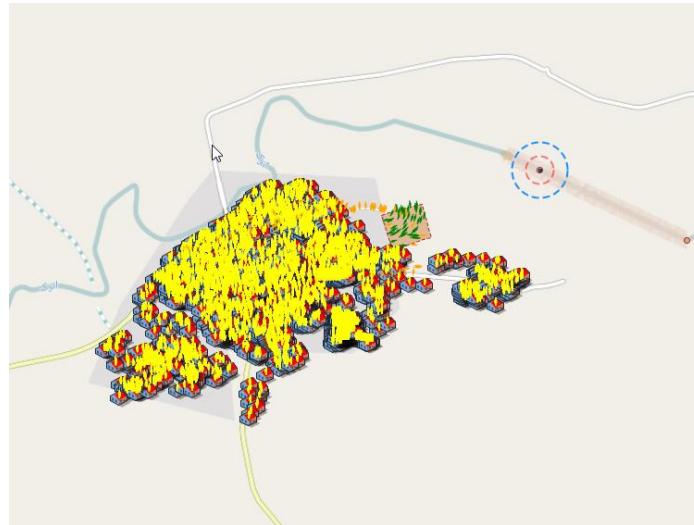


Disaster Simulation: An AnyLogic Agent-Based Approach

Example:
Flood simulation and evacuation
GIS Environment



Ali Asgary
ADERSIM
York University
2017

Lesson 5: Defining Agents' Behavior 3- Person agent

- Defining population of agents' behavior using statechart
- Emergency evacuation simulation.

Defining a simple statechart for sensor agent

We want people to receive the flood warning issued by the sensor and react to it by moving to evacuation zone if their houses are located in the flood zone.

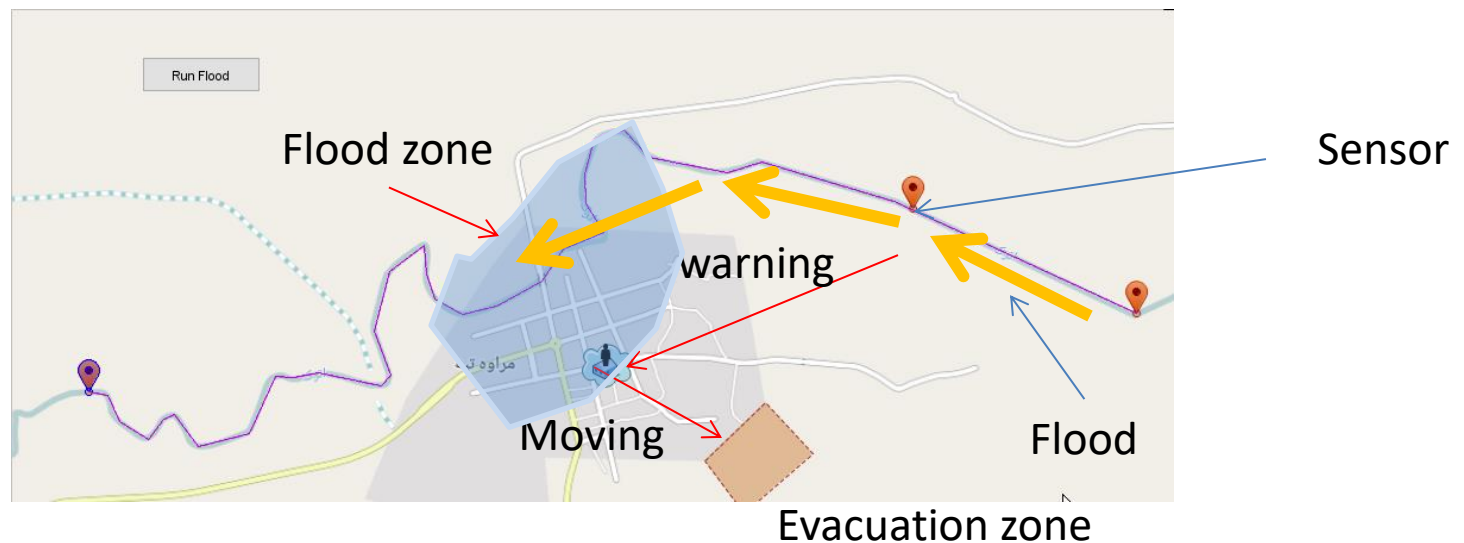
To do this we use statecharts to define different states of the Person agent.

We assume that Person has two main states:

Normal: when water level is below certain level (no flood).

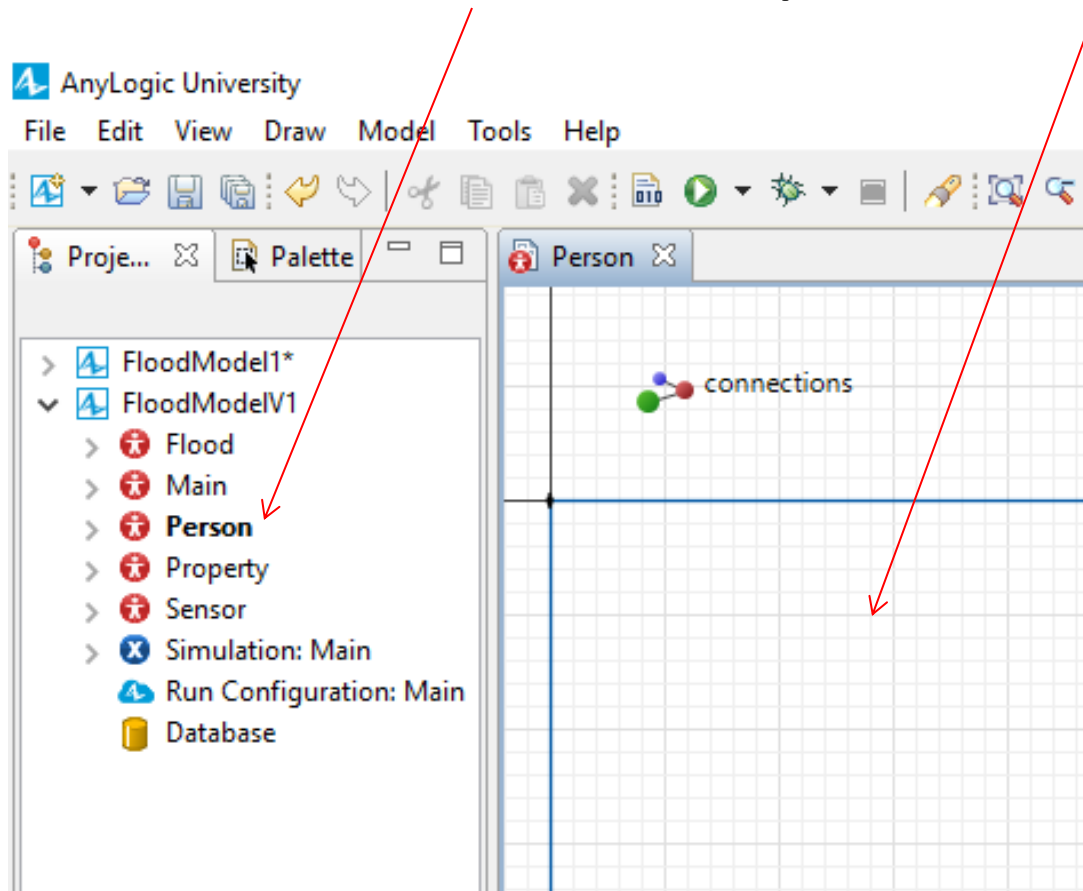
Flooding: when water level is above the normal level and flood warning is issued.

As soon as people receive the warning and their homes are located in the flood zone they start preparing and then moving to the pre designated evacuation zone. They will remain there till flood warning is lifted. When flood warning is lifted by the sensor, people who have taken shelter in the evacuation zone return to their homes.



