

[Tutorials](#) > [Supply Chain GIS \(AB + DE\)](#)




## Phase 4. Sending orders from the retailer

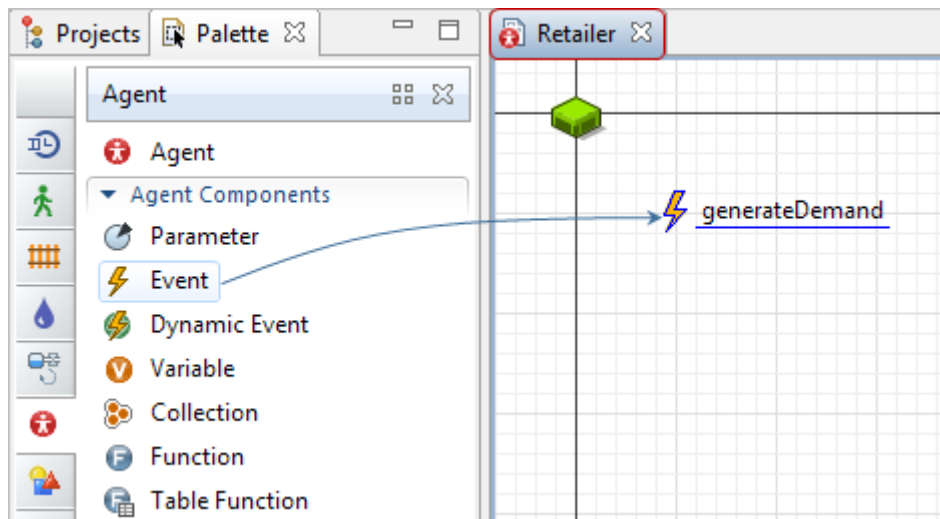
In this phase, we will model how any retailer can send a request to the distributor. We will tell a retailer how to form an order and then where to send it.

Retailers do not order the product delivery just once, and we want to model a supply chain that functions repeatedly.

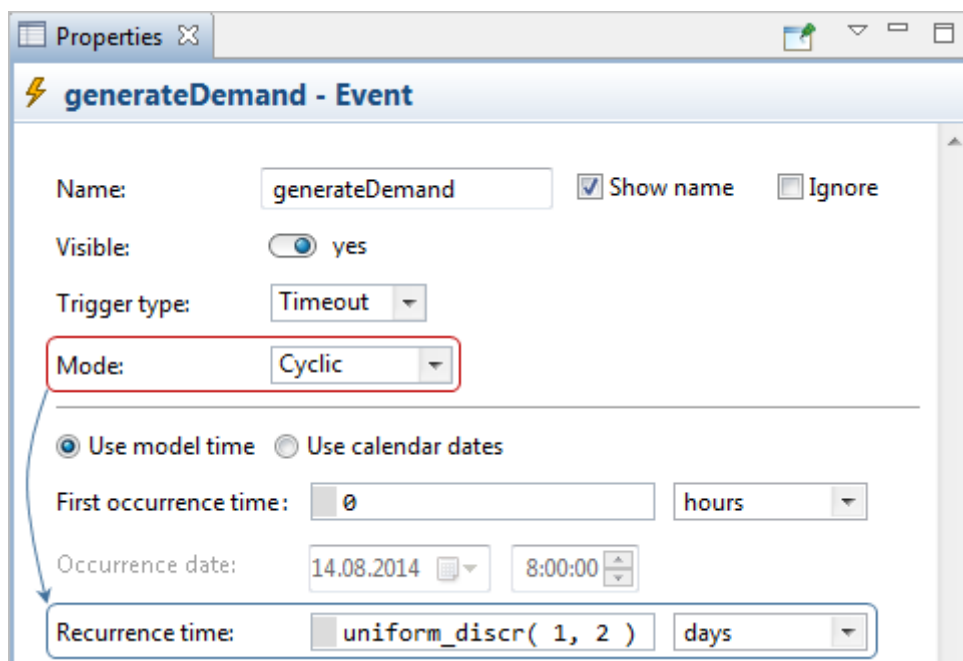
The best option to create a request like this is to use the **Event** object. The events in AnyLogic do not just happen, we can also assign to an event an action to perform.

