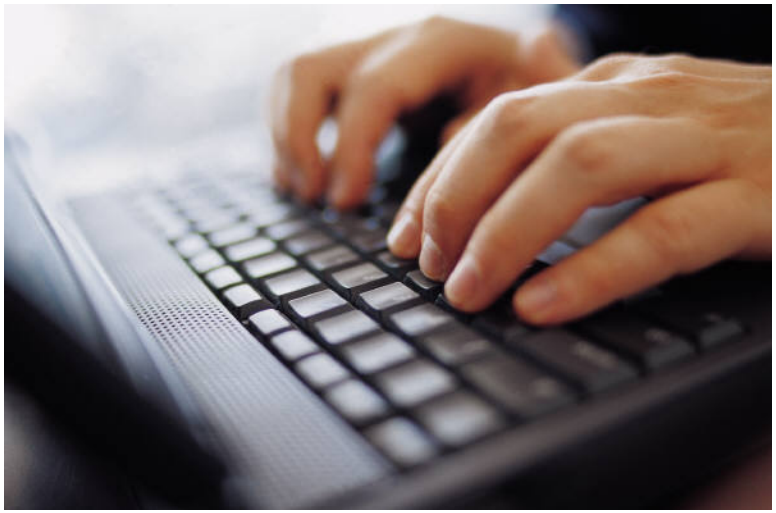
Overview of the AnyLogicInterface

Nathaniel Osgood

11-5-2009

Hands on Model Use Ahead



Load model: TBv1.alp

The AnyLogic User Interface

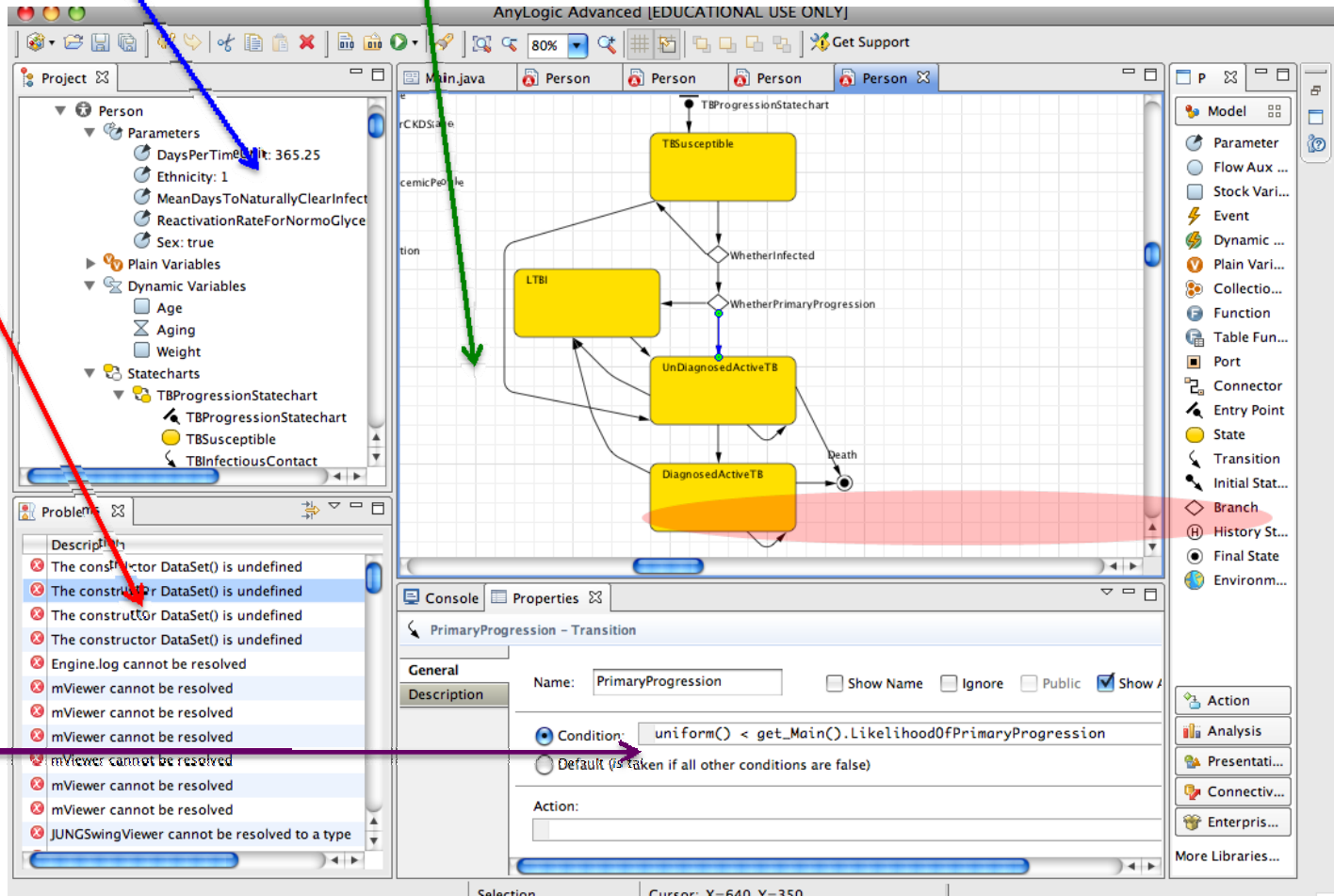
Common Configuration

The Project View (provides overview of projects & components)

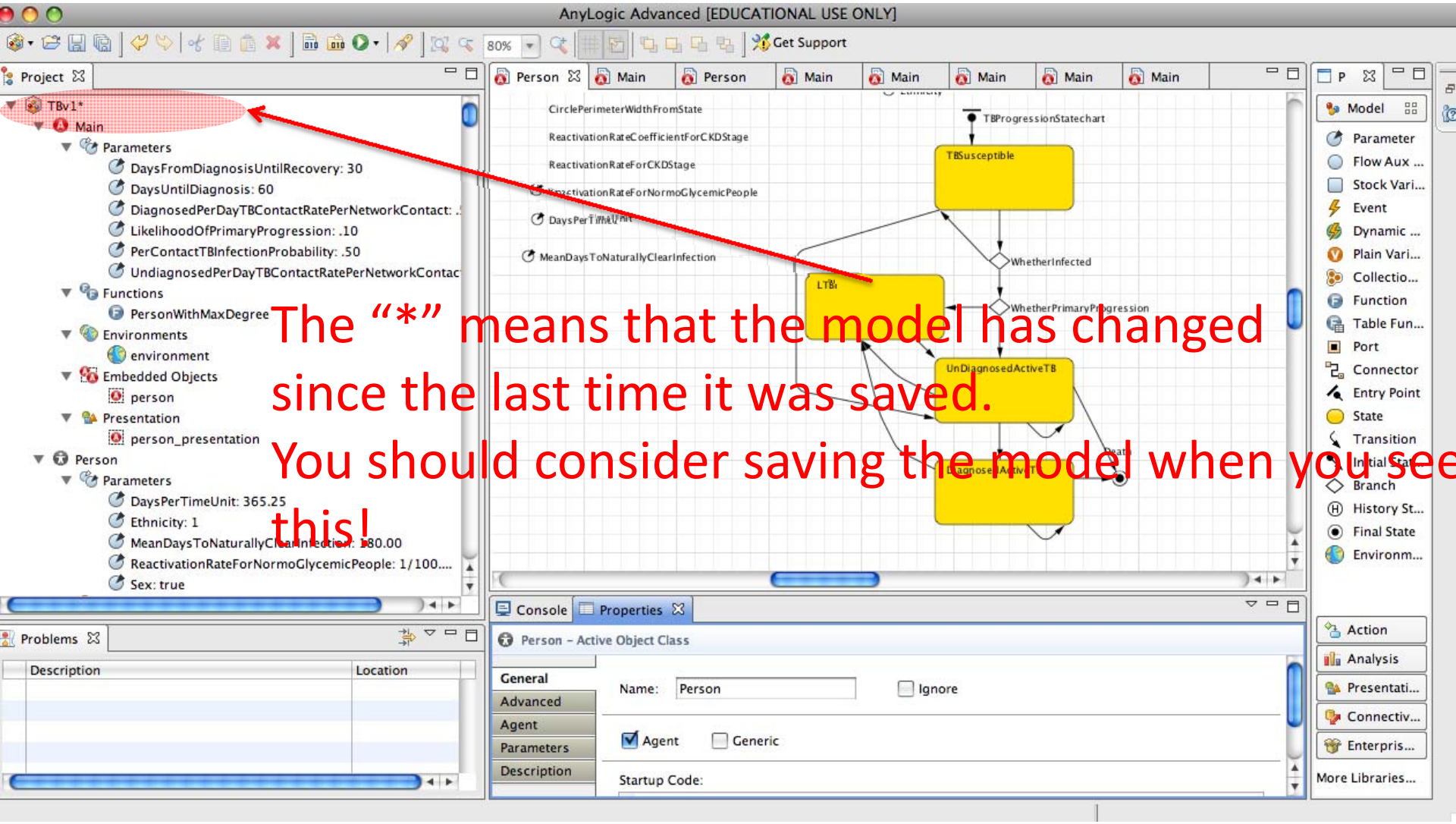
Palettefor currently selected item

Problem area (indicates problem Building/ running Model)

Properties area (shows info on selected element in project or palette window)



The “Project View” – Hierarchically Shows the Project Components



The “*” means that the model has changed since the last time it was saved.

You should consider saving the model when you see this!

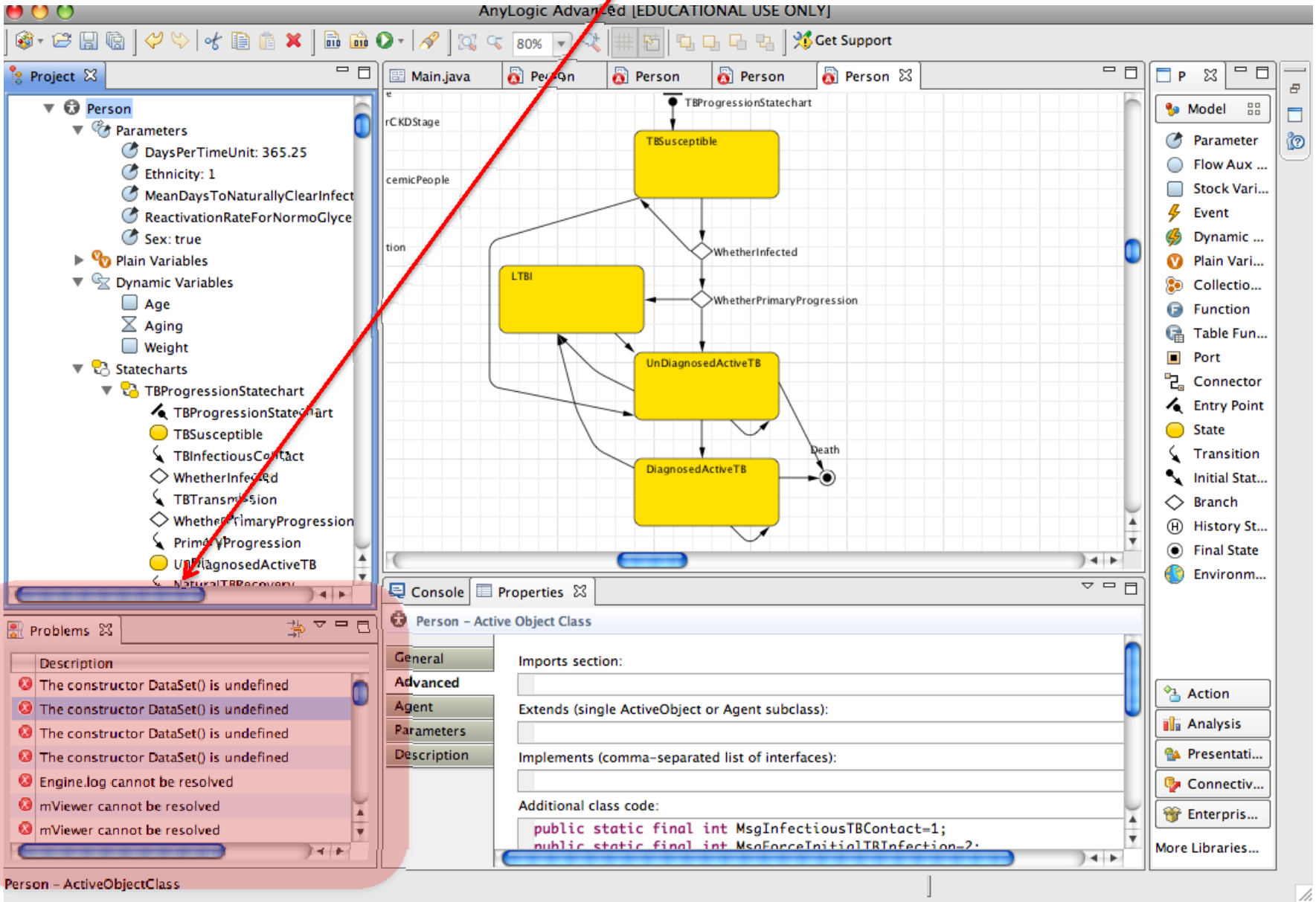
A Critical Distinction: Design vs. Execution time

