M-Files Gantt View

User Guide

App Version:	1.1.0
Author:	Joel Heinrich
Date:	02-Jan-2013

Contents

1 Introduction

The M-Files Gantt view is an HTML and JavaScript based application that is built upon the UI Extensibility Framework introduced in M-Files version 9. It converts standard M-Files views into interactive, customizable Gantt charts using each object's metadata to determine where and how it appears in the chart.

1.1 Requirements

- Internet Explorer 9 or greater
- M-Files 9.0 or greater
    - additional functionality available with M-Files 9.0.3424.0 or greater

2 Basic Use

2.1 Activation

To use the Gantt chart, it must be installed in the vault by an administrator using the Server Administrator tool. The installation instructions can be found from the M-Files Help (section “Applications”).

Aften the installation of the application, users are prompted to allow the application to run when they login to the vault next time.

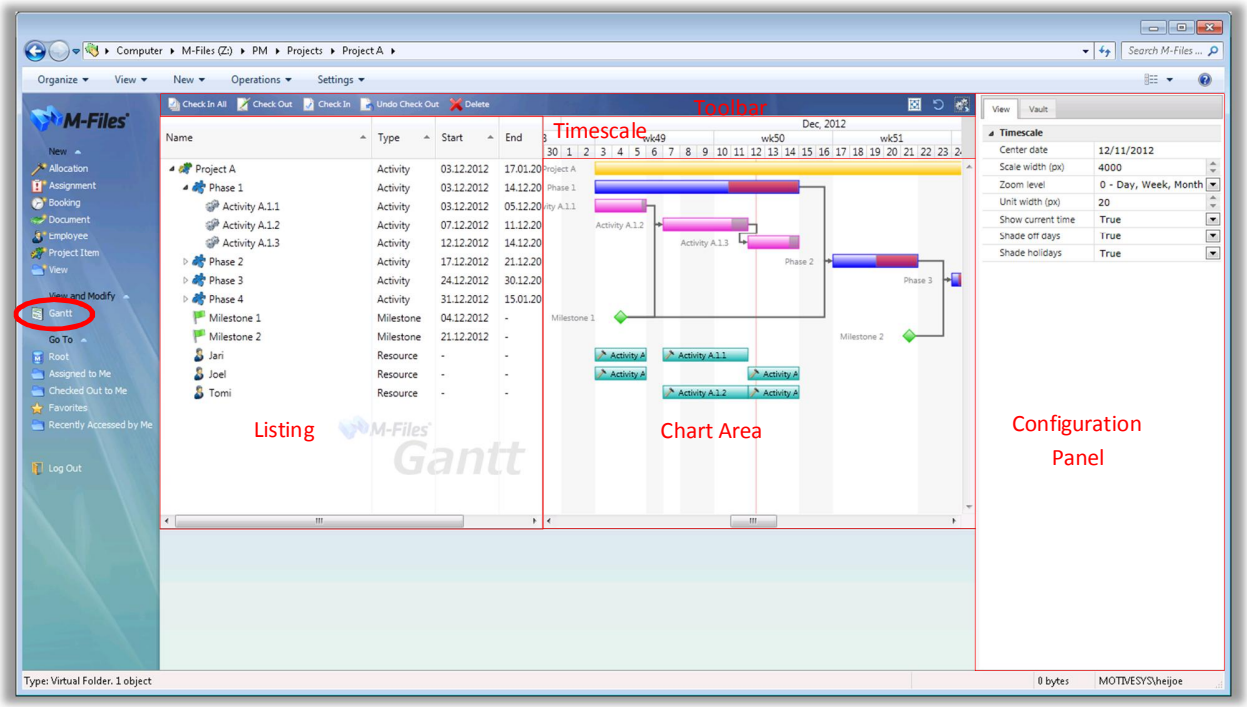
After the activation has taken effect, the "Gantt" button will be visible in the Task Pane under the "View and Modify" menu when in a view with at least one object visible.

The Gantt View can be toggled on or off by clicking the "Gantt" button on the Task Pane.

2.2 Layout

The Gantt view interface consists of the following parts:

- **Toolbar -** Contains some basic commands for items as well as some commands to refresh and auto-zoom the chart and toggle the configuration panel.
- **Listing -** Lists M-Files objects in a fashion similar to the default M-Files listing, however the behavior is not identical. The top level items shown in the Gantt listing are always the same as the items that appear in the current view listing.
- **Timescale -** Displays the current dates which are visible in the chart area.
- **Chart Area -** Renders information about the listing items according to data and time properties and their configuration.
- **Configuration Panel -** Provides full control over how each item in the current view and vault are displayed in the Gantt chart.



2.3 Navigation

To control what dates you are viewing, you can configure the *Center date* and Zoom *level* from the configuration panel-> Timescale section.

You can also use the "Zoom to Fit" option from the toolbar, which attempts to automatically select a *Center date*, *Zoom level* and *Unit width* so that all items are visible.

2.4 Interaction

2.4.1 Mouse shortcuts

If you spin the mouse wheel while holding down the Control button on the keyboard, the chart will dynamically zoom in and out.

If you spin the mouse wheel while holding down the Shift button on the keyboard, the chart will pan left and right.

2.4.2 Rescheduling Items

If a user has edit rights for an activity or milestone, the whole item or individual ends can be dragged in the chart area to reschedule or change their duration.

2.4.3 Time span Highlighting

A span of time can be highlighted in the chart area by clicking/selecting the most detailed units visible in the timescale.

2.5 Validation

The Gantt chart does not validate data. The underlying vault structure should do this, as items displayed in the Gantt chart can also be modified through the standard M-Files interface. There are instances where the Gantt chart does try to detect issues and inform users.

2.5.1 Precedence Violation

If a dependency rule is violated (an item starts before the "Comes before" item has ended) the dependency line will turn red.

2.5.2 Resource Over-Utilization

If a resource has overlapping allocations, both allocation items will glow red.

3 Item Mapping

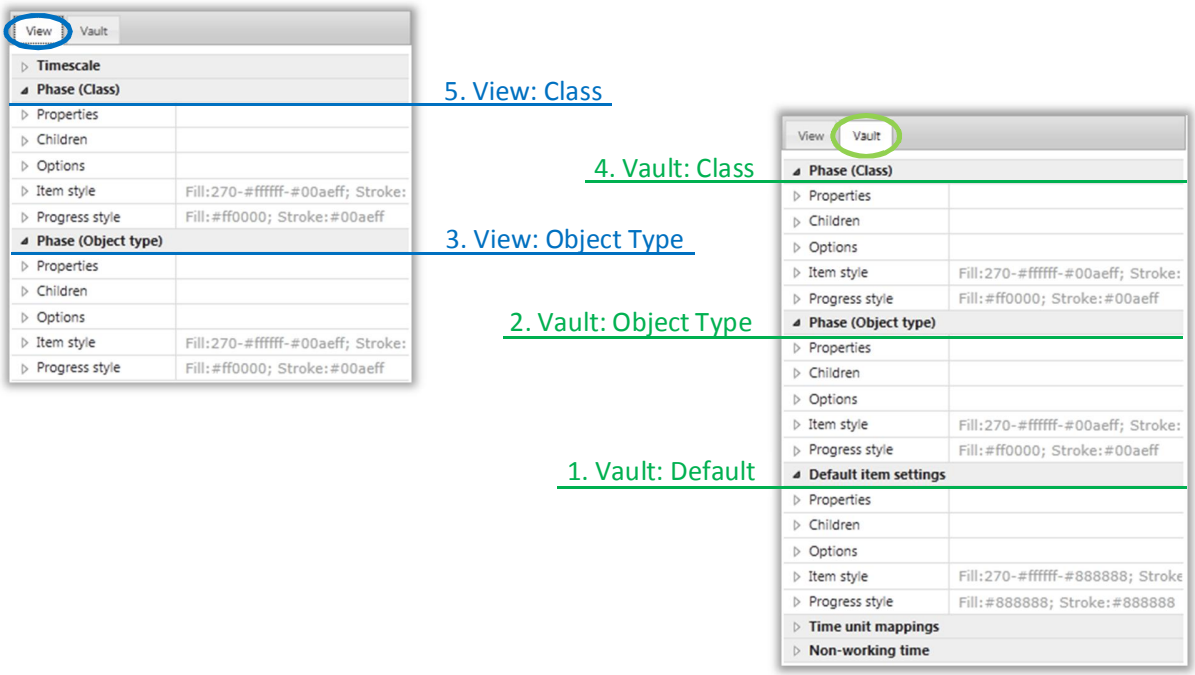
In order for the application to render an M-Files object in the Gantt View, some configuration must be provided indicating how it should be displayed. Each specification indicating how certain M-Files objects should be displayed in the Gantt View is called an *Item Map*.