### To create a request for the product delivery

1. In the **Projects** view, double-click the  **Retailer** agent type to open its diagram. There you will find the animation figure we have selected for it in the **New agent** wizard.
2. Drag the  **Event** object from the  **Agent** palette. Type its name in the built-in editor: `generateDemand`.




3. For instance, we want our distributors to order more product every other day. We do not need to create a new event for each time it happens, we will use the special event mode called **Cyclic**. Then, since the mode is cyclic, we need to specify the **Recurrence time**.

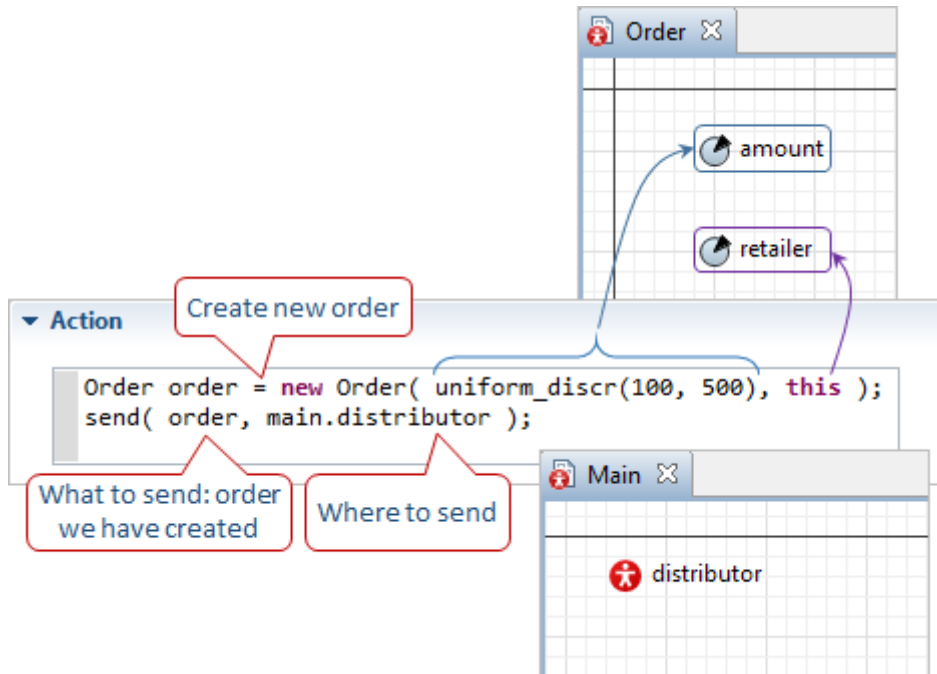


Here, we use one of the standard distributions that AnyLogic provides, [uniform\\_discr\(\)](#). See below for an easy way to define the probability distributions..

4. We have defined when the event should occur. Now let us tell AnyLogic what to do when the event occurs. Expand the **Action** properties section. It contains the edit box where we can type the code that will be called.

First of all, we create an order of a special kind by calling the constructor of this type of agent. We use the parameters we have created before for the  `Order` to define what kind of order it is.



Then, we send the order that has been created to our distribution center.