- The computational elements of Anylogic support both design & execution time presence &behaviour
  - Design time: Constructing the model, running builds
  - Execution time ("Runtime"): Simulating the model
- It is essential to be clear on what behavior & information is associated with which times
- Generally speaking, design-time elements are created to support certain runtime behaviors

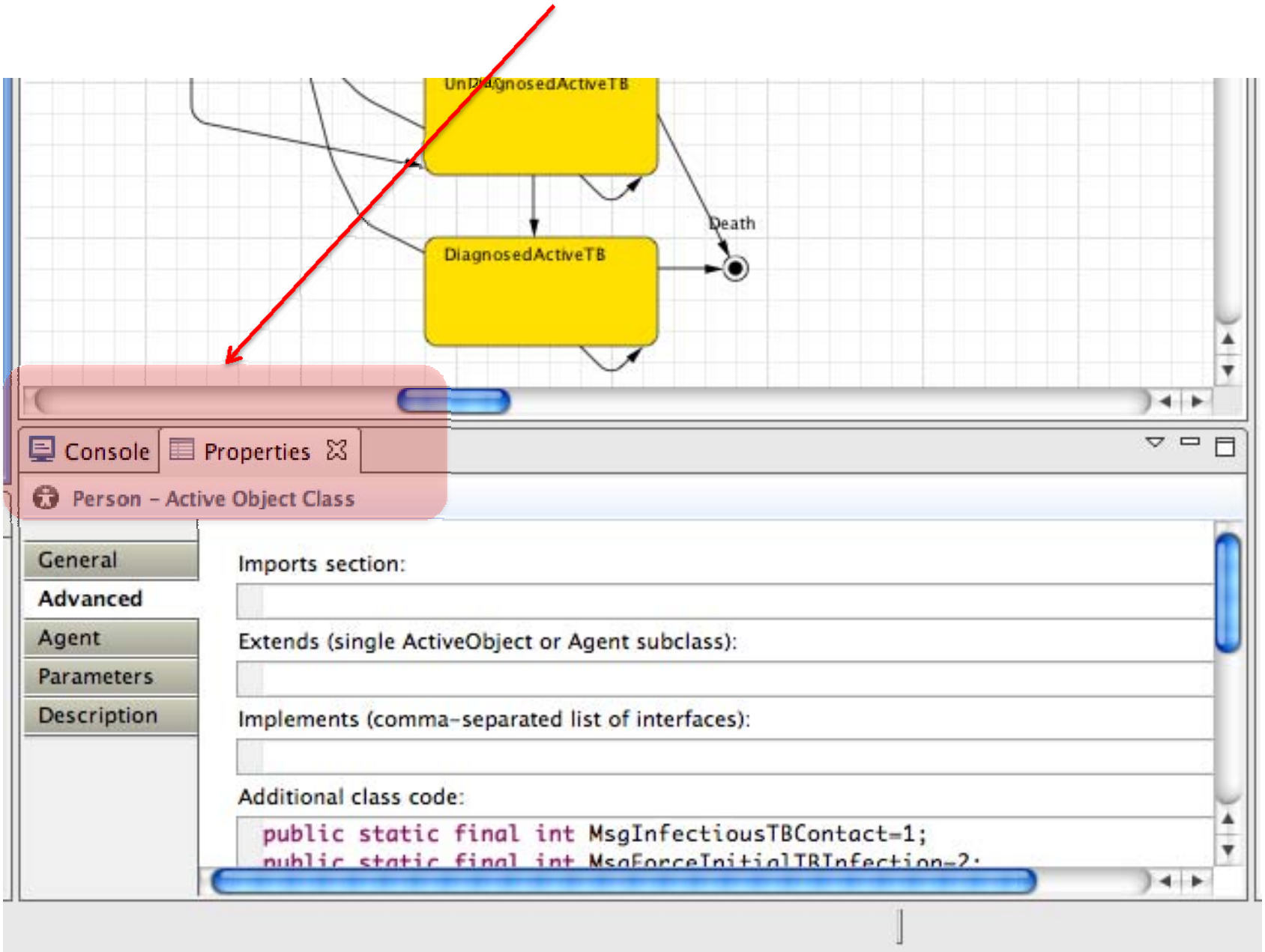
The Notion of a “Build”

- A “Compiler” is a tool to convert from a program’s specification (e.g. state charts, Action diagrams, etc.) to a representation that can be executed
- Normally a compiler is applied to each of several components of a program (e.g. classes)
- AnyLogic’s “build” process applies a compiler to the components of the AnyLogic model

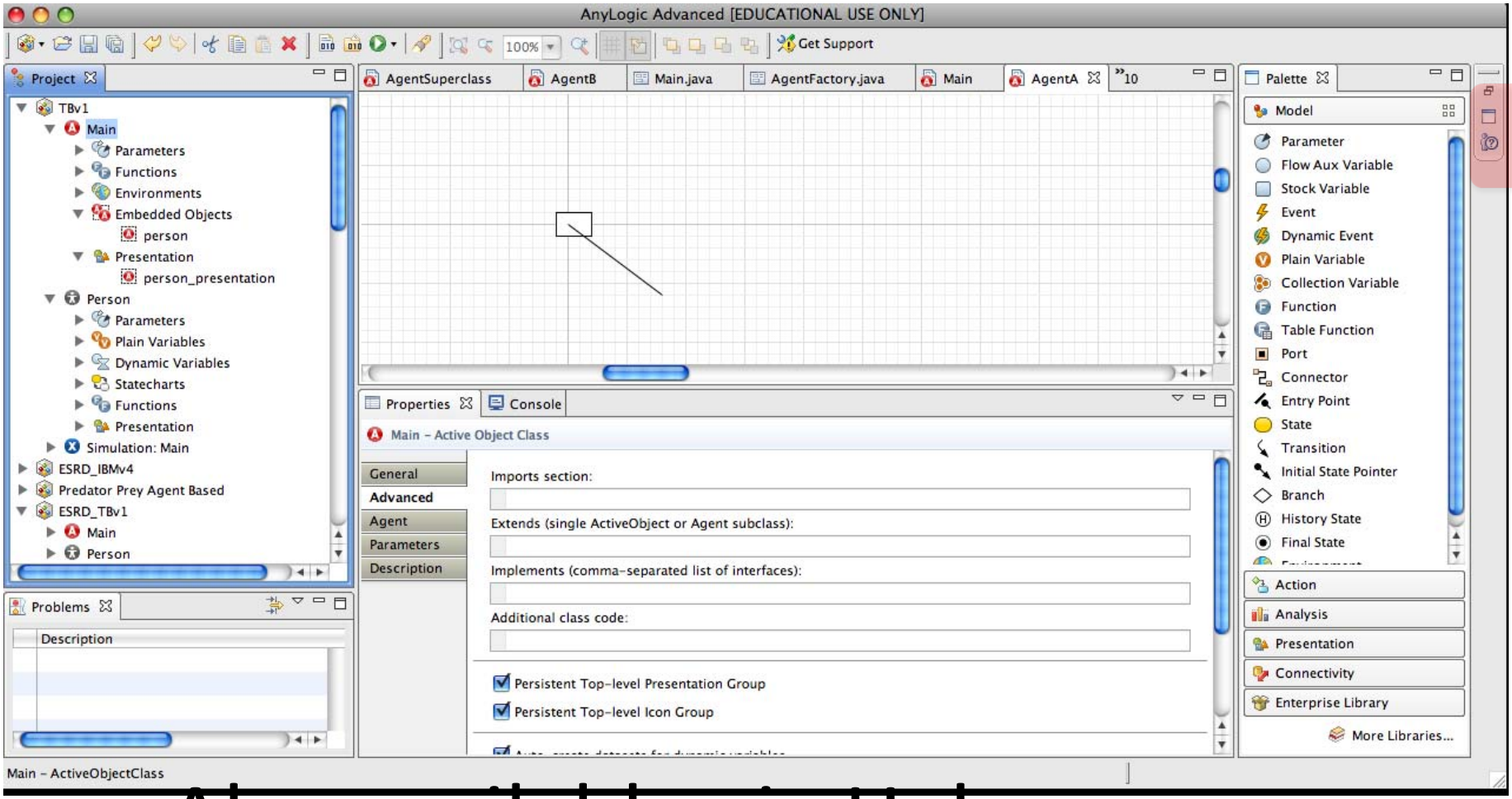
The “Problems View”



Multiple Tabs (switch among tasks)



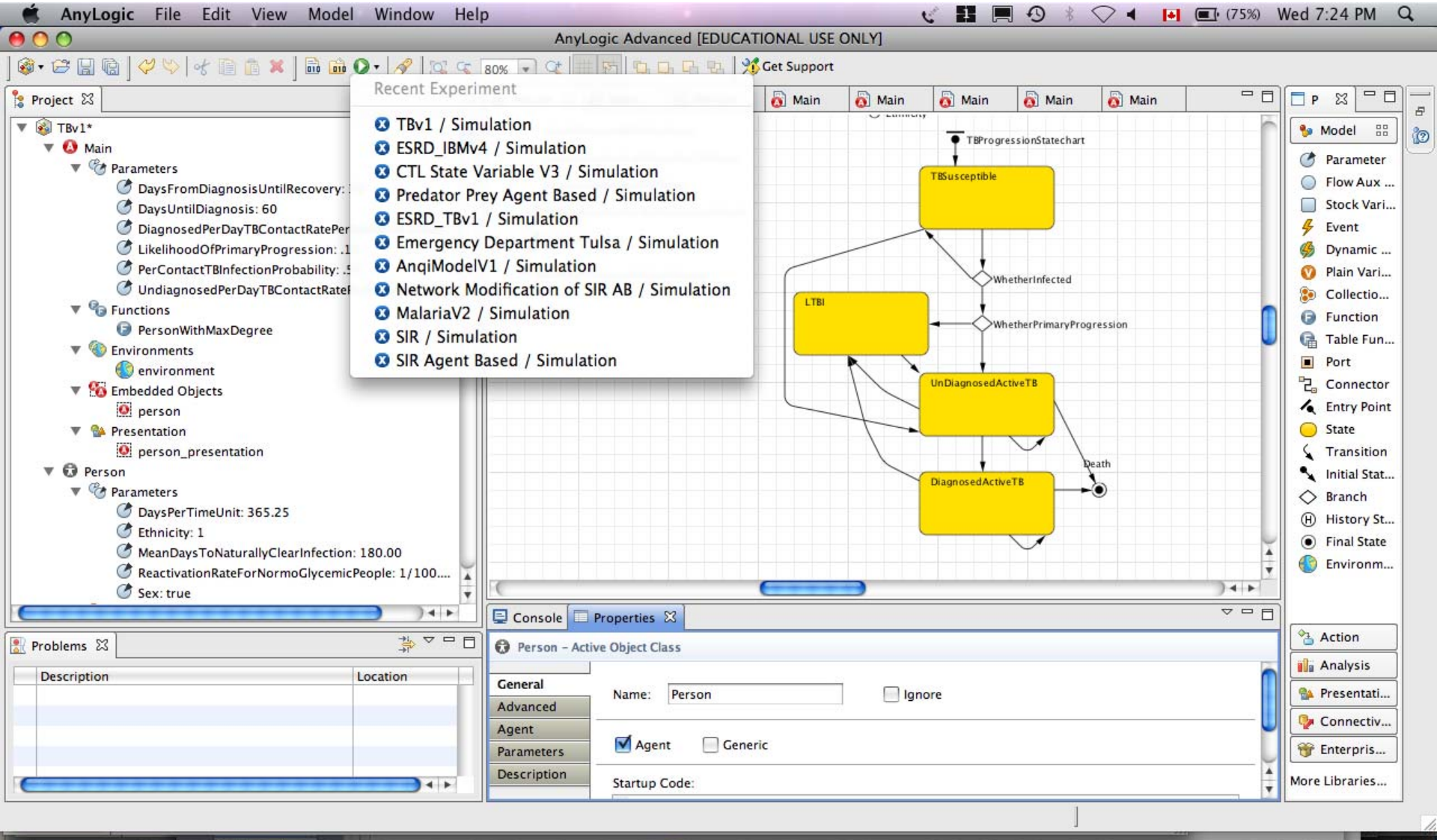
Displaying the Splash Screen (to Access Samples)



