# TTMSFNCGanttProject

The `TTMSFNCGanttProject` can be used for filling up the `TTMSFNCGanttChart` via the Project property.

This project can be used to work with different projects on the same instance of the `TTMSFNCGanttChart`.

In this control you build the structure of the project with the different tasks and connect them with the necessary relations. Configure the work time that is taken into account to set the start and end time of the tasks based on the day of the week, working hours and defined holidays.

!!! warning "Breaking Change Between v1.2.0 and v1.2.1"

```
The Task property **UseWorkingHours** has been changed to **WorkTimePolicy**.
By default this was set to True and this behavior remains unchanged with the new
property.
However, if UseWorkingHours was previously set to **False**, this setting was
stored in the project file and will now cause an error when attempting to open
the project.
To resolve this issue, you can safely ignore the error and set the
WorkTimePolicy property of the task to **whAllTime** to replicate the original
behavior.
```

## Properties

| **Property name** | **Description** |
|---|---|
| ProjectName | The name of the project. |
| ProjectId | A unique identifier for the project. |
| Description | A description of the project. |
| DefaultPriorityLevel | The default priority level assigned to new tasks (default = 2). |
| `States` | Collection of available task states in the project (e.g. Not Started, In Progress, Completed). |
| `WorkTime` | Defines working hours and workdays for the project. |
| `Tasks` | The collection of tasks belonging to the project. |
| Options | Defines configuration options for the project. |

## Methods

| **Method name** | **Description** |
|---|---|
| Create | Constructor. Initializes a new instance of the project. |
| Destroy | Destructor. Cleans up the project and its resources. |
| ClearTasks | Removes all tasks from the project. |

| Method name | Description |
|---|---|
| CalculateTasks | Recalculates task dates, durations, and dependencies. |
| SetSampleData | Fills the project with sample tasks and data for demonstration purposes. |
| Assign | Copies the content of another project instance into this one. |
| BeginUpdate | Starts an update block. Prevents immediate refresh until `EndUpdate` is called. |
| EndUpdate | Ends an update block and applies changes since `BeginUpdate`. |
| DoOnWorkTimeChanged | Executes when the project work time changes. |
| DoOnStatesChanged | Executes when the task states collection changes. |
| GetNextTask | Returns the next task after a given task. If `ASameParent = True`, only considers siblings under the same parent. |
| GetPreviousTask | Returns the previous task before a given task. If `ASameParent = True`, only considers siblings under the same parent. |
| AddTask | Adds a new task to the project. Parameters include `Name`, `Description`, `StartDate`, `Duration`, `WorkTimePolicy`, `Progress`, and `State`. |
| InsertTask | Inserts a new task at a specific WBS (Work Breakdown Structure) position. Similar parameters as `AddTask`. |
| GetTasksCount | Returns the total number of tasks in the project. |
| SetDefaultStates | Resets task states to default values (e.g. Not Started, In Progress, Completed). |
| GetTaskByWBS | Returns a task by its WBS identifier. |
| GetTaskByInternalId | Returns a task by its internal system-generated ID. |
| GetTaskByTaskId | Returns a task by its assigned Task ID. |
| GetTaskByTaskName | Returns a task by its name. |
| GetTaskByCount | Returns a task at a specific index (zero-based). |
| GetTaskCountByTask | Returns the number of child tasks for a given task. |
| DeleteTaskByTaskId | Deletes a task identified by its Task ID. Returns `True` if successful. |
| DeleteTask | Deletes the specified task object. Returns `True` if successful. |

| Method name | Description |
|---|---|
| Sort | Sorts the tasks in the project based on a property. **Parameters:** • `AProperty` *(string, default `'wbs'`).* • `ARecurse` *(Boolean, default `False`)* — If `True`, sorts recursively into subtasks. • `ACaseSensitive` *(Boolean, default `True`)* — Case-sensitive sorting if `True`. • `AAscending` *(Boolean, default `True`)* — Sort ascending (`True`) or descending (`False`). |

For the property name of the Sort method the same values can be used as for the Column Name: *wbs, taskname, taskdescription, startdate, enddate, duration, statename, progress, cost, dependencies, datastring, datainteger, databoolean.*

### Events

| Event name | Description |
|---|---|
| OnChanged | Event that is triggered when the tasks, states or worktime of the project is changed. |

## TTMSFNCGanttTaskState

The task states are the different milestones of a task in a project that can be defined.