3.1 Inheritance

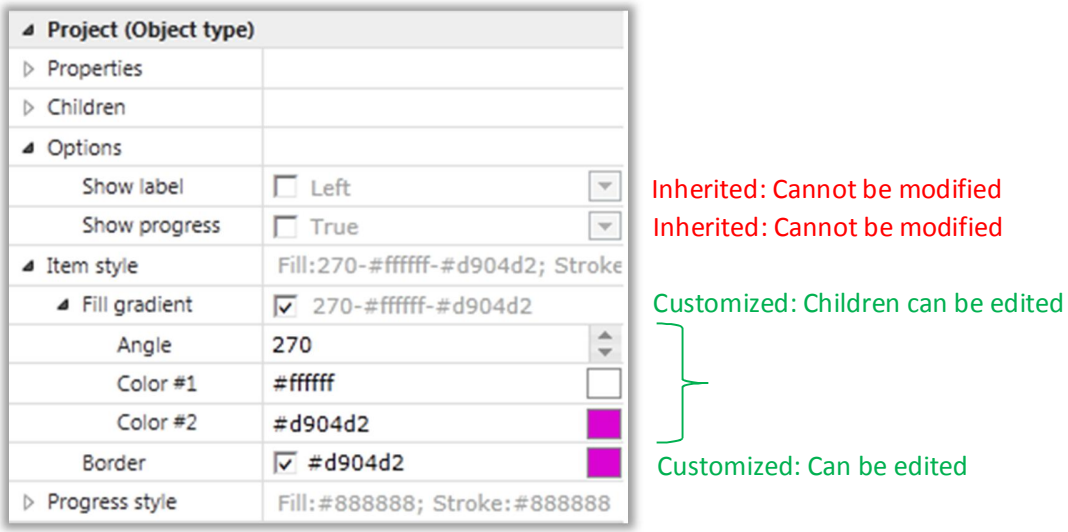
Item Maps can be specified for Object Types or Classes at either the Vault or View level. Additionally, a default Item Map can be specified for the entire vault.

Item Maps inherit settings from one another in a set order to reduce the amount of configuration required to setup a View. This provides, for example, an administrator the ability to define Vault specific defaults, and then allows a user to override those defaults in their own personally defined view.

All the levels of Item Maps that contribute to the way a specific type of item are shown below; numbered in order of priority. An option specified at a higher level (number), will override an option specified at a lower level.



The checkboxes next to settings in an Item Map control whether it is explicitly defined or inherited. The unchecked state indicates the value is inherited, and the setting cannot be modified, until it has been checked.



3.2 Properties

The properties section of an Item Map determines how M-Files meta-data translates to Gantt specific properties.

3.2.1 Render As

The value of the *Render As* property determines how the Gantt chart should render the objects. Items can be rendered as:

- **Activities** - a line or bar that spans a period between the *Start Date* and *End Date*.
- **Milestones** - a diamond at one point in time corresponding to the *Start Date*.
- **Recurring Events/Milestones** - either an activity or milestone that recurs at a specific interval between the *Start Date* and *End Date*. Each occurrence is drawn as a milestone unless *Duration* and *Duration Unit* can be properly resolved. The interval between occurrences is determined by *Recurrence Interval* and *Recurrence Unit*.
- **Resource** - a placeholder in the listing, where children activities or "(resource) allocations" will be rendered in line. Also allows inline creation of allocations.

3.2.2 Start Date

The value resolved from the *Start Date* property determines the starting point of an Activity, the placement of a Milestone, and the start of the first occurrence of a Recurring Item.

3.2.3 End Date

The value resolved from the *End Date* property determines the end of an Activity and the end of the period in which a Recurring Item's occurrences can occur. This property is ignored for Milestones.

3.2.4 Start Time

The value resolved from the *Start Time* property adds time specificity to the *Start Date*.

3.2.5 End Time

The value resolved from the *End Time* property adds time specificity to the *End Date*.

3.2.6 Progress (% Done)

The value resolved from the *Progress* property indicates what percent of an activity has been completed. This property is ignored for Milestones and Recurring items. If the option *Show Progress* is "True" the portion of an activity that has not yet been completed will be masked over by a progress bar. The progress bar can be styled in another section of the *Item Map*.

3.2.7 Duration

The value resolved from the *Duration* property is primarily used to determine how long each occurrence of a Recurring Activity should last. When an *End Date* value cannot be resolved for an activity but *Duration* and *Duration Unit* are present; they will be used to calculate the end date.

3.2.8 Duration Unit

The value resolved from the *Duration Unit* property is used to quantify the time value of the *Duration* property. The selected M-Files property should refer to a Value List, whose items have been mapped to predefined time units in the *Vault: Time Unit Mappings* section of the configuration.

3.2.9 Recurrence Interval

The value resolved from the *Recurrence* property is used to determine the time between each occurrence of a Recurring item.

3.2.10 Recurrence Unit

The value resolved from the *Recurrence Unit* property is used to quantify the time value of the *Recurrence* property. The selected M-Files property should refer to a Value List, whose items have been mapped to predefined time units in the *Vault: Time Unit Mappings* section of the configuration.

3.2.11 Comes After (Dependencies)

The value resolved from the *Comes After* property is used to draw dependency lines between items. The selected M-Files property should be a select list that refers to other objects. When those objects are displayed in the same chart as the item being mapped, a line will be drawn between the two. The arrow end of the dependency line will always point to the item with the chosen property.

The dependencies are universal, and not subject to location in the hierarchy or relationship to other items, unless constrained by the property definition itself.

3.2.12 Allocation

This section determines how items representing the allocation of resources should be created dynamically. If the values are properly defined, after selecting a span of time on a resource line, the selection can be double-clicked, and a new object's meta-data card will appear with the resource and times predefined.

This section does not control which allocations are displayed in-line with a resource. To affect this, object types and/or classes representing resources must be rendered as resources, and the allocation items defined as children.

3.2.12.1 Object Type

The *Object Type* value determines the type of object that is used to store resource allocation information. In general, all classes of this object type should refer to the resource, start and stop dates and possibly the item that the resource is being allocated to.

3.2.12.2 Property

The *Property* value determines which property definition a resource will be pre-populated in when dynamically creating a new resource allocation.

3.2.13 Label

The *Label* value determines what label will be displayed for an item on the chart. Note that this setting does not affect the name or title of the item in the listing. A combination of literal text, and placeholders which reference local item property values can be used similar to a property definition's simple concatenation automatic value options.

Place holders are simply property definition id's between two '%' (percent signs). The default vault is "%0%" - which refers to the built in "Name or Title" property definition.

The label value will not be visible unless the *Show label* option in the *Options* section is set to 'No'.

3.2.14 Icon

The *Icon* setting allows an icon to be displayed within an activity bar. If the '(default)' value is selected, the items default icon will be displayed. If a property definition is selected, the value list item's icon that corresponds to current value will be displayed (if there is one). If an item has multiple values specified for the property to be used to resolve the icon, no icon will be displayed.

3.3 Children

The hierarchy visible in the Gantt view is fully configurable. The *children* settings determine which items will be displayed directly below an item (or in the case or resources, in-line with the item). All child definitions are processed in parallel, i.e. all resolved children will be displayed side by side; they do not represent different levels of the hierarchy.

3.3.1 Refer to Parent

If children items contain a property that refers to their parent, they should be defined in the *Refer to Parent* section.

3.3.1.1 Property

The *property* setting indicates which property the child item's use to refer to their parent. If an item is found with this property, and a value that refers to the parent item, it will be displayed as the parent's child.

3.3.1.2 Class Filter

Sometimes too many items refer to a parent via a certain property. The *Class Filter* setting allows the child items to be filtered, so only items with a certain class will be displayed as a child.

3.3.2 Defined in Parent

If child items are defined in a property of the parent item itself, they should be specified in the *Defined in Parent* section.

3.3.2.1 Property

The *Property* setting indicates which property of the parent item, contains references to the children that will be displayed.