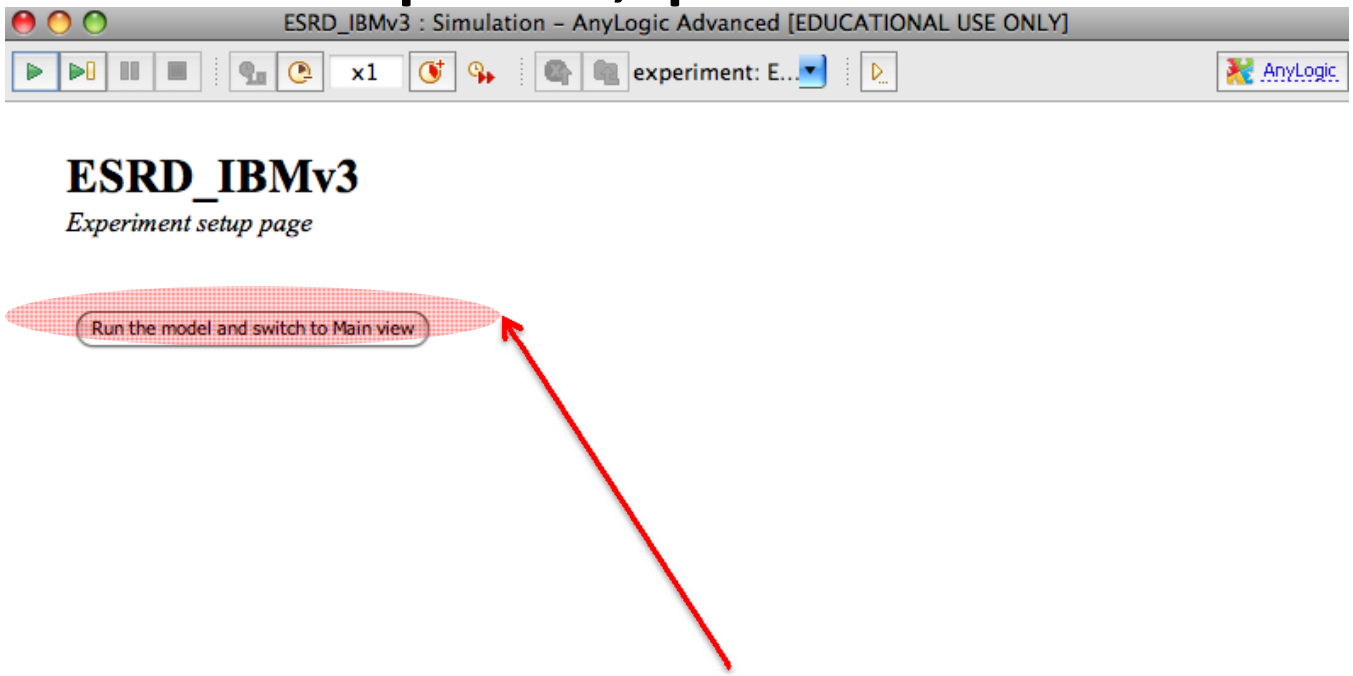
Also available via Help menu

Displaying the Splash Screen (to Access Samples)

Running a Model

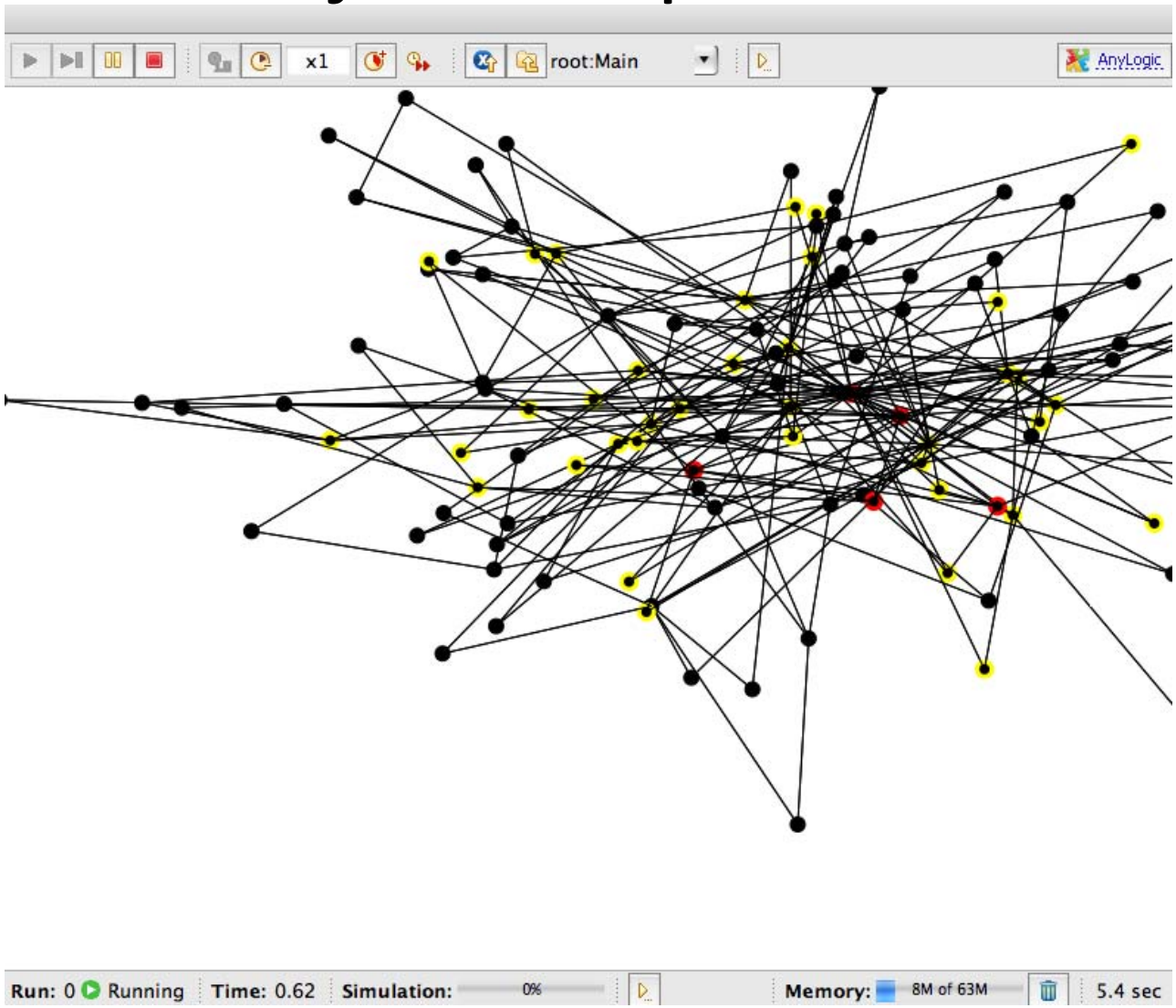


Experiment Set up
(Use to set speed, parameters via UI)

