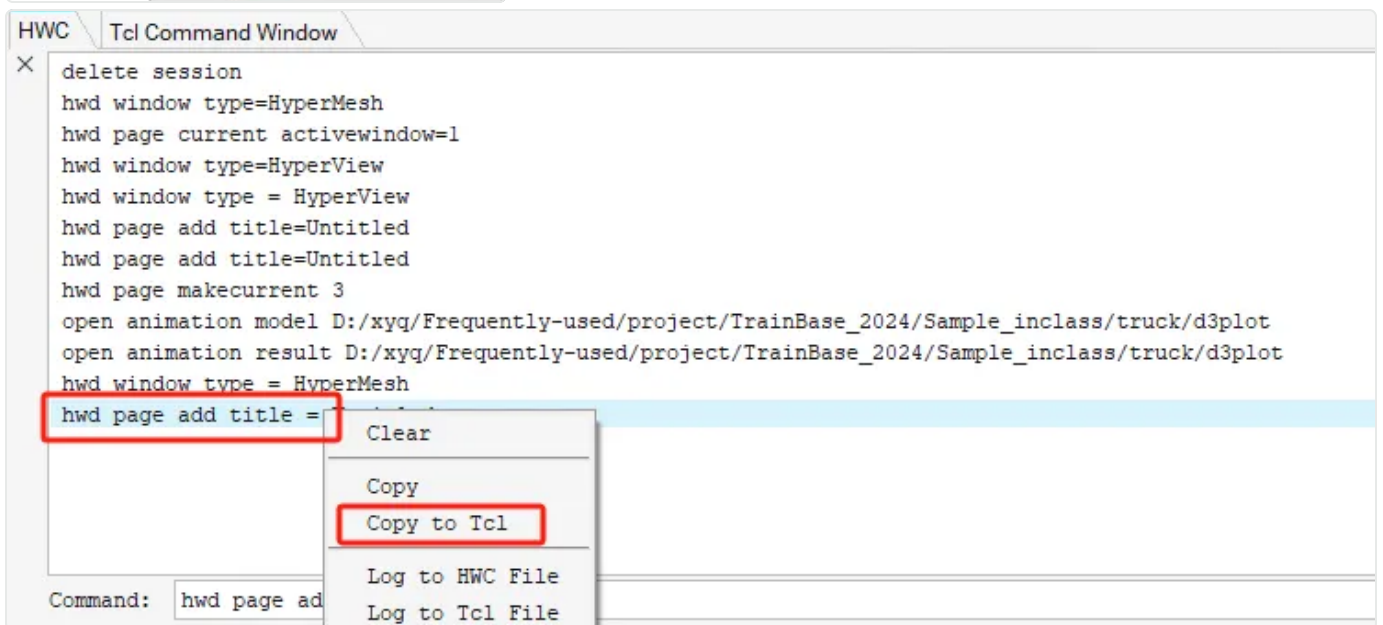
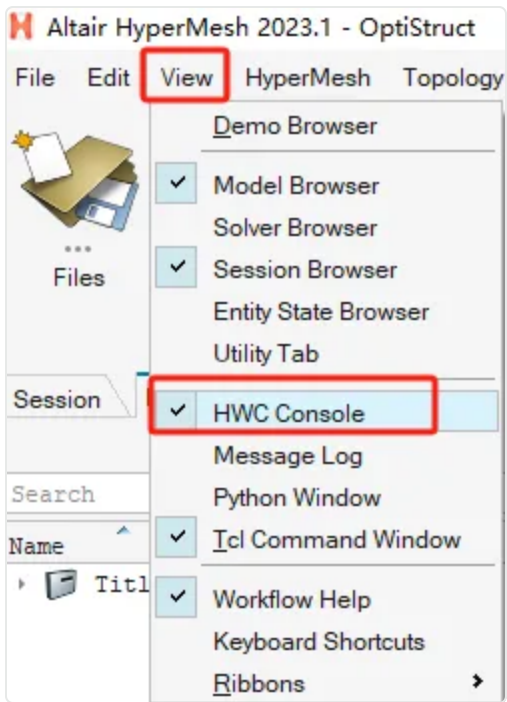
## 例2: 新增结果

```
Tcl |
1 proc HwiLoadModelAndResult {model result} \
2 {
3     set t [clock clicks]
4     hwi OpenStack
5     hwi GetActiveClientHandle cln$t
6     # 设置模型文件
7     cln$t AddModel $model
8     cln$t GetModelHandle mdl$t 1
9     # 设置结果文件
10    mdl$t SetResult $result
11    # 刷新界面
12    cln$t Draw
13    hwi CloseStack
14 }
```



