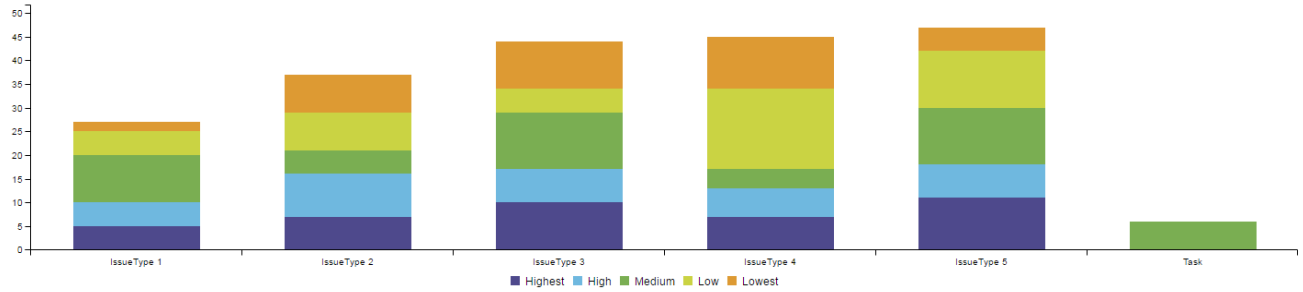
Group by chart

Group by chart shows any issues and group by any value. The groups on the Y-axis are always grouped by name.

E.g. Two project components with the same name are shown as one (project A: Component "COMP", project B: Component "COMP" there will only be one COMP group)

Chart preview



Parameters

Parameter		Type	Default value
UI	**Code**		
JQL	`JQL`	JQL Autocomplete	
Group By X	`GroupByX`	Group By Picker	Project
Group By Y	`GroupByY`	Group By Picker	Issue Type
Chart Type	`ChartType`	Chart Type Picker	Bar
Value	`Value`	Value Field Picker	Issue
Options • Accumulate Groups • Accumulate Values	`AccumulateGroups`	Boolean	True
	`AccumulateValues`	Boolean	False

Layout Data

Used layout: **Easy Group By**.

```
function formatTooltipAsHours(value, ratio, id, index)
{
    return value.toFixed(2) + ' h';
}

function formatTooltipAsHoursWithDays(value, ratio, id, index)
{
    var hours = parseInt(value);
    var days = parseInt(hours / 24);
    value = value - days * 24;
    if (days > 0)
    {
        return days + 'd, ' + value.toFixed(2) + ' h';
    }
    return value.toFixed(2) + ' h';
}

var c3arg = {
    onrendered: updateFrameHeight,
    data: chartData,
    axis: {
        x: {
            type: 'category', // this is needed to load string x value
            label: {
                text: chartData.custom.xLabel,
                position: 'outer-left'
            }
        },
        y: {
            label: chartData.ytype
        }
    }
};

if (chartData.custom && chartData.custom.tooltip)
{
    var tooltipFunction = eval(chartData.custom.tooltip);
    c3arg.tooltip = {
        format: {
            value: tooltipFunction
        }
    };
}

c3.generate(c3arg);
```

Script Data

```
import java.lang.reflect.Field;
import java.math.BigDecimal;

import org.apache.lucene.document.Document;

import com.atlassian.jira.component.ComponentAccessor;
import com.atlassian.jira.issue.DocumentIssueImpl;
import com.atlassian.jira.issue.Issue;
import com.atlassian.jira.jql.parser.JqlParseException;
import com.atlassian.jira.jql.parser.JqlQueryParser;
import com.atlassian.query.Query;

import com.decadis.jira.xchart.api.ChartParam;
import com.decadis.jira.xchart.api.model.Period
import com.decadis.jira.xchart.api.util.DateUtils;
```

```
//function that clears the group name in case it is null or empty
def clear = { String string -> (string == null || string.trim().length() == 0 ) ? "-" : string; }

def metaCountGroup = chartBuilder.newDataCollector();

JqlQueryParser jqlQueryParser = ComponentAccessor.getComponent(JqlQueryParser.class);
Query query = null;

try
{
  query = jqlQueryParser.parseQuery(JQL);
} catch (JqlParseException e)
{
  throw new IllegalArgumentException("Bad JQL: " + JQL);
}

def groupValueExtractorX = chartBuilder.getGrouper(GroupByX);
def groupValueExtractorY = chartBuilder.getGrouper(GroupByY);

def valueExtractor = chartBuilder.getValueExtractor(Value);
if ( valueExtractor == null )
{
  throw new IllegalArgumentException("Valuepicker for " + Value + " is not supported.");
}

Field documentField;
try
{
  documentField = DocumentIssueImpl.class.getDeclaredField("document");
  documentField.setAccessible(true);
  for ( Issue issue : chartBuilder.getFilterUtils().performSearch(query, user) )
  {
    def document = (Document) documentField.get(issue);
    BigDecimal value = valueExtractor.get(issue, document);

    for ( String groupX : groupValueExtractorX.getGroups(document) )
    {
        for ( String groupY : groupValueExtractorY.getGroups(document) )
        {
            groupX = groupValueExtractorX.getResolvedValue(groupX, issue);

            groupY = clear(groupY);
            groupX = clear(groupX);

            metaCountGroup.addValue(value, groupY, groupX);
        }
    }
  }
} catch (Exception e){}

if ( AccumulateValues ) {
    metaCountGroup.accumulateGroups();
}
else {
    metaCountGroup.fillMissingValues();
}

def chartData = chartBuilder.newChartData(groupValueExtractorY.getGroupName());

for ( String grpKey : metaCountGroup.keySet() )
{
  chartData.addGroupName(grpKey, groupValueExtractorY.getResolvedValue(grpKey, null));
}

chartData.setXType(groupValueExtractorX.getGroupName());
chartData.setYType(groupValueExtractorY.getGroupName());
chartData.setType(ChartType);

chartBuilder.getChartUtil().transformResult(metaCountGroup, chartData, AccumulateGroups);

return chartData;
```

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