3.4 Options

3.4.1 Show Label

The *Show label* setting controls if and where an item's label will be displayed in the chart. The value of the label can be configured in the *Properties* section.

Possible values are:

- **No** - Hides the label of each item
- **Left** - Display a label to the left of each item (default)

- **Inside** - Displays a label inside each item (only works with activities)
- **Right** - Displays a label to the right of each item

3.4.2 Show Progress

The *Show Progress* option controls whether the progress bar overlay is rendered for activities. If set to true, the activity must still resolve a progress value for the overlay to be drawn. The style of the progress overlay can be configured in the separate *Progress Style* section.

3.5 Item Style

The style of each item type can be configured.

3.5.1 Fill Gradient

The *Fill Gradient* section contains settings that control the basic color of an item. By default this is a gradient, whose angle and colors can be specified. If a solid color (no gradient) is desired, give Color #1 and Color #2 the same value.

3.5.2 Border

The *Border* setting controls the item's border color.

3.6 Progress Style

The progress overlay style for each item type can be configured.

3.6.1 Fill / Fill Opacity

The *Fill* setting controls the main color of an item's progress overlay. The *Fill Opacity* setting controls how much of the underlying default color will be visible.

3.6.2 Border / Border Opacity

The *Border* setting controls the border color of an item's progress overlay. The *Border Opacity* setting controls how much of the underlying default color will be visible.

4 Chart Options

4.1 Time Unit Mapping

Value lists items can be directly mapped to known time units. These value list items can then be used in items to specify interval and duration units which are understood by the chart.

Add each value list that contains items that can refer to time units, and then map each of those items to the matching time unit.

4.2 Non-Working Time

Non-working time can be shaded in the chart, but those time need to be specified.

4.2.1 Days Off (Weekly)

If there are days each week that are usually not worked (like weekends) then those days should be specified in the *Days Off* section. By default, Saturday and Sunday are selected.

4.2.2 Holidays

In addition to weekly days off, specific dates can be specified as holidays. The dates must be entered by hand in the format "yyyy-mm-dd" or "mm-dd". If no year is specified, the date will be considered a holiday in every year. For example, a value of "12-25" can be used for Christmas as it falls on the same date every year. Variable date holidays such as Easter will need to be entered on a per year basis.

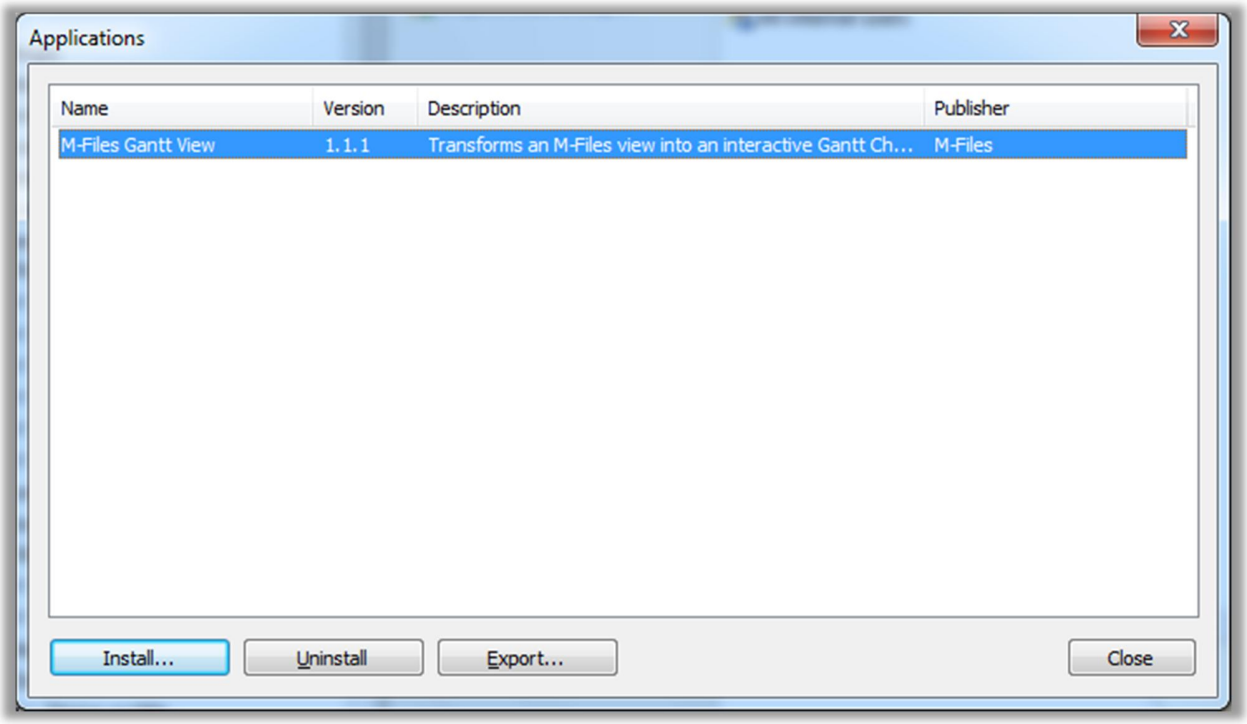
5 Step by Step Setup

This section explains how to setup a vault from scratch. It discussing the different choices users have and best practices. We will start with a new, empty vault called "Gantt Demo". Create it, and a client connection to it, now.

5.1 Install the Application

To install the Gantt View application:

1. Right-click on the "Gantt Demo" vault icon
2. Select Applications
3. Click the "Install..." button
4. Navigate to, and open the application "Gantt.zip".

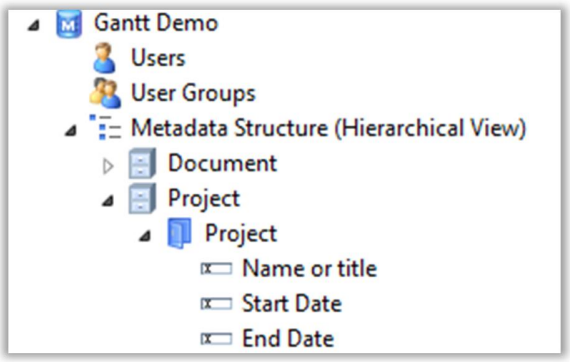


5.2 Projects

5.2.1 M-Files Admin Setup

Gantt charts are most commonly used for Project Management, so let's setup some projects in our new vault.

Typically, projects last multiple days, so we'll define two mandatory date properties to capture this information on the default class; *Start Date* and *End Date*.



5.2.2 Client Content Setup

Now we should have enough information to use the Gantt chart application, so let's log in to the vault via the client, create a couple projects, and then create a view where we can see them all.

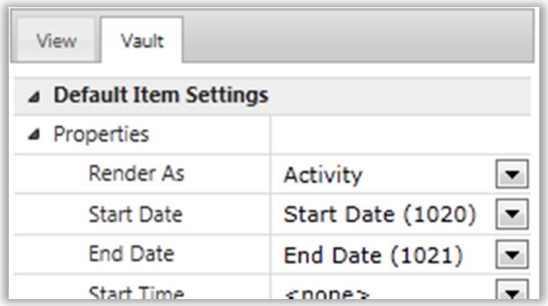
Note: You may have to allow the Gantt application to run the first time you log in. You may also have to activate it by navigating to Settings->Applications and checking the box next to it.

After navigating to the new view, you should see the "Gantt" button in the Task Pane (it will only appear if you enter a view with visible items, so make sure you create some projects first).

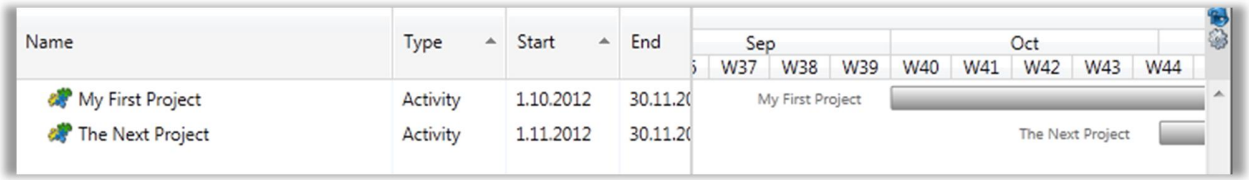
5.2.3 Default Property Setup

Click the Gantt button, and wait for the interface to load. You'll notice that projects appear in the listing, but they aren't drawn anywhere, because no Item Maps have been defined yet. Let's setup the Vault's default Item Map, so we have some general vault-level rules for rendering things. This will require "Common View" editing rights.

