



Starting and Working with the
featherlite Planning Rich Client Reference
Implementation

Author: Michael Gatto
Type: 5 minute tutorial
Date: 2011-02-18
Version: 0.1.0

Abstract

The goal of 5 minute tutorial is to describe how to start and how to work with the reference implementation of the *featherlite* planning client, and show what information is shown in it. By means of this tutorial, we'll also hint to the features of the *featherlite* platform.

Contents

1 Overview	1
2 Prerequisites	1
3 Starting the Server and the Reference Client	1
3.1 Starting it all up	1
3.2 Shutting it all down	2
4 Working with the Reference Client	2
4.1 Orders	2
4.2 Resources	4
4.3 Inspector View	5
4.4 Gantt Chart View	5
4.5 Violation View	7
5 Going Further	7

1 Overview

This short tutorial is meant to be completed in 5 minutes. Here, we show how to start the server side and the client side of *featherlite*'s planning reference implementation. Then, we plan several orders of a predefined model, and show what data can be viewed from the client.

In Planning mode, *featherlite* allows to plan complex business processes, which are intertwined: for instance, one could plan on how to build a bicycle by a given due date. We can configure *featherlite* such that to build a bicycle we require two rims, a frame, brake pads, cables and so on. We know that building the bicycle requires different steps (like first mounting the cables for the brakes and the brake pads to the frame, then mounting the front wheel, and so on). So, we can configure *featherlite* to plan for these actions in sequence, taking the time they take and the required resources into account (like the number of workers needed, which might enforce that only a given number of bicycles can be built in parallel), and of the raw material (brake pads, frames, and so on). We can configure *featherlite* in such a way that it will even spawn purchasing orders (which again take time) given that some resource is not available (e.g., we ran out of brake pads and must order some). So in planning mode, *featherlite* allows to build complex plans and schedules from many domains (from building a bicycle to, say, building a timetable for an exam session).

2 Prerequisites

To be able to complete this tutorial, you need to have downloaded the *featherlite* Planning Reference Implementation from the website, www.featherlite-framework.com. Keep in mind that as these are executable applications, you need to download the one specific to your platform (Windows, Linux or Apple, in the 32 or 64 bit version).

3 Starting the Server and the Reference Client

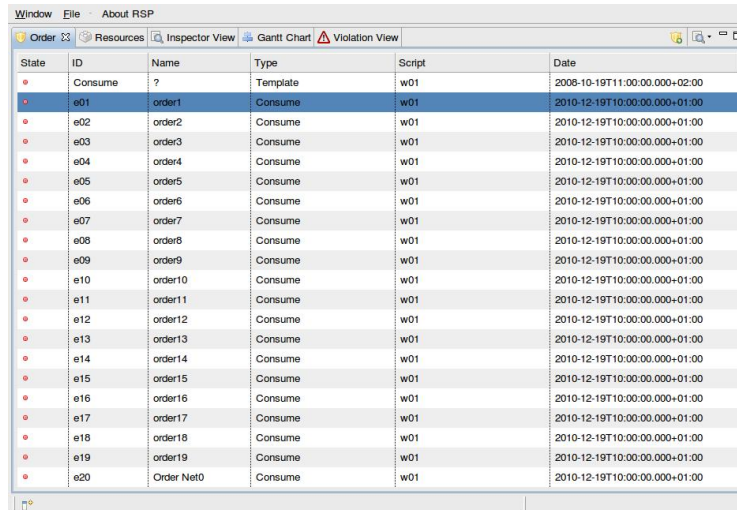
Let us start with the basics: starting and stopping the server and the client.

3.1 Starting it all up

We must first start the server. We do this by simply double-click on the executable file called "PlanningServer"¹. Depending on the platform, you may see a terminal opening and displaying the logging information of the server. If you cannot see any logging information, you can expect the server to be running within 5 to 15 seconds on current consumer hardware.

¹ If you have access to a terminal, you may also start the server from the command line.

To start the client, simply double-click on the executable file called “PlanningClient”. Note that the server needs to be running in order to ensure the correct communication with the server. While starting, the client displays a splash-screen. Once startup is complete, the user interface shown in Figure 1 appears.



State	ID	Name	Type	Script	Date
•	Consume	?	Template	w01	2008-10-19T11:00:00.000+02:00
•	e01	order1	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e02	order2	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e03	order3	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e04	order4	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e05	order5	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e06	order6	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e07	order7	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e08	order8	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e09	order9	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e10	order10	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e11	order11	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e12	order12	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e13	order13	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e14	order14	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e15	order15	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e16	order16	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e17	order17	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e18	order18	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e19	order19	Consume	w01	2010-12-19T10:00:00.000+01:00
•	e20	Order Net0	Consume	w01	2010-12-19T10:00:00.000+01:00

Figure 1: The reference planning client at startup.

