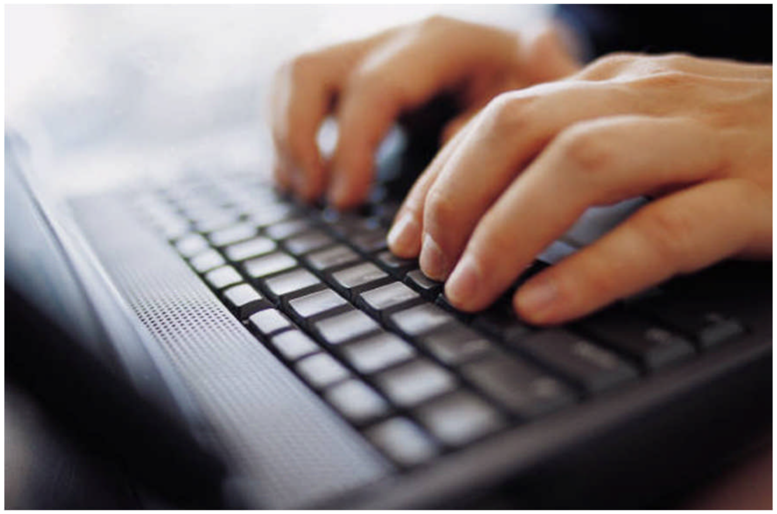
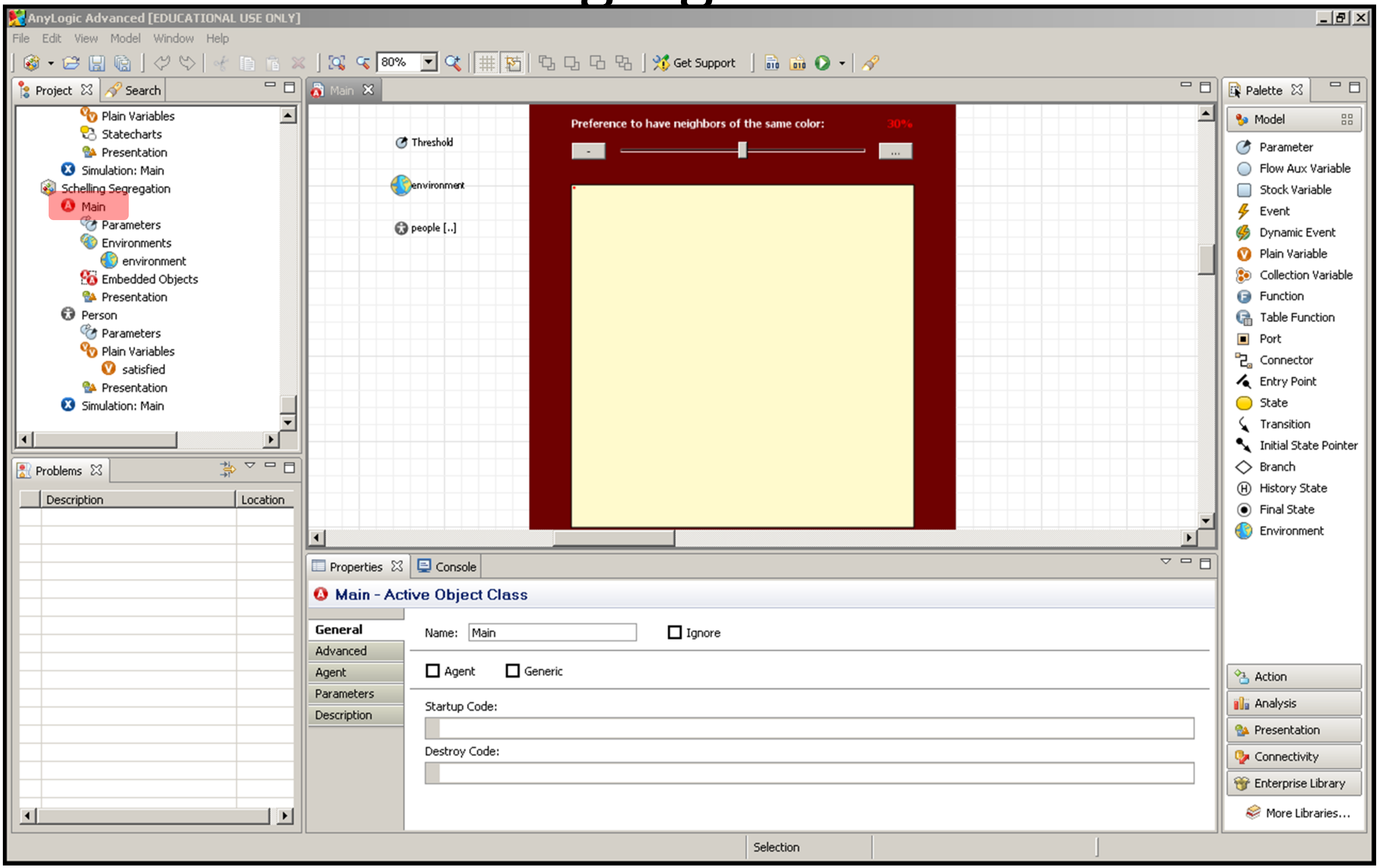
Hands on Model Use Ahead

