Gantt-Diagramm

AID 087a DE

Autor: FA, MW, KN. Version 10.1. Zuletzt geändert 19.09.2017

Version	Änderungen
10.1	Anpassung der Formatierungen
10.0	Anpassung an ADITO4.6
1.0	Letzter Stand vor Übernahme in Versionierung

Inhaltsverzeichnis

1. Gantt-Diagramm 4

2. Ansichtsoptionen 5

2.1. Optionen in der Toolbar bei Klick auf eine Aktivität 5

2.2. Kontextmenü 5

2.3. Symbolleiste Gantt-Diagramm 5

3. Technik 7

3.1. Aufbau 7

3.2. Zur Verfügung stehende Prozesse im changeProcess 7

3.3. Beispiel 7

3.4. Aufbau des Gantt-Diagramms mit AditoGanttAdditional 10

3.5. Aktualisieren eines Gantt-Diagramms mit swing.updateGantt 13

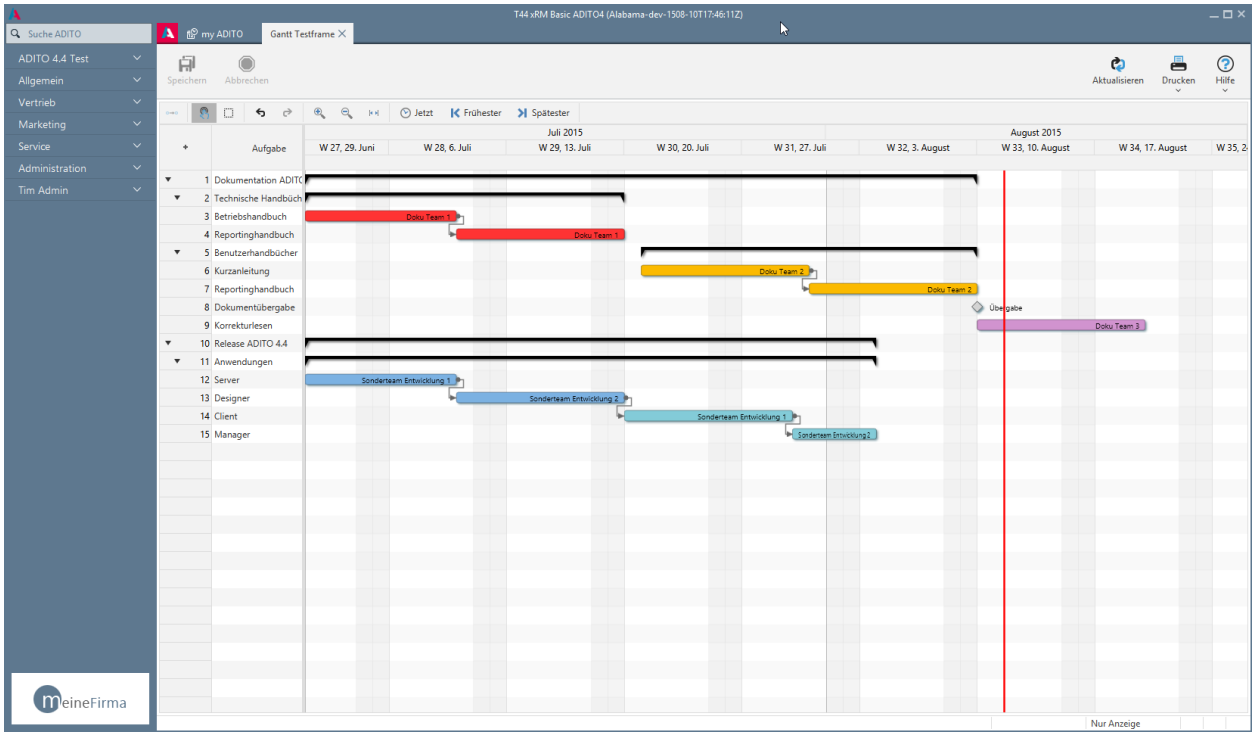
3.5.1. Hinzufügen einer Aktivität 13

3.5.2. Bearbeiten einer Aktivität 14

3.5.3. Löschen einer Aktivität 15

1. Gantt-Diagramm

Ab ADITO 4.4 ist die Planungskomponente "Gantt-Diagramm" enthalten.



