Parameter Section Level

This is the third level and is described by the abstract class **ParameterSection**. Implementing classes must override a few methods and are described in the javadoc. The important members in the API are listed below:

Parameter Section Members	Description
public String getSectionKey()	This returns a unique identifier for the Section within a Panel.
public String getName()	The name of the Section. This is rendered as the heading of a section.
public String getParameterSectionClassName()	This returns a CSS class to be applied to the Section. This should be overridden as the default implementation returns null.
public List<Parameter> getParameters()	Implement this method to get a list of ParameterSection objects within this Panel. This should never return a null value.
public String getDynamicKey() public void setDynamicKey (String dynamicKey)	These methods are used for working with Dynamic Parameters, which is discussed in the section on Helper Objects.
public GeneralPanelOptions getSectionOptions()	General Options are discussed in the section on Helper Objects. These options control how the UI is rendered.
public List<ParameterDisplayRule> getDisplayRules()	Display Rules are discussed in the section on Helper Objects. They are used to determine whether a panel should be displayed or not based on user input.
public Map<String, ?> getData()	This method is abstract at this level. Implementations should return the data object for the level, containing data relevant to the Parameters. This is used by ParameterPanel's toJSON() method to construct its data object.
protected ParameterValueLoader parameterValueLoader	This protected member variable contains the Value Loader associated with this section. It is used to load option values of a Parameter, when the user selects a related parameter. For example, reloading a country dropdown, based on the user's selection of region. Value Loaders are discussed in detail in the section on Helper Objects.
public JSONObject toJSON()	This method converts everything in the Section to a org.json.JSONObject (Jackson). The main attributes of the object are name, sectionKey, parameterSectionClassName, parameters, displayRules, dynamicKey and generalOptions. • parameters holds a JSONArray of Parameter instances returned by getParameters(). Each item is converted to a JSONObject before it is put into the JSONArray. • displayRules holds a list of ParameterDisplayRule, the details of which are discussed in the section on Helper Objects. • generalOptions holds an instance of GeneralPanelOptions, discussed in the section on Helper Objects. If a value loader is set, the JSON object will also contain the classname of the loader object in the valueLoaderClassName attribute.

Implementation

Yellowfin comes included with **ETLStepConfigSection**, an implementation of **ParameterSection** which can be used with Data Transformation steps. Related methods of use include:

Methods	Description
public Map<String, ?> getData()	Returns member variable data which contains data from the parameters in this section.
public void setData(Map<String, ?> data)	Replaces the class member data with the object passed into this method.
public void setMappedParameterDataObject (String key, Object value)	Adds the specified key and value to the data object.

public void setMappedParameterData(String key, String value)	Adds the key and value to the data object. When the value is a JSON String, it is converted to an equivalent Java object. For example: - "{key1: value1, key2: value2}" is converted to Map<String, Object> - "[1, 2, 3]" is converted to List<Object> - "true" is converted to a Boolean. This is done using the method convertParameterToJSON() in the ParameterSection class.

When this implementation is used, it applies some standard Yellowfin styling to the section, as shown in the example below:

```
ETLStepConfigSection section = new ETLStepConfigSection("SECTION_KEY", "SectionName");
section.addParameter(rowCountParam);                                 // Parameter containing option key
ROW_COUNT_KEY
section.setMappedParameterDataObject("ROW_COUNT_KEY", rowCountInteger);
```