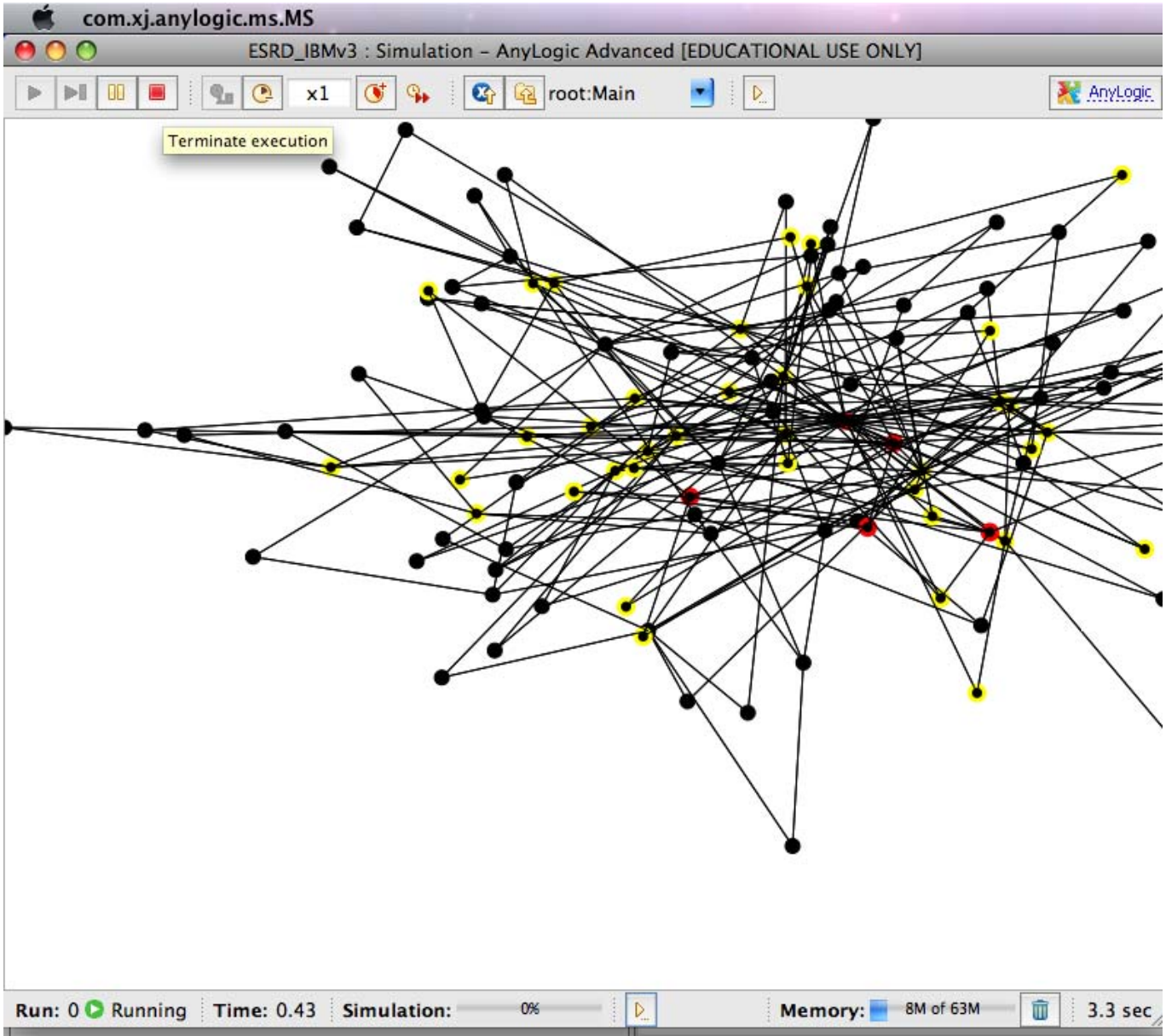


Press this button to switch to the model presentation display

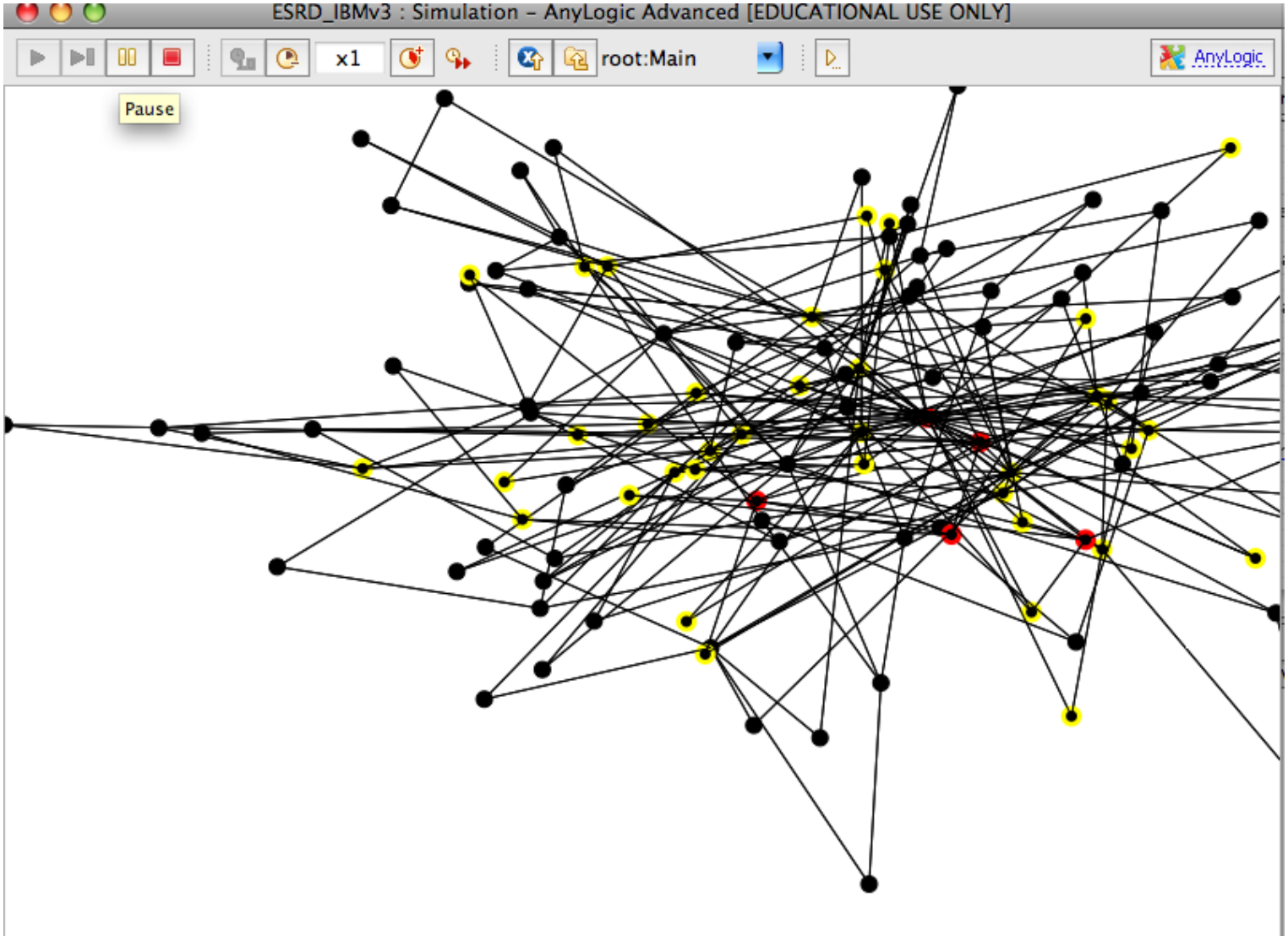
Presentation of the Model Main Object in Operation