Diese Komponente wird über die ADITO-Anpassungssprache JDito völlig frei definiert, lässt sich also für viele Verwendungszwecke einsetzen. Selbstverständlich können die Daten auf Benutzerebene verändert werden.

Dieses Dokument ist nur für die Swing-Anwendung gültig.

2. Ansichtsoptionen

Mit der STRG-Taste plus Mausrad kann der angezeigte Bereich vergrößert oder verkleinert werden. Durch Klick auf den freien Bereich und ziehen mit der Maus kann man sich durch das Gantt-Diagramm navigieren.

Ein Klick auf eine Aktivität kann diese verschieben oder größer- bzw. kleiner ziehen.

Durch Klick der rechten Maustaste kann ein Kontextmenü aufgerufen werden. Dieses Kontextmenü kann über den ADITO-Designer definiert werden. (Siehe "Kontextmenü" weiter unten)

2.1. Optionen in der Toolbar bei Klick auf eine Aktivität

In Beziehung setzen mit: Ermöglicht es, eine Beziehung zu einer anderen Komponente zu setzen. Nach Klick auf diesen Menüpunkt beginnt die ausgewählte Aktivität zu blinken, man kann dann ein Ende einer anderen Aktivität anklicken, um eine Beziehung aufzubauen

2.2. Kontextmenü

Über die Eigenschaft `popup` kann ein Kontextmenü definiert werden. In diesem stehen die folgenden Variablen zur Verfügung:

Variable	Bemerkung
$local.path	Angabe des Pfads innerhalb des Daten-Arrays im Gantt-Diagramm, in dem sich die Werte geändert haben.
$local.data	Gesamtes Daten-Array des Gantt-Diagramms
$local.type	Typ der Änderung; ADDED, REMOVED oder CHANGED
$local.time	Zeitpunkt des Mauscursors

2.3. Symbolleiste Gantt-Diagramm

Von links nach rechts:

- In Beziehung setzen mit: Setzt Aktivitäten miteinander in Beziehung.
- Scrollen: Scrollt durch Mausklick.
- Rechteckselektion: Erlaubt, mit einer Rechteckselektion Aktivitäten und Meilensteine zu selektieren.
- Rückgängig: Nimmt den letzten Schritt zurück.
- Wiederherstellen: Stellt den zurückgenommenen Schritt wieder her.
- Zoom +: Vergrößert die Ansicht.
- Zoom -: Verkleinert die Ansicht.

- Alles anzeigen: Zoomt so, dass alle Aktivitäten angezeigt werden können.
- Jetzt: Setzt die Mitte der Komponente auf das jetzige Datum.
- Frühester: Wechselt zur frühesten Aktivität.
- Spätester: Wechselt zur spätesten Aktivität.
- (Bei ausgewählter Beziehung): Beziehung Ende-Anfang.
- (Bei ausgewählter Beziehung): Beziehung Ende-Ende.
- (Bei ausgewählter Beziehung): Beziehung Anfang-Anfang.
- (Bei ausgewählter Beziehung): Beziehung Anfang-Ende.

3. Technik

3.1. Aufbau

Das Gantt-Diagramm ist im Aufbau einer TreeTable ähnlich. Über ein strukturiertes Objekt wird das Gantt-Diagramm aufgebaut. Änderungen der Benutzer werden in dieses Objekt geschrieben und stehen im `changeProzess` zur Verfügung. Dort kann das Objekt verarbeitet und ggf. wieder gespeichert werden.