1. Click the "Gear" icon in the upper right corner of the chart to show the configuration panel
2. Select the "Vault" tab
3. In the "Default item settings" section, set the following properties
    a. Render As = Activity
    b. Start Date = Start Date
    c. End Date = End Date



After setting just those three properties, your items should now be rendered on the chart, and the "Start" and "End" columns in the listing should be displaying the "Start Date" and "End Date" values found in your projects' meta-data.

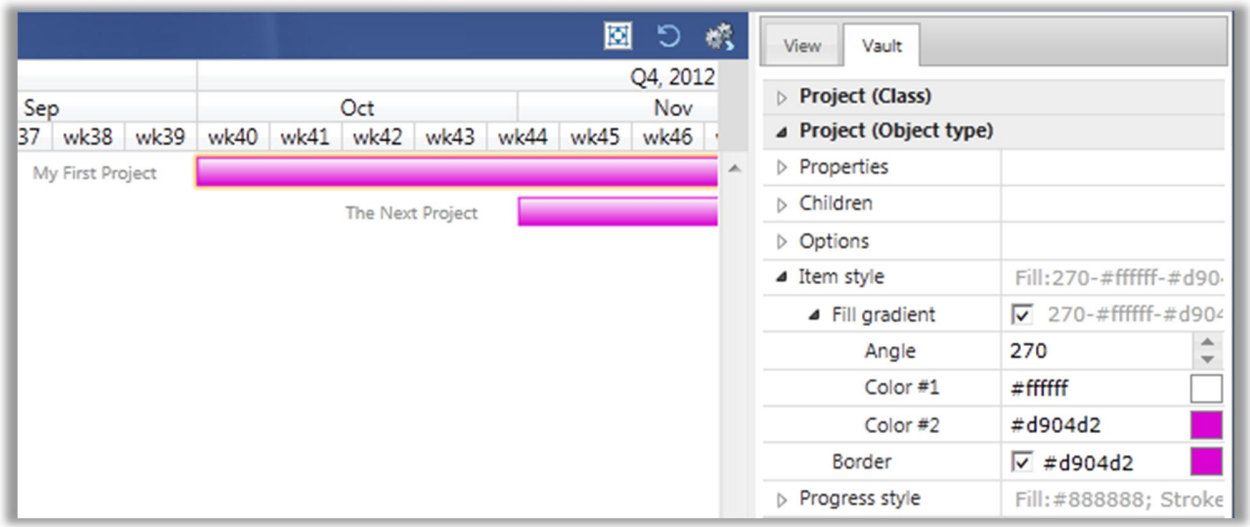


Note: You might need to navigate around the chart to find them depending on the dates you chose (use "Shift-Mouse wheel" to pan, and "Ctrl-Mouse wheel" to zoom in and out) or simply click the "Zoom to Fit" button in the toolbar.

5.2.4 Custom Colors

Now that we've setup these defaults, the Gantt chart will be able to correctly display any new object that uses our "Start Date" and "End Date" meta-data properties. The default gray color is pretty boring though, and we don't want to mistake a project on the chart with other types of objects that we create later, so let's specify a color that will be specific to projects.

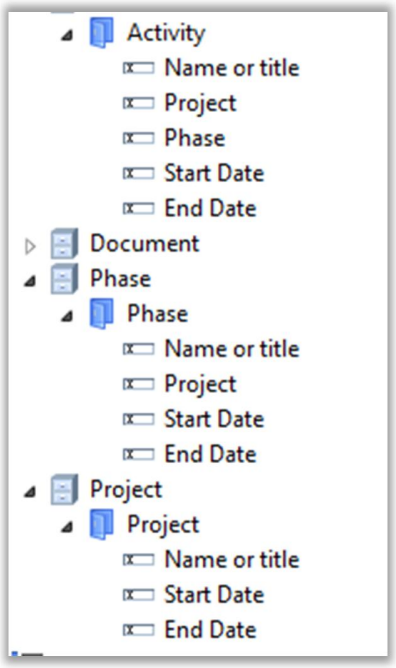
1. With the configuration panel open, click on a project in the Gantt listing
2. Make sure the "Vault" tab is selected
3. Navigate to the *Project (Object type): Item Style* section
4. Then with the *Fill Gradient* and *Border* settings
    a. Check the boxes to indicate we are going to override the default settings
    b. Specify your preferred color(s)



5.3 Phases & Activities

Being able to see when projects start and stop is a great first step, but most projects consist of much more detail. Let's break-down the projects into phases, and the phases into specific activities so we can visualize all these details.

5.3.1 M-Files Admin Setup



Let's create two new object types to represent the *Phases* and *Activities*.

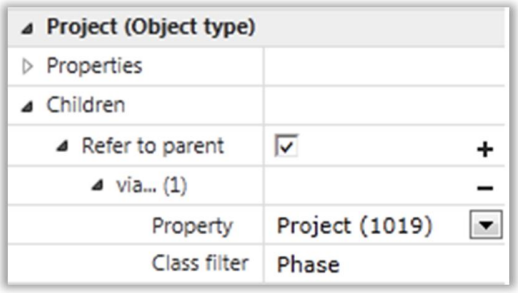
We'll give their default classes both the *Start Date* and *End Date* properties we gave the projects', and then we'll define the hierarchy manually by giving them both the default Project property, and then giving the Activity class the Phase property as well.

5.3.2 Children (Refer to Parent via Property…)

Now that we have some new objects that provide our project break-down information, we need to configure the Gantt chart to display them where we want. There is a clear hierarchy we've setup, which is Project -> Phase -> Activity.

So let's first define phases as the children of projects:

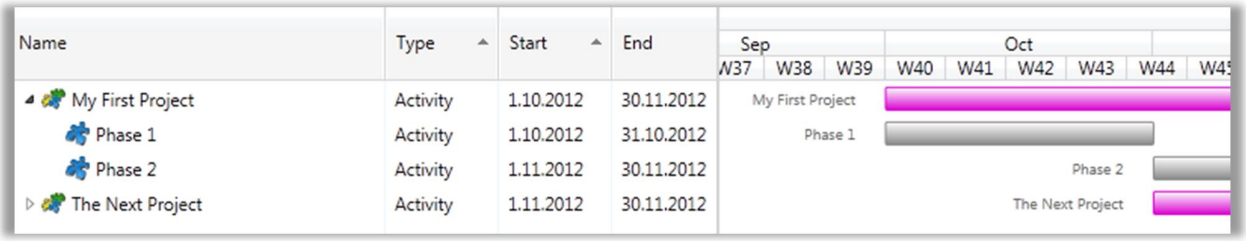
1. Return to the Gantt chart and, with the configuration panel open, select a Project
2. Navigate to the configuration section *Vault: Project (Object Type): Children*
3. Check the box next to "Refer to parent"
4. Click the plus (+) icon to create a new definition
5. Expand the "Refer to parent" property and the newly created definition "via… (1)"
6. Choose default "Project" property definition for the Property setting
   Note: We do this because a Phase relates to its parent project through its Project property
7. Select only "Phase" for the Class Filter



Note: We do this so Activities won't also appear as children directly under the Project (they also refer to Projects through the Project property).

Now, if you refresh the chart by pressing F5 or the blue icon in the upper, right corner you'll see tree arrow icons appear next to the projects in the listing. This is because they now have a definition for containing children.

Let's create a couple Phases that break down the first project in the listing, and then expand the tree.