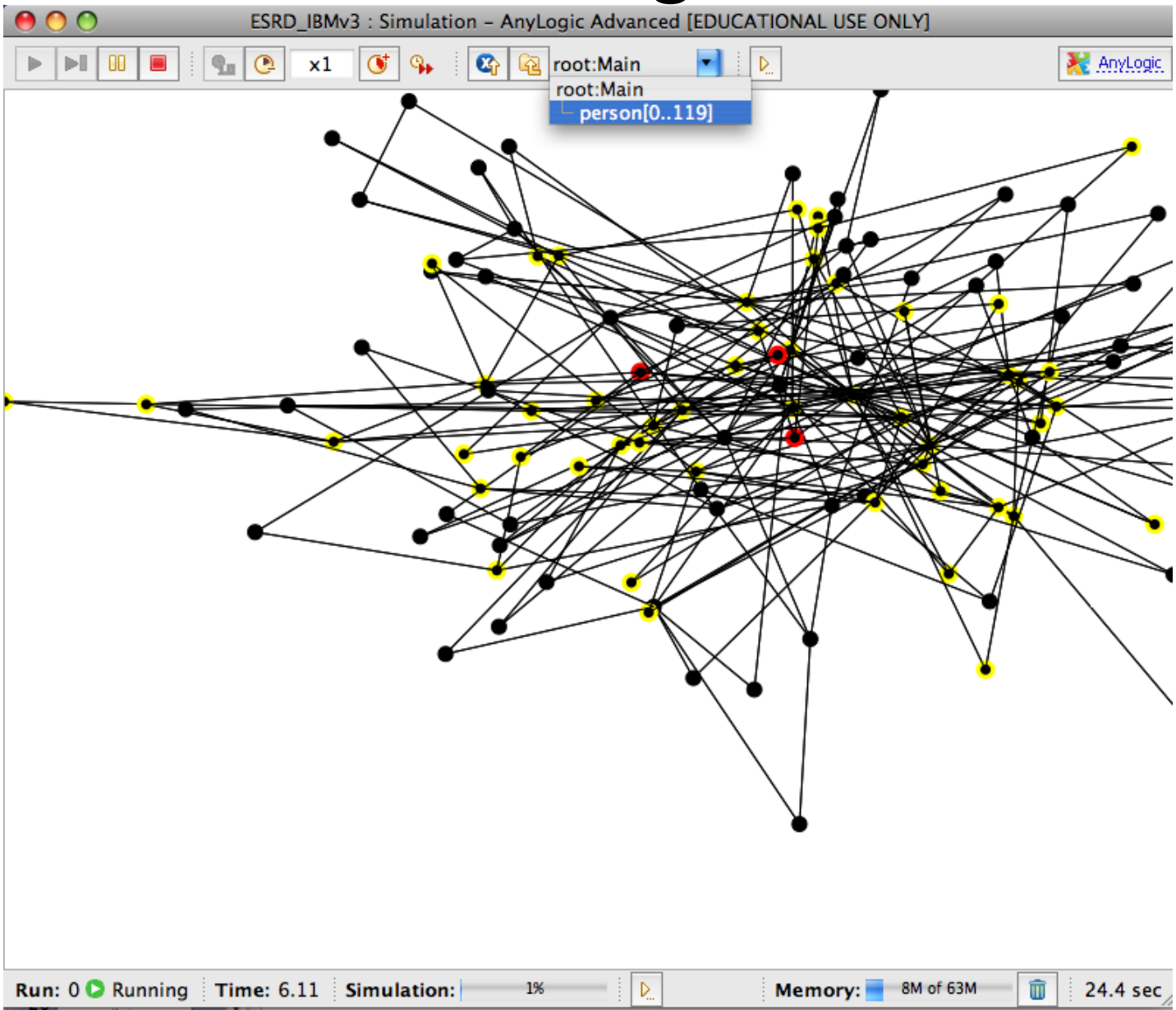
Terminating Model Execution



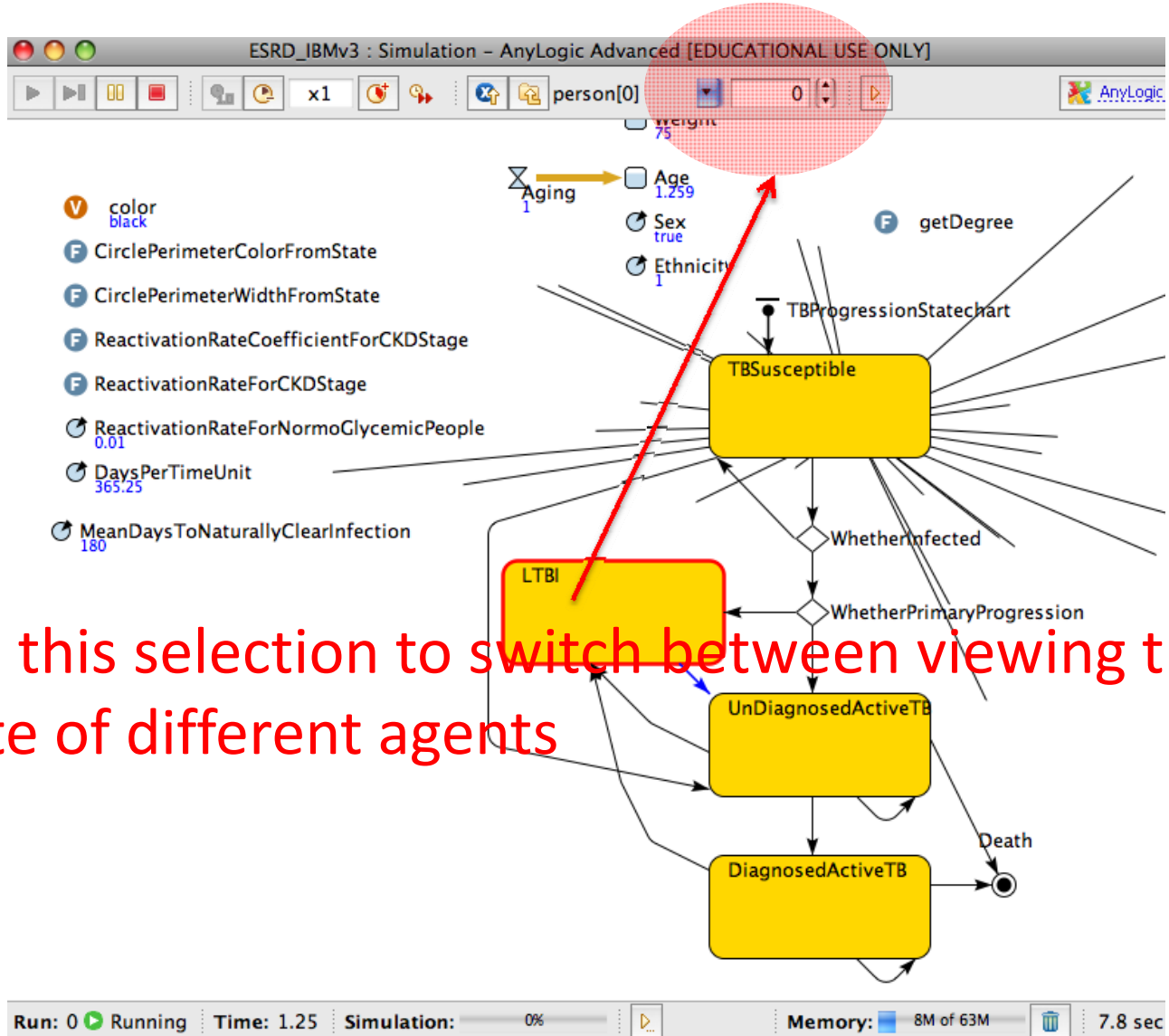
Pausing the Model



Drill Down from the Model to Particular Agents

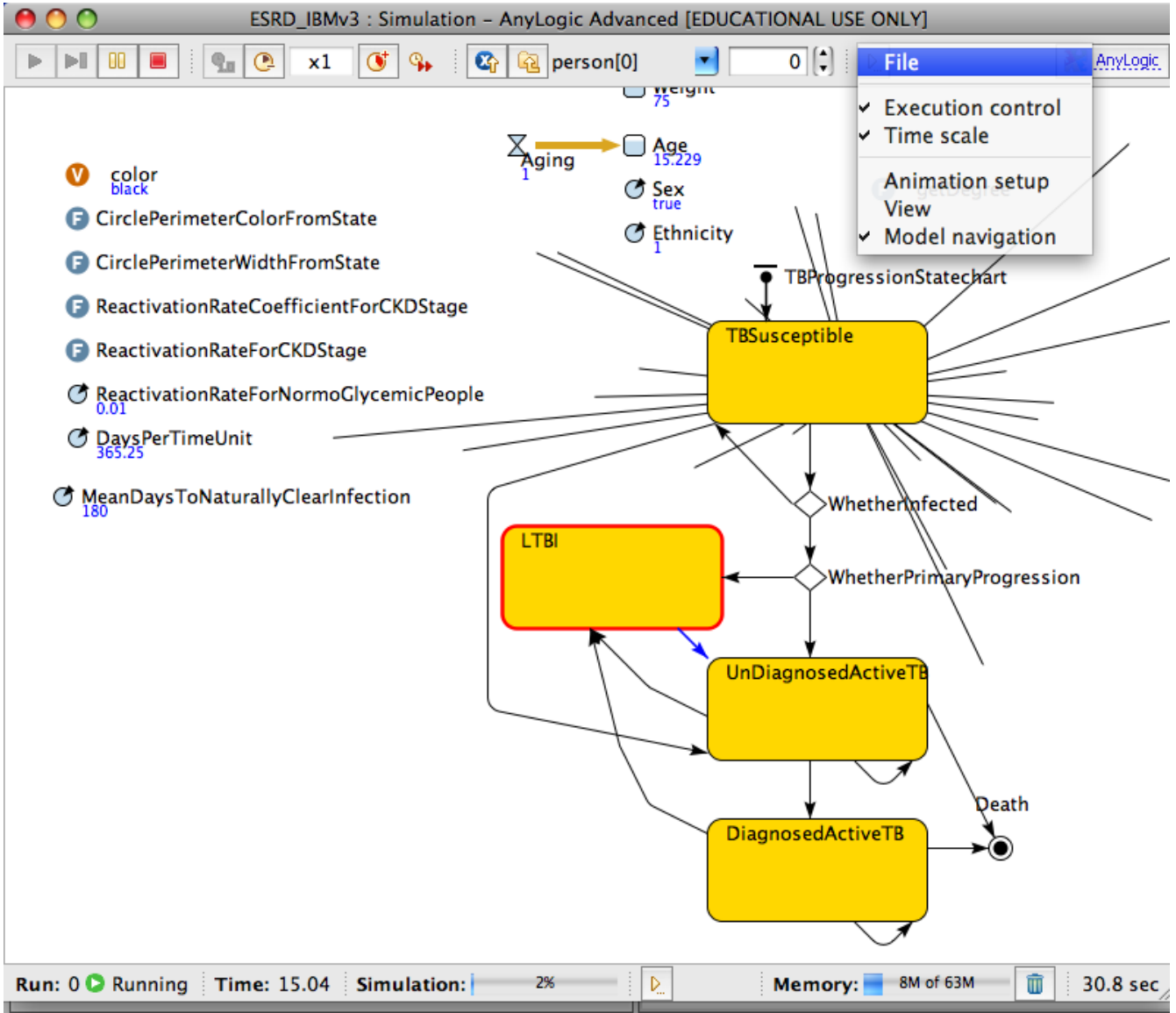


View of Agent State

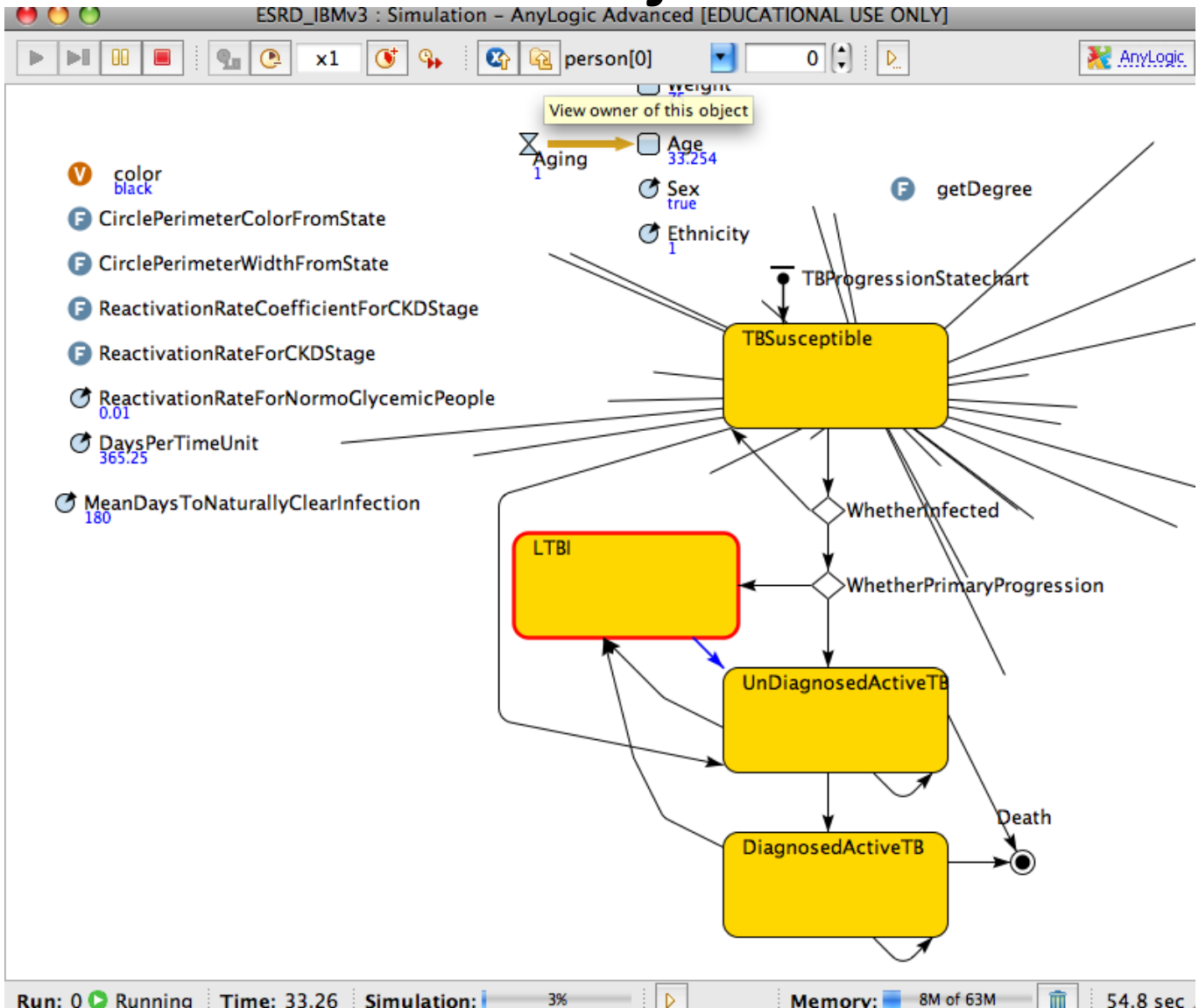


Use this selection to switch between viewing the State of different agents

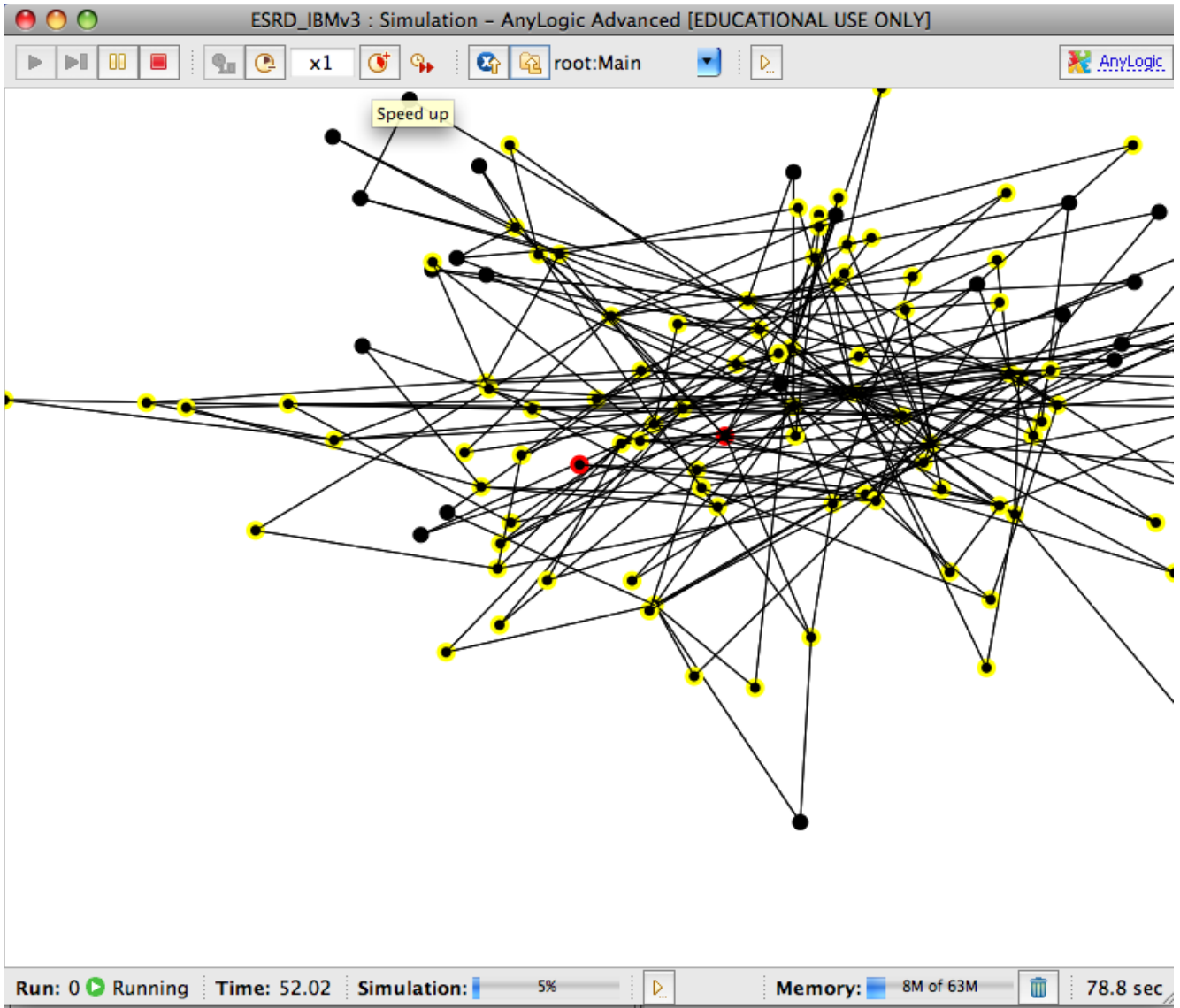
Customizing the Model Running User Interface



