Group ordering in scripted charts

The chart results on the axes are always sorted by name. To change this behavior in scripted charts, it's possible to add a custom Comparator to the result.

All provided group collectors use the standard Jira sorting method. E.g. Custom Field options are sorted by standard ordering. Statuses, priorities, and resolutions appear in the order they were sorted in the Jira Issues configuration.

X-axis ordering

The x-axis can be sorted by calling the method **metaCountGroup.orderX(Comparator comparator);** In most cases, it might be enough to use the Comparator provided by the **groupValueExtractor.getComparator().**

Here is one example of sorting groups on the x-axis. The values on the x-axis are the issues statuses:

```
...

def metaCountGroup = chartBuilder.newDataCollector();

def groupValueExtractorX = chartBuilder.getGrouper("status");

... data collection ...

metaCountGroup.orderX(groupValueExtractorX.getComparator())

....
```

Y-axis ordering

To sort the groups on the y axis requires this to be the last operation before data is transmitted to the chart builder or to create a new object. This is necessary because the method returns a new sorted object and does NOT sort the current object itself. The method to call is **metaCountGroup. getOrdered(Comparator comparator);**

Here is an example of how to sort the y-axis:

```
...

def metaCountGroup = chartBuilder.newDataCollector();

def groupValueExtractorY = chartBuilder.getGrouper("customfield_17006");
...
// This has to be done as last operation, because a new ordered metaCountGroup is returned
// It is not enough to call metaCountGroup.getOrdered(groupValueExtractorY.getComparator().reversed()); as
this will not change the current metaCountGroup
chartBuilder.getChartUtil().transformResult(metaCountGroup.getOrdered(groupValueExtractorY.getComparator()),
chartData, true);
return chartData;
```

Reversed ordering

To reverse the ordering of an axis, call **comparator.reversed()**

In code this should look like the following example:

```
...
metaCountGroup.orderX(groupValueExtractorX.getComparator().reversed())
....
```

Custom coparator

It is possible to write a complete new Comparator e.g. to sort Strings by a leading number.

Therefore it is necessary to write the comparator, then call the ordering functions and use a new instance of the custom comparator as a parameter.

The following example shows the implementation of the comparator and how to use it:

```
...
import java.util.Comparator;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
...

// define the custom comparator
class CustomComparator implements Comparator<String>
{
  Pattern p = Pattern.compile("-?[\\d\\.]+");

  public int compare(String a, String b)
  {

    if ( a == null && b == null )
    return 0;
    if ( a == null )
    return 1;
    if ( b == null )
    return -1;
    Matcher ma = p.matcher(a);
    if ( ma.find() )
    {
      Matcher mb = p.matcher(b);
      if ( mb.find() )
      {
        String sa = ma.group();
        String sb = mb.group();

        try
        {
          return Double.compare(Double.parseDouble(sa), Double.parseDouble(sb));
        } catch (NumberFormatException e)
        {
          logger.error(e);
        }

      }
    }
    return a.compareTo(b);
  }
};

def metaCountGroup = chartBuilder.newDataCollector();

... data collection ...

metaCountGroup.orderX(new CustomComparator());

....
```

Complex comparator

In some cases, it can be useful to order the groups by related objects instead of string comparison. The standard GroupValueExtractor Comparator compares the related objects.

To use a custom complex comparator the following steps are required:

1. Create a custom **GroupComparator<T> implements Comparator<String>**
2. Add all relevant objects to the Comparator while receiving the results with **comparator.addValue(String groupName, T t)**
3. At the end of data collection, a call to the relevant order method depending on the axis either **metaCountGroup.orderX(Comparator comparator)** or **metaCountGroup.getOrdered(Comparator comparator)**;

```
package com.decadis.jira.xchart.sorting;

import java.util.Comparator;
import java.util.HashMap;

import org.apache.log4j.Logger;

import com.atlassian.jira.issue.customfields.option.Option;
import com.atlassian.jira.issue.customfields.statistics.OptionComparator;

public class GroupComparator<T> implements com.decadis.jira.xchart.api.GroupComparator<T>
{

    private static final Logger log = Logger.getLogger(GroupComparator.class);
    Comparator<T> comp;
    HashMap<String, T> map = new HashMap<String, T>();

    public GroupComparator(Comparator<T> comp)
    {
        this.comp = comp;
    }

    @Override
    public void addValue(String value, T object)
    {
        map.put(value, object);
    }

    /*
     * (non-Javadoc)
     * @see
     * com.decadis.jira.xchart.sorting.GroupComparatorApi#compare(java.lang.String, java.lang.String)
     */
    @Override
    public int compare(String a, String b)
    {
        if ( a == null && b == null )
            return 0;
        if ( a == null )
            return -1;
        if ( b == null )
            return 1;
        if ( a.equals(b) )
            return 0;

        // here is the comparison of the related objects
        if ( map.containsKey(a) && map.containsKey(b) )
        {
            return comp.compare(map.get(a), map.get(b));
        }

        if ( map.containsKey(a) )
        {
            return 1;
        }
        if ( map.containsKey(b) )
        {
```

```
            return -1;
        }

        return a.compareTo(b);
    }

    // Simple String Comparator
    public static GroupComparator<String> getStringComparator()
    {
        return new GroupComparator<String>(null)
        {
            @Override
            public int compare(String a, String b)
            {
                if ( a == null && b == null )
                    return 0;
                if ( a == null )
                    return -1;
                if ( b == null )
                    return 1;
                if ( a.equals(b) )
                    return 0;
                return a.compareTo(b);
            }
        };
    }

    // Custom Field Option Comparator
    public static GroupComparator<Option> getOptionComparator()
    {
        return new GroupComparator<Option>(new OptionComparator());
    }

}
```

Related examples

Title
Using Jira Software specific classes and Pickers in Scripts
Story status category grouped by Epic and custom value
Simple Timeseries Chart
Simple Table Report
Simple Scripting Example
Report - Lucene Group By
Open issues with average
Issues in specific status (Period)
Group ordering in scripted charts
Gantt Diagram
Customers in a Google Map
Created vs. resolved with trend

- Comments count by user in JQL result
- Block Search
- 2Y Axes Chart
- Simple External Database Chart

If you still have questions, feel free to refer to our support team.