Easy Excel

# 客户端接口设计

客户输入元素

读取excel

DataSource

ReadDataSource

怎么读 配置源

Configuration

Table

Limit

Filter

Column1 y1 key1

Columns2 y2 key2

Columns…n y…n key…n

Point1 x1 y1 key1

Point1 x2 y2 key2

Point…n x…n y…n key…n

读取之后怎么组装结果

客户怎么方便获取结果

写入excel

DataSource

WriteDataSource

硬编码接口

SimpleReader

.build(InputStream)

.select(query)

.select(tableQuery, pointQuery)

.from(sheet)

.limit(startRow, size)

.addFilter(first, second, more)

.execute()

SimpleWriter

.build(OutputStream output)

.build(OutputStream output, boolean useXlsx)

.build(OutputStream output, InputStream template)

.insert(insert)

.insert(tableInsert, pointInsert)

.into(sheetName/sheetIndex)

.addData(List tableData)

.addData(Map pointData)

.addData(List tableData, Map pointData)

.addFilter(first, second, more)

.addFilter(String pointName, first, second, more)

.execute()

StandardReader

.build(InputStream)

.select(first, second, more)

.addFilter(String sheet, Integer tableIndex, Collection<Filter> filters)

.addFilter(String sheet, String pointName, Collection<Filter> filters)

StandardWriter

.build(OutputStream output)

.build(OutputStream output, boolean useXlsx)

.build(OutputStream output, InputStream template)

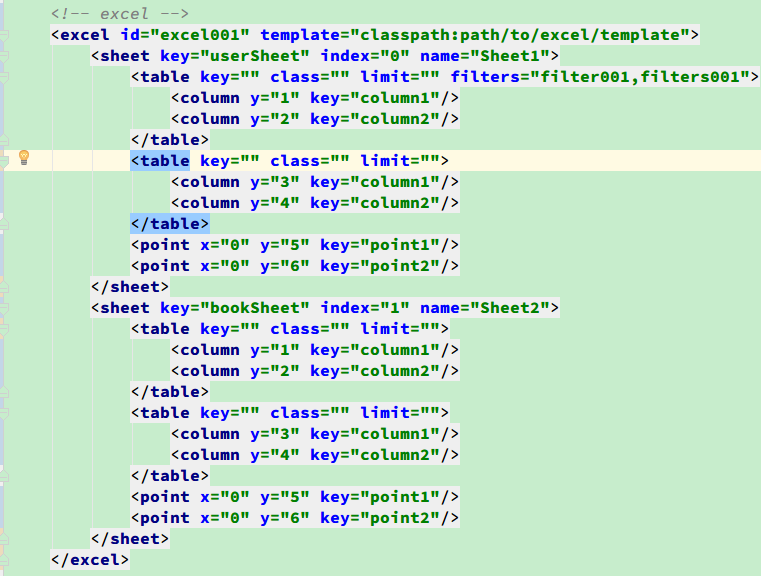
.insert(first, second, more)

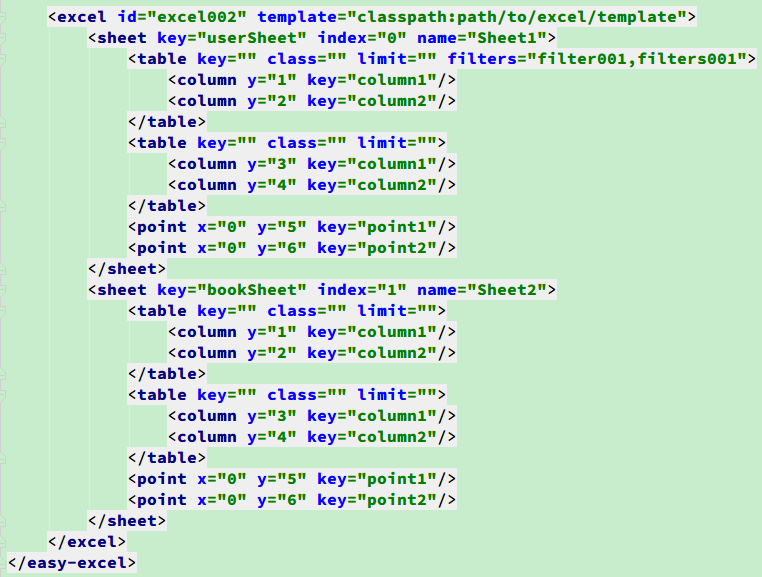
.addData(String sheet, Integer tableIndex, List tableData)