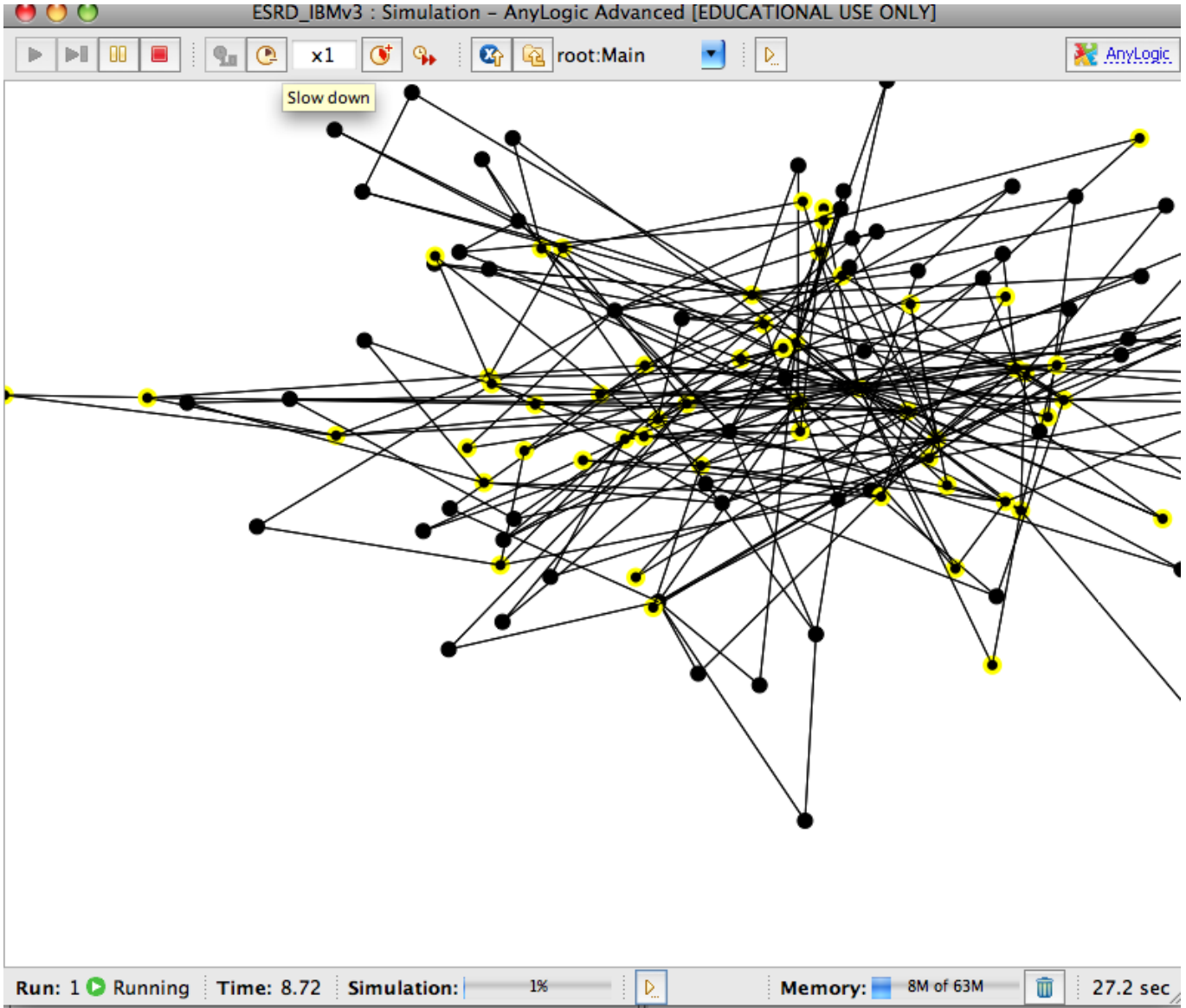
Switching Back to View the Main Object



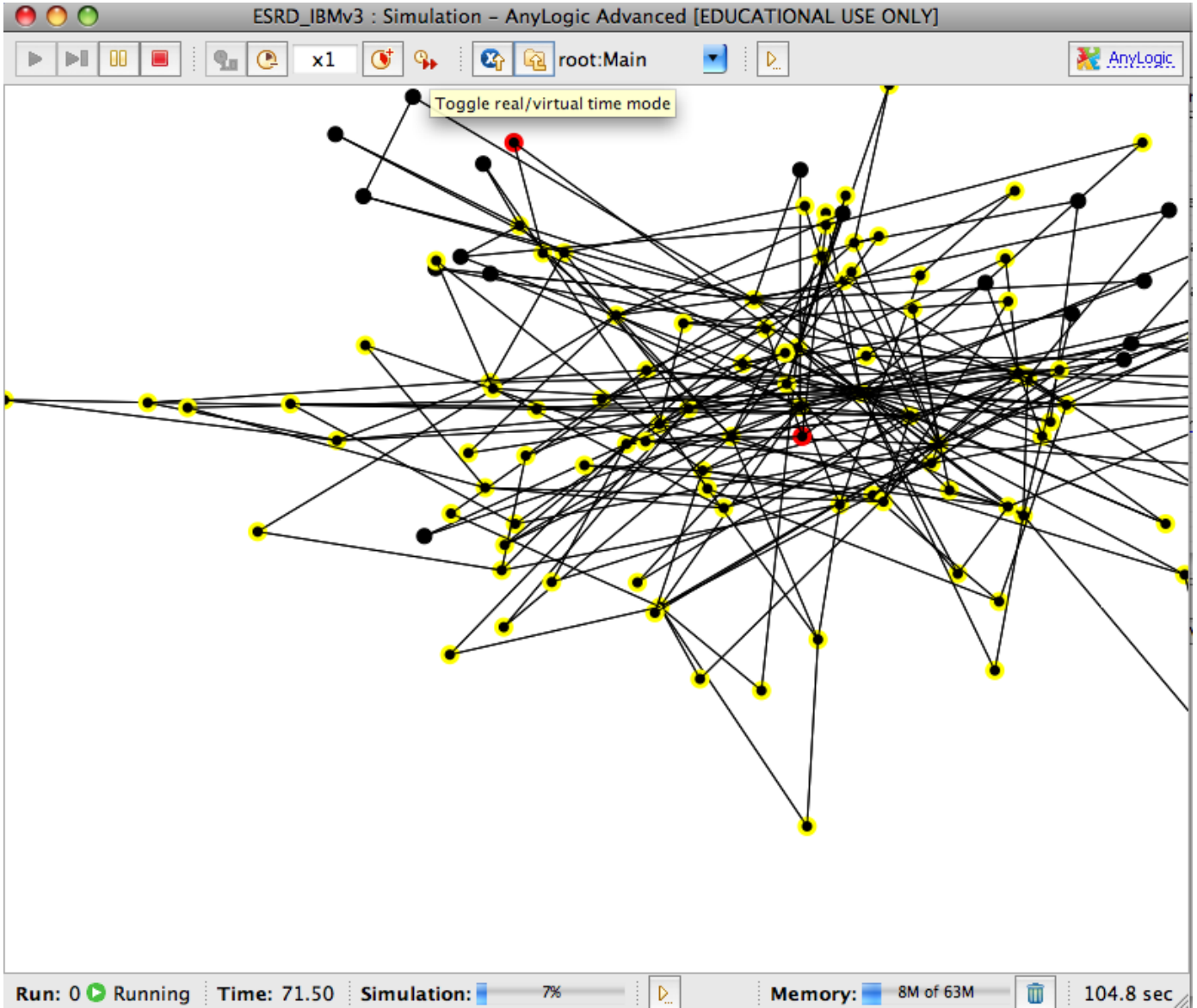
Controlling Simulation Speed (Speeding up)



Controlling Simulation Speed (Slowing Down)

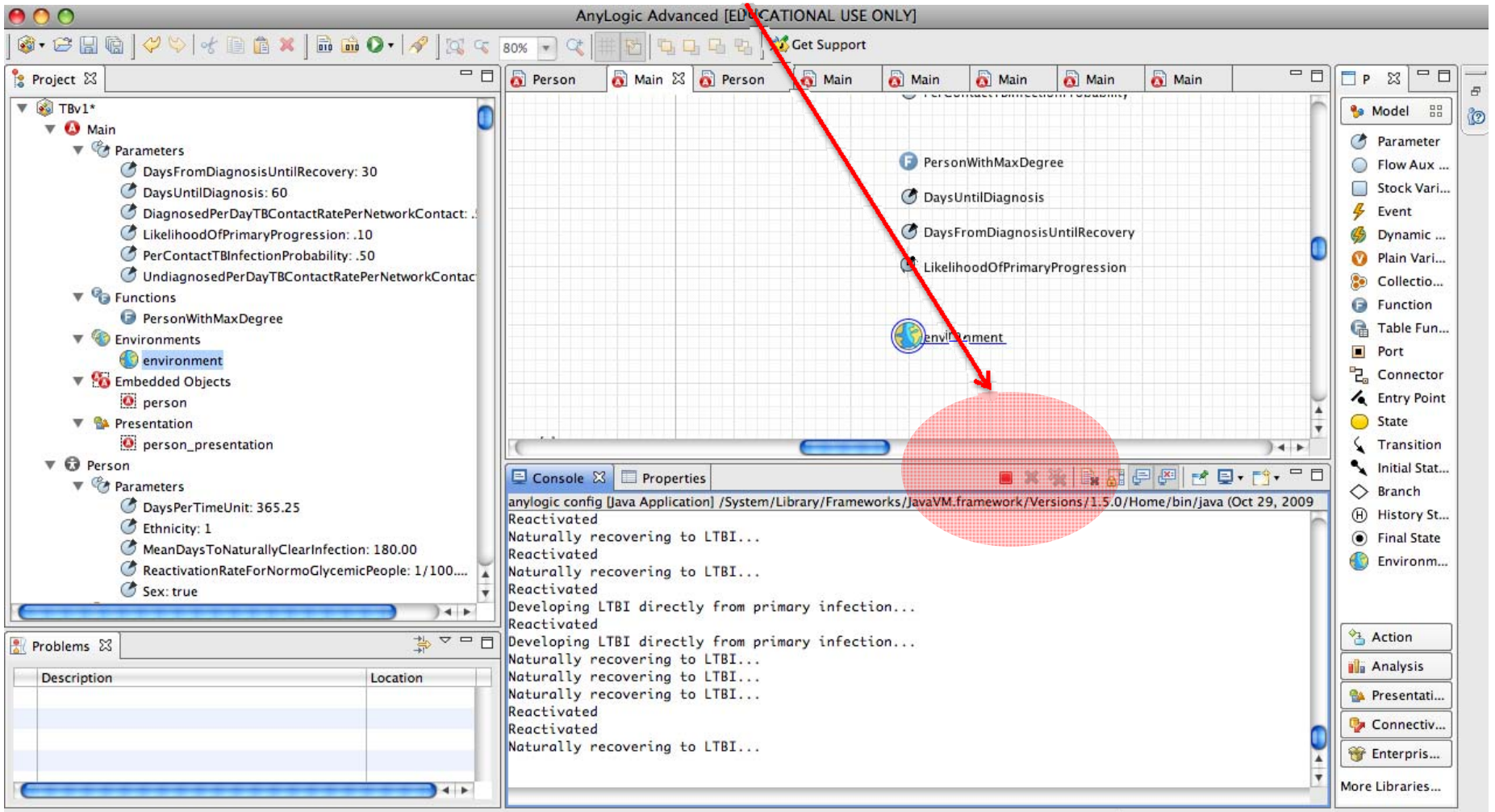


Toggling between Maximum and a Throttled Speed



Another Way to Terminate a Simulation

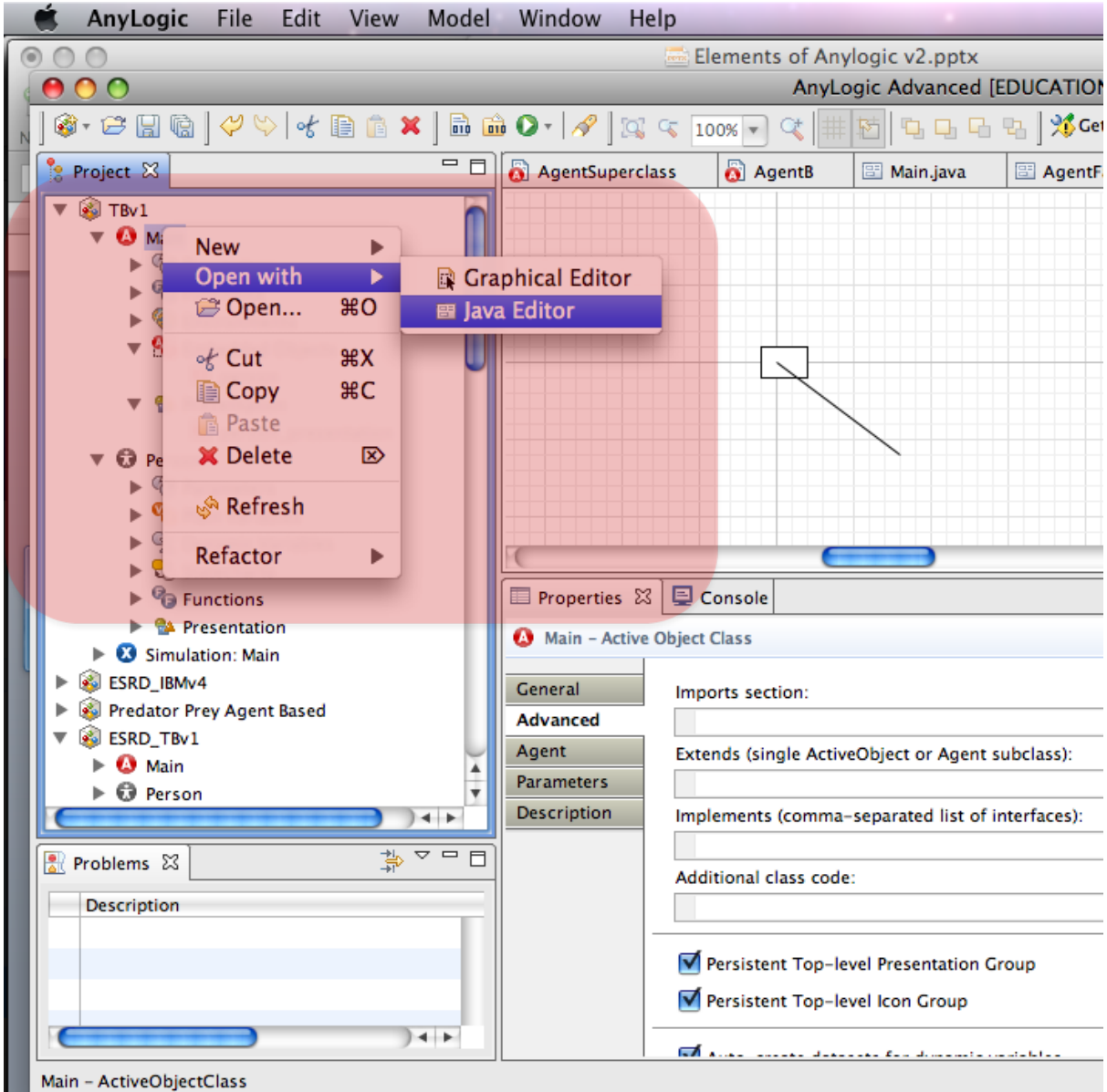
Use this Console “stop’” button to terminate the simulation



Inspecting the Java code

- As a step towards creating an executable representation of the code, AnyLogic creates a Java representation
  - If you want to see the Java code for a model, you will need to do a “build”
- Sometimes it can be helpful to look at this Java code
  - To find errors about which AnyLogic may be complaining
  - Advanced: To see how things are being accomplished or “work”