3.2. Zur Verfügung stehende Prozesse im changeProcess

Variable	Bemerkung
$local.path	Angabe des Pfads innerhalb des Daten-Arrays im Gantt-Diagramm, in dem sich die Werte geändert haben.
$local.data	Gesamtes Daten-Array des Gantt-Diagramms
$local.type	Typ der Änderung; ADDED, REMOVED oder CHANGED

3.3. Beispiel

Folgender Code befüllt ein Gantt-Diagramm über Rückgabe des Objektes, welches das Gantt-Diagramm erwartet.

```
import("system.util");
import("system.result");
import("system.datetime");
import("aditoGantt");

var stuff = "-" // vars.getString("$comp.STUFF"); // Kann auch Wert
aus einer Komponente abfragen bei Bedarf

if(stuff != "")
{

    var data = {};

    data.rows = [
        _newRow("Dokumentation ADITO4.4", null, null, [
            _newRow("Technische Handbücher", null, null, [
                _newRow("Betriebshandbuch", -52429, "Doku Team 1",
undefined, false, datetime.toLong("01.07.2015", "dd.MM.yyyy"),
datetime.toLong("10.07.2015", "dd.MM.yyyy")),
```

```
                _newRow("Reportinghandbuch", -52429, "Doku Team 1",
undefined, false, datetime.toLong("10.07.2015", "dd.MM.yyyy"),
datetime.toLong("20.07.2015", "dd.MM.yyyy"))

            ]),

            _newRow("Benutzerhandbücher", null, null, [

                _newRow("Kurzanleitung", -280064, "Doku Team 2",
undefined, false, datetime.toLong("21.07.2015", "dd.MM.yyyy"),
datetime.toLong("31.07.2015", "dd.MM.yyyy")),

                _newRow("Reportinghandbuch", -280064, "Doku Team 2",
undefined, false, datetime.toLong("31.07.2015", "dd.MM.yyyy"),
datetime.toLong("10.08.2015", "dd.MM.yyyy"))

            ])

        ]),

        _newRow("Dokumentübergabe", null, "Übergabe", undefined,
true, datetime.toLong("10.08.2015", "dd.MM.yyyy"),
datetime.toLong("10.08.2015", "dd.MM.yyyy")),

        _newRow("Korrekturlesen", -3042353, "Doku Team 3",
undefined, false, datetime.toLong("10.08.2015", "dd.MM.yyyy"),
datetime.toLong("20.08.2015", "dd.MM.yyyy")),

        _newRow("Release ADITO 4.4", null, null, [

            _newRow("Anwendungen", null, null, [

                _newRow("Server", -8670750, "Sonderteam Entwicklung
1", undefined, false, datetime.toLong("01.07.2015", "dd.MM.yyyy"),
datetime.toLong("10.07.2015", "dd.MM.yyyy")),

                _newRow("Designer", -8670750, "Sonderteam
Entwicklung 2", undefined, false, datetime.toLong("10.07.2015",
"dd.MM.yyyy"), datetime.toLong("20.07.2015", "dd.MM.yyyy")),

                _newRow("Client", -8139816, "Sonderteam Entwicklung
1", undefined, false, datetime.toLong("20.07.2015", "dd.MM.yyyy"),
datetime.toLong("30.07.2015", "dd.MM.yyyy")),

                _newRow("Manager", -8139816, "Sonderteam Entwicklung
2", undefined, false, datetime.toLong("30.07.2015", "dd.MM.yyyy"),
datetime.toLong("04.008.2015", "dd.MM.yyyy"))

            ]),

        ])

    ];


    data.markers = [new Marker("Urlaubsphase Team1", 1437256800000,
1438293600000, -280064)];


    data.relations = [

                new Relation(util.getNewUUID(),
data.rows[0].rows[0].rows[1].activities[0].id,
data.rows[0].rows[1].rows[0].activities[0].id, "END_TO_START"),
```

```
                new Relation(util.getNewUUID(),
data.rows[0].rows[0].rows[0].activities[0].id,
data.rows[0].rows[0].rows[1].activities[0].id, "END_TO_START")
            ];
    data.preferences = new TimeScalePreferences(null, null,
1437545686046, false, new ZoomRange(4.0, "WEEKS"));
    //
    data.treetableHeader = new TreeTableHeader([new
TreeTableColumn("Aufgabe"), new TreeTableColumn("Aufgabe2")]);


    result.object(data);


}


    /**
     * @return {Row}
     */
    function _newRow(pColumn, pActivityColor, pCaption, pChildren,
pMilestone, pStartTime, pEndTime)
    {
        var act = new Activity(util.getNewUUID(), pStartTime,
pEndTime, 0,pActivityColor, pCaption, "");
        var milestone = undefined;
        if(pMilestone)
            milestone = new Activity(util.getNewUUID(), pStartTime,
pStartTime, 0, pActivityColor, "", "");


        //new TreeTableCell(pColumn)
        return new Row(util.getNewUUID(), pChildren, [act, milestone
] , [new TreeTableCell(pColumn)], true);
    }
```