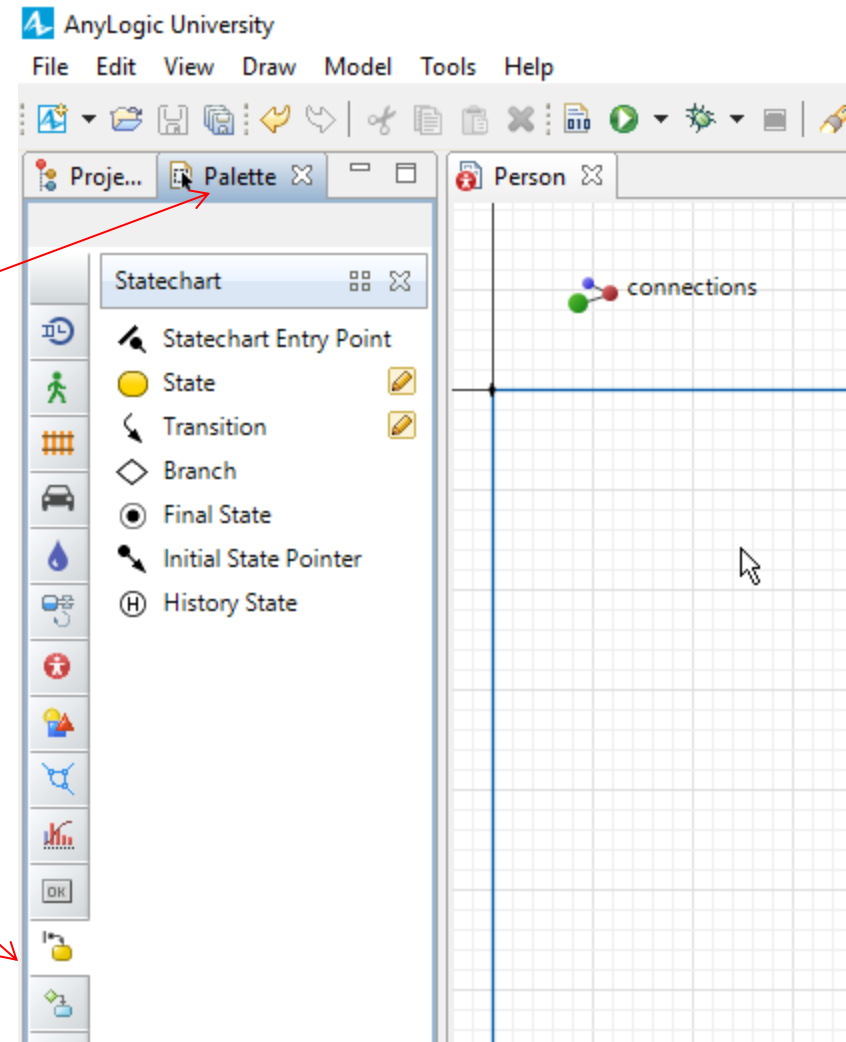
Open Person window

Double click on Person to open its window.



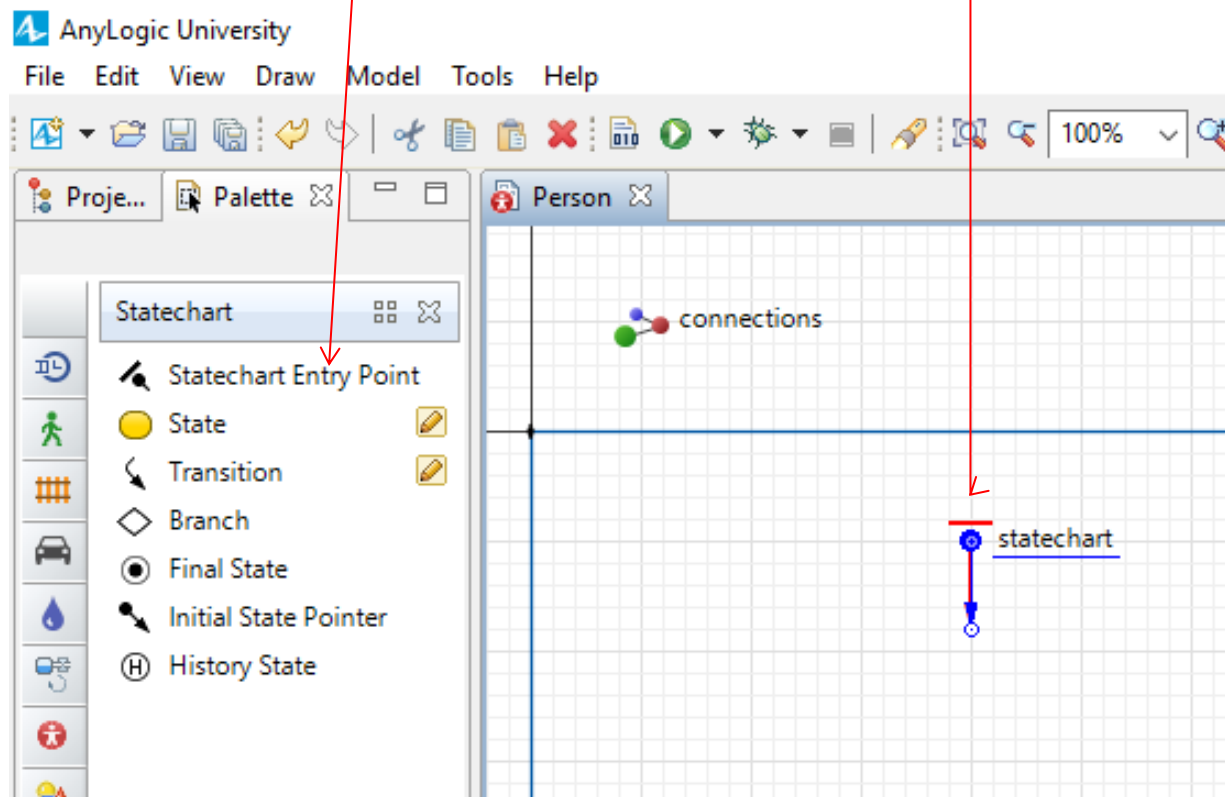
Expand Statechart tools

Click on the Palette tab and click on Statechart tools.



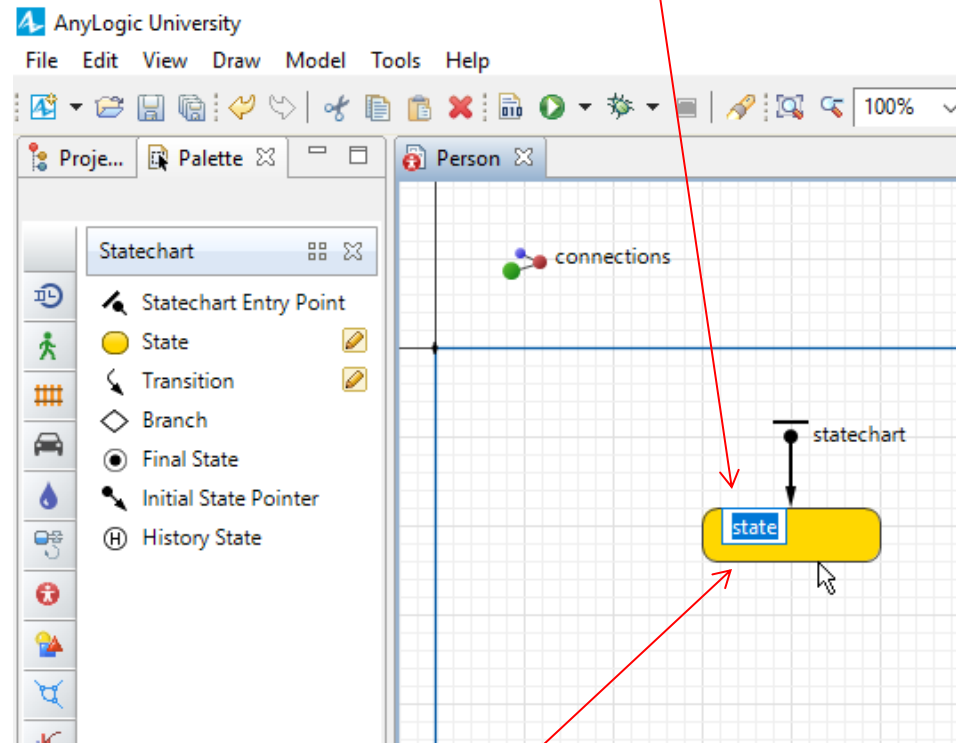
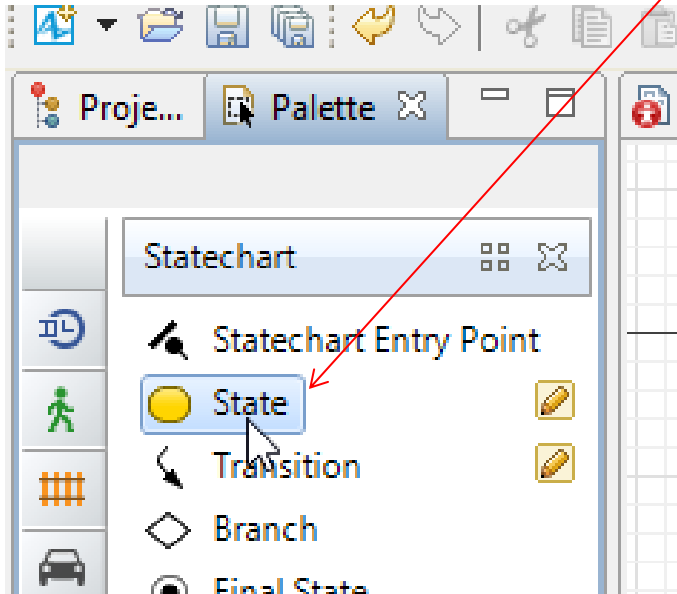
Creating statechart for Person

To create a statechart for Person you need to drag and drop a Statechart Entry Point to your Person window.



