Story status category grouped by Epic and custom value

This chart displays the status categories from all Stories under an Epic grouped by a custom value.

The status categories are also grouped on the y axis by Epic and will be displayed as individual donut charts within the table.

The number of tickets within the status category **Done** will be displayed as percentage values in the center of each donut.

In the following example, the Stories are grouped by **Priority**.

approve Download Scripted Chart Bundle (Jira < 8)

approve Download Scripted Chart Bundle (Jira >= 8)

Chart preview

Parameters

Name	Type	Default
JQL	JQL autocomplete	
Epic Link Field ID	Text	
GroupBy	Group By Picker	

Layout Script

JavaScript Template

```
const $container = $('#chart');


const $table = $('<table />').addClass('table table-bordered').css('width', 'auto').appendTo($container);
const $thead = $('<thead />').appendTo($table);
const $tbody = $('<tbody />').appendTo($table);

const data = chartData.data;
const statusCategories = chartData.statusCategories;


// create header row
const $tr = $('<tr />').css('background', '#f5f5f5').append('<td />').appendTo($thead);
// create header columns
$.each(data[Object.keys(data)[0]], function (groupKey) {
    $('<th />').addClass('nowrap text-center').text(chartData.groups[groupKey]).appendTo($tr);
});

// set color pattern
const colors = {
    'done': '#8cc152',
    'indeterminate': '#ffce54',
    'new': '#ed5556'
};

/**
 * Calculate sum
 *
 * @param obj
 * @returns {number}
 */
const sum = function (obj) {

    let keys = Object.keys(obj);

    if(keys.length){
        return keys.reduce(function (sum, key) {
            return sum + parseFloat(obj[key]);
        }, 0);
    }

    return 0;
}

/**
 * Create donut status chart
 *
 * @param status
 * @returns {void|*|jQuery}
 */
const getStatusChart = function (status) {
    let $element = $('<div />').addClass('text-center status-chart-wrap');
    let $chart = $('<div />').appendTo($element);

    let total = sum(status);

    // return empty element, if status has no data
    if (total === 0) {
        return $element;
    }

    // calculate percents
    let title = Math.floor((status.done / total) * 100) + '%';

    // timeout is necessary, becaouse we should create the chart after the dom element are created
    setTimeout(function () {
        c3.generate({
            bindto: $chart[0],
            data: {
                colors: colors,
                columns: [
```

```
                    ['done', status['done'] || 0],
                    ['indeterminate', status['indeterminate'] || 0],
                    ['new', status['new'] || 0]
                ],
                names: statusCategories,
                type: 'donut'
            },
            size: {
                height: 90,
                width: 90
            },
            legend: {
                show: false
            },
            donut: {
                title: title,
                label: {
                    show: false
                },
                width: 13
            }
        });
    }, 50);

    $('<small />').text('Total: '+total).appendTo($element);

    return $element;
}

$.each(data, function (epicKey, groups) {

    // create table row
    let $tr = $('<tr />').appendTo($tbody);

    // create first column label
    let $strong = $('<a />').attr({
        href : chartData.contextPath +'/browse/'+ epicKey
    }).text(chartData.epicData[epicKey] +" ["+epicKey + "]");

    // create first column
    $('<td />').append($strong).appendTo($tr);

    // create charts columns
    $.each(groups, function (groupName, status) {
        $('<td />').addClass('text-center').append(getStatusChart(status)).appendTo($tr);
    });
});
```