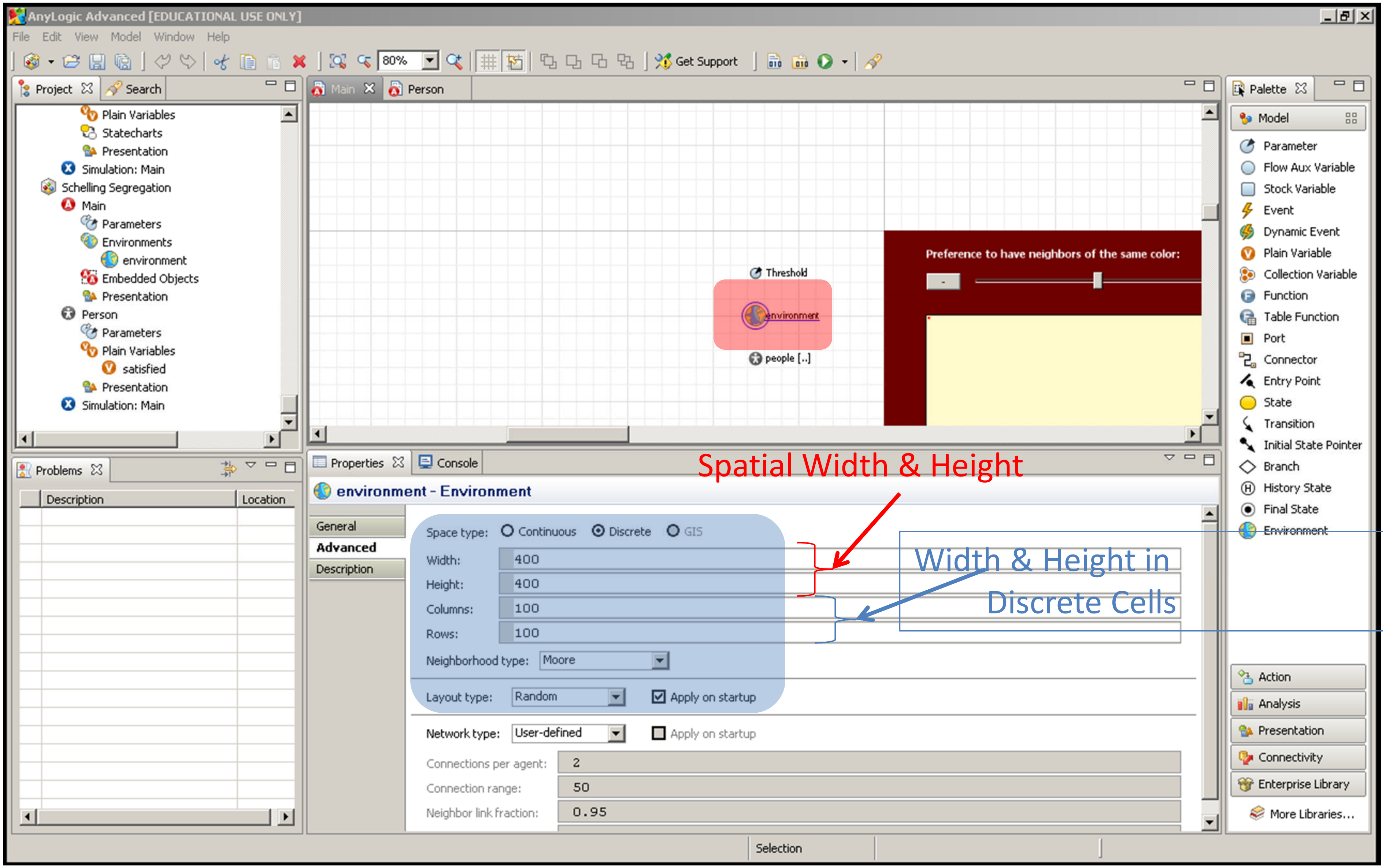


Load model: Schelling Segregation.alp

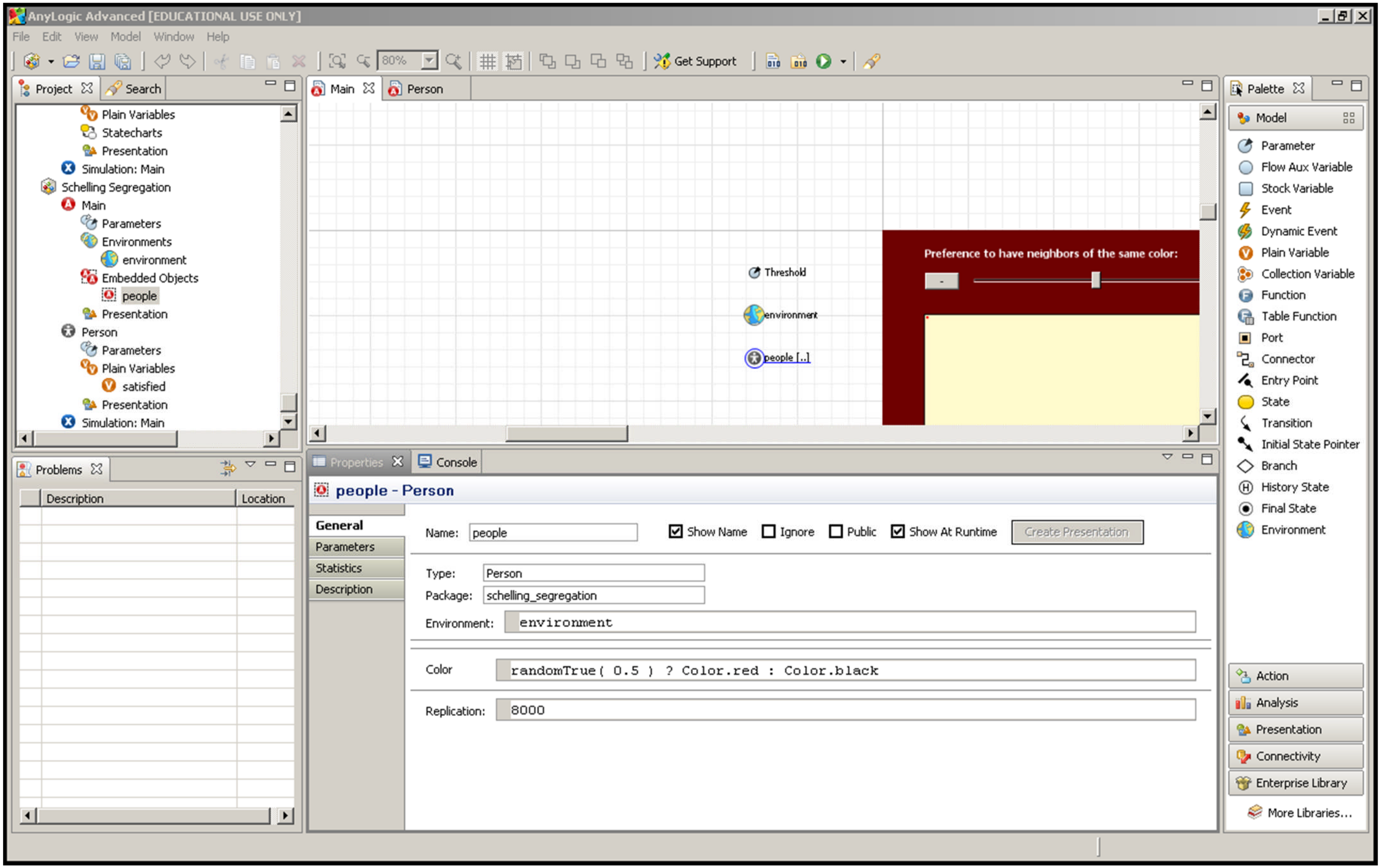
A Model to Examine the Emergence of Segregation