Adding states

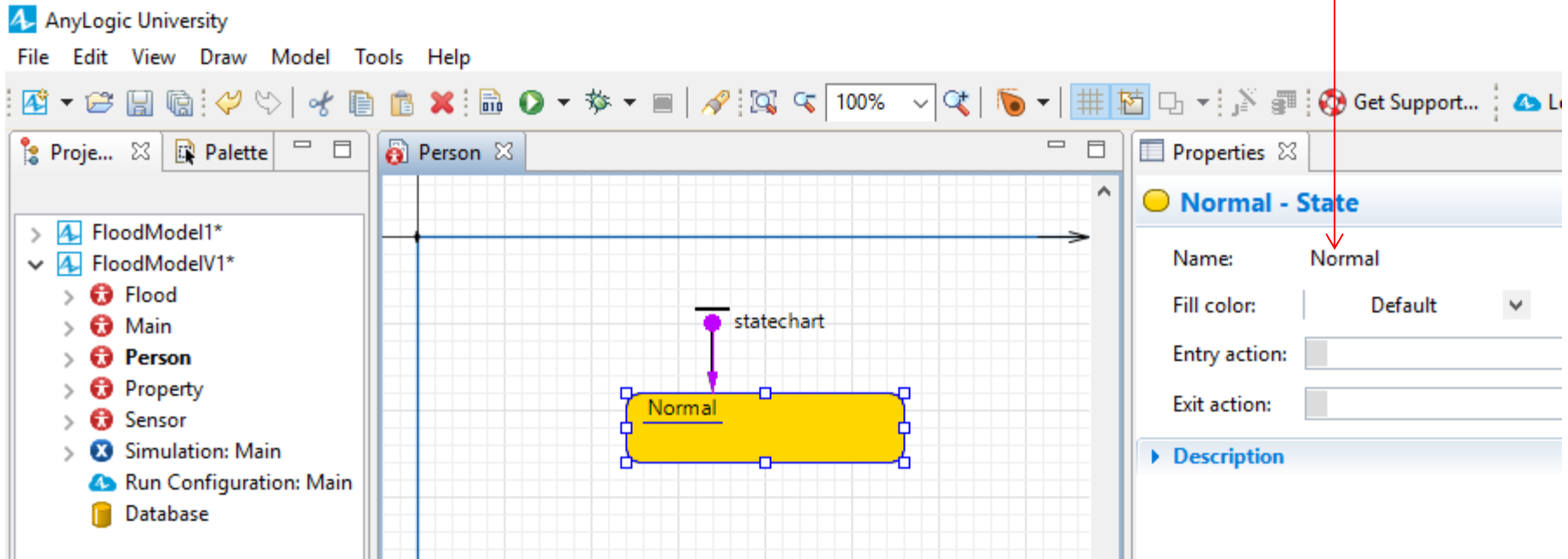
You can add a state by dragging and dropping a state from the Statechart tools to your Person space.



Make sure that the state is connected to the Statechart Entry Point

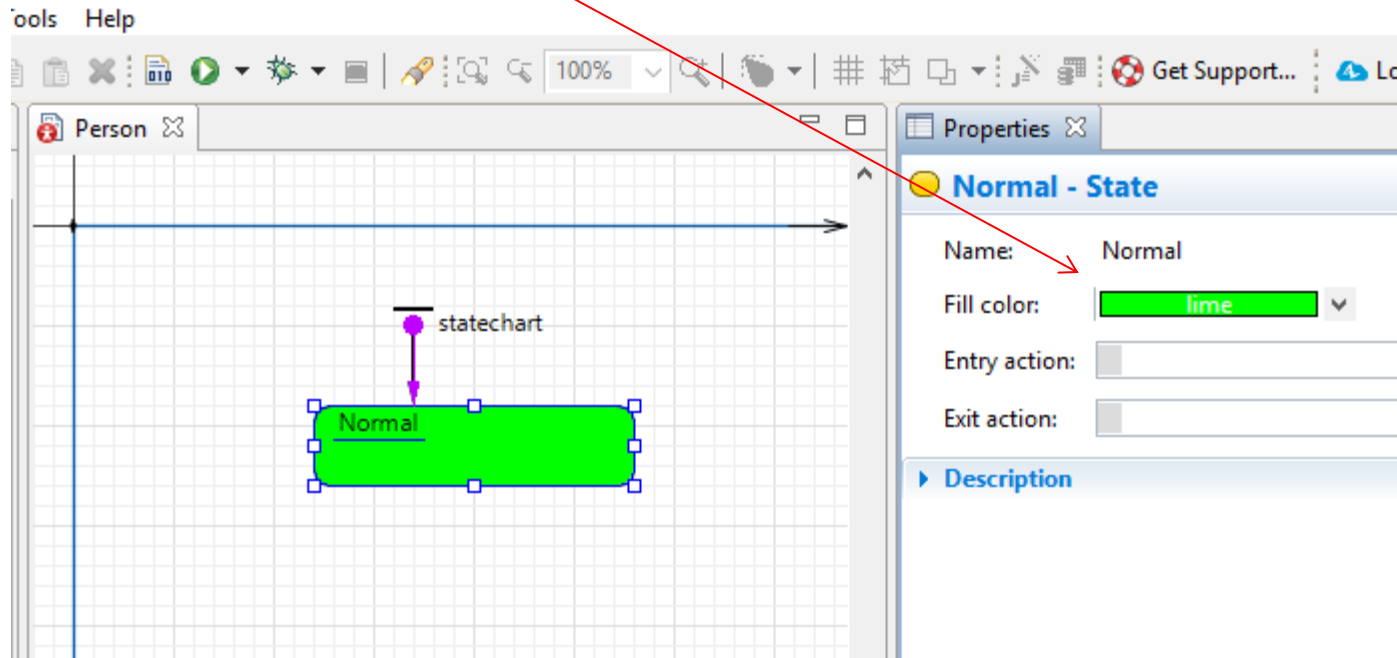
Change the state name to Normal

Click on the state that you just created and in the property window change its name to Normal



Change the state color

Using the state property window change the Fill color to lime.



Transition from Normal state to Flooding state

If Person receives flood warning from the Sensor its state changes from Normal to Flooding.

For this to happen we drag and drop a transition to the Normal state as shown:

The screenshot displays a statechart editor interface. On the left, a palette titled 'Statechart' contains various elements: Statechart Entry Point, State, Transition, Branch, Final State, Initial State Pointer, and History State. The 'Transition' element is highlighted. In the center, a statechart diagram shows a state named 'Normal' (a green rounded rectangle) with a transition arrow pointing to it from a 'statechart' entry point. A red arrow points from the 'Transition' element in the palette to the transition arrow on the statechart. On the right, a 'Properties' window for the selected 'transition - Transition' shows the following configuration:

- Name: transition
- Triggered by: Message
- Message type: String
- Fire transition: Unconditionally, On particular message, If expression is true
- Message: "floodWarning"
- Action: (empty field)
- Guard: (empty field)

At the bottom of the Properties window, there is a 'Description' section.

Define properties of the transition

Change the Name to: flooding

Change the Triggered by to: Message

Change Message type to: String

Select Fire transition: On particular message

For the Message write: "flood warning"

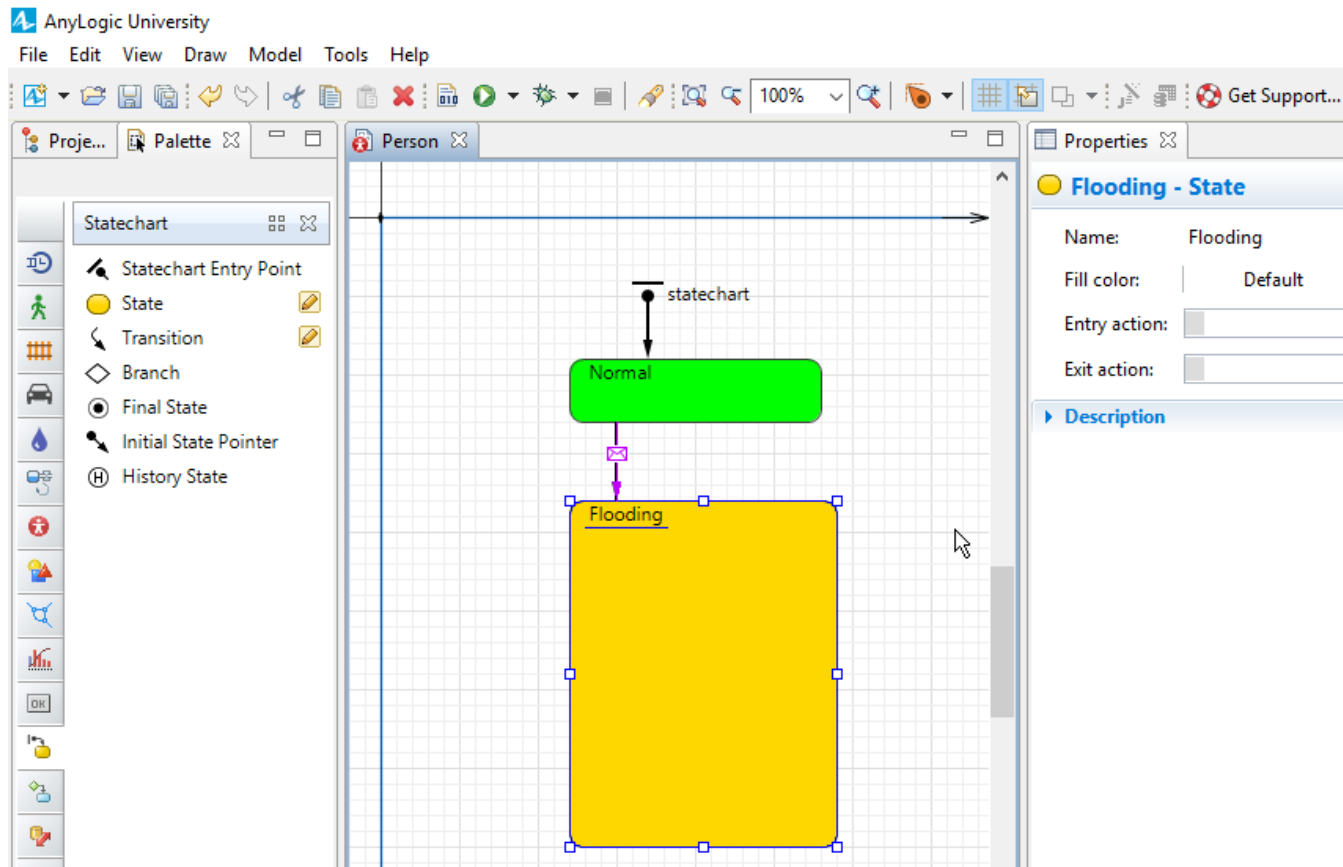
The screenshot displays a statechart editor interface. On the left, a statechart is visible with a state named 'Normal' (highlighted in green) and a transition triggered by a message icon. On the right, the 'Properties' window for the 'flooding - Transition' is open, showing the following configuration:

- Name: flooding
- Triggered by: Message
- Message type: String
- Fire transitions: On particular message
- Message: "floodWarning"
- Action: (empty)
- Guard: (empty)

Red arrows from the text above point to these specific settings in the Properties window.

Add the Flooding state

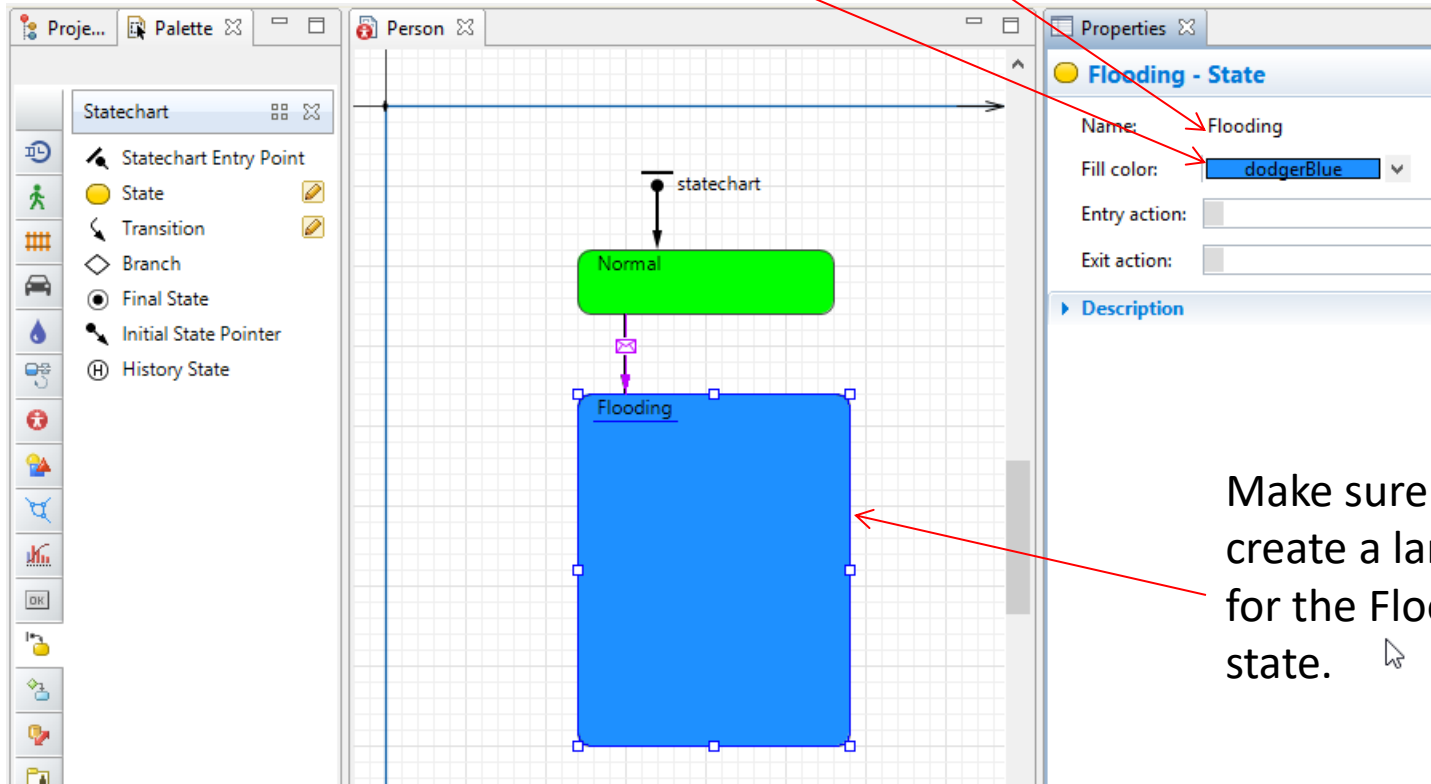
Drag another state from the Statechart tools and drop it into the Person window and attach it to the flooding transition as shown



Change the properties of the Flooding state

Change the name to: Flooding

Change the Fill Color to blue



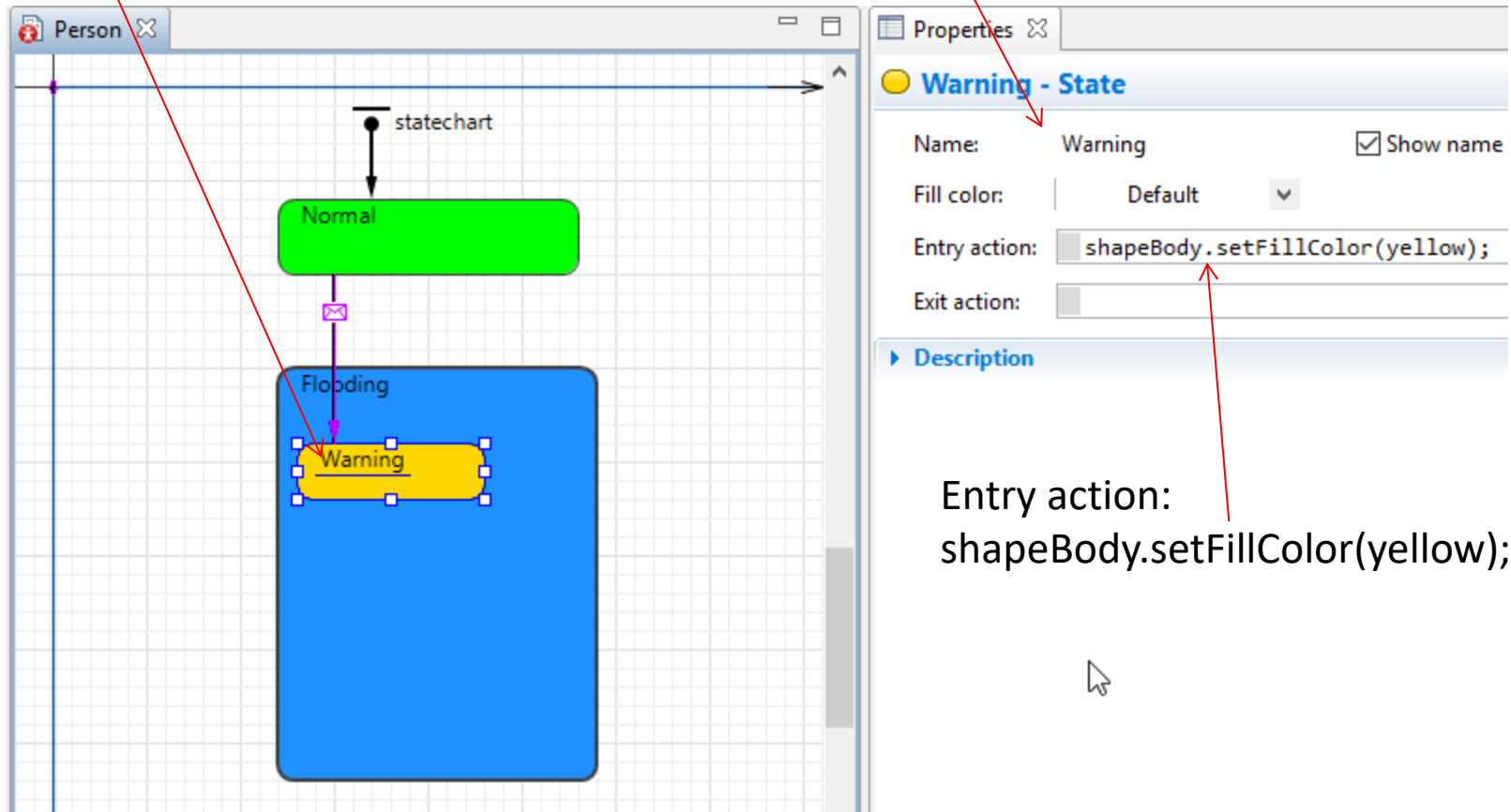
Make sure that you create a large area for the Flooding state.