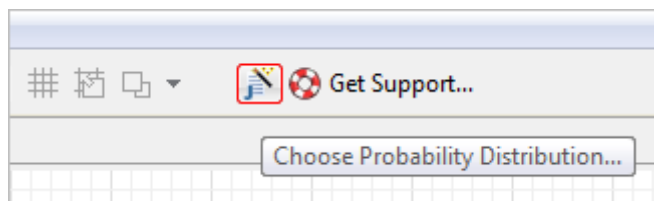


5. We have set up the  `Retailer` agent type logic.

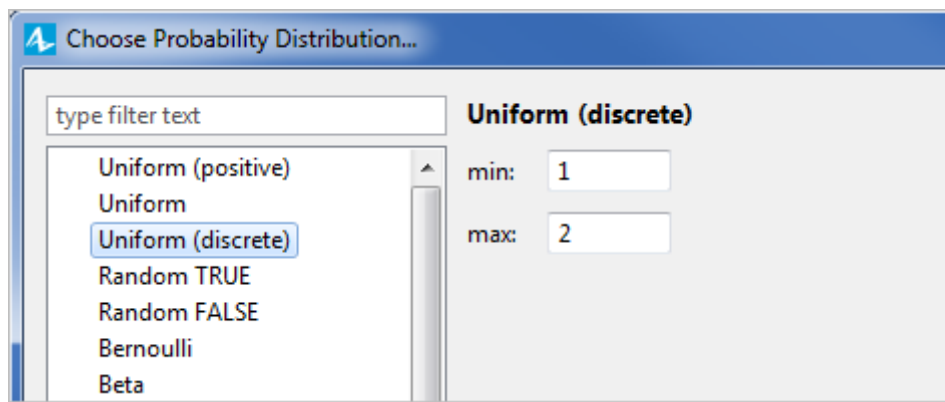
Different probability distribution functions can be very useful, but it may not be very convenient to write them. Sometimes you might want to learn more about the differences between them before you can choose which one to use. To make it more convenient, AnyLogic provides you a special code wizard, always available from the AnyLogic toolbar.

### To add a probability distribution function

1. Click in the edit box of some parameter where you want to add a probability distribution function.
2. Click the button  located on the AnyLogic toolbar next to the  **Get Support** button.







3. The wizard will pop up. Here you can select what function exactly you need, find the information about it, and define its arguments in an easy way.





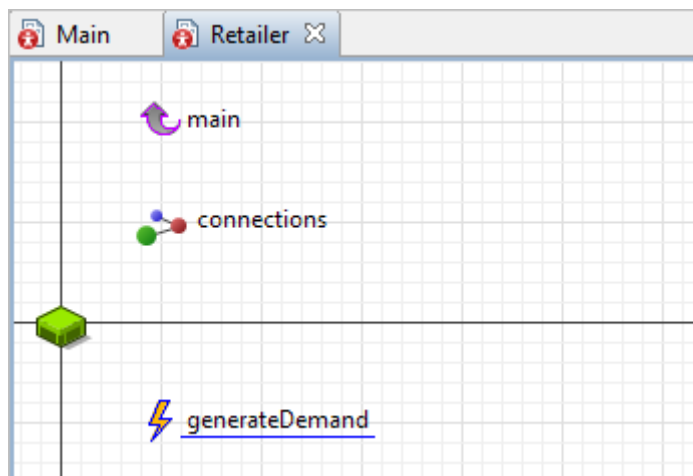
4. When you are finished, click **OK**, and AnyLogic will paste the created function into that edit box.

In Java, when you are writing code in, for instance, `objectA`, you should use a dot "." to access a method or a parameter of some other `objectB` after this object's name: `objectB.size()`;

In the code that we write to define the event's action, we need to access the  `distributor` agent that lives on  `Main` in our model while editing the  `Retailer` diagram.

You can think of the agent types as of some sort of containers, and to access internal elements of these containers, we provide you a special object called  **Link to upper level agent** that is present by default on every agent type's diagram that you create. It allows you to access the agent types like any other model object. The link object is located above the X axis, [move the diagram](#) down to find it.

In the figure below, you can also see that the  `generateDemand` event is highlighted in the blue color, as AnyLogic usually highlights those objects you select in the graphical editor, and the  `main` link is highlighted in the purple color: in this way AnyLogic highlights the objects that you refer to in the currently selected object.



Read more about accessing objects and their methods while writing the Java code in the article [Where am I and how do I get to...](#)

In the next phase, we will define how the distributor processes the order that it receives from a retailer.

**Reference model:** [Supply Chain GIS - Phase 4](#)

◀◀ [Phase 3. Placing agents in GIS space](#)

▶▶ [Phase 5. Processing orders at the distributors](#)