.addData(String sheet, Map pointData)

## 配置文件







<easy-excel>

<!-- share filter -->

<filter id="filter001" class="com.xxx.filter.XxxFilter"/>

<filter id="filter002" class="com.xxx.filter.XxxFilter"/>

<filter id="filter003" class="com.xxx.filter.XxxFilter"/>

<filter id="filter004" class="com.xxx.filter.XxxFilter"/>

<filters id="filters001">

<filter ref="filter001"/>

<filter class="com.xxx.filter.XxxFilter"/>

</filters>

<filters id="filters002">

<filter ref="filter002"/>

<filter ref="filter003"/>

<filter class="com.xxx.filter.XxxFilter"/>

</filters>

<!-- excel -->

<excel id="excel001" template="classpath:path/to/excel/template">

<sheet key="userSheet" index="0" name="Sheet1">

<table key="" class="" limit="" filters="filter001,filters001">

<column y="1" key="column1"/>

<column y="2" key="column2"/>

</table>

<table key="" class="" limit="">

<column y="3" key="column1"/>

<column y="4" key="column2"/>

</table>

<point x="0" y="5" key="point1"/>

<point x="0" y="6" key="point2"/>

</sheet>

<sheet key="bookSheet" index="1" name="Sheet2">

<table key="" class="" limit="">

<column y="1" key="column1"/>

<column y="2" key="column2"/>

</table>

<table key="" class="" limit="">

<column y="3" key="column1"/>

<column y="4" key="column2"/>

</table>

<point x="0" y="5" key="point1"/>

<point x="0" y="6" key="point2"/>

</sheet>

</excel>

<excel id="excel002" template="classpath:path/to/excel/template">

<sheet key="userSheet" index="0" name="Sheet1">

<table key="" class="" limit="" filters="filter001,filters001">

<column y="1" key="column1"/>

<column y="2" key="column2"/>

</table>

<table key="" class="" limit="">

<column y="3" key="column1"/>

<column y="4" key="column2"/>

</table>

<point x="0" y="5" key="point1"/>

<point x="0" y="6" key="point2"/>

</sheet>

<sheet key="bookSheet" index="1" name="Sheet2">

<table key="" class="" limit="">

<column y="1" key="column1"/>

<column y="2" key="column2"/>

</table>

<table key="" class="" limit="">

<column y="3" key="column1"/>

<column y="4" key="column2"/>

</table>

<point x="0" y="5" key="point1"/>

<point x="0" y="6" key="point2"/>

</sheet>

</excel>

</easy-excel>

## Annotation注解

PojoReader.build(InputStream input)