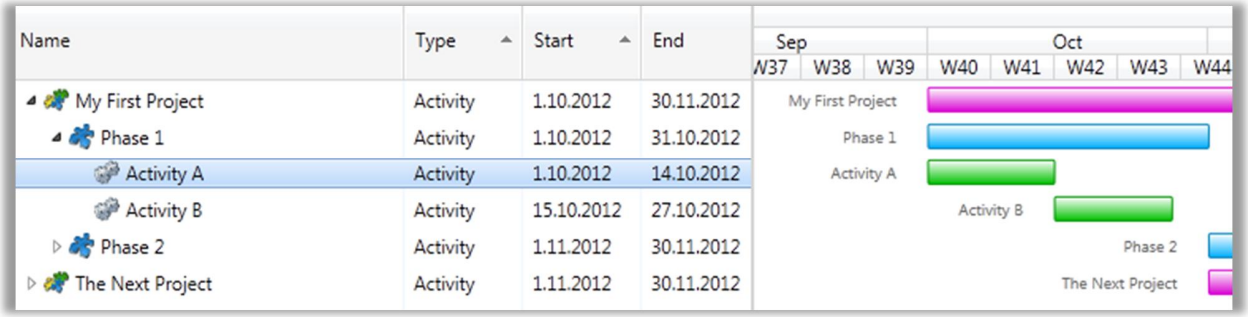


Next, let's configure Activities as the children of Phases in the same way:

1. Select a Phase
2. Navigate to the configuration section *Vault: Phase (Object Type): Children*
3. Check the box next to "Refer to parent"
4. Click the plus (+) icon to create a new definition
5. Expand the "Refer to parent" property and the newly created definition "via… (1)"
6. Choose the default "Phase" property definition for the Property setting
7. Select only "Activity" for the Class Filter

Again, refresh the chart, define a couple activities and expand the related phase.

You've probably noticed already that both the Phases and Activities are rendered with the dreary, default gray. Go ahead and define distinctive colors for them just as we did for the projects.



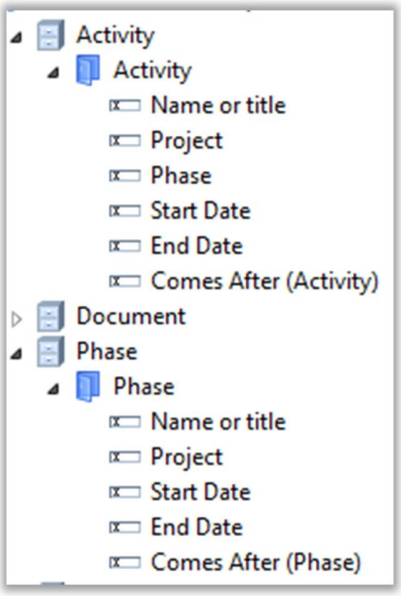
5.3.3 Dependencies

Often times, the activities and planned phases of a project must come in a specific order. For instance, it could be presumed in the screen shot above that "Phase 1" must be complete before "Phase 2" can begin; and it could also be presumed that "Activity A" must be completed before "Activity B" can begin.

Before we can configure the chart to express this, we first need to create some new properties, so we can define these dependencies in our meta-data.

1. Create a new property definition for the Phase class
    a. Name: "Comes After (Phase)"
    b. Type: "Choose from list (multiselect)"
    c. Show values from the following list: "Phases"

2. Create a new property definition for the Activity class
    a. Name: “Comes After (Activity)”
    b. Type: “Choose from list (mulitselect)”
    c. Show values from the following list: "Activities"



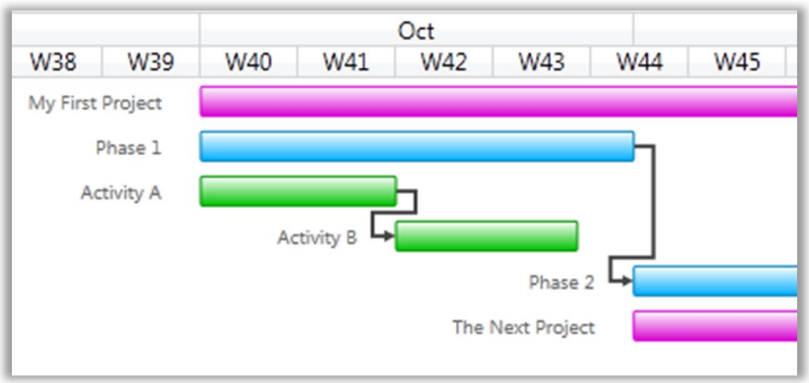
Back in the Gantt chart, we’ll specify the actual dependencies of our items:

3. Edit “Phase 2”
    a. Set “Comes After (Phase)” to “Phase 1”
4. Edit “Activity B”
    b. Set “Comes After (Activity)” to “Activity A”

And now, we’ll tell the Gantt chart what those properties mean.

5. Select a Phase in the Listing
6. Navigate to the configuration section *Vault: Phase (Object type): Properties*
7. Check the box next to “Comes After”
8. Choose the “Comes After (Phase)” property
   Note: You might need to close and reopen the configuration panel to refresh the property list
9. Select an Activity in the Listing
10. Navigate to the configuration section *Vault: Activity (Object type): Properties*
11. Check the box next to “Comes After”
12. Choose the “Comes After (Activity)” property

If configured correctly, the dependency lines will be drawn as illustrated in the screen shot below.



5.3.4 Progress

Gantt charts generally show events and activities as they are planned. It can be useful, however to know how reality measures up. To illustrate this, progress information can be shown for items rendered as Activities.

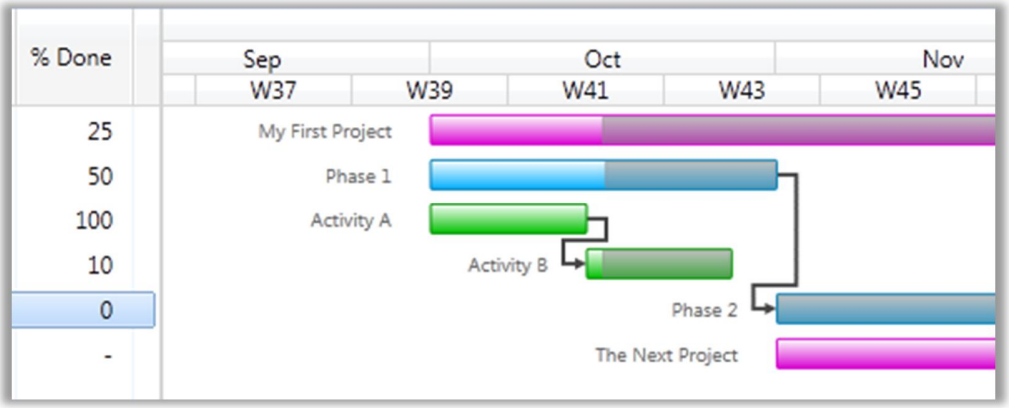
To demonstrate this, let's first create the requisite property definition, so we can store this information in the meta-data.

1. Create a new property definition
    a. Name: "Progress (%)"
    b. Type: "Number (integer)"
2. Add the property to all of the classes we've created
    a. Project
    b. Phase
    c. Activity

Next, let's specify at the default vault level what our new property definition represents (this way we won't have to specify it for each of the three object types individually).

3. Navigate to the configuration section *Vault: Default Item Settings: Properties*
4. Set Progress = "Progress (%)"

Now that the property is wired up, specify the progress for some of the items. It should work for any object type.

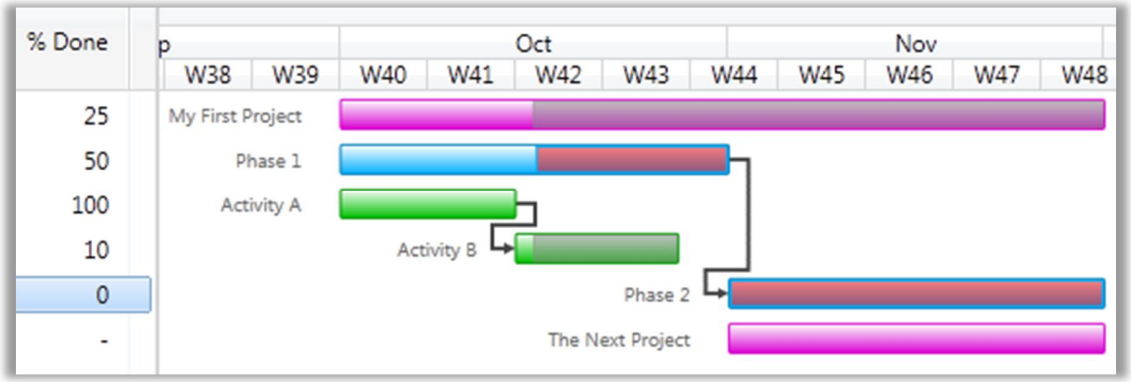


5.3.5 Progress Style

The default style of the progress bar should suit most situations, and having it rendered consistently makes it easier for users to become familiar with its meaning. There may be times however, when the progress bar should be styled for emphasis, or simply to make it more visible atop a chosen color.