Requesting Viewing of Java Code



AnyLogic: Above & Below the "Hood"

- One of AnyLogic's greatest strengths is the presence of diverse & powerful *declarative* mechanisms for building models
  - These let you focus on the "what" you are modeling, rather than "how" it will be implemented
  - AnyLogic will take care of figuring out the "*how*"
  - This is in contrast to writing code in a general purpose computer language, which requires specifying more of the *how*
- For Anylogic, declarative mechanisms include statecharts, stock & flow diagrams, "action" flow charts & process maps
- Other familiar declarative mechanisms include spreadsheet formulas &vensim stock & flow diagrams.
- For most interactions with AnyLogic, you will be able to specify your intentions using these declarative mechanisms
- On occasion, you will need to write & look at Java code

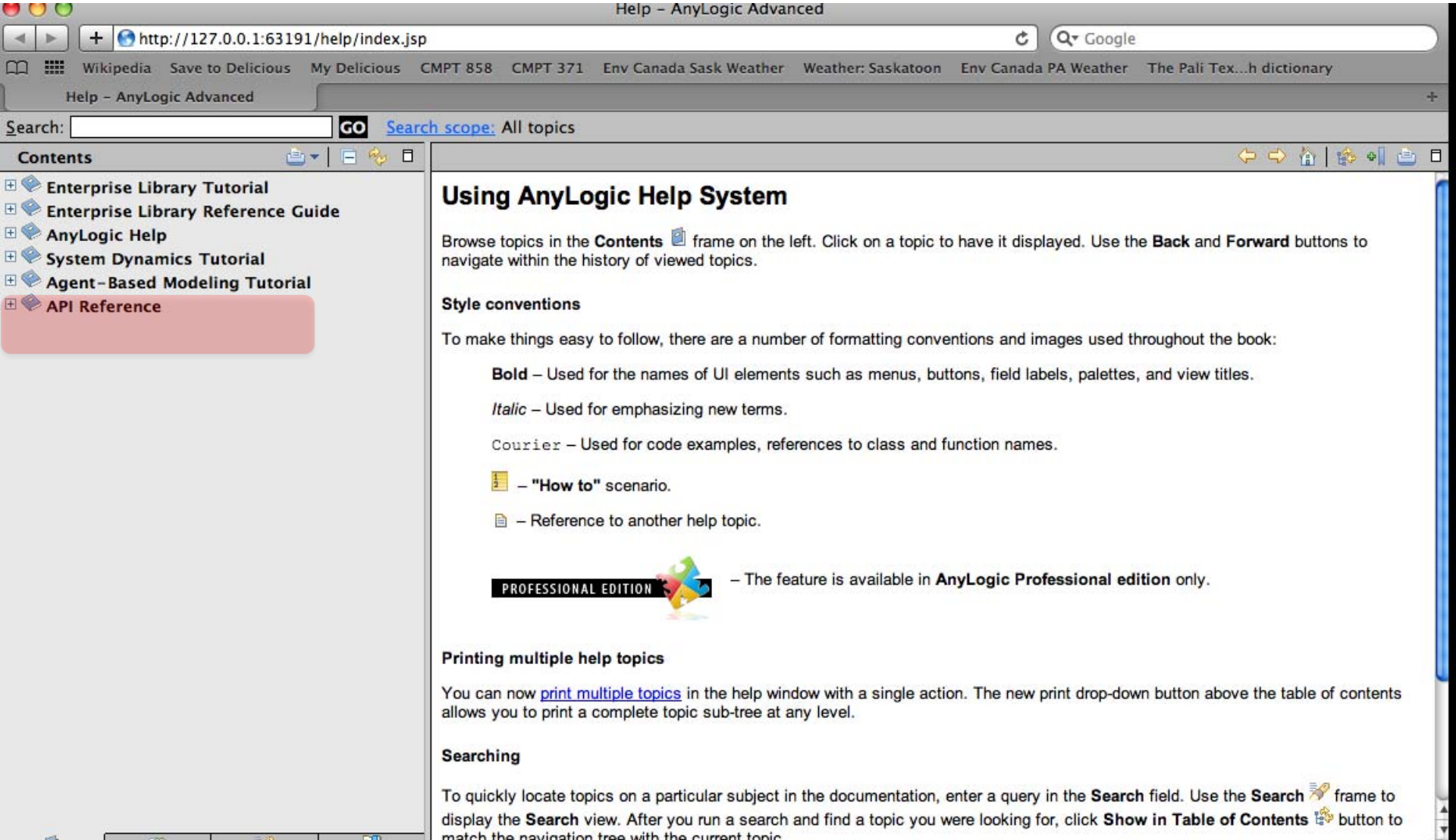
The Notion of a Code “Library”

- A “library” lets third parties (e.g. xjtek) share compiled code they have developed with others
- The classes built into our AnyLogic projects (e.g. Agent, ActiveObject, NetworkResourcePool, etc.) are contained in the library
- The available libraries that come with AnyLogic& Java have many additional components that can offer tremendous additional functionality
  - By tapping into this functionality, we can avoid having to write code ourselves
- To use a library, you need to know what is in it!

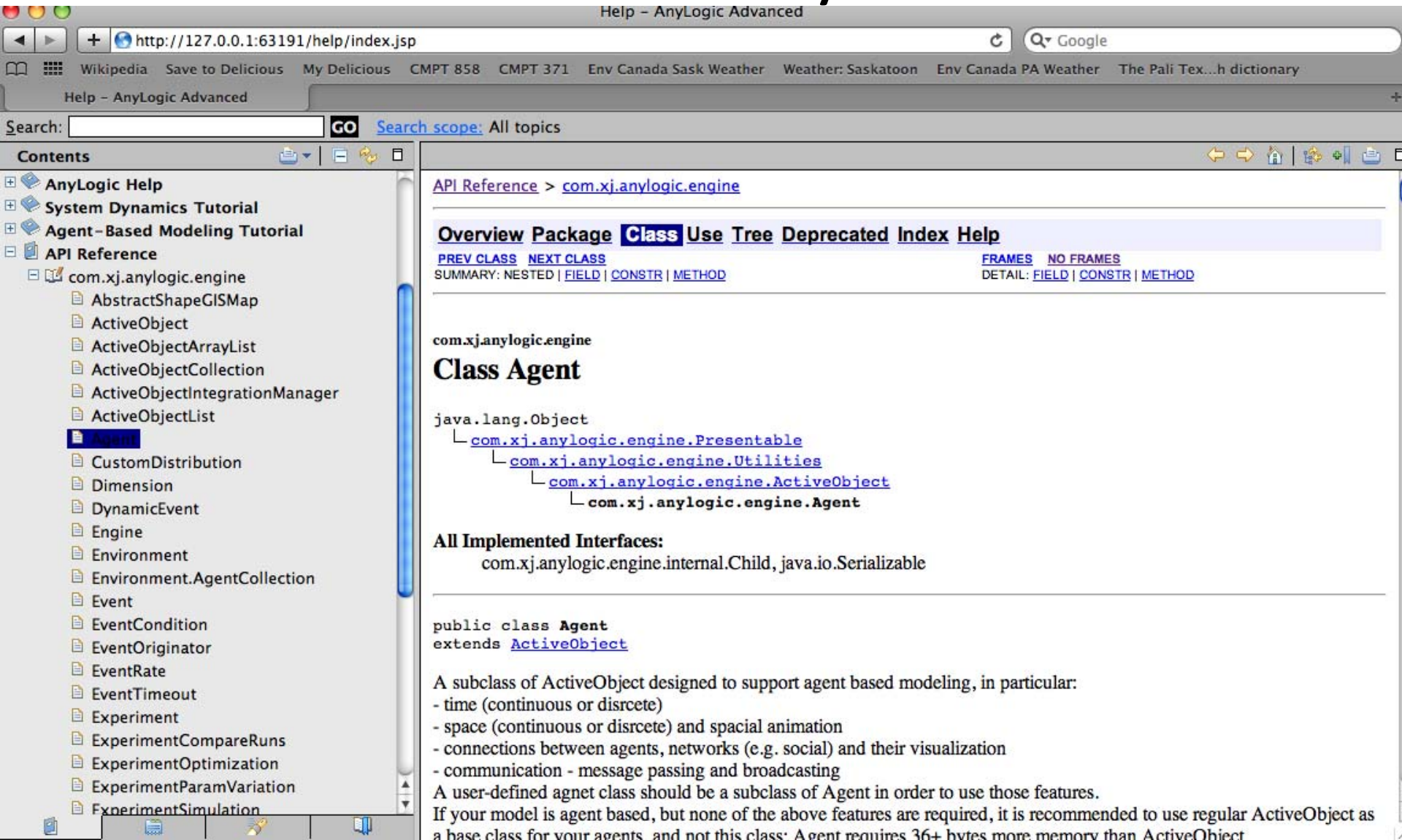
Getting to the AnyLogic Help

- Choose “Help”/”Help Contents”

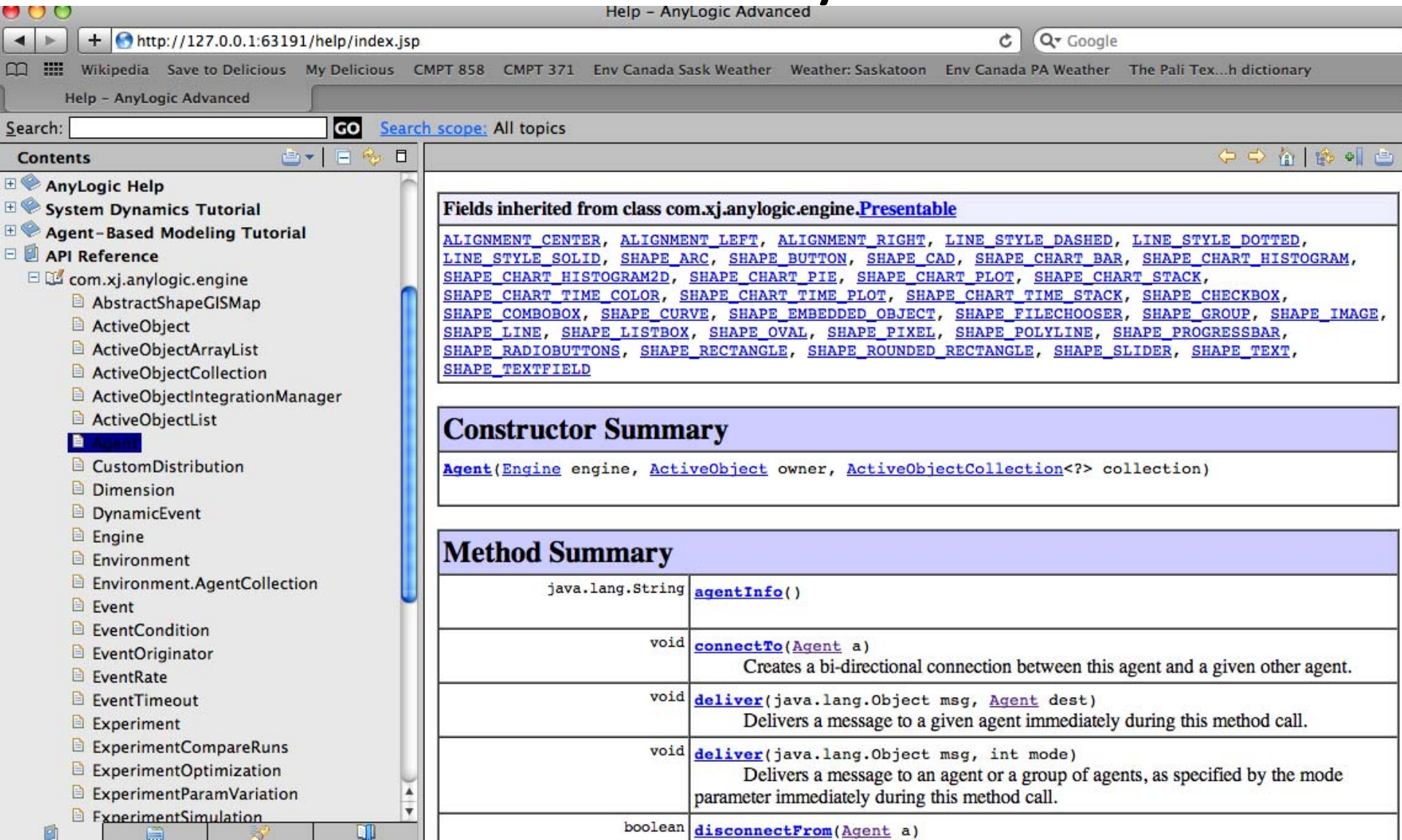
Getting Information on the Anylogic (Java) Libraries