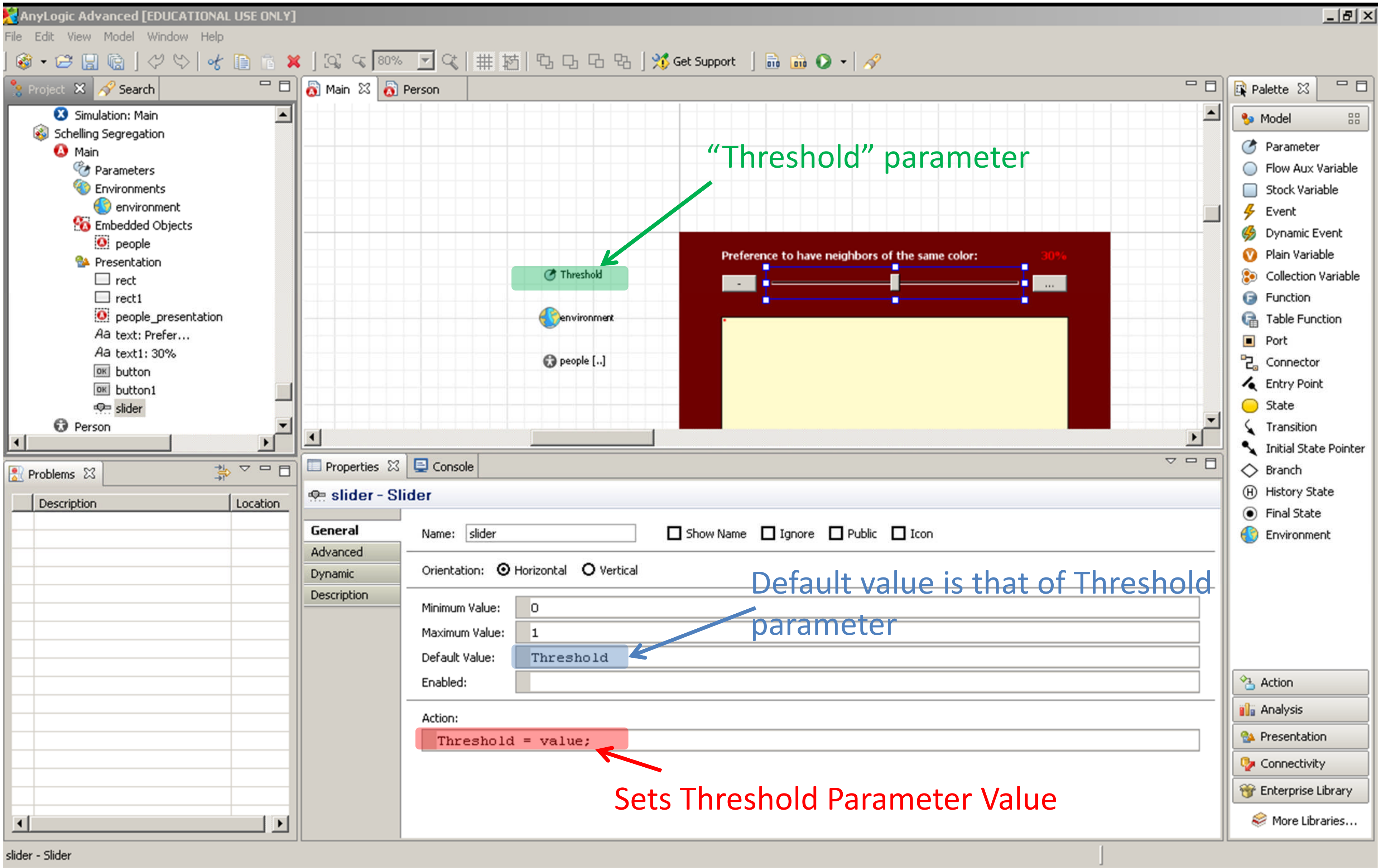
A Discrete Spatial Environment with Random Agent Positioning