## HWC介绍

1. 2019之后版本可以通过HWC命令实现图形界面的快速调整，版本越高支持的功能越多；
2. 支持绝大部分用户操作图形界面的动作，例：载入结果,加载云图，创建note或者measure；
3. 在HWC面板中将命令选中右键复制为tcl命令；
4. 在tcl console中运行HWC命令；
5. 主要支持的功能：载入结果,加载云图，创建注释，视角调整，截图，组件/单元显示或隐藏；



在新版本后处理二次开发的流程思路

**HWI缺陷：流程复杂，需要逐级建立handle，代码量大**

**HWC缺陷：HWC命令不会报错，也没有任何返回值**

HWI和HWC相结合的方式，HWC没有返回值，利用HWI查询结果文件数据，通过HWC处理图形内容。

1. **新建页面**，设置页面布局，调整窗口类型；
2. 加载结果文件；
3. Subcase,Simulation以及Animation设置；
4. 云图设置；
5. 添加note/measure；
6. **读取数据**，保存数据
7. 截图

## 在HV中读取数据

方法1：读取measure中的值

```

1  ▾ proc GetMeasureData {measure_id} \
2  ▾ {
3  ▾     set t [clock clicks]
4
5     hwi OpenStack
6
7     # 获取client handle
8     hwi GetActiveClientHandle cln$t
9
10    # 获取measure handle
11    cln$t GetMeasureHandle m$t $measure_id
12
13    # 刷新界面
14    cln$t Draw
15
16    # 提取id,值以及类型
17  ▾    set lst_entity_id [m$t GetEntityList]
18  ▾    set lst_value      [m$t GetValueList]
19  ▾    set lst_type       [m$t GetEntityTypeList]
20
21    # 自行处理数据
22  ▾    set data [list $lst_entity_id $lst_value $lst_type]
23
24    hwi CloseStack
25
26    return $data
27 }

```

## 方法2：读取set的内容

### 1. 创建set,hwi 中的selectionset 就是 hwc 中的sset

```
1 hwc sset create node set_new_node
```

### 2. 获取set的id

由于hwc创建的sset没有返回值，无法直接得到它的id



```

1 ▾ proc GetSetMaxId {args} \
2 ▾ {
3 ▾     set t [clock clicks]
4
5     hwi OpenStack
6
7     # 获取client handle
8     hwi GetActiveClientHandle cln$t
9
10 ▾     cln$t GetModelHandle model$t [cln$t GetActiveModel]
11
12     # 获取selection set的id列表
13 ▾     set lst_selection_set_id [model$t GetSelectionSetList]
14
15
16     hwi CloseStack
17
18 ▾     set lst_selection_set_id [lsort -real -increasing $lst_selection_set_i
19     d]
20 ▾     set set_max_id [lindex $lst_selection_set_id end]
21
22     return $set_max_id
23 }
24
25 ▾ proc HwcCreateSSet {set_name set_type} \
26 ▾ {
27     hwc sset create $set_type $set_name
28
29 ▾     set new_set_id [GetSetMaxId]
30
31     return $new_set_id
32 }

```

### 3. 查询set的内容

```

1  ▾ proc Query {selection_set_id {queryopts "component.name"}} \
2  ▾ {
3  ▾     set t [clock clicks]
4      hwi OpenStack
5          hwi GetActiveClientHandle cln$t
6  ▾     cln$t GetModelHandle model$t [cln$t GetActiveModel]
7          model$t GetQueryCtrlHandle query$t
8          query$t SetSelectionSet $selection_set_id
9          query$t SetQuery $queryopts
10         query$t GetIteratorHandle it$t
11 ▾     for {it$t First} {[it$t Valid]} {it$t Next} {
12 ▾         set data [it$t GetDataList];
13         puts d:$data
14     }
15     hwi CloseStack
16 }