3.4. Aufbau des Gantt-Diagramms mit AditoGanttAdditional

Das Objekt zum Aufbau des Gantt-Diagramms ist komplex. Für die tägliche Arbeit und das leichte Übernehmen einer zweidimensionalen Struktur in ein Gantt-Objekt stellt ADITO die Funktionsbibliothek `aditoGanttAdditional` zur Verfügung.

Grundsätzlich muss dieser Funktionsbibliothek aber auch noch ein Objekt übergeben werden, dieses Objekt ist jedoch einfacher aufgebaut als das Objekt, welche das Gantt-Diagramm zur Rückgabe im `contentProcess` überarbeitet.

Folgendes Codebeispiel nutzt die Funktion `buildGantt`:

```
import("system.result");
import("system.translate");
import("system.eMath");
import("system.db");
import("aditoGantt");
import("aditoGanttAdditional");
import("lib_sql");


//                          0                                          1
2          3           4        5           6

var sqlstr = "select relation_id,  " + concat(["firstname",
"lastname"]) + ", campaign_id, name, date_start, date_end,
description "
            + "from campaignparticipant "
            + "join relation on relation_id = relationid "
            + "join pers on pers_id = persid "
            + "join campaign on campaign_id = campaignid ";


var firstdate = db.cell("select date_start from campaign order by
date_start asc");
var lastdate = db.cell("select date_end from campaign order by
date_end desc");



  var data = db.table( sqlstr);
    var tree = {
        root: {
            ids: []
        }
    };
```

```
    for ( var i = 0; i < data.length; i++)
    {
        var pid = data[i][0];
        var id = data[i][2];
        if (tree[pid] == undefined)
        {
            tree[pid] = {
                ids: [],
                name: data[i][1],
                count: 0,
                tooltip: data[i][1]
            };
            tree.root.ids.push(pid);

        }
        var start = eMath.absInt( data[i][4] );
        var end = eMath.absInt( data[i][5] );

        if ( start != "" && end != "" && start <= end )
        {
            tree[id] = {
                name: data[i][3],
                act: [{
                    id: data[i][2],
                    start: start,
                    end: end,
                    color: "-3355444",
                    //color: (data[i][7] == "" ? "-3355444":
data[i][9]),
                    caption: data[i][6],
                    tooltip: data[i][6]
                }],
                tooltip: ""
            };
            tree[pid].ids.push(id);
        }
    }
```

```
 var getTreeTableCell = function( pID )
    {
        return new TreeTableCell( tree[pID].name, tree[pID].tooltip
);
    }
    var getChild = function(pID, pIndex)
    {
        return tree[pID].ids[pIndex];
    }
    var getChildCount = function(pID)
    {
        return tree[pID].ids != undefined ? tree[pID].ids.length :
0;
    }
    var getId = function(pID)
    {
        return pID;
    }
    var isExpanded = function()
    {
        return true;
    }
    var getAct = function(pID, pActIndex)
    {
        return tree[pID].act[pActIndex];
    }
    var getActCount = function(pID)
    {
        return tree[pID].act != undefined ? tree[pID].act.length :
0;
    }
    var getActId = function(pAct)
    {
        return pAct.id;
    }
    var getActDuration = function(pAct)
    {
```

```
        return [pAct.start, pAct.end];
    }

    var getActColor = function(pAct)
    {
        return pAct.color;
    }
    var getActCaption = function(pAct)
    {
        return pAct.caption;
    }
    var getActToolTip = function(pAct)
    {
        return pAct.tooltip;
    }

    data = buildGantt("root", 1, getTreeTableCell, getChild,
getChildCount, getId, isExpanded, getAct, getActCount, getActId,
getActDuration, null, getActColor, getActCaption, getActToolTip );

    data.preferences = new TimeScalePreferences(null, null,
firstdate, false, new ZoomRange( (lastdate - firstdate) /
datetime.ONE_DAY, "DAYS"));
    data.treetableHeader = new TreeTableHeader([new TreeTableColumn(
translate.key("Kampagne") + " / " + translate.key("Stufen"))]);
    result.object(data);
```