3.2 Shutting it all down

To shut down the reference client, open the File menu, and select the “Shutdown server and client” menu entry. As is apparent from the menu description, this action shuts down both the server side and the client. In the dialog that appears you can confirm the shut down, or cancel the action.

4 Working with the Reference Client

Once we have the application running, we’ll proceed with analyzing an order, plan it, and see the repercussions of the planning steps on the resources affected by this order, and see a graphical representation of the planned orders.

4.1 Orders

The orders specify some type of action, as for instance producing an item. In this view, orders are shown row-wise; for each order, we can see its planning state as an icon, its id, its name and its type. Moreover, we can see what script is used to plan the order. The script defines what actions are undertaken at planning, and thus what resources are affected by the order (Does the order occupy some facility? Does the order use

some raw material?). Finally, each order specifies a date by which it should be completed. In this model, the orders will be scheduled in a just-in-time fashion, such as to be completed as near as possible to the specified date.

We have a look at the orders by selecting order *e01*, opening the context-menu using right mouse button click and selecting the “Show Order Detail” menu entry. The two tabs show information we have stored on the order, as shown in Figure 2. In this order, we

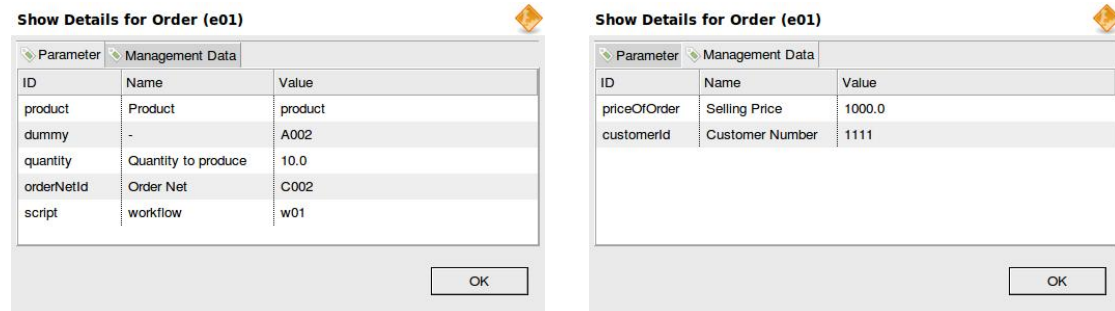


Figure 2: View of the details of an order. On the left, the view of the parameters. On the right, the view of the management data.

can for instance see what quantity is to be produced with this order (10.0 units). In the management data, we can see a customer number for this order, and a selling price. This information can be used in specific actions, for instance to calculate the total cost of fulfilling an order.

We now proceed planning this order. To this aim, select order *e01*, open the context menu by clicking on the order with the right mouse button, and select the “Plan Order” menu entry. A green check mark should now have appeared at the left of the order, in substitution of the red dot marker, as shown in Figure 3.

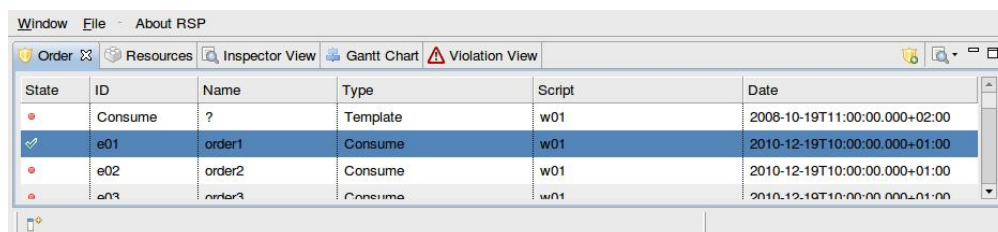


Figure 3: The order *e01*, which has now been planned, illustrated by the green check mark at the left of the order.

Now, select orders *e02* through *e07*, and plan them by opening the context menu on the selected orders, and selecting the “Plan Order” menu entry. A green check appears when the orders have been planned. We now proceed with determining what effect the planning of these orders has on other components of our model.

4.2 Resources

Switch to the “Resource” tab of the reference client. The view shown in Figure 4 appears.