@Table

sheetName

sheetIndex

limit

@Column

y

@Hash

sheetName

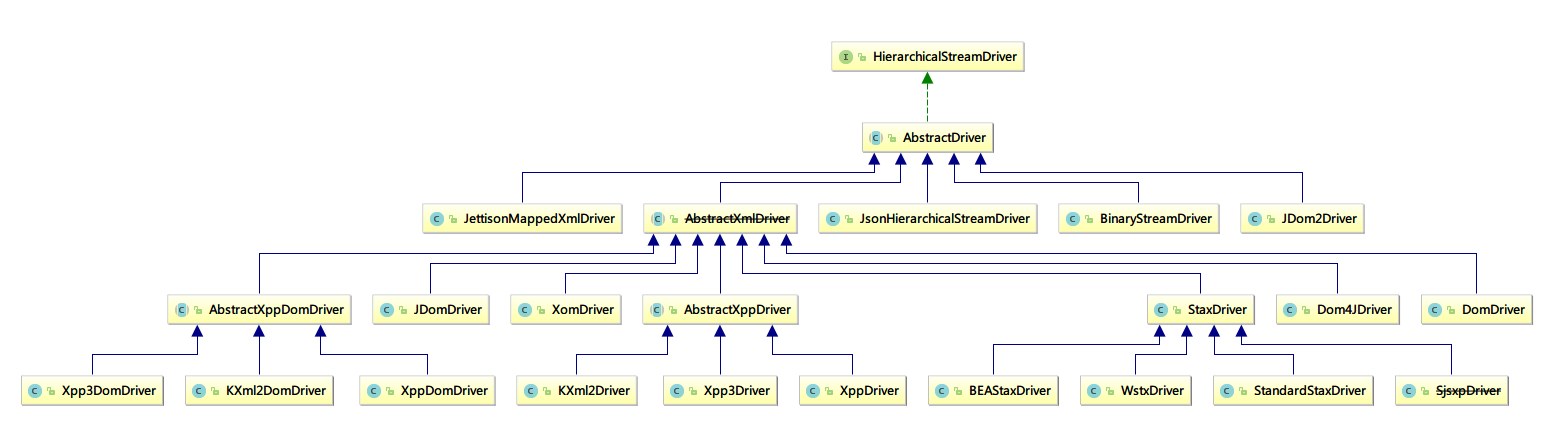
sheetIndex

@Point

x

y

# 服务端功能设计



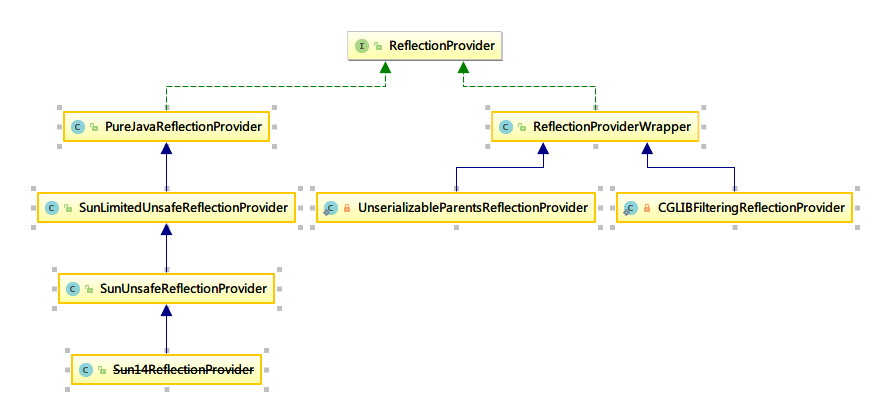
HierarchicalStreamDriver

提供从一个流当中读取或者写入对于XStream

也就是赋予XStream对象能够从Stream中读取或者写入数据

createReader(Reader in)

createWriter(Writer out)



反射提供者

接口：

根据类型创建实例

newInstance(Class type)

观察字段序列化

visitSerializableFields(Object object, Visitor visitor)

给对象的字段赋值

writeField(Object, object, String fieldname, Object value, Class definedIn)

获取字段类型

getFieldType(Object object , String fieldname, Class definedIn)

判断字段在Class中是否定义

fieldDefinedInClass(String fieldname, Class type)

观察者定义

interface Visitor{

// name:字段名称

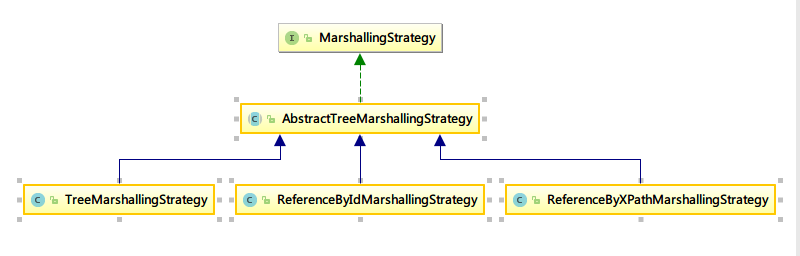
// type: 字段类型

// definedIn: 字段所属类

// value: 字段值

void visit(String name, Class type, Class definedIn, Object value)