3.5. Aktualisieren eines Gantt-Diagramms mit swing.updateGantt

3.5.1. Hinzufügen einer Aktivität

```
import("system.swing");
import("system.util");
import("system.vars");
import("aditoGantt");

//Daten und Pfad an der ausgewählten Stelle
var data = vars.get("$local.data");
var path = vars.get("$local.path");
var time = vars.get("$local.time");
```

```
//Navigiere zur Ebene, in der die Activity hinzugefügt wird
for (let i = 0; i < path.length; i++)   data = data[path[i]];

//Alle IDs der Ebene + neue ID für Zeitabschnitt
var ids = util.getNewUUID();

[...hier werden spezifisch Daten abgefragt und die Variablen
gefüllt...]

//Neue Activity erstellen mit den gewünschten Daten
var newActivity = new Activity(ids, from_Date, till_Date, null,
color, description, "Geplante Stunden: "+ duration/60);

//vorhandenes data-Objekt erweitern um neue Activity
data.activities[data.activities.length] = newActivity;

//Gantt-Diagramm mit aktualisiertem Objekt updaten
swing.updateGantt([data], "$comp.GanttChart");
```

3.5.2. Bearbeiten einer Aktivität

```
import("system.swing");
import("system.vars");
import("aditoGantt");

//Daten und Pfad an der ausgewählten Stelle
var data = vars.get("$local.data");
var path = vars.get("$local.path");

//Navigiere zur Ebene, in der die Activity aktualisiert wird
for (let i = 0; i < path.length; i++)   data = data[path[i]];

//Alle IDs der Ebene + neue ID für Zeitabschnitt
var ids = data.id;

[...hier werden spezifisch Daten abgefragt und die Variablen
gefüllt...]

//Vorhandene Activity aktualisieren mit den gewünschten Daten
```

```
var existingActivity = new Activity(ids, from_Date, till_Date, null,
color, description, "Geplante Stunden: "+ duration/60);


//Gewünschte Activity innerhalb des Gantt-Diagramms updaten
swing.updateGantt([existingActivity], "$comp.GanttChart");
```

3.5.3. Löschen einer Aktivität

```
import("system.swing");
import("system.vars");


//Daten und Pfad an der ausgewählten Stelle
var data = vars.get("$local.data");
var path = vars.get("$local.path");


var row;
//Navigiere zur Ebene, in der die Activity gelöscht wird
for (let i = 0; i < path.length; i++)
{
    //Speichere die Ebene, aus der die Activity gelöscht wird, für
späteres Aktualisieren
    if(i == path.length - 3)    row = data[path[i]];
    data = data[path[i]];
}


//Alle Activitys überprüfen, bei Übereinstimmung mit gewählter
Activity diese löschen
for (let j = 0; j < row.activities.length; j++)
if(row.activities[j].id == data.id)    row.activities[j] = null;


//Übergeordnete Ebene aktualisieren
swing.updateGantt([row], "$comp.GanttChart");
```