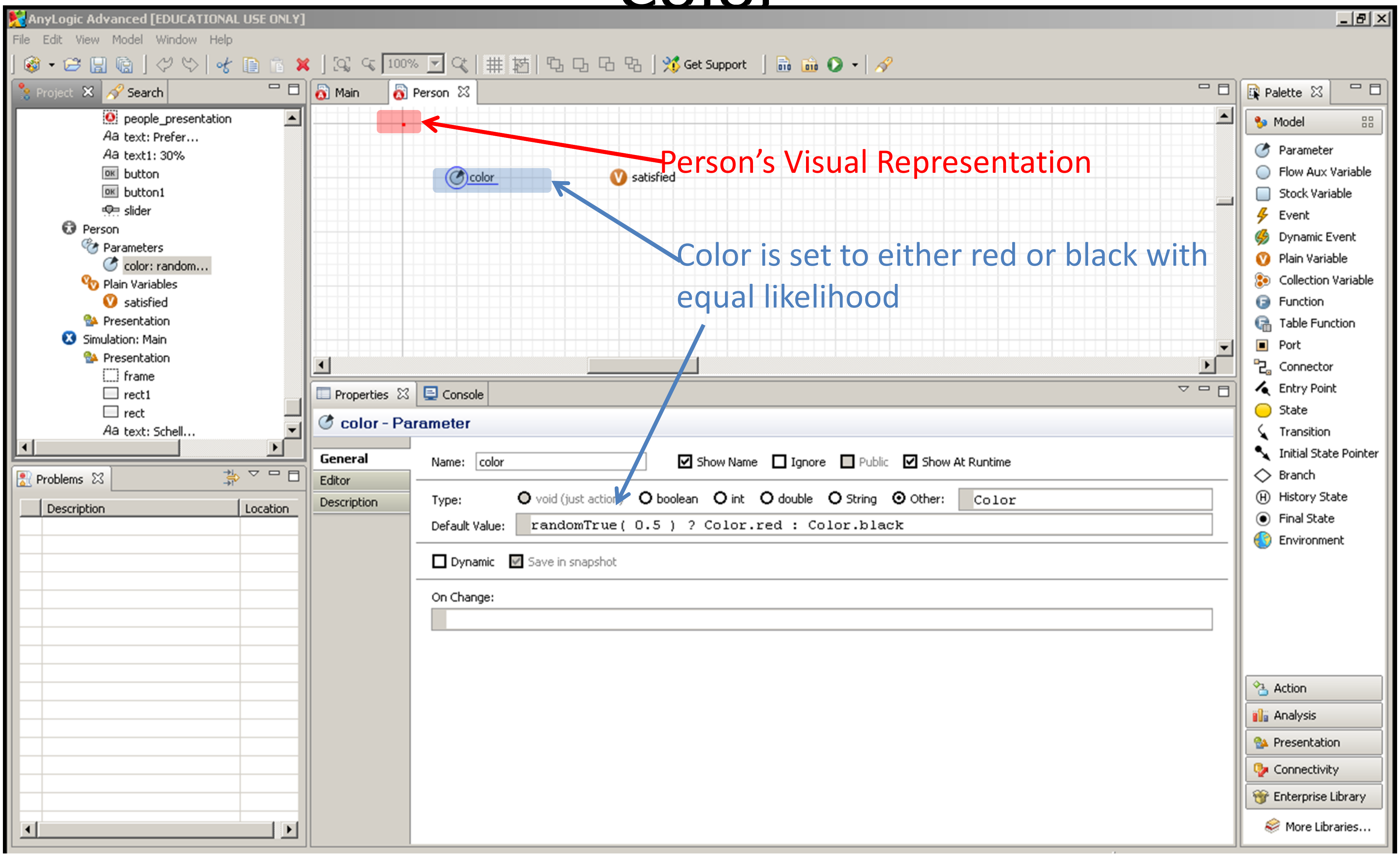
Population Dependence on the Population



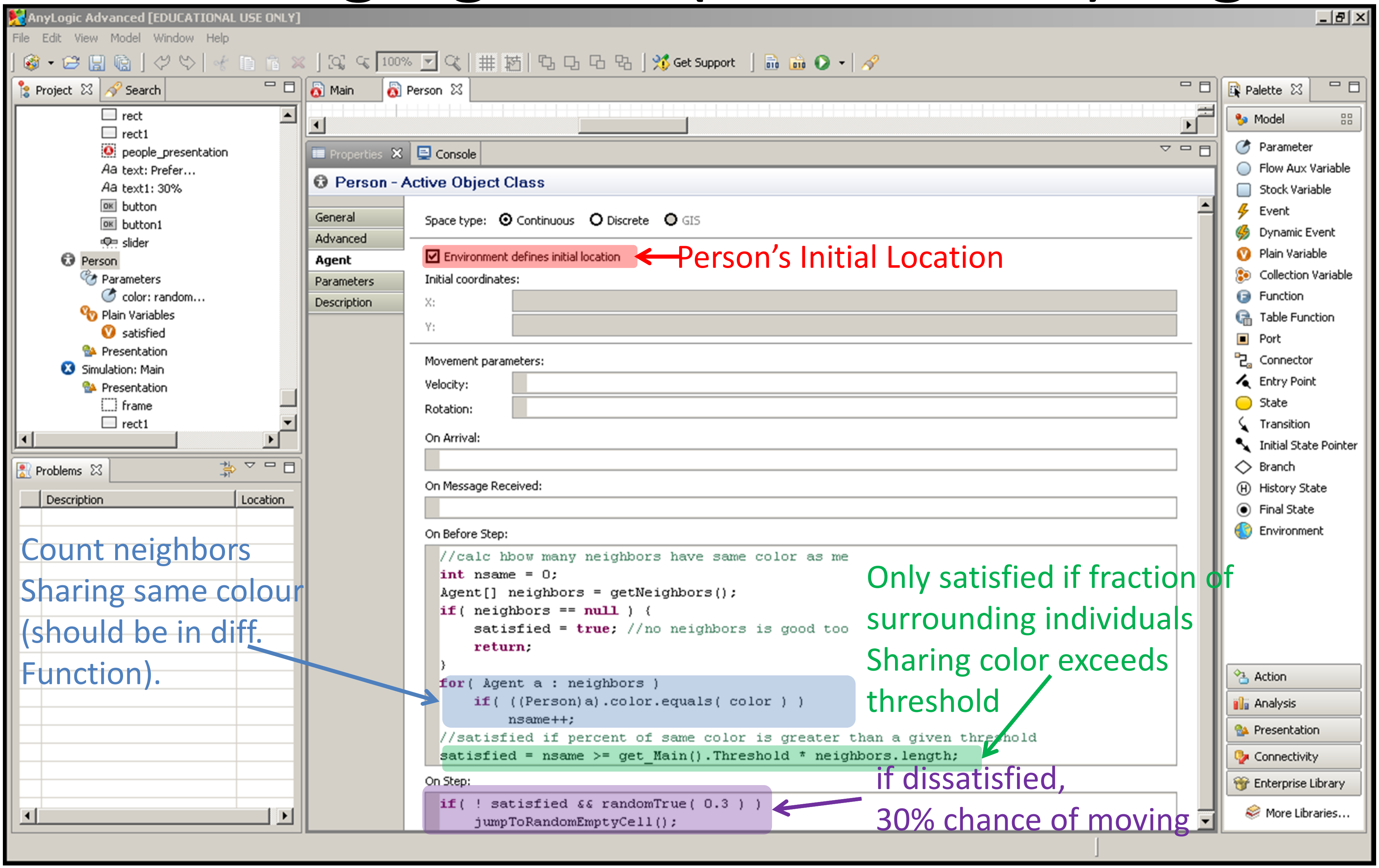