ID	Name	Type	Planning Policy	Placement Policy
Product	Product	Template	DefaultPlanning	ExtendedProductionPlacement
Machine	Facility	Template	NoPlanning	ExtendedProductionPlacement
RawMaterial	RawMaterial	Template	NoPlanning	ExtendedProductionPlacement
raw1	Raw Material 1	RawMaterial	NoPlanning	ExtendedProductionPlacement
raw2	Raw Material 2	RawMaterial	NoPlanning	ExtendedProductionPlacement
raw3	Raw Material 3	RawMaterial	DefaultPlanning	ExtendedProductionPlacement
product	Product	Product	DefaultPlanning	ExtendedProductionPlacement
facility_01	Facility 1	Machine	NoPlanning	ExtendedProductionPlacement
facility_02	Facility 2	Machine	NoPlanning	ExtendedProductionPlacement
facility_03	Facility 3	Machine	NoPlanning	ExtendedProductionPlacement

Figure 4: The resource view of the *featherlite* Planning reference implementation.

ID	Name	Type	Order Net	State	Start	End
useFacility03	Use Facility 03	Use	e06	PLANNED	2010-12-17T08:00:00.000+01:00	2010-12-17T10:00:00.000+01:00
useFacility03	Use Facility 03	Use	e01	PLANNED	2010-12-17T13:00:00.000+01:00	2010-12-17T15:00:00.000+01:00
useFacility03	Use Facility 03	Use	e04	PLANNED	2010-12-17T10:00:00.000+01:00	2010-12-17T12:00:00.000+01:00
useFacility03	Use Facility 03	Use	e03	PLANNED	2010-12-17T11:00:00.000+01:00	2010-12-17T13:00:00.000+01:00
useFacility03	Use Facility 03	Use	e02	PLANNED	2010-12-17T12:00:00.000+01:00	2010-12-17T14:00:00.000+01:00
useFacility03	Use Facility 03	Use	e07	PLANNED	2010-12-16T15:00:00.000+01:00	2010-12-17T09:00:00.000+01:00
useFacility03	Use Facility 03	Use	e05	PLANNED	2010-12-17T09:00:00.000+01:00	2010-12-17T11:00:00.000+01:00

Figure 5: The view “Show Tasks” that opens through the context menu of the resource view.

The shown resources have a different types: there are “Templates”, “Product”, “RawMaterial” and “Machine” types. Select the resource with ID “facility03”. This resource represents the use of some machine during the production process of the orders we have planned. In this view, we can see (beyond the ID, name and type of the resource) what type of planning policy is used (here, “NoPlanning”, hence no particular planning policy) and what placement policy (here, the ExtendedProductionPlacement policy, which causes tasks to be extended over non-working periods defined by a work calendar). Open the context menu of “facility03”, and select the “Show Tasks” menu entry. In the opening view, shown in Figure 5, you can see the tasks that are “placed” on the “facility03” machine. Each row of the table contains one such task together with some associated information: for instance, we can see that task “useFacility03” of the order *e01* we planned has executed, on this facility, during a specific start and end time.

By opening the context menu “Show State Variables” we can see the occupancy of the facility03 (see Figure 6). Here, we can see the values varying over time.

Edit Values of State (facility_03)

ID	Name	Actual State	Date	Value
Capacity	Occupancy Fac	0.0	1970-01-01T01:00:00.000+01:00	0.0
			2010-12-16T15:00:00.000+01:00	1.0
			2010-12-17T08:00:00.000+01:00	2.0
			2010-12-17T14:00:00.000+01:00	1.0
			2010-12-17T15:00:00.000+01:00	0.0

OK

Figure 6: The state variable showing the usage of the facility03 resource. You can see that first the usage is zero, then it grows to one, then to two, back to one and finally, to zero again. This reflects the occupancy of the resource due to the planned orders.

4.3 Inspector View

The inspector view allows to browse through all objects instantiated at runtime. This view can be used to see all states of these objects, and is generally used by developers for debugging, or to see how to access the different fields of the object at hand. You can try and click around and see what data is displayed; however, an explanation of this view is beyond the scope of the 5 minutes.

4.4 Gantt Chart View

The Gantt chart is probably the most interesting view of the reference client. Here, you can see how the tasks of the different orders are scheduled on the resource. To make things clearer, return to the order view, select order *e06*, open the context menu, and select the “Mark Order Net” menu entry. Switch back to the Gantt chart view. You should now see some tasks in yellow, as shown in Figure 7.