For instance, if the 0%-complete "Phase 2" item in the chart above was seen on its own, it might not be clear that the whole bar is shaded. Let's make the lack of progress stand out in red, and border it with more blue so it's clear that it's not a red item, but an incomplete blue item:

1. Select a Phase from the listing
2. Navigate to the configuration section *Vault: Phase (Object type): Progress Style*
3. Check the boxes next to all the Progress Style settings and edit their values



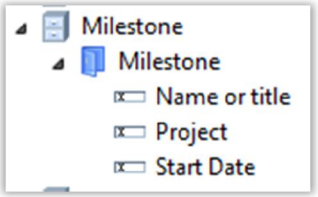
5.4 Milestones

So far we've only mapped items that correlate to a span of time, i.e. comprised of both start and end dates. But, it is also possible to display single, significant moments in time, by rendering an item as a milestone.

5.4.1 M-Files Admin Setup

Before we begin rendering Milestones in the chart, let's create a Milestone object type in M-Files:

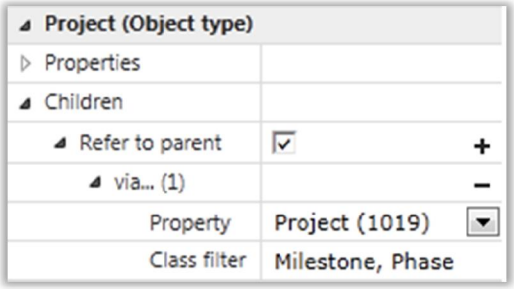
1. Create a new "Milestone" object type in M-Files Admin
2. Add the Project property as required
3. Add the Start Date property as required



5.4.2 Client side

Returning to the client now, let's first specify that we wish to see the milestones as children of their respective project(s).

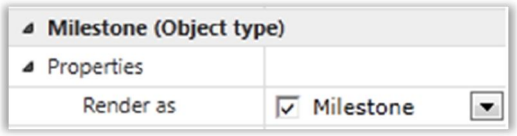
1. Select a Project from the listing
2. Navigate to the configuration section
   Vault: Project (Object Type): Children: Related To: 0
3. Add "Milestone" to the *Class Filter* setting



Note: we can use the existing definition because the relationship is based on the same property

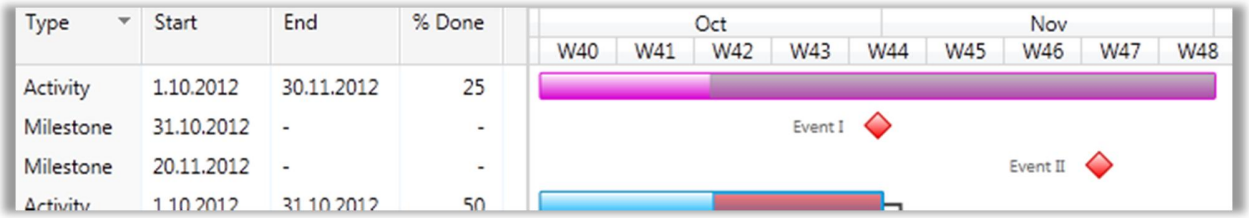
Now let's create a couple Milestones, refresh the chart, and expand the project that contains them. The Milestones should now appear in the listing, but they aren't rendered anywhere. This is because they inherited the *Render As* value of "Activity", but they don't have an *End Date* so the Gantt chart cannot resolve how to display

them. Let's fix this:



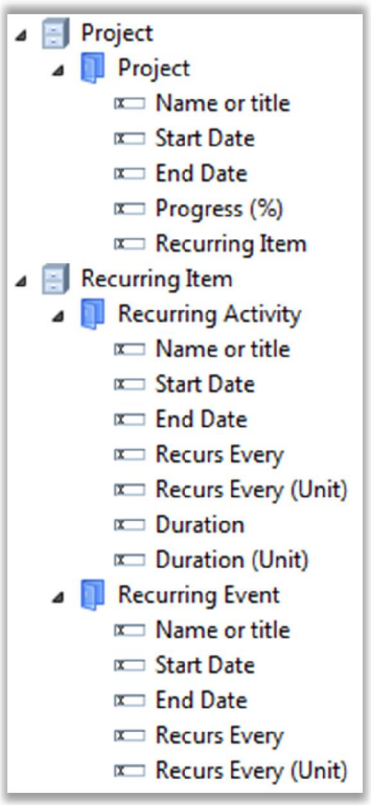
4. Select a Milestone from the listing
5. Navigate to the configuration section *Vault: Milestone (Object Type): Properties*
6. Check the box next to "Render As"
7. Set the "Render As" value to "Milestone"

With that change, the milestones should now appear in the chart. Let's change their color while we have the configuration section open, and apply a quick sort on the listing's "Type" column.



5.5 Recurring Items

Many projects have events or activities that recur on a set schedule. For instance, there might be a progress review meeting once a week, or an internal audit every month. It can be useful to display this information in the Gantt chart so users are reminded of the events, and use that knowledge, for instance, to schedule things around them.

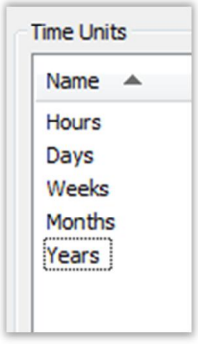


5.5.1 Server Setup

Recurring items are quite complex, and must convey a lot of information to the Gantt chart in order to be rendered correctly.

First, let's setup a value list to help communicate the recurrence interval and duration of our items.

1. Create a new "Time Units" value list
   a. users shouldn't need to add new values to the list
2. Add the following contents:
   a. Hours
   b. Days
   c. Weeks
   d. Months
   e. Years



Next, let's build a new object type and some classes:

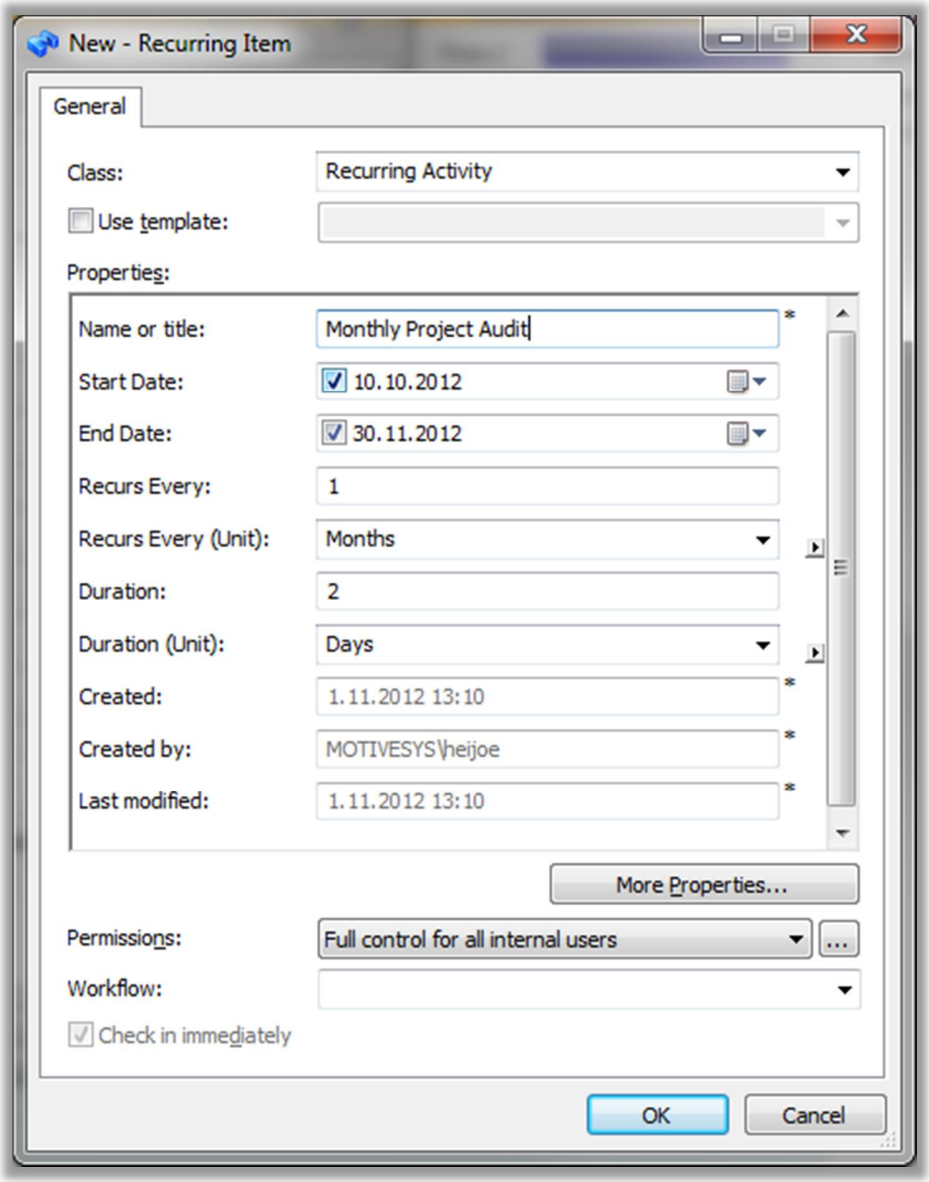
3. Create a new "Recurring Item" object type
    a. Don't show the creation command in the task area
4. Rename the default class to "Recurring Activity"
5. Add a new class called "Recurring Event"
6. Add the following required properties to both classes:
    a. Start Date
    b. End Date
    c. A new integer property "Recurs Every"
    d. A new single select property "Recurs Every (Unit) " based on "Time Units"
7. Add the following required properties to the "Recurring Activity" class only:
    a. A new integer property "Duration"
    b. Add a single select property "Duration (Unit) " based on "Time Units"
8. Add the default (auto-generated) "Recurring Item" property to the "Project" class