Data Script

Groovy Script for Jira < 8

```
import com.atlassian.jira.component.ComponentAccessor
import com.atlassian.jira.config.StatusCategoryManager
import com.atlassian.jira.issue.CustomFieldManager
import com.atlassian.jira.issue.DocumentIssueImpl
import com.atlassian.jira.issue.Issue
import com.atlassian.jira.issue.fields.CustomField
import com.atlassian.jira.issue.link.IssueLink
import com.atlassian.jira.issue.search.SearchResults
import com.atlassian.jira.issue.search.providers.LuceneSearchProvider
import com.atlassian.jira.jql.parser.JqlQueryParser
import com.atlassian.jira.rest.api.util.ErrorCollection
import com.atlassian.jira.web.bean.PagerFilter
```

```
import com.atlassian.query.Query
import com.decadis.jira.xchart.api.GroupValueExtractor
import com.decadis.jira.xchart.api.ValueExtractor
import com.decadis.jira.xchart.calculating.ValueExtractorBuilder
import com.decadis.jira.xchart.grouping.GroupValueExtractorFactory
import org.apache.commons.lang.StringUtils
import org.apache.lucene.document.Document


import java.lang.reflect.Field

/**
 * Get issues on JQL
 *
 * @param jqlString
 *
 * @return
 */
static List<Issue> getIssues(final String jqlString) {
    final LuceneSearchProvider luceneSearchProvider = ComponentAccessor.getComponent(LuceneSearchProvider.
class)
    final JqlQueryParser jqlQueryParser = ComponentAccessor.getComponent(JqlQueryParser.class)

    Query query = jqlQueryParser.parseQuery(jqlString)
    SearchResults searchResults = luceneSearchProvider.searchOverrideSecurity(query, null, PagerFilter.
getUnlimitedFilter(), null)

    return searchResults.getIssues()
}

/**
 *
 * @param issue
 * @param groupValueExtractor
 *
 * @return
 */
static Map<String, String> getGroups(final Issue issue, final GroupValueExtractor<?> groupValueExtractor) {
    final Map<String, String> result = new TreeMap<>()

    try{
        Field documentField = DocumentIssueImpl.class.getDeclaredField("document")
        documentField.setAccessible(true)
        Document document = (Document) documentField.get(issue)

        for (String value : groupValueExtractor.getGroups(document)) {
            if (StringUtils.isNotBlank(value)) {
                result.put(value, groupValueExtractor.getResolvedValue(value, issue))
            }
        }
    }
    catch (Exception e){

    }

    return result
}

/**
 * fill all empty values with empty map
 *
 * @param data
 * @param groups
 *
 * @return
 */
static void fillMissingValues(final Map<String, Map<String, Map<String, Integer>>> data, final Map<String,
String> groups)
{
    for (Map.Entry<String, Map<String, Map<String, Integer>>> epics : data.entrySet()){
        Map<String, Map<String, Integer>> group = epics.getValue()
```

```
        for(String key : groups.keySet())
        {
            if(!group.containsKey(key)){
                group.put(key, new TreeMap<>())
            }
        }
    }
}


final StatusCategoryManager statusCategoryManager = ComponentAccessor.getComponent(StatusCategoryManager.
class)

// create group value extractor
final GroupValueExtractor<?> groupValueExtractor = GroupValueExtractorFactory.Create(GroupBy, "");

Map<String, Object> result = new HashMap<>()
Map<String, String> groups = new TreeMap<>()
Map<String, String> epicData = new TreeMap<>()
Map<String, Integer> statusMap = new TreeMap<>()
Map<String, String> statusCategoriesMeta = new HashMap<>()
Map<String, Map<String, Map<String, Integer>>> data = new HashMap<>()

// get all epic issues
List<Issue> epics = getIssues(JQL);

// mapping status category values
statusCategoryManager.getStatusCategories().each {
    if (!"undefined".equalsIgnoreCase(it.getKey())){
        statusMap.put(it.getKey(), 0)
        statusCategoriesMeta.put(it.getKey(), it.getTranslatedName())
    }
}

for (Issue epic : epics)
{
    Map<String, Map<String, Integer>> groupData = new TreeMap<>()

    for (Issue story : getIssues("cf["+ Epic_Link_FieldID +"] = " + epic.getKey()))
    {
        Map<String, String> groupsTemp = getGroups(story, groupValueExtractor)

        groups.putAll(groupsTemp)

        for(String groupName : groupsTemp.keySet())
        {
            Map<String, Integer> statusMapTemp = groupData.get(groupName)

            if(statusMapTemp == null)
            {
                statusMapTemp = new TreeMap<>(statusMap)
            }

            String statusCategoryKey = story.getStatus().getStatusCategory().getKey()

            if(statusCategoryKey != null)
            {
                statusMapTemp.put(statusCategoryKey, ++statusMapTemp.get(statusCategoryKey))
            }

            groupData.put(groupName, statusMapTemp)
        }
    }

    data.put(epic.getKey(), groupData)
    epicData.put(epic.getKey(), epic.getSummary())
}

fillMissingValues(data, groups)
```

```
result.put("groups", groups)
result.put("data", data)
result.put("statusCategories", statusCategoriesMeta)
result.put("epicData", epicData)
result.put("contextPath", ComponentAccessor.getApplicationProperties().getString("jira.baseurl"))

return result
```