This view shows each resource and associated state variables (e.g., Facility 1, followed by the Machine Occupancy state variable). The tasks associated with the order we selected (order *e06*) are shown marked yellow. Now, we can see that each such order spawns 8 tasks: three occupying the machines (Facility 1 through 3), a task producing some product (symbolized by the upward-slope triangle), two tasks consuming raw material 1 and 2, and two tasks affecting raw material 3, one producing it, and one consuming it.

The shown shaded area represents points in time when the facility cannot be occupied (as it is outside the work period of the facility). To see how this affects the planning and how the tasks are interconnected, select the leftmost task placed on Facility 01 by clicking on it with the left mouse button and, while keeping the mouse button pressed, drag the task close to (but not within) the shaded area at the left. A configuration similar to the one shown in Figure 8 should result. Here, you can see that due to the planning policy, the related task on facility two now spans over the night period, whereas the task on Facility 3 starts and ends before the rest period (the shaded area). These results are due to the placement policy which is used to plan these tasks (the ExtendedProductionPlacement policy).

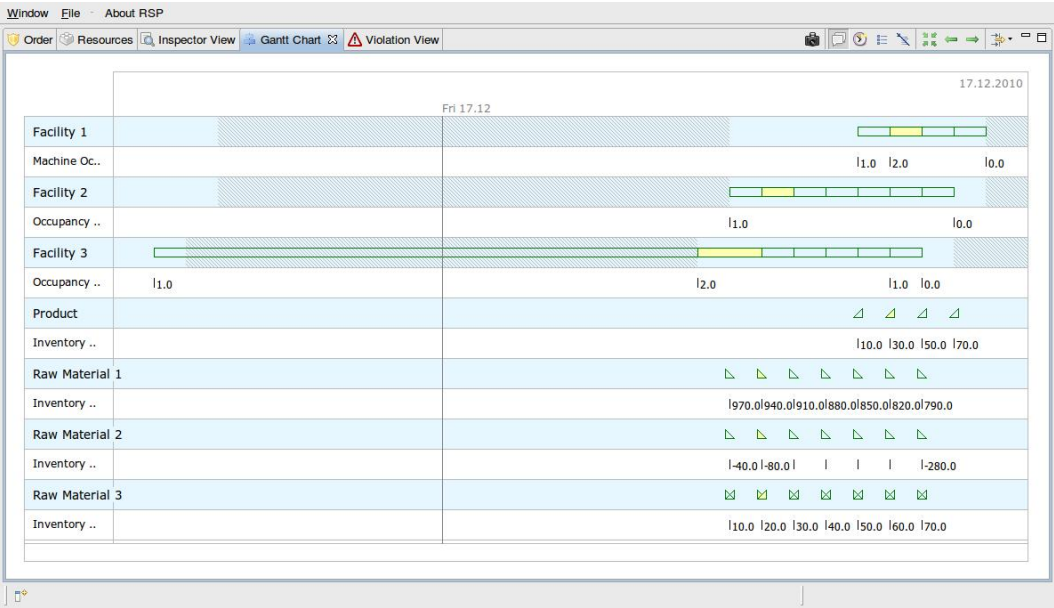


Figure 7: The Gantt chart view, with order e06 marked.