You can retrieve the states on the collection level in 2 ways:

- GetStateByName
- GetStateByPriorityLevel

### Properties

| Property name | Description |
|---|---|
| Fill | The Fill settings of how the tasks in this state are drawn in the `TTMSFNCGanttChart` timeline. |
| Font | The Font settings how the name of the tasks in this state are drawn in the `TTMSFNCGanttChart` timeline. |
| Name | The name of the state, which is used to retrieve the correct information of the state. |
| PriorityLevel | The importance of the state to the project. A high number will get priority to set the state of the parent task. |
| ProgressFill | The Fill settings how the progress of the tasks in this state are drawn in the |

| Property name | Description |
|---|---|
| | TTMSFNCGanttChart timeline. |
| ProgressStroke | The Stroke settings how the progress of the tasks in this state are drawn in the TTMSFNCGanttChart timeline. |
| Stroke | The Stroke settings how the tasks in this state are drawn in the TTMSFNCGanttChart timeline. |
| Tag | Can be used as preferred, is not used in the internal code. (Default value is 0.) |

## TTMSFNCGanttTask

The task is the most important part of the Gantt Chart. The tasks can be subdivided in different other tasks, if that is the case we will refer to this as the Parent Task with its SubTasks.

### Methods

| Method name | Description |
|---|---|
| CountSubTasks | Returns the number of all the subtasks. |
| SetTaskStateByName | Sets the state of the task based on the name. |
| AddSubTask | Adds a sub task to this task. |
| AddNextTask | Adds a task which will start the moment this task ends. (This is just the planned start, and does not create a dependency.) |
| AddNextTaskPrevious | Adds a task which will end the moment this task starts. (This is just the planned start, and does not create a dependency.) |
| AddNextTaskSameEndDate | Adds a task which will end the moment this task ends. (This is just the planned start, and does not create a dependency.) |
| AddNextTaskSameStartDate | Adds a task which will start the moment this task starts. (This is just the planned start, and does not create a dependency.) |
| AddDependentTask | Adds a task which will start the moment this task starts. (This is just the planned start, and does not create a dependency.) |
| AddSubEventMarker | Adds an event marker as child to this task. |
| AddNextEventMarker | Adds an event marker which will start the moment this task ends. (This is just the planned start, and does not create a dependency.) |
| AddNextEventMarkerPrevious | Adds an event marker which will end the moment this task starts. (This is just the planned start, and does not create a dependency.) |

| Method name | Description |
| --- | --- |
| AddNextEventMarkerSameEndDate | Adds an event marker which will end the moment this task ends. (This is just the planned start, and does not create a dependency.) |
| AddNextEventMarkerSameStartDate | Adds an event marker which will start the moment this task starts. (This is just the planned start, and does not create a dependency.) |
| AddDependentEventMarker | Adds an event marker which will start the moment this task starts. (This is just the planned start, and does not create a dependency.) |
| MoveTaskTime | Moves the task over the defined period. |

## Properties

| Property name | Description |
| --- | --- |
| Cost | The predicted or final cost to complete the task. |
| Description | Detailed information of the task. |
| IsEventMarker | Indicates if the task is an event. |
| Locked | Indicates wether the start and end date can be changed on this task. |
| Name | The name of the task. |
| PlannedDuration | The `Duration` duration of the task not taking into account the work time. |
| PlannedStart | The moment the task should start. |
| Progress | The progress on the task, number between 0 and 100. |
| ScheduledEnd | When the task should and based on the PlannedStart and PlannedDuration where the current work time and dependencies are taken into account. |
| ScheduledStart | When the task should and based on the current work time and dependencies. |
| SubTasks | The different tasks into which this task is divided. This makes this task a Parent Task. |
| TaskAppearance | The appearance properties of the task. |
| TaskDependencies | The `Dependencies` define the relationships to other tasks. |
| TaskId | An ID that can be given to the task. |
| WorkTimePolicy | Specifies how working hours and working days are considered when calculating time-related operations. Possible values are WorkTimeAndDays, WorkTimeOnly (neglects |

| Property name | Description |
|---|---|
| | working days), WorkDaysOnly (neglects the working hours) and AllTime. By default the value is whWorkTimeAndDays and will use the configured working hours and work days. |
| Hint | The text that will be shown when the ShowHint property of the parent control is active and the text is not empty. |
| Tag | Can be used as preferred, is not used in the internal code. (Default value is 0.) |
| WBS | The Work Breakdown Structure show the position in the project based on its parent tasks. |
| WBSPart | Defines a custom part of the Work Breakdown Structure (WBS) for the task. This value is appended to the automatically generated WBS hierarchy, allowing you to extend or customize the task's WBS code. *By default the Index of the item is set*. |

## TTMSFNCGanttWorkTime

The moments that should be taken into account to calculate the begin and end of tasks and to show in the timeline of the `TTMSFNCGanttChart`.