Adding Warning, Evacuating and Evacuated states inside the Flood state

Once a Person is in the Flooding state he/she may be at Warning state, Evacuating state or Evacuated state. Therefore we will add these three states inside the Flooding state.

Drag and drop a state inside the Flooding state and change its properties as shown:

Name: Warning



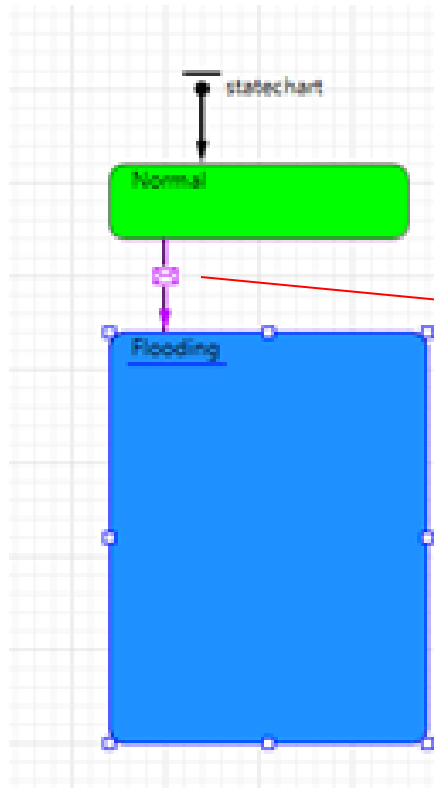
The image shows a statechart editor interface. On the left, a statechart is displayed on a grid. It has a root state 'statechart' with a transition to a green state 'Normal'. From 'Normal', there is a transition to a blue state 'Flooding'. Inside the 'Flooding' state, there is a yellow state 'Warning'. The 'Warning' state is currently selected and highlighted with a yellow border and fill. On the right, the 'Properties' panel is open for the 'Warning - State'. It shows the following configuration:

- Name: Warning Show name
- Fill color: Default
- Entry action: `shapeBody.setFillColor(yellow);`
- Exit action: (empty)

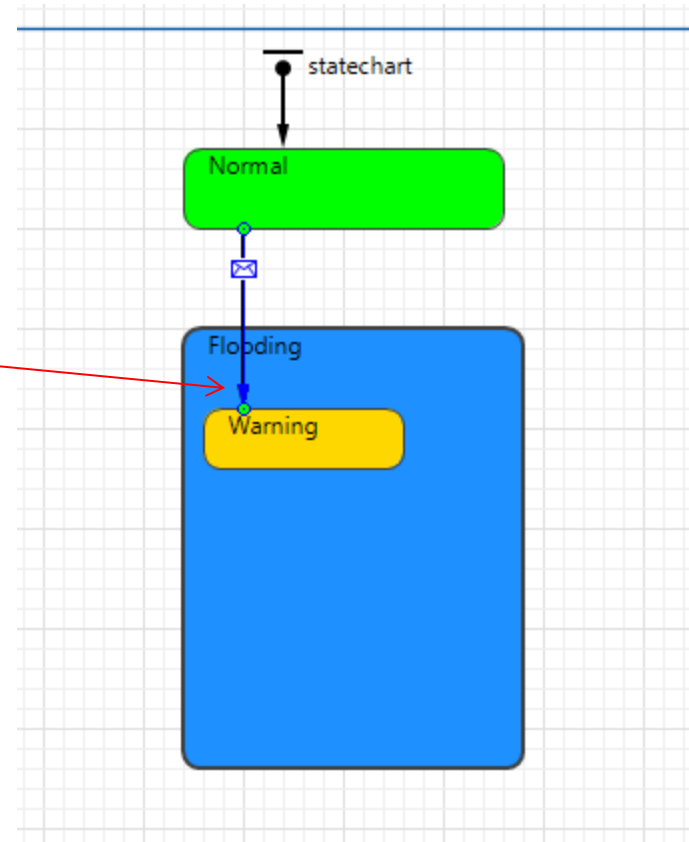
Below the Properties panel, the text 'Entry action: shapeBody.setFillColor(yellow);' is displayed, with a red arrow pointing from the entry action field in the Properties panel to this text.

Make sure that you extend the Flooding transition to be connected to the warning state.

Before



After



Add transition from Warning to the next state (Evacuating)

Change the Triggered by to: Timeout

The screenshot displays the AnyLogic University interface. On the left, a statechart diagram for a 'Person' object is shown. It starts at a 'statechart' entry point leading to a green 'Normal' state. A transition leads to a blue 'Flooding' state, which contains a yellow 'Warning' state. A red arrow points to a transition from 'Warning' to an unlabeled state, with a red arrow from the 'Timeout' property in the right pane pointing to this transition. The right pane shows the 'transition - Transition' properties: Name: transition, Triggered by: Timeout, Timeout: uniform (1, 30) minutes, and Action: (empty). A 'Description' section is also visible.

transition - Transition

Name: transition Show name Ignore

Triggered by: Timeout

Timeout:

Action:

Guard:

Description

Add the following code for the Timeout:
uniform (1, 30).

This means that each Person randomly takes
between 1 to 30 minutes to start evacuating

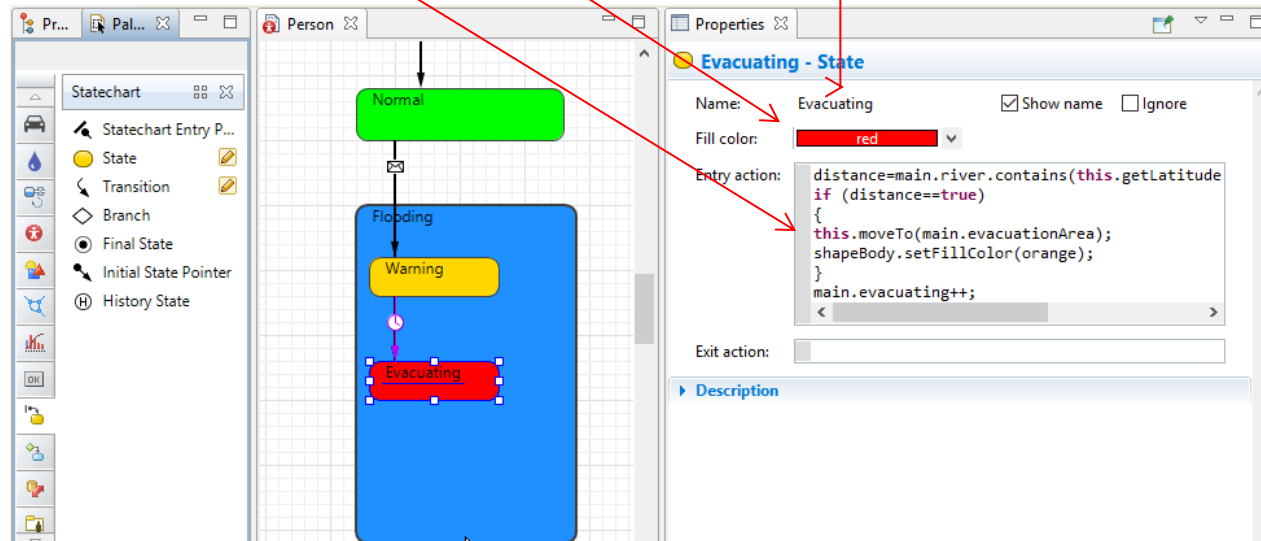
Drag and drop another state from the statechart pallet to the Person window as shown here:

Change its name to: Evacuating

Change its Fill color to: Red

Add the following lines of codes to the Entry action section:

```
distance=main.river.contains(this.getLatitude(), this.getLongitude(), 200.0);  
if (distance==true)  
{  
  this.moveTo(main.evacuationArea);  
  shapeBody.setFill-color(orange);  
}  
main.evacuating++;
```



In the previous slide we use the code to identify people who are living within 200 meters distance of the river.

```
distance=main.river.contains(this.getLatitude(), this.getLongitude(), 200.0);
```

Those who are within the 200 meters distance evacuate the area and move to evacuation zone.

```
if (distance==true)  
{  
this.moveTo(main.evacuationArea);
```

We also change the color of these persons to orange

```
shapeBody.setFillColor(orange);  
}
```

We also create a variable in the main calling it evacuating to measure number of people who are evacuating

Drag and drop another transition to the Person window and attach it to the Evacuating state as shown.

Change its triggered by to : Agent Arrival

The screenshot displays the AnyLogic University interface. The main workspace shows a statechart for a 'Person' object. The statechart starts at a 'statechart' entry point leading to a 'Normal' state (green rounded rectangle). From 'Normal', a transition leads to a 'Flooding' state (blue rounded rectangle). Inside the 'Flooding' state, there is a 'Warning' state (yellow rounded rectangle) and an 'Evacuating' state (red rounded rectangle). A transition is being added to the 'Evacuating' state, indicated by a red arrow from the 'Transition' icon in the left palette. The Properties panel on the right shows the configuration for 'transition1 - Transition':

- Name: transition1
- Triggered by: Agent arrival (selected in a dropdown menu)
- Action: (empty text field)
- Guard: (empty text field)

The 'Description' section is also visible but empty.