Slider Input Sets Parameter Value



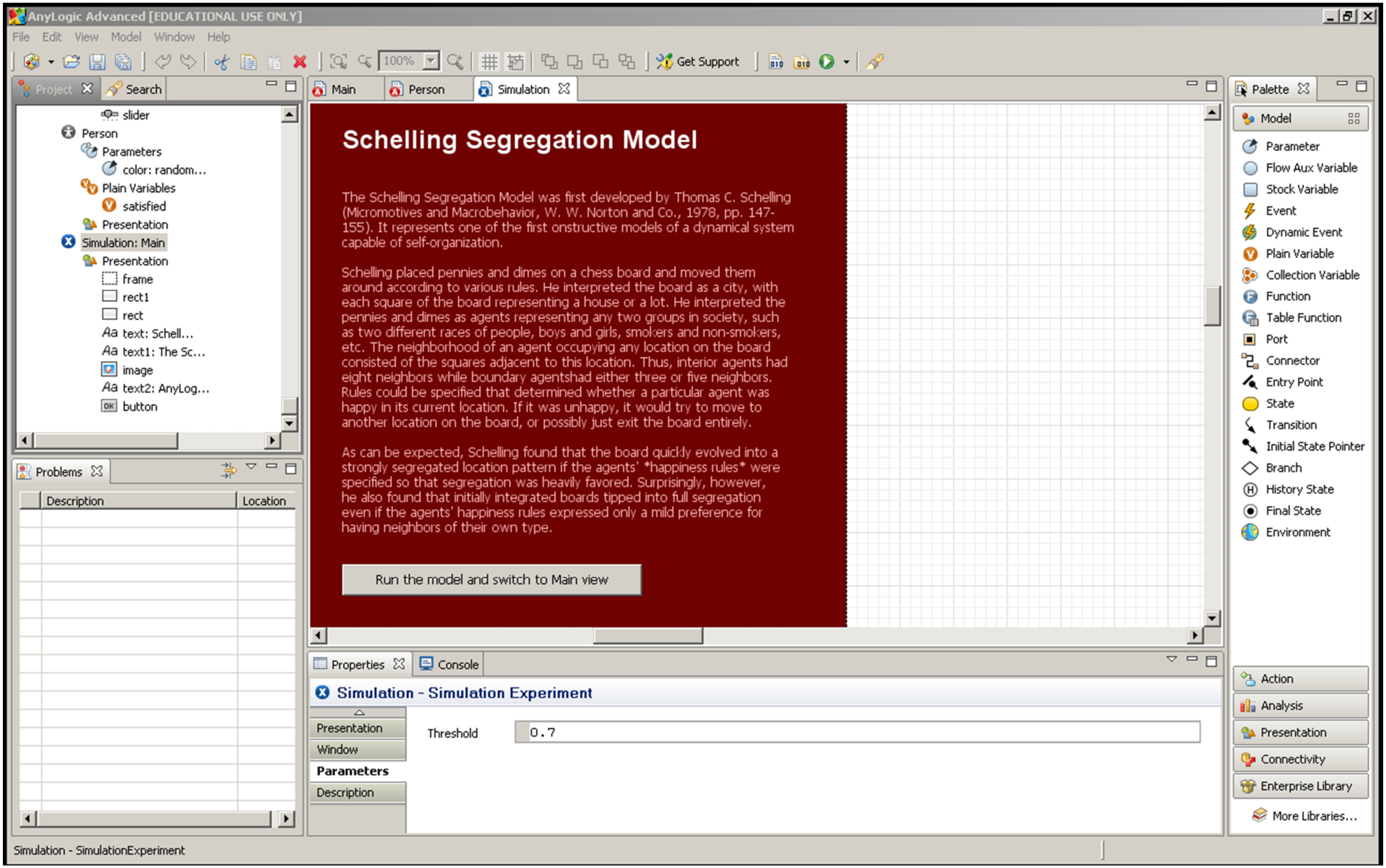
Person is Assigned a Randomly Picked Color