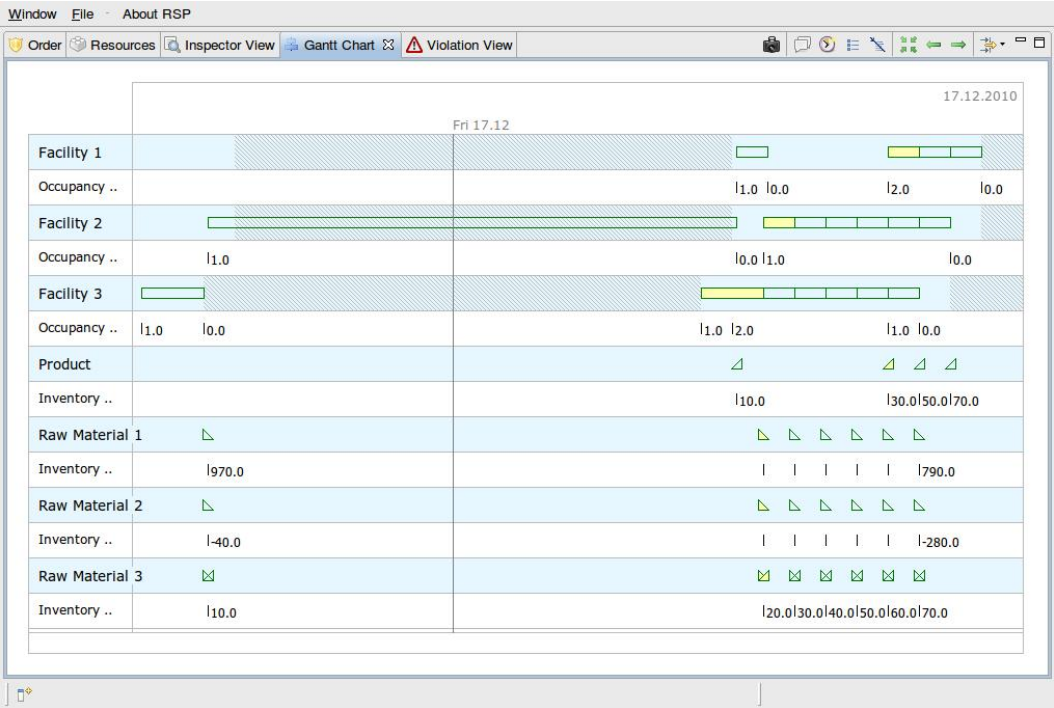


Figure 8: The resulting configuration after the leftmost task on Facility 1 has been dragged close to the shaded area.

ID	Type	Start	End
raw2	CapacityViolation	2010-12-16T16:11:11.707+01:00	2010-12-17T10:00:00.000+01:00
raw2	CapacityViolation	2010-12-17T10:00:00.000+01:00	2010-12-17T11:00:00.000+01:00
raw2	CapacityViolation	2010-12-17T11:00:00.000+01:00	2010-12-17T12:00:00.000+01:00
raw2	CapacityViolation	2010-12-17T12:00:00.000+01:00	2010-12-17T13:00:00.000+01:00
raw2	CapacityViolation	2010-12-17T13:00:00.000+01:00	2010-12-17T14:00:00.000+01:00
raw2	CapacityViolation	2010-12-17T14:00:00.000+01:00	2010-12-17T15:00:00.000+01:00
raw2	CapacityViolation	2010-12-17T15:00:00.000+01:00	-

Figure 9: The violation view, showing the violations on the capacity of the resource *raw2*.

You can now shift some other tasks around and see the effects on the Gantt chart. Note that you cannot shift tasks later in time because of their due date. If you do, an error message appears.

4.5 Violation View

The violation view serves to see when constraints set on resources have been violated. *featherlite* has two general mechanisms for constraint violation, specified by the placement policy used for the resource: hard constraints prevent any task to be planned if the constraint is violated (if, e.g., a facility can handle only one task at a time, and we attempt to plan two tasks at the same time); soft constraints do not prevent a task to be planned, but violations to these bounds can be checked. Here (see Figure 9), the constraint of raw material “*raw2*” has several violations, specified over time intervals. If we look at the details of one of these entries by selecting it, opening its context menu with a right-button mouse click, and selecting the “Show Details” menu entry, we see the view shown in Figure 10. Here, (we chose the second entry of the table) we can see that the capacity was violated by going negative by 80 units, whereas the constraint (which is not enforced) requires the inventory to be between zero and 10^7 units.

5 Going Further

In this tutorial, we used the reference planning client and used some basic features of the *featherlite* platform, without explaining too much what happens behind the scenes.

We looked at standard orders, which have an associated script to update and occupy the resources it affects. *featherlite* allows to define many different types of orders, as



Parameter

ID	Name	Value
min	Minimum	0.0
max	Maximum	1.0E7
value	Value	-80.0

OK

Figure 10: The violation view, showing the violations on the capacity of the resource *raw2*.

of your needs. These orders can be individually configured with parameters and other data at runtime. Every order can be configured to use a different business logic. So, you can for instance create orders for some raw materials, some production process for an item, and orders using complex business logic which you can define and implement yourself, and configure the *featherlite* platform to use.

We also saw several different types of resources. The resources are again objects that can be configured at runtime with policies and other standard data. The *featherlite* framework takes care of the other aspects, like persisting them to a database without any user interaction, using functions of other available *featherlite* plug-ins.

The effect we saw when shifting the tasks in the Gantt chart was due to a so-called placement policy. Several standard predefined placement policies are available in the *featherlite* framework. Moreover, you can add your own and configure *featherlite* to use them in a very simple fashion. You might need to associate further actions as a result of planning (like sending an email to notify the necessity of buying some raw material). This business logic can also be executed through policies, and some standard functions (like sending encrypted emails) are available in the *featherlite* Utils plug-in. Of course, the standard planning functions are all available in the base plug-ins.

This tutorial also showed how a GUI can look like. This implementation serves as a reference for your customized interfaces.