Groovy script for Jira >= 8

```
import com.atlassian.jira.component.ComponentAccessor
import com.atlassian.jira.config.StatusCategoryManager
import com.atlassian.jira.issue.CustomFieldManager
import com.atlassian.jira.issue.DocumentIssueImpl
import com.atlassian.jira.issue.Issue
import com.atlassian.jira.issue.fields.CustomField
import com.atlassian.jira.issue.link.IssueLink
import com.atlassian.jira.issue.search.SearchResults
import com.atlassian.jira.jql.parser.JqlQueryParser
import com.atlassian.jira.rest.api.util.ErrorCollection
import com.atlassian.jira.web.bean.PagerFilter
import com.atlassian.query.Query
import com.decadis.jira.xchart.api.GroupValueExtractor
import com.decadis.jira.xchart.api.ValueExtractor
import com.decadis.jira.xchart.calculating.ValueExtractorBuilder
import com.decadis.jira.xchart.grouping.GroupValueExtractorFactory
import org.apache.commons.lang.StringUtils
import org.apache.lucene.document.Document
import com.atlassian.jira.bc.issue.search.SearchService;

import java.lang.reflect.Field

/**
 * Get issues on JQL
 *
 * @param jqlString
 *
 * @return
 */
 List<Issue> getIssues(final String jqlString) {
        SearchService serchService = ComponentAccessor.getComponentOfType(SearchService.class);
       SearchService.ParseResult parseResult =
    serchService.parseQuery(ComponentAccessor.getJiraAuthenticationContext().getLoggedInUser(),
    jqlString);

        if (!parseResult.isValid())
        {
            throw new Exception("jql is not valid. jql was "+jqlString);
        }

        Query query = parseResult.getQuery();
          SearchResults sr =
    serchService.search(ComponentAccessor.getJiraAuthenticationContext().getLoggedInUser(),
    query, PagerFilter.getUnlimitedFilter());
            return sr.getResults();

}

/**
 *
 * @param issue
 * @param groupValueExtractor
 *
 * @return
 */
static Map<String, String> getGroups(final Issue issue, final GroupValueExtractor<?> groupValueExtractor) {
    final Map<String, String> result = new TreeMap<>()

    try{
```

```
        Field documentField = DocumentIssueImpl.class.getDeclaredField("document")
        documentField.setAccessible(true)
        Document document = (Document) documentField.get(issue)

        for (String value : groupValueExtractor.getGroups(document)) {
            if (StringUtils.isNotBlank(value)) {
                result.put(value, groupValueExtractor.getResolvedValue(value, issue))
            }
        }
    }
    catch (Exception e){

    }

    return result
}

/**
 * fill all empty values with empty map
 *
 * @param data
 * @param groups
 *
 * @return
 */
static void fillMissingValues(final Map<String, Map<String, Map<String, Integer>>> data, final Map<String,
String> groups)
{
    for (Map.Entry<String, Map<String, Map<String, Integer>>> epics : data.entrySet()){
        Map<String, Map<String, Integer>> group = epics.getValue()

        for(String key : groups.keySet())
        {
            if(!group.containsKey(key)){
                group.put(key, new TreeMap<>())
            }
        }
    }
}


final StatusCategoryManager statusCategoryManager = ComponentAccessor.getComponent(StatusCategoryManager.
class)

// create group value extractor
final GroupValueExtractor<?> groupValueExtractor = GroupValueExtractorFactory.Create(GroupBy, "");

Map<String, Object> result = new HashMap<>()
Map<String, String> groups = new TreeMap<>()
Map<String, String> epicData = new TreeMap<>()
Map<String, Integer> statusMap = new TreeMap<>()
Map<String, String> statusCategoriesMeta = new HashMap<>()
Map<String, Map<String, Map<String, Integer>>> data = new HashMap<>()

// get all epic issues
List<Issue> epics = getIssues(JQL);

// mapping status category values
statusCategoryManager.getStatusCategories().each {
    if (!"undefined".equalsIgnoreCase(it.getKey())){
        statusMap.put(it.getKey(), 0)
        statusCategoriesMeta.put(it.getKey(), it.getTranslatedName())
    }
}

for (Issue epic : epics)
{
    Map<String, Map<String, Integer>> groupData = new TreeMap<>()

    for (Issue story : getIssues("cf["+ Epic_Link_FieldID +"] = " + epic.getKey()))
    {
```

```
        Map<String, String> groupsTemp = getGroups(story, groupValueExtractor)

        groups.putAll(groupsTemp)

        for(String groupName : groupsTemp.keySet())
        {
            Map<String, Integer> statusMapTemp = groupData.get(groupName)

            if(statusMapTemp == null)
            {
                statusMapTemp = new TreeMap<>(statusMap)
            }

            String statusCategoryKey = story.getStatus().getStatusCategory().getKey()

            if(statusCategoryKey != null)
            {
                statusMapTemp.put(statusCategoryKey, ++statusMapTemp.get(statusCategoryKey))
            }

            groupData.put(groupName, statusMapTemp)
        }
    }

    data.put(epic.getKey(), groupData)
    epicData.put(epic.getKey(), epic.getSummary())
}

fillMissingValues(data, groups)

result.put("groups", groups)
result.put("data", data)
result.put("statusCategories", statusCategoriesMeta)
result.put("epicData", epicData)
result.put("contextPath", ComponentAccessor.getApplicationProperties().getString("jira.baseurl"))

return result
```

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.