Finding out Information Interfaces for Library Elements 1



Finding out Information
Interfaces for Library Elements 2



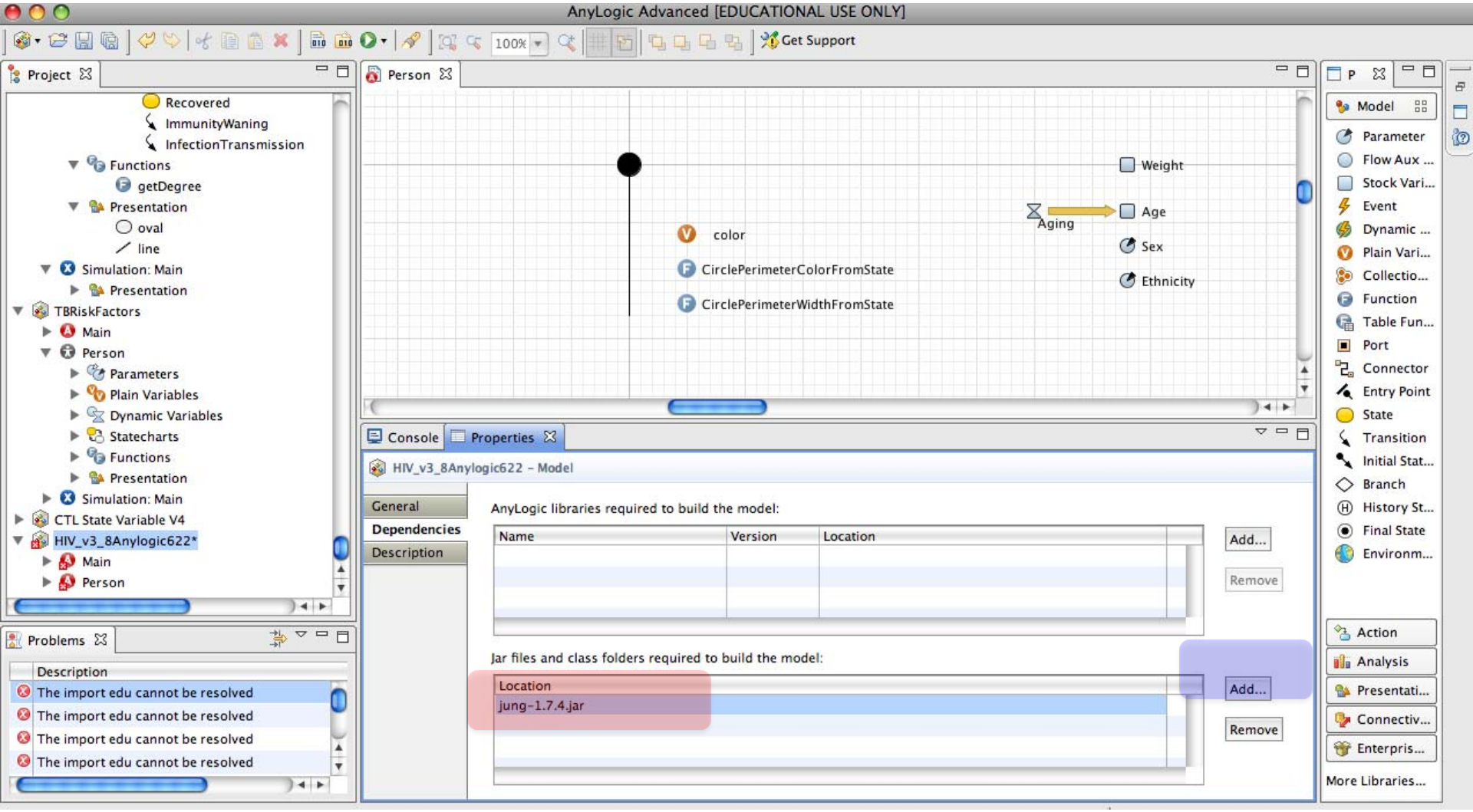
Using Libraries

- There are two major libraries that can be used “automatically”: Java libraries &AnyLogic libraries
- To use an object in the Java libraries, you will use an “import” statement

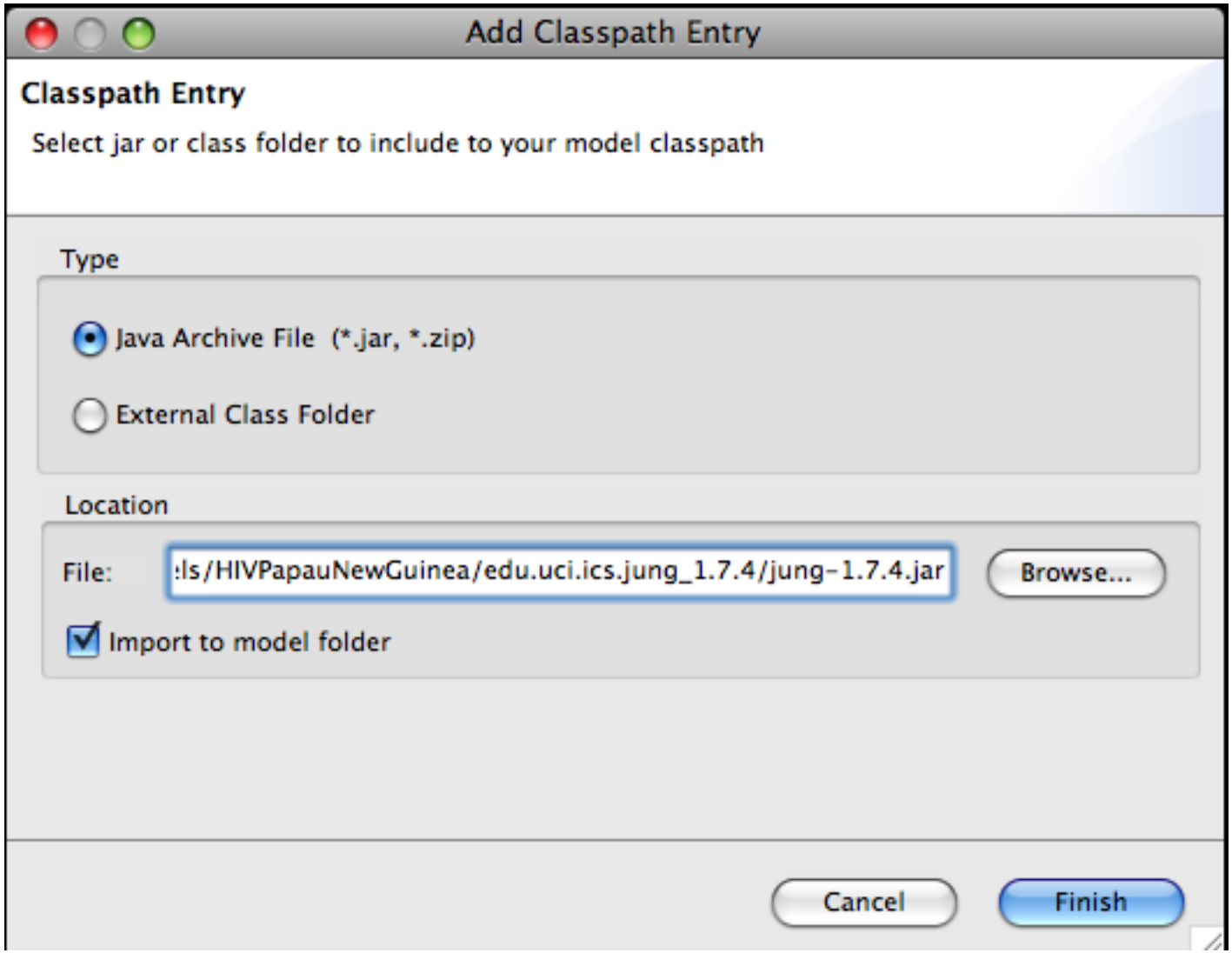
Using External Libraries

- There are tremendous numbers of 3rd party libraries available for Java
- The functionality associated with these libraries is incredibly diverse
- Many of these libraries are available for free; others are sold
- It is very easy to make use of the functionality of 3rd party libraries from AnyLogic
  - In order to do this, AnyLogic needs to “know about” the external library.

Adding External Libraries 1



Adding External Libraries 2



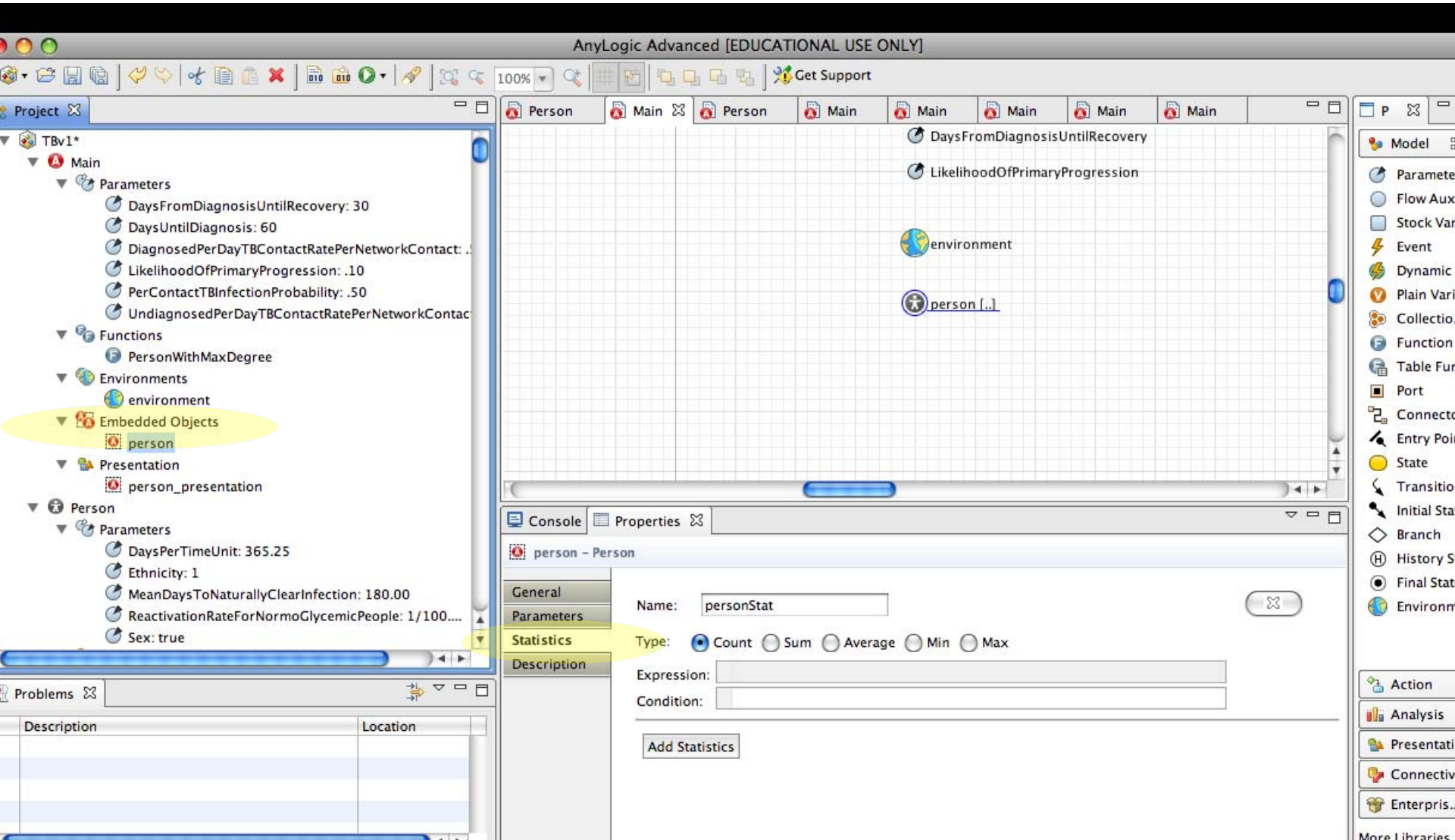
Recording of Results

- A frequent modeler need is to record some components of model state over time
  - State variables (e.g. stocks)
  - States of agents
  - Summaries of model state
  - We informally term this a “trajectory file”
- In contrast to Vensim (& other system dynamics packages), trajectory recording is not automatic
- AnyLogic does allow for
  - Definition of *DataSets* that record recent values of parameters
  - Statistics summarizing model state
  - Reporting on values of data sets as a graph or table