Drag and drop another state to the Person window and attach it to the hanging transition as shown.

Change its name to: Evacuated

Change its Fill color to: green

Add the following codes to the Entry action:

```
shapeBody.setFill-color(green);  
main.evacuated++;
```

The screenshot displays a statechart editor interface. On the left, a project tree shows a hierarchy of folders and files, including 'FloodModel1' and 'FloodModelV1*'. The main workspace shows a statechart with a 'statechart' root. The states are: 'Normal' (green), 'Flooding' (blue), 'Warning' (yellow), 'Evacuating' (red), and 'Evacuated' (green). The 'Evacuated' state is currently selected. On the right, the 'Properties' panel for the 'Evacuated - State' is visible. It shows the state name 'Evacuated', a checked 'Show name' option, a 'Fill color' dropdown set to 'green', and an 'Entry action' field containing the code: `shapeBody.setFill-color(green); main.evacuated++;`. The 'Exit action' field is empty. Below the properties is a 'Description' section. Red arrows from the text above point to the statechart, the state name, the fill color, and the entry action code in the Properties panel.

First line changes the color of the person animation to green.

Second line adds this person to the people who have evacuated.

Drag and drop a transition to the Person window and attach it to the Warning state and Normal state as shown.

Change its triggered by to : Message

Change Message type to: String

Select On particular message for Fire transition

Change the Message to: "Normal"

AnyLogic University

File Edit View Draw Model Tools Help

Projects Palette

FloodModel1

- Flood
- Main
- Person
- Property
- Sensor
- Simulation: Main
- Run Configuration: Main
- Database

FloodModelV1*

- Flood
- Main
- Person
 - Presentation
 - scale
 - person
 - shapeBody
 - Statecharts
 - Links to agents
 - Property
 - Sensor
 - Simulation: Main
 - Run Configuration: Main
 - Database

Person

statechart

Normal

Flooding

Warning

Evacuating

Evacuated

Properties

transition2 - Transition

Name: transition2 Show name

Triggered by: Message

Message type: String

Fire transition: Unconditionally On particular message If expression is true

Message: "normal"

Action: shapeBody.setFillColor(black);

Guard:

Description

Type the following line of code in the Action section:
`shapeBody.setFillColor(black);`

Drag and drop a transition to the Person window and attach it to the Evacuating state and Normal state as shown.

Change its triggered by to : Message

Change Message type to: String

Select On particular message for Fire transition

Change the Message to: "Normal"

AnyLogic University
File Edit View Draw Model Tools Help

Projects Palette

Statechart

- Statechart Entry Point
- State
- Transition
- Branch
- Final State
- Initial State Pointer
- History State

Person

statechart

Normal

Flooding

Warning

Evacuating

Evacuated

transition4 - Transition

Name: transition4 Show name

Triggered by: Message

Message type: String

Fire transition: Unconditionally On particular message If expression is true

Message: "normal"

Action: `this.moveTo(Y, X);
shapeBody.setFill-color(black);
main.evacuating--;`

Guard:

Description

Type the following lines of code in the Action section:

```
this.moveTo(Y, X);  
shapeBody.setFill-color(black);  
main.evacuating--;
```

Drag and drop a transition to the Person window and attach it to the Evacuated state and Normal state as shown.

Change its triggered by to : Message

Change Message type to: String

Select On particular message for Fire transition

Change the Message to: "Normal"

AnyLogic University

File Edit View Draw Model Tools Help

Projects Palette

Statechart

- Statechart Entry Point
- State
- Transition
- Branch
- Final State
- Initial State Pointer
- History State

Person

statechart

Normal

Flooding

Warning

Evacuating

Evacuated

transition4 - Transition

Name: transition4 Show name

Triggered by: Message

Message type: String

Fire transition: Unconditionally On particular message If expression is true

Message: "normal"

Action: `this.moveTo(Y, X);
shapeBody.setFill-color(black);
main.evacuated--;`

Guard:

Description

Type the following lines of code in the Action section:

```
this.moveTo(Y, X);  
shapeBody.setFill-color(black);  
main.evacuated--;
```


- Drag and drop a variable from the Palette and change its name to: distance.
- Change its type to: boolean
- Set its initial value to: false

The screenshot displays the AnyLogic University interface. On the left, the 'Agent' palette shows various components, with 'Variable' selected. A red arrow points from the 'Variable' icon to a 'distance' variable in the statechart. The statechart for the 'Person' agent shows a sequence of states: 'Normal' (green), 'Flooding' (blue), 'Warning' (yellow), 'Evacuating' (red), and 'Evacuated' (green). Transitions are indicated by arrows between these states. On the right, the 'Properties' window for the 'distance' variable is shown. It has the following settings: Name: distance, Visible: yes, Type: boolean, and Initial value: false. Red arrows from the list above point to these specific settings.

- Drag and drop another variable from the Palette and change its name to: X
- Change its type to: double
- Set its initial value to: getLongitude()

The screenshot displays the AnyLogic University interface. On the left, the 'Agent' palette shows various components, with 'Variable' selected. A red arrow points from the 'Variable' icon to a variable named 'X' in the workspace. The workspace shows a statechart for a 'Person' agent with states: Normal (green), Flooding (blue), Warning (yellow), Evacuating (red), and Evacuated (green). A red arrow points from the 'Initial value' field in the Properties panel to the 'getLongitude()' method call. The Properties panel on the right shows the following configuration for the 'X - Variable':

- Name: X
- Visible: yes
- Type: double
- Initial value: getLongitude()
- Access: public
- Static:
- Constant:
- Save in snapshot:
- System dynamics units:

- Drag and drop another variable from the Palette and change its name to: Y
- Change its type to: double
- Set its initial value to: getLatitude()

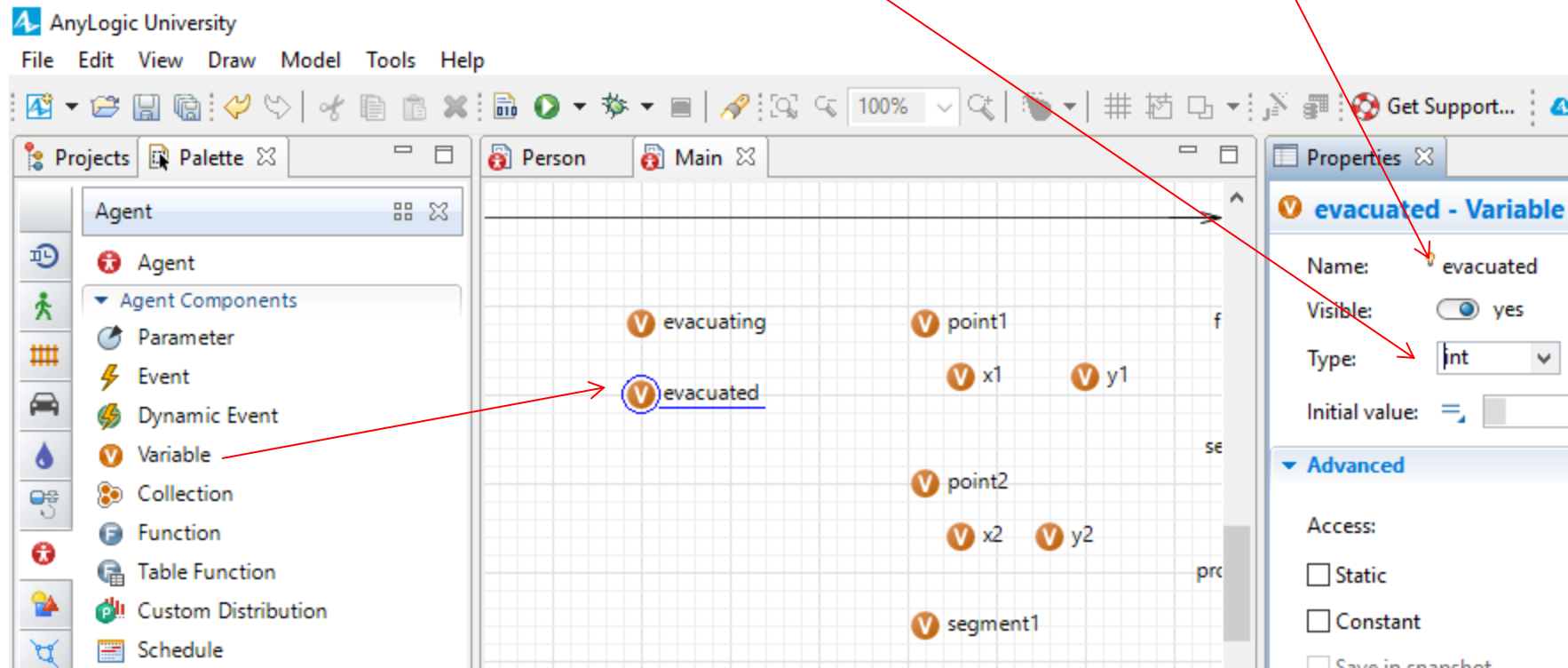
The screenshot displays the AnyLogic development environment. On the left, the 'Palette' shows various components, with the 'Variable' icon highlighted. In the center, a statechart for a 'Person' agent is visible, featuring states: 'Normal' (green), 'Flooding' (blue container), 'Warning' (yellow), 'Evacuating' (red), and 'Evacuated' (dark green). A variable 'Y' is shown in the palette. On the right, the 'Properties' window for the 'Y - Variable' is open, showing the following configuration:

- Name: Y
- Visible: yes
- Type: double
- Initial value: getLatitude()
- Access: public
- Static:
- Constant:
- Save in snapshot:
- System dynamics units:

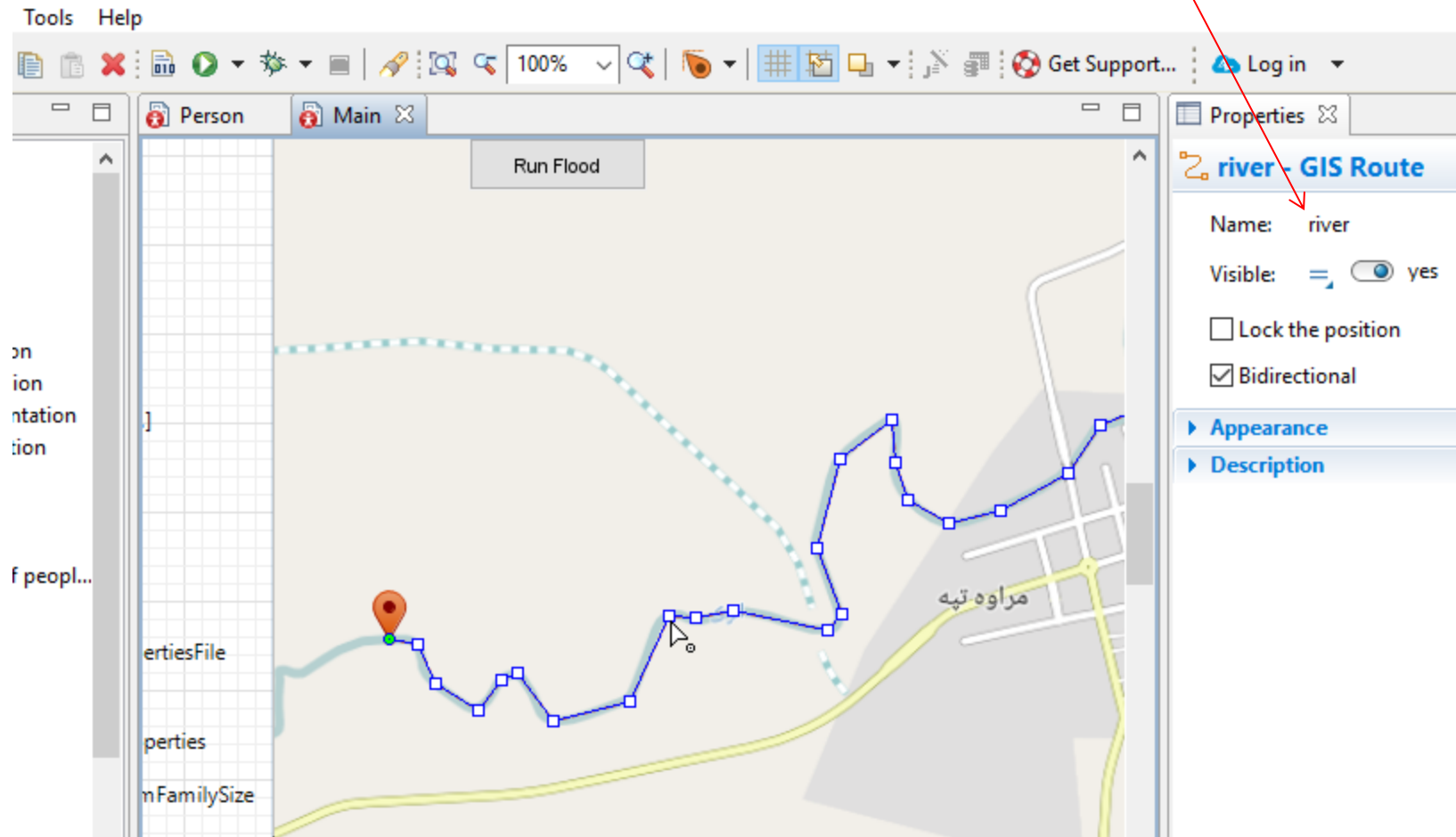
- Double click on the Main and open the main window
- Drag and drop a variable from the Palette and change its name to: evacuating
- Change its type to: int

The screenshot displays the AnyLogic University software interface. On the left, the 'Agent' palette is open, showing various components. A red arrow points from the 'Variable' component in the palette to the 'evacuating' variable in the main workspace. The main workspace shows a grid with several variables: 'evacuating', 'point1', 'x1', 'y1', 'point2', 'x2', 'y2', 'segment1', and 'path'. A red arrow points from the 'evacuating' variable in the workspace to the 'Properties' panel on the right. The 'Properties' panel shows the variable's name as 'evacuating', its type as 'int', and its access as 'public'. The 'Advanced' section is also visible, with options for 'Static', 'Constant', 'Save in snapshot', and 'System dynamics units'.

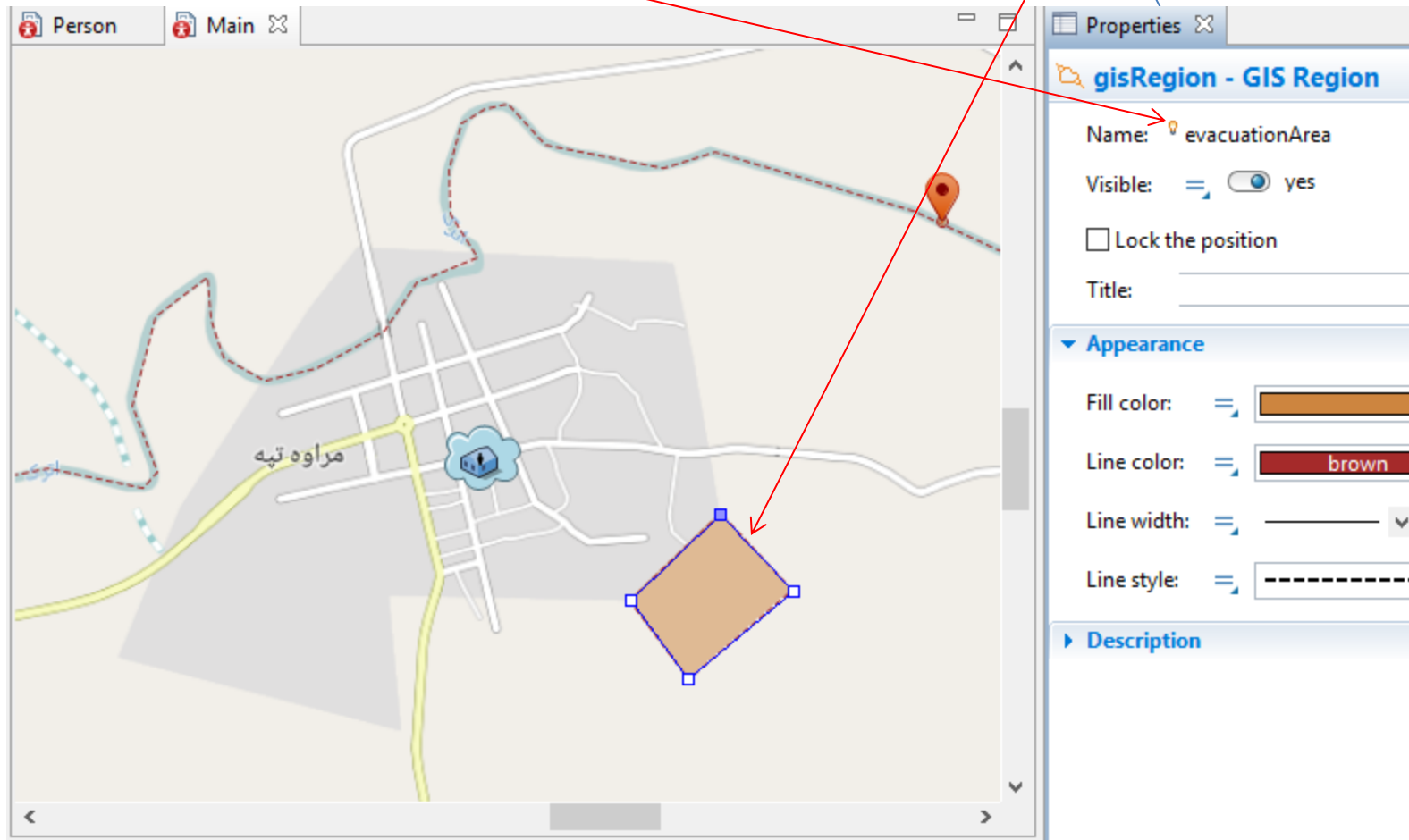
- Drag and drop another variable from the Palette to Main and change its name to: evacuated
- Change its type to: int