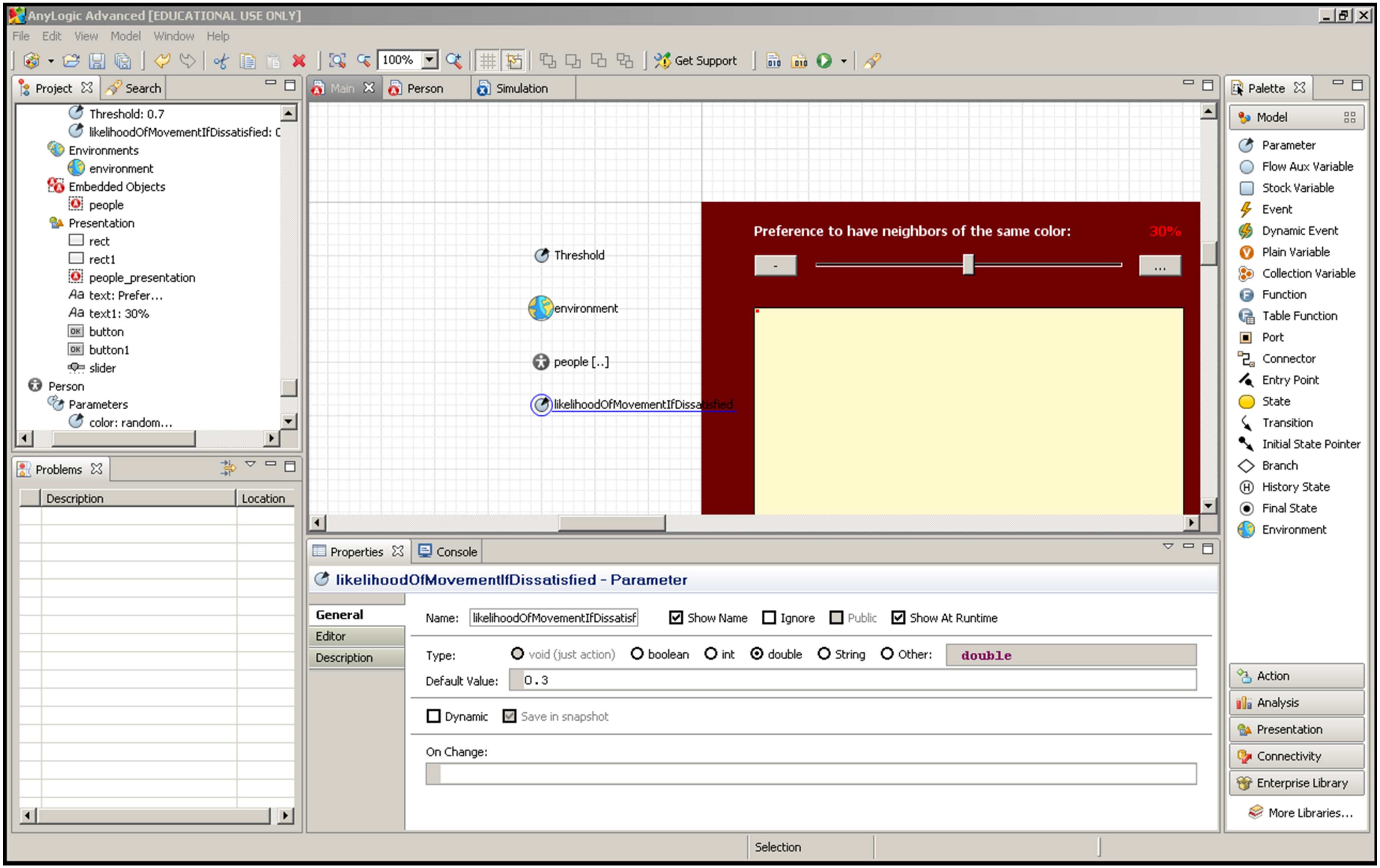
Core Segregation (Movement) Logic



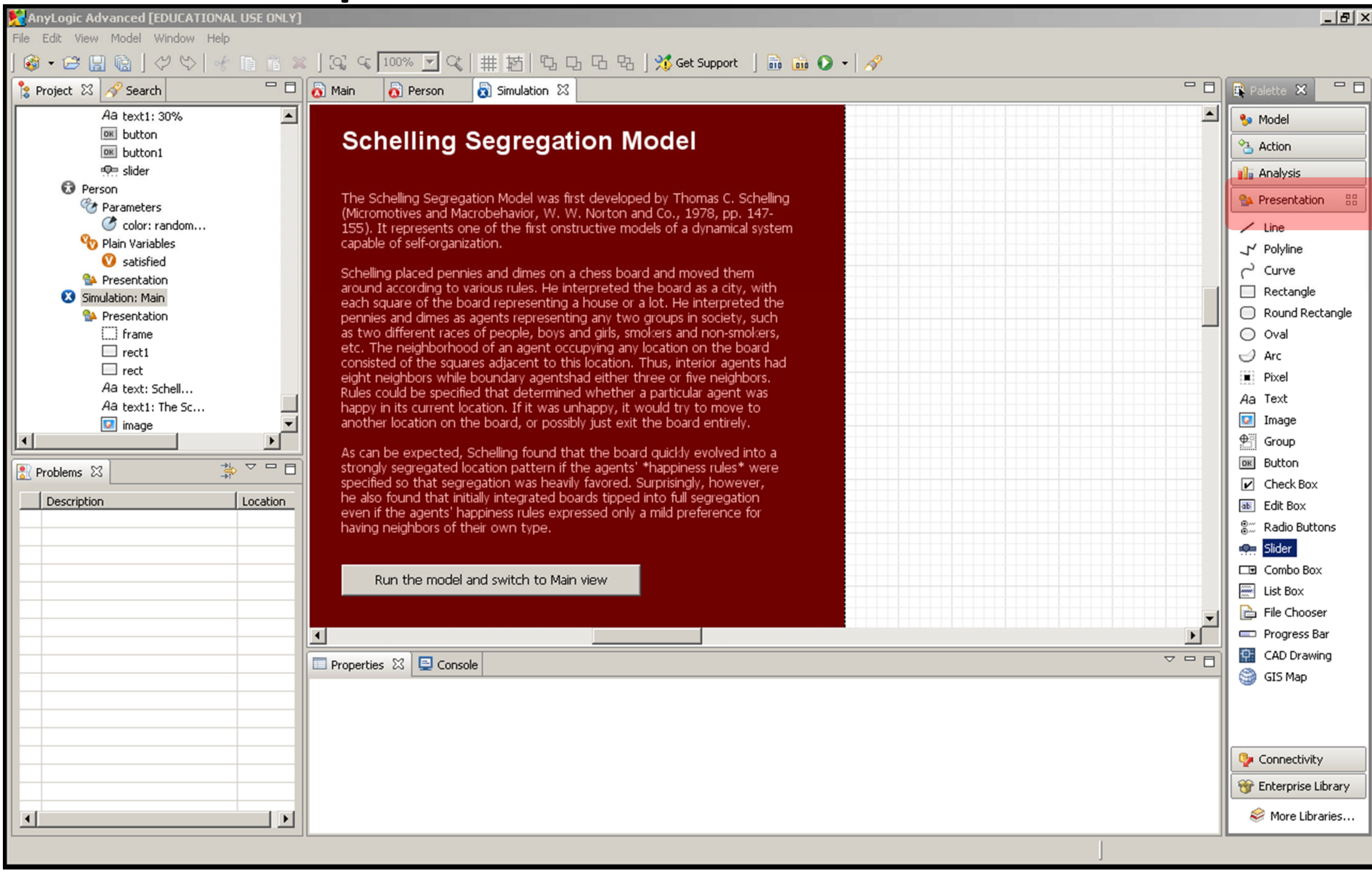
Experiment: Simulation Sets Parameter Assumptions



