

Using Alf for State Behaviors

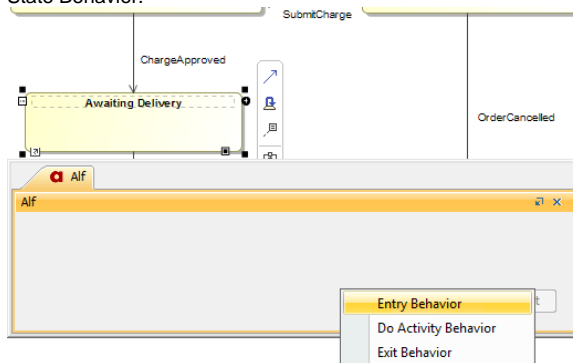
A State in a State Machine can have an entry Behavior (which is executed when the State is entered), an exit Behavior (which is executed when the State is exited) and/or a do-activity Behavior (which is executed asynchronously while the State is active). You can use the [Alf editor](#) to create an Alf body for, or edit the existing Alf body of, any one of these kinds of Behaviors.

Related pages

- [The Alf editor](#)
- [The Alf compiler](#)

To create an Alf body for a State behavior

1. Select the State (in the Model Browser or on a State Machine diagram) and open the Alf editor window (select **Windows > Alf**), if it isn't already open.
2. In the Alf editor window, press **Create**. A menu will pop up with available options for creating a State Behavior.



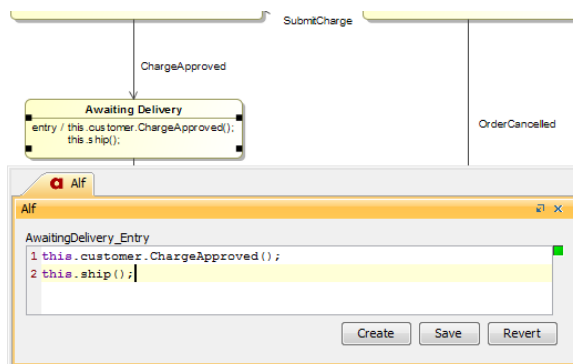
3. Select **Entry Behavior**, **Do Activity Behavior** or **Exit Behavior** as appropriate.
4. Enter the Alf code for the Behavior and press **Save** to compile and save it.

i The Alf code for a State Behavior can access attribute values of the context Classifier of the State Machine containing its State (using Alf *this* expressions). It can also [access the data](#) in the Event occurrence that triggered its execution (e.g., the attribute data of a received Signal).

To edit the Alf code for a State Behavior

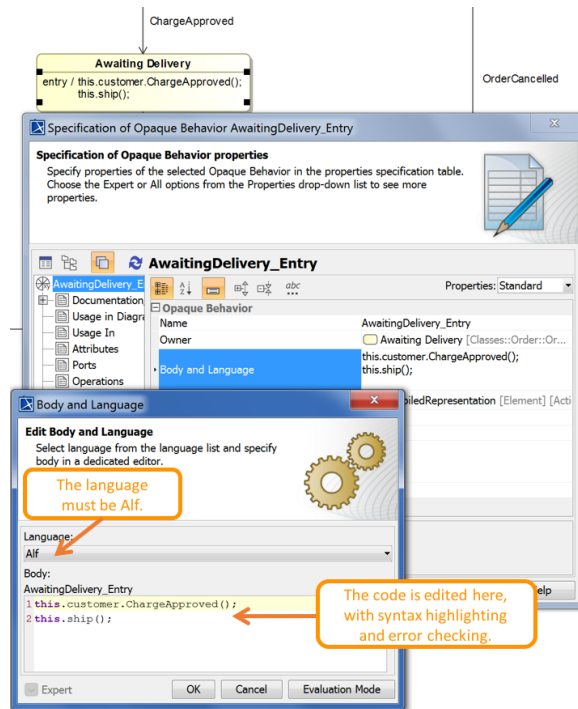
- Select the Behavior (in the Model Browser or as represented on a State Machine diagram) and open the Alf editor window (select **Windows > Alf**), if it isn't already open. The Alf code will appear in the window.

! To edit a State Behavior on a State Machine diagram, be sure to select the specific Behavior that you want to edit (entry, do or exit), and not on the State itself



i Note that, even while you are editing an existing State Behavior, the **Create** button stays active if there are remaining State Behaviors that could be created. For instance, while you are editing an entry Behavior, you can press **Create** in order to create an exit Behavior (if the State does not already have one).

If you create a State Behavior as an Opaque Behavior, then Alf code for it can also be entered or edited directly in the **Body and Language** property for it in the Specification window for the State. If you open the Edit Body and Language window, then the Alf code may be edited just as in the Alf editor window. However, instead of **Save** and **Revert** buttons, this window has **OK** and **Cancel** buttons, either of which will close it. If you press **OK**, then the text is saved, and, if it has no errors, compiled. If you press **Cancel**, the Alf text is *not* saved.



Editing Alf code for a State Behavior in an Opaque Behavior body



To be recognized as Alf code, the **Body** of an Opaque Behavior must have the **Language** Alf. If your project was created using the Alf project template, then Alf will be the default language. Otherwise, select Alf from the **Language** menu (you may have to scroll upwards in the menu to find the selection for *Alf*).



To make Alf the default language for Opaque Behaviors

1. Select **Options > Project**.
2. On the left, under **Default model properties**, select **Opaque Behavior**.
3. Click on the **Language** field and then on the small + button on the right.
4. Enter *Alf* (spelled and capitalized in exactly that way) and click on **OK**.
5. Click on **OK** to close the options window.

If the **Opaque Behavior Display Mode** symbol property of a State is set to Body, then any State Behaviors that are Opaque Behaviors can also be edited directly on a State Machine diagram showing the State and its behaviors. Since such editing does not use the Alf editor, no parsing or constraint checking happens while Alf code is being edited. The code is compiled once you finish editing it, and, if there are compilation errors, they will be recorded in an error annotation on the Opaque Behavior.