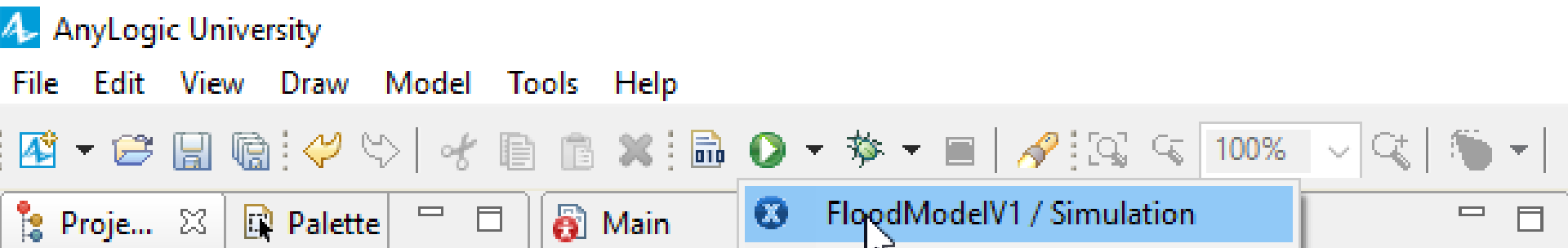
- While on the Main. Click on the river line and change its name from gisRoute to river



- While on the Main. Click on the gisRegion and change its name from gisRegion to: evacuationArea



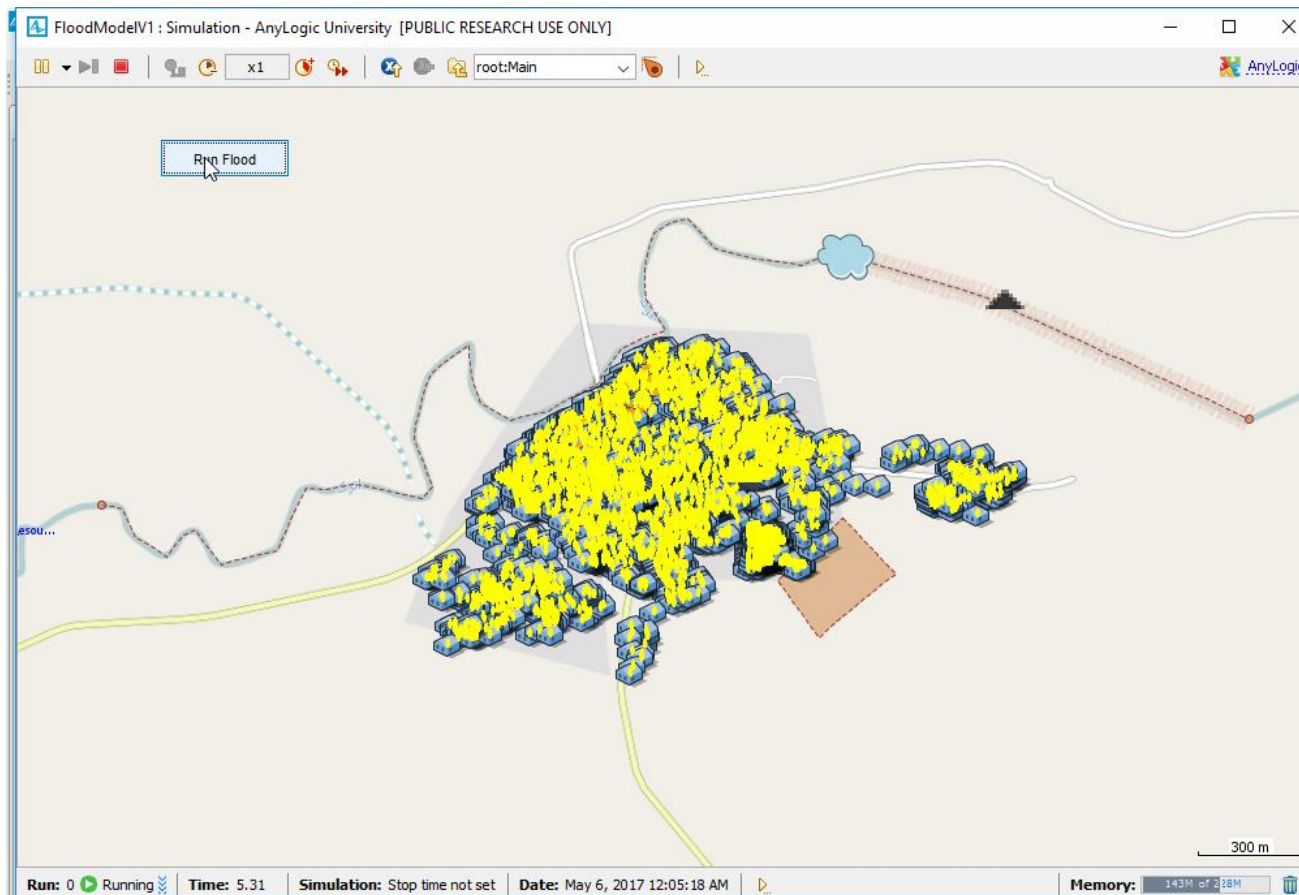
Run the model



Results

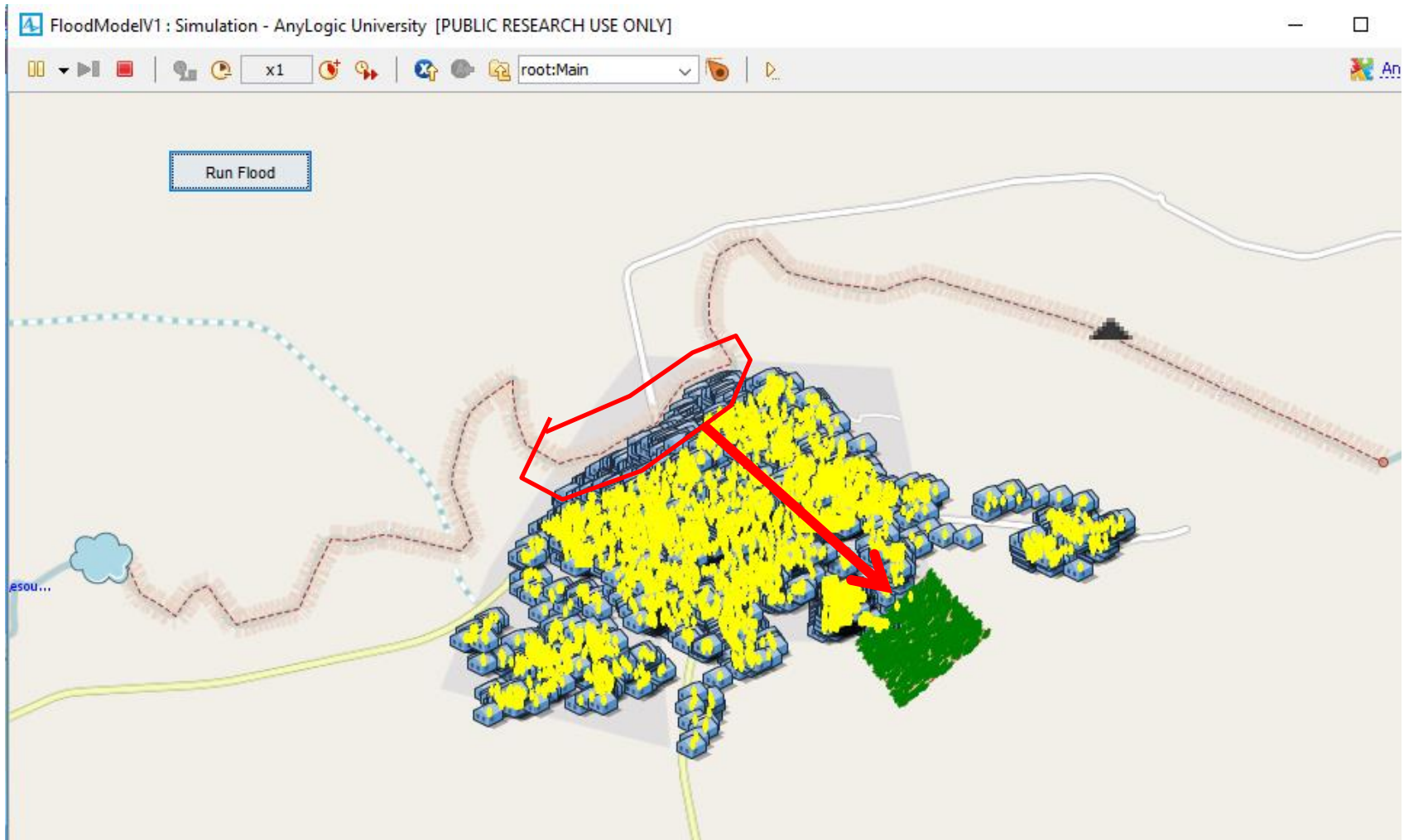
If you do not see any error, you will see that after flood reaches to the Sensor, sensor symbol increases. It also sends the warning to people that we will see them in the next lesson.

People who receive the warning are shown in yellow and then people who are close to the river start evacuating.



Results

- People who are evacuating move to the evacuationArea



- This was another way of creating behavior for an agent using statecharts.
- We will create more complicated statecharts in the future.
- We will explain how to extract data and graph from this simulation in lesson 5.
- Save your project.