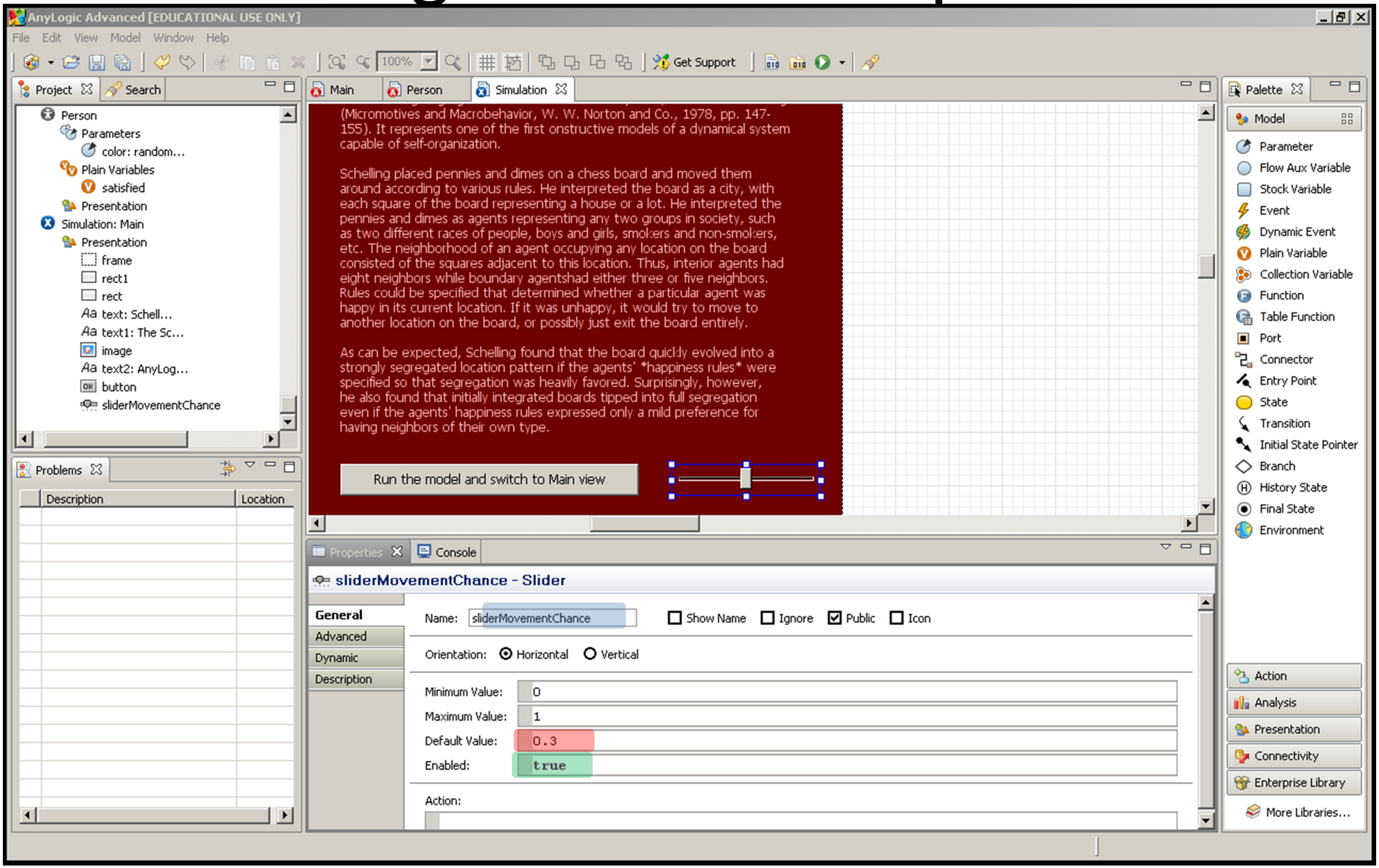
Add a Parameter to Main



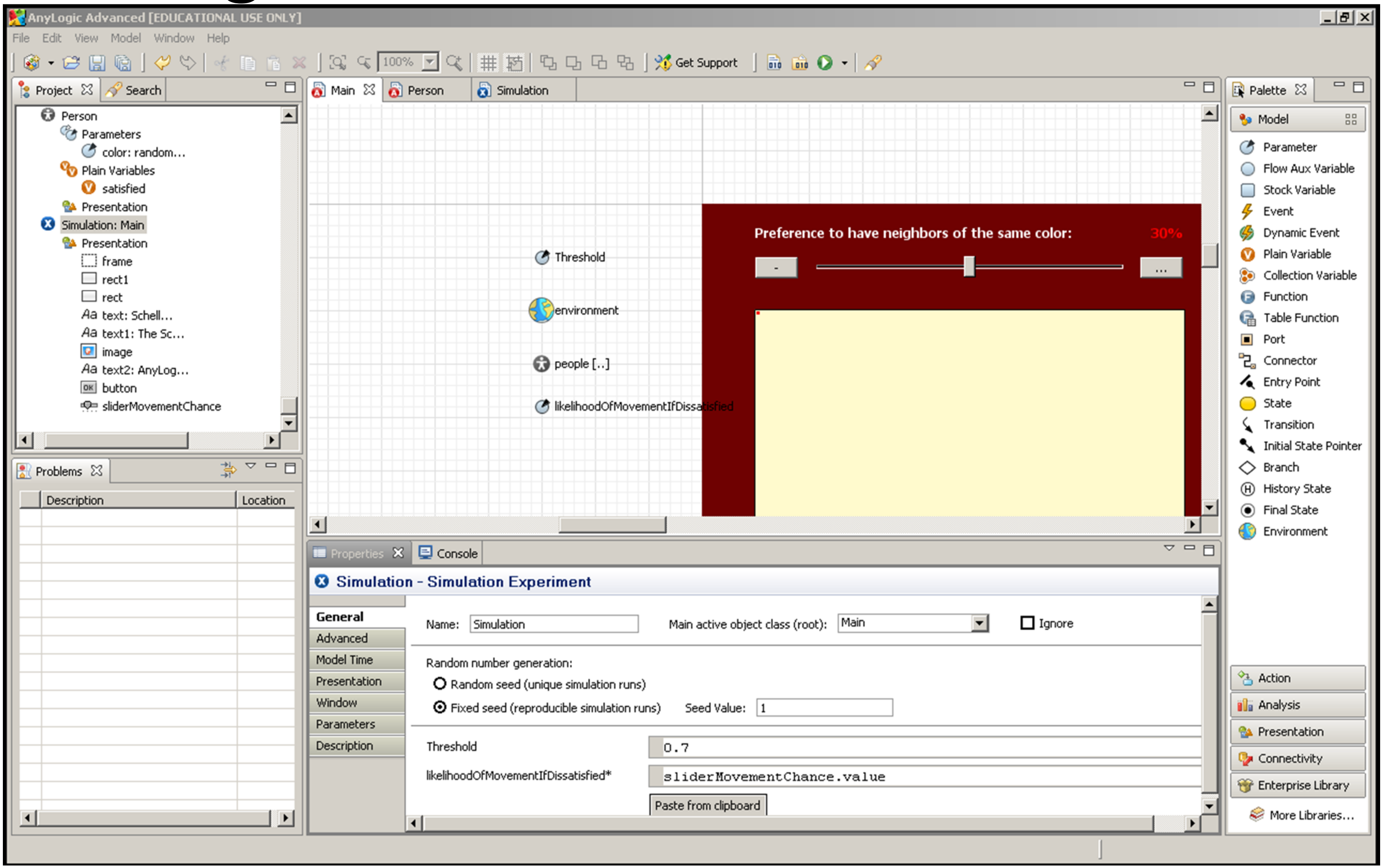
Experiment: Add a Slider!