```

## HG中添加曲线

### 1. 获取文件中的数据

```

1 ▾ proc HGGetFileData {file} \
2 ▾ {
3 ▾     set t [clock clicks]
4
5     hwi OpenStack
6         hwi GetSessionHandle sess$t
7         sess$t GetDataFileHandle dataf$t $file
8
9         # 获取第一个subcase
10 ▾     set subcase [dataf$t GetSubcase]
11
12         # 获取所有的subcase
13 ▾     # set lst_subcase [dataf$t GetSubcaseList]
14
15         # 设置subcase
16         # dataf$t SetSubcase
17
18 ▾     set lst_datatype [dataf$t GetDataTypeList]
19
20         # RequestList 和 ComponentList 的查询都是与datatype相关的
21 ▾     set datatype [lindex $lst_datatype 1]
22
23 ▾     set lst_request [dataf$t GetRequestList $datatype]
24 ▾     set lst_component [dataf$t GetComponentList $datatype]
25
26     hwi CloseStack
27 }

```

## 2. 添加曲线

```
1 ▾ proc ::postcommontool::HWIAddCurveByFile {resfile subcase x_datatype y_datatype y_request y_component} \
2 ▾ {
3 ▾     set t [clock clicks]
4     hwi OpenStack
5     hwi GetSessionHandle sess$t
6     sess$t GetProjectHandle prj$t
7 ▾     prj$t GetPageHandle pg$t [prj$t GetActivePage]
8 ▾     pg$t GetWindowHandle win$t [pg$t GetActiveWindow]
9     win$t GetClientHandle cln$t
10
11     # 新增曲线
12 ▾     set curve_id [cln$t AddCurve]
13
14     # 获取新增曲线的handle
15     cln$t GetCurveHandle mycur$t $curve_id
16
17     # 获取曲线向量handle: x,y,u,v
18     mycur$t GetVectorHandle mycurx$t x
19     mycur$t GetVectorHandle mycury$t y
20
21     # 设置曲线数据类型: file,math,values
22     mycurx$t SetType file
23     mycury$t SetType file
24
25     mycurx$t SetFilename $resfile
26     mycury$t SetFilename $resfile
27
28     mycurx$t SetSubcase $subcase
29     mycurx$t SetDataType $x_datatype
30     # mycurx$t SetRequest Time
31     # mycurx$t SetComponent Time
32
33     mycury$t SetSubcase $subcase
34     mycury$t SetDataType $y_datatype
35     mycury$t SetRequest $y_request
36     mycury$t SetComponent $y_component
37
38     # 计算曲线并刷新界面
39     cln$t Recalculate
40     cln$t Autoscale
41     cln$t Draw
42     hwi CloseStack
43
44     return $curve_id
```

### 3. 读取曲线数据

```
1  ▾ proc GetCurveData {curve_id} \  
2  ▾ {  
3  ▾     set t [clock clicks]  
4      hwi OpenStack  
5          hwi GetSessionHandle sess$t  
6          sess$t GetProjectHandle prj$t  
7  ▾     prj$t GetPageHandle pg$t [prj$t GetActivePage]  
8  ▾     pg$t GetWindowHandle win$t [pg$t GetActiveWindow]  
9      win$t GetClientHandle cln$t  
10  
11     cln$t GetCurveHandle curve$t $curve_id  
12     curve$t GetVectorHandle curve_x$t x  
13     curve$t GetVectorHandle curve_y$t y  
14  
15     # Get Value List  
16  ▾     set lst_x_value [curve_x$t GetValuesList]  
17  ▾     set lst_y_value [curve_y$t GetValuesList]  
18  
19     hwi CloseStack  
20  ▾     return [list $lst_x_value $lst_y_value]  
21  }
```

Tcl |