5.5.2 Creating a Recurring Activity

When creating the "Recurring Item" object type, we specifically chose not to "show the creation command in the task area", and we also chose not to provide the reference to a project in the Recurring Item's meta-data, but instead placed a property in the Project class so that it could reference Recurring Items. We'll address how to map this relationship to the Gantt chart in the next section, but it also effects how we'll create new Recurring Items:

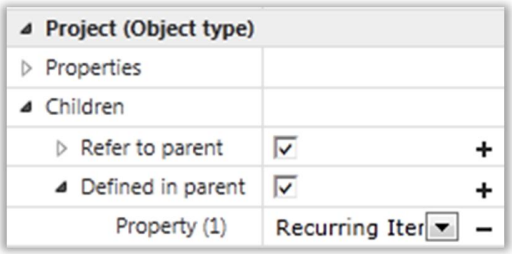
1. Double-click a project to open its meta-data card
2. Press the small arrow button next to the Recurring Item property and select "New (Recurring Item)..."
3. Select the "Recurring Activity" class and specify all its properties

In the metadata card shown below, we've indicated that our project has a Monthly Audit, on the 10th and 11th of every month. Starting in October and ending in November.



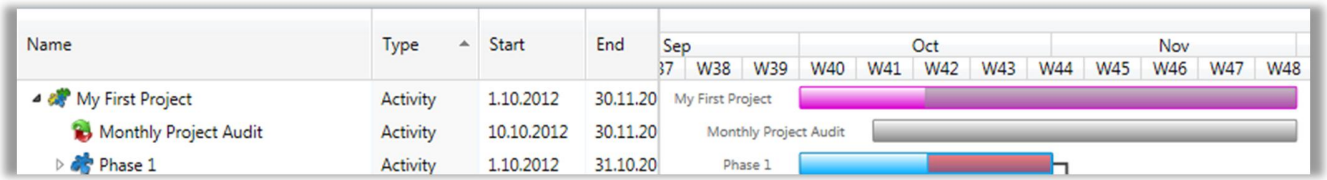
5.5.3 Children Revisited (Defined in Parent)

Now we are ready to specify where Recurring Items should be shown. Unlike the other items we've mapped thus far, Recurring Items do not reference their parents, rather, in this particular setup, the parent has a property that references its child.



1. Select a Project in the listing
2. Navigate to the configuration section
   Vault: Project (Object Type): Children
3. Check the box next to "Defined in parent"
4. Click the "plus" (+) icon
5. Expand the "Defined in parent" node
6. Set definition "Property (1)" to "Recurring Item"

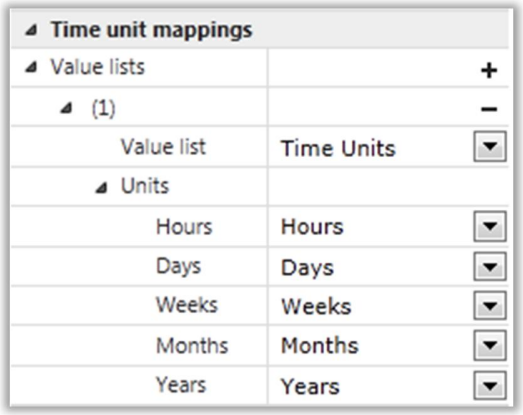
Now, the items found in a project's "Recurring Item" property will be displayed as its children as well. When we refresh the chart and open our project, we see that that is indeed the case, but our Recurring Activity is being render as a normal activity, and is not using any of the recurrence information we provided.



5.5.4 Time Unit Mappings

Before wiring up the Recurring Items themselves, let's wire up the newly created "Time Units" value list, so the Gantt chart can recognize their meaning. The literal values of the value list items aren't used here, so that translations of the list items are supported.

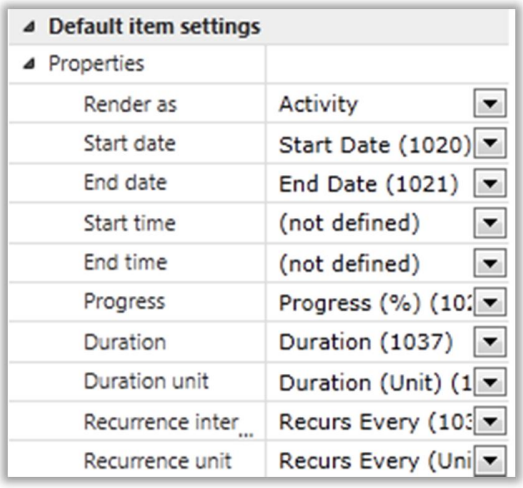
1. Navigate to the configuration section *Vault: Time Unit Mappings*
2. Click the "plus" (+) icon, and expand the Value Lists and new definition "1" node
3. Select "Time Units" as the Value List, then expand the Units node
4. Map each unit with the corresponding value list item



5.5.5 Wiring up Recurring Items

To render our Recurring Item correctly, let's first wire up our newly defined properties at the vault's default level, so they can be reused if we ever create other objects that can recur.

1. Navigate to the configuration section *Vault: Default Item Settings: Properties*
2. Set the properties:
    a. Duration to "Duration"
    b. Duration Unit to "Duration (Unit)"
    c. Recurrence Interval to "Recurs Every"
    d. Recurrence Unit to "Recurs Every (Unit)"

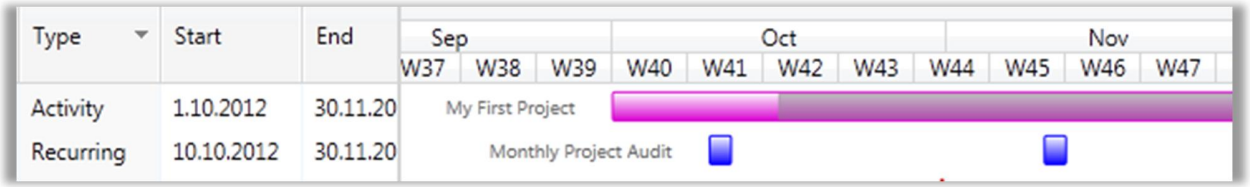


Next, let's override the *Render As* property in for the specific object type.

3. Select a Recurring Item in the listing
4. Navigate to the configuration section
   Vault: Recurring Item (Object type): Properties
5. Check the box next to *Render As*
6. Choose "Recurring" for the *Render As* property

Now our recurring item should be displayed correctly, with a two-day event appearing in both October and November.

A quick color change and sort should result in something similar to this:



5.6 Resources

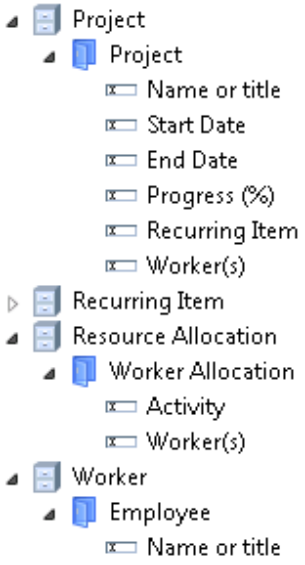
Practically all activities of a project require resources. A resource can be anything from a worker, or piece of equipment to an office or manufacturing space. The Gantt chart can be configured to list resources and display when they've been allocated (reserved).

For our vault, let's define a set of workers for a project, and then allocate them to it's various activities.

5.6.1 Server Setup

Before we configure the chart, we'll need to add two new object types to our vault structure; Workers (our resource) and Resource Allocations (where we define when and for what a resource has been reserved).

1. Create a "Worker" object type
   a. Set the default class to "Employee"
2. Create a "Resource Allocation" object type
   a. Set the default class to "Worker Allocation"
      i. Add the default "Activity" property and set it as the name
      ii. Add the default "Worker" property (I've renamed it to "Worker(s)")
      iii. Remove the default "Name or title" property
3. Add the default "Worker" property to the "Project" class



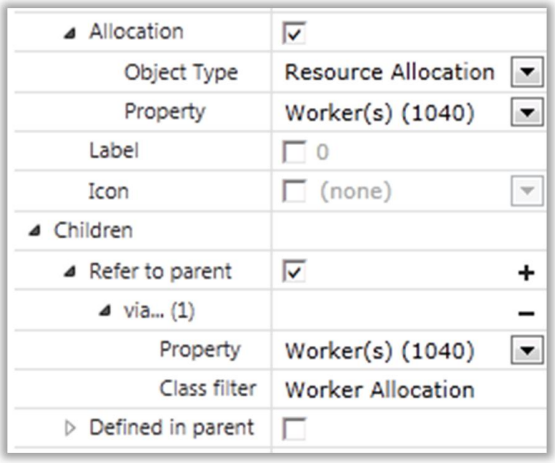
5.6.2 Configuration

Back in the Gantt View, let's first configure our Project to display our resources.

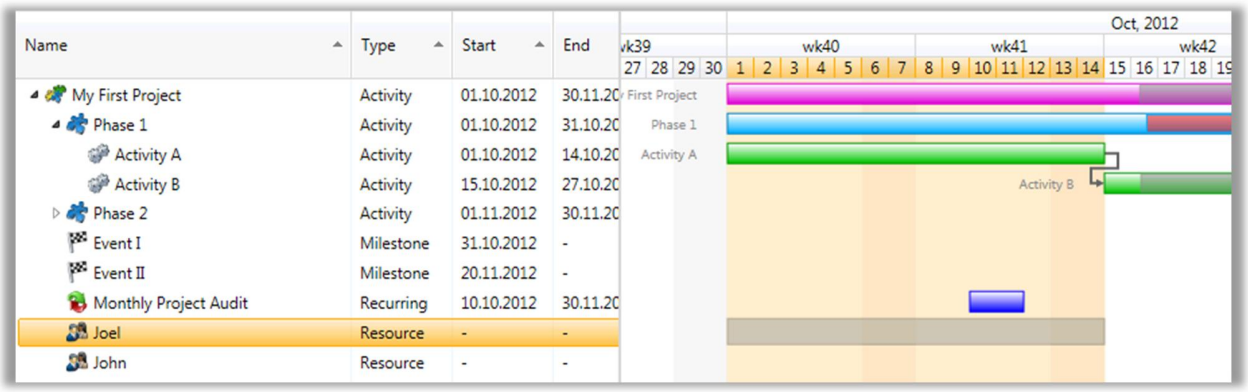
1. Open the configuration panel
2. Select a Project from the listing
3. Navigate to the configuration section
   Vault: Project (Object type): Children: Defined in parent
4. Add a new child property, and set the value to our new default "Worker" property definition
5. Then double-click the project and add a couple workers to it (you'll have to create some employees on the fly, because there aren't any yet)

After you've saved the changes you should see the new workers appear underneath your project. By default, the workers are being rendered as activities though. Let's continue with the configuration so they're treated as resources.

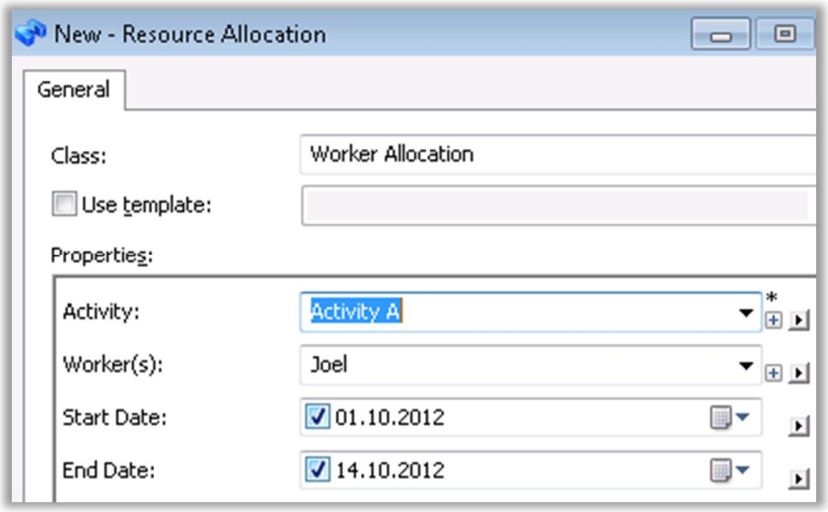
6. Select a Worker from the listing
7. Navigate to the configuration section
   Vault: Worker (Object type): Properties
   a. Locally define the *Render As* setting to "Resource"
   b. Locally define the Allocation settings to
      i. Object Type = "Resource Allocation"
      ii. Property = "Worker(s)"
8. Next, navigate to the configuration section
   Vault: Worker (Object type): Children: Refer to Parent
9. Add a new child definition where
   a. Property = "Worker(s)"
   b. Class filter = "Worker Allocation"



Now, we should be ready to view (and create) allocations for our workers. To begin we'll allocate Joel to Activity A. We can highlight the time span of Activity A so that it's easy to select the same time period for the allocation. Select the time period on the same line as Joel and then double click the gray selection bar.



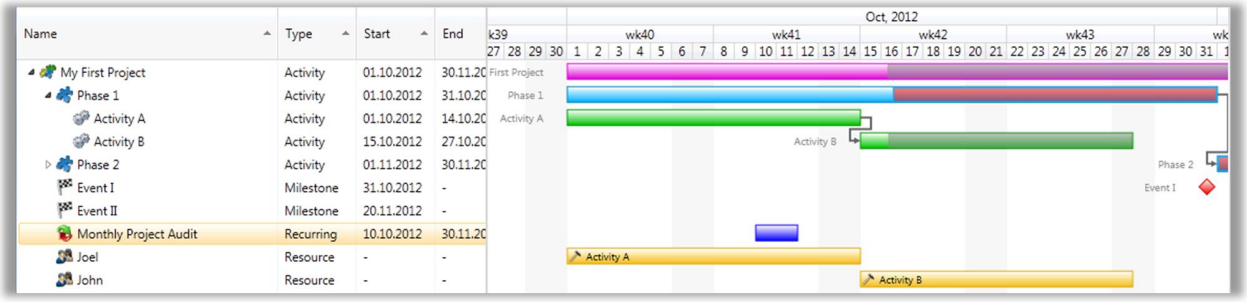
A new "Resource Allocation" object's metadata card will then appear, and our worker, Joel, as well as the correct start and stop dates will be pre-filled. Select the Activity you wish to allocate the worker to on these dates (in our case Activity A), and click OK.



Now that we've successfully created a resource allocation, we can configure how it will look. Besides standard options like color, we should consider the label position as well. Allocation objects are rendered in-line with their resource so labels positioned outside of the object can interfere with other resources that may be rendered. Let's move the label inside the allocation bars and add an icon while we're at it.

1. Select the newly created resource allocation item from the chart area
2. Navigate to the configuration section
   Vault: Resource Allocation (Object type): Options
3. Set Show Label = "Inside"
4. Navigate to the configuration section
   Vault: Resource Allocation (Object type): Properties
5. Set Icon = "(default)"

After setting a color, and allocating John to Activity B, the chart is updated accordingly:



6 Technical support

Please find the contact information for technical support and M-Files professional services at www.m-files.com/support.