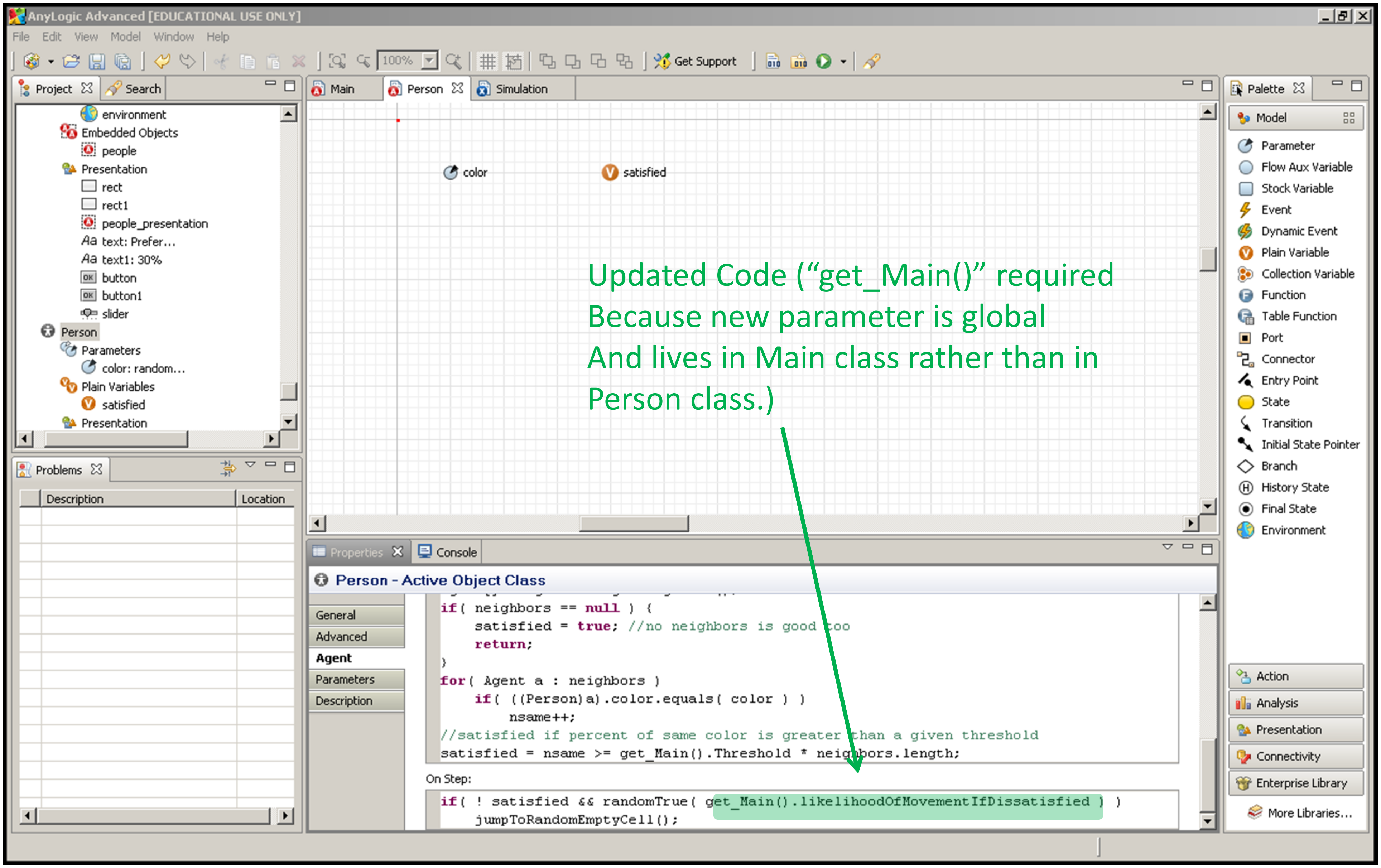
Setting the Slider Properties



Setting Value for Parameter from Slider



Modify Person's Behavior to Depend on New Parameter



Movement in Discrete Space

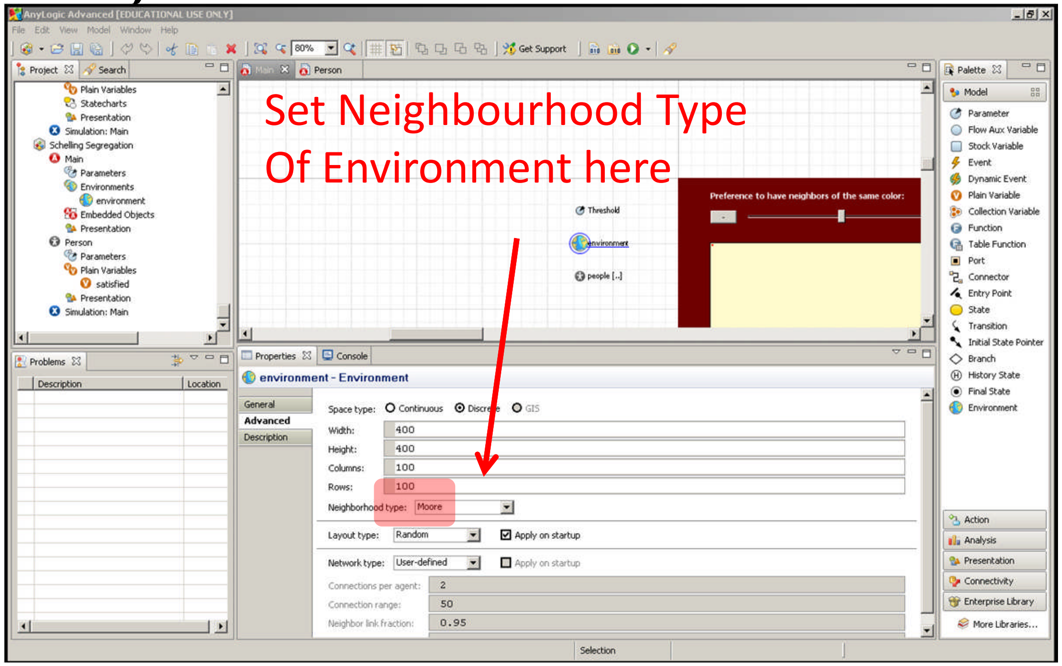
- jumpToCell(int row, int column)
  - Jumps to a particular unoccupied cell
  - Precondition: destination cell is unoccupied
- moveToNextCell(int direction)
  - Moves agent into neighbouring cell in a given direction
  - Directions: NORTH, SOUTH, EAST, WEST, NORTHEAST, NORTHWEST, SOUTHEST,SOUTHWEST
    - Precondition: destination cell is unoccupied
- jumpToRandomEmptyCell
  - Jumps to randomly selected empty cell (returning true), returns false if no empty cell can be located

Discovery in Discrete Space

- int []findRandomEmptyCell
  - Returns row & column of an unoccupied cell
- Getting agents in cell or direction
  - getAgentAtCell(int row, int column)
  - getAgentNextToMe(int direction)
  - getNeighbors()

Neigbourhood Models

- Moore: Cardinal directions
  - NORTH,SOUTH,EAST, WEST
- Euclidean
  - NORTH, SOUTH, EAST, WEST, NORTHEAST, NORTHWEST, SOUTHEST,SOUTHWEST



Important Distinction

- Suppose an agent is moving in discrete 2D space and need to be concerned about moving into the same cell as another agent
- We can readily prevent this agent from moving into another cell currently occupied
- But canwe prevent this agent from colliding with another agent that wishes to move into the same cell?
  - To answer this, we need to be clear about the model of time used by agents

Two Key Models of Time in Anylogic: Synchronous Time

- Here, agents all change in lockstep, separated by fixed "time steps"
- When computing agent behavior (to determine agent state in the next timestep), our enquiries about agent state (e.g. using *getAgentAtCell* or *getAgentNextToMe*) need to ask about the situation in the current timestep
  - We gather needed information regarding current state in "onBeforeStep", and changes are performed in "OnStep".
- This is similar to what we saw in System Dynamics – the changes over the next small interval of time (Δt) depend on the current value of the stocks
  - These changes are then applied at once, and all stocks are updated

Enabling Synchronous Time

- Unless enable the steps, the various handlers for synchronized time (e.g. “On before step”, “On step”, “On after step”) etc.) are executed
  – Both environment and agents have “On before step” and “On after step” handlers
  – “On before step” for environments is executed before the corresponding method for agents
  – “On after step” for environments is executed after the corresponding method for agents
- Synchronous time can be enabled via the environment “General” page
  – Click checkbox “Enable steps”

Two Key Models of Time in Anylogic: Asynchronous Time

- Here, every agent is updated at a different time
- No two agents are typically likely to be updated at exactly the same time, so when considering the state of other agents they "see" the last situation where the other agent has been updated

Synchronization & Discrete Agent Movement

- In Synchronous mode, it is difficult to know if two agents will collide using data on the current timestep
  - Even if we know where the other object was during the current timestep, it's possible it will move into the cell we wish to occupy in the next timestep
- It is simpler to handle this asynchronously
  - Here, we can have each agent update at slightly different times, and observe the location of the other agents at the current time – without any significant chance that they will move to the same place at the same time.
- This issue only arises for discrete agent movement, as this is the only case where cells only contain 1 agent