### Properties

| Property name | Description |
|---|---|
| Holidays | Collection of `Holidays`. |
| WorkingDays | Set of days of the week on which the project progresses. |
| WorkingHours | Collection of TTMSFNCGanttTimeSpans with a start and end time. |

## TTMSFNCGanttHoliday

The days and moments in a year that the project isn't worked on.

### Properties

| Property name | Description |
|---|---|
| EndDate | When the holiday ends. |
| Name | The name of the holiday. |
| StartDate | When the holiday starts. |
| Teg | Can be used as preferred, is not used in the |

| Property name | Description |
|---|---|
| | internal code. (Default value is 0.) |
| UseTime | Indicates whetheher the time of the start and end date should be taken into account. |

## TTMSFNCGanttDuration

A timespan used int `TTMSFNCGanttChart` that is used for the task duration and delay on dependencies.

### Properties

| Property name | Description |
|---|---|
| DurationType | The unit of the duration. Can be seconds, minutes, hours or work days. |
| Value | Numerical value of the duration (Double). |
| Tag | Can be used as preferred, is not used in the internal code. (Default value is 0.) |

## TTMSFNCGanttTaskDependency

Used to define relationships between different `Tasks`.

### Properties

| Property name | Description |
|---|---|
| Delay | The `Duration` that indicates how long the task should be delayed. |
| DependencyType | How the task relates to the other. Can be Start-to-Finish, Finish-to-Start, Finish-to-Finish, Start-to-Start. |
| DependsOnWBS | The WBS value of the task it depends on. (Is not updated automatically when tasks are moved or deleted.) |
| Tag | Can be used as preferred, is not used in the internal code. (Default value is 0.) |

## Code Example

```
procedure TTMSFNCGanttProject.SetSampleData;
var
  dt: TDateTime;
  t, st, st2: TTMSFNCGanttTask;
begin
  BeginUpdate;
```

```
    SetDefaultStates;
    WorkTime.SetSampleData;

    dt := IncDay(Now, -9);
    t := AddTask('Assessment', 'First checks to see if project can start');
    st := t.AddSubTask('Files Review', 'Check if the customers checklist and
files were retrieved and filled out correctly.', EncodeDateTime(YearOf(dt),
MonthOf(dt), DayOf(dt), HourOf(WorkTime.WorkDayStart),
MinuteOf(WorkTime.WorkDayStart), SecondOf(WorkTime.WorkDayStart), 0), 2,
gdtWorkDays, whWorkTimeAndDays, 100, States[4]);
    st2 := st.AddDependentTask('Map Site', 'Plan out the different pages needed
for the site.', gtdFinishToStart, 3, gdtWorkDays, 0, gdtSeconds,
whWorkTimeAndDays, 20, States[3]);

    t := AddTask('Design', 'Start visual design of site.');
    dt := IncDay(Now, -4);
    st := t.AddSubTask('Mockups', 'Create the mockups for the style of the
site.', EncodeDateTime(YearOf(dt), MonthOf(dt), DayOf(dt),
HourOf(WorkTime.WorkDayStart), MinuteOf(WorkTime.WorkDayStart),
SecondOf(WorkTime.WorkDayStart), 0), 3, gdtWorkDays, whWorkTimeAndDays, 0,
States[2]);
    st.TaskDependencies.AddDependency(st2,gtdFinishToStart,1, gdtWorkDays);

    st.AddDependentTask('Slice and Code', 'Create blocks of the necessary visual
code.', gtdStartToStart, 5, gdtWorkDays, 1, gdtWorkDays, whWorkTimeAndDays, 0,
States[1]);

    t := AddTask('Development', 'Start visual design of site.');
    dt := IncDay(Now, +5);
    st := t.AddSubTask('Page Templates', 'Develop the different pages.',
EncodeDateTime(YearOf(dt), MonthOf(dt), DayOf(dt),
HourOf(WorkTime.WorkDayStart), MinuteOf(WorkTime.WorkDayStart),
SecondOf(WorkTime.WorkDayStart), 0), 5, gdtWorkDays, whWorkTimeAndDays, 0,
States[0]);
    st2 := st.AddDependentTask('Scripts', 'Program all the scripts.',
gtdFinishToFinish, 4, gdtWorkDays, 0, gdtSeconds, whWorkTimeAndDays, 0,
States[0]);
    st2.AddDependentTask('Testing', 'Test for all use cases.', gtdFinishToStart,
2, gdtWorkDays, 1, gdtWorkDays, whWorkTimeAndDays, 0, States[0]);
  EndUpdate;
end;
```