}



MarshallingStrategy编排策略

解析编排

unmarshal(

Object root, // 跟对象

HierarchicalStreamReader reader, // 可层级化读

DataHolder dataHolder, // 数据持有者

ConverterLookup lookup, // 转换观察者

Mapper mapper // 映射器

)

编排

marshal(

HierarchicalStreamWriter writer,

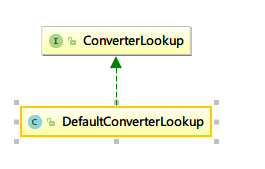
Object obj,

ConverterLookup, lookup,

Mapper mapper,

DataHolder dataHolder

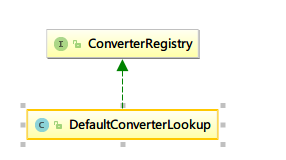
)



转换器查找

查找指定类型的转换器

Converter lookupConverterForType(Class type)

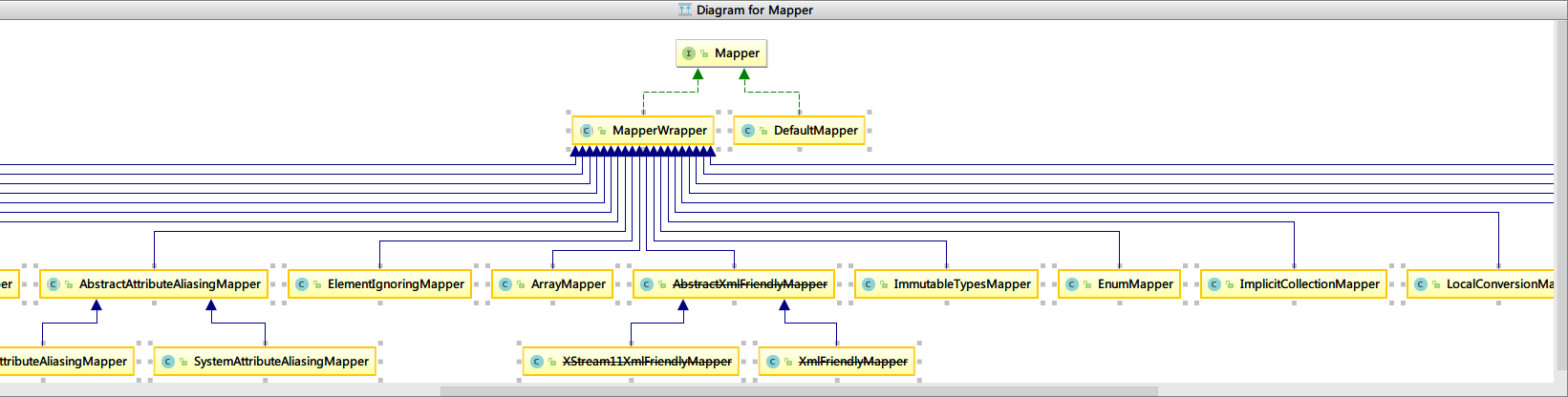


转换器注册

ConverterRegistry

// 注册一个转换器

RegisterConverter(Converter converter, int priority)



Mapper 映射器接口

// class -> element name

String serializedClass(Class type)

// element name -> class

realClass(String elementName)

// 成员变量 -> element name

serializedMember(Class type, String memberName)

// element name -> 成员变量

String realMember(Class type, String serialized)

是否是可变值类型

Boolean isMmutableValueType(Class type)

是否能够反射

Boolean isReferenceable(Class type)

// 属性 映射 别名

String aliasForAttribute(String attribute)

// 别名 映射 属性

String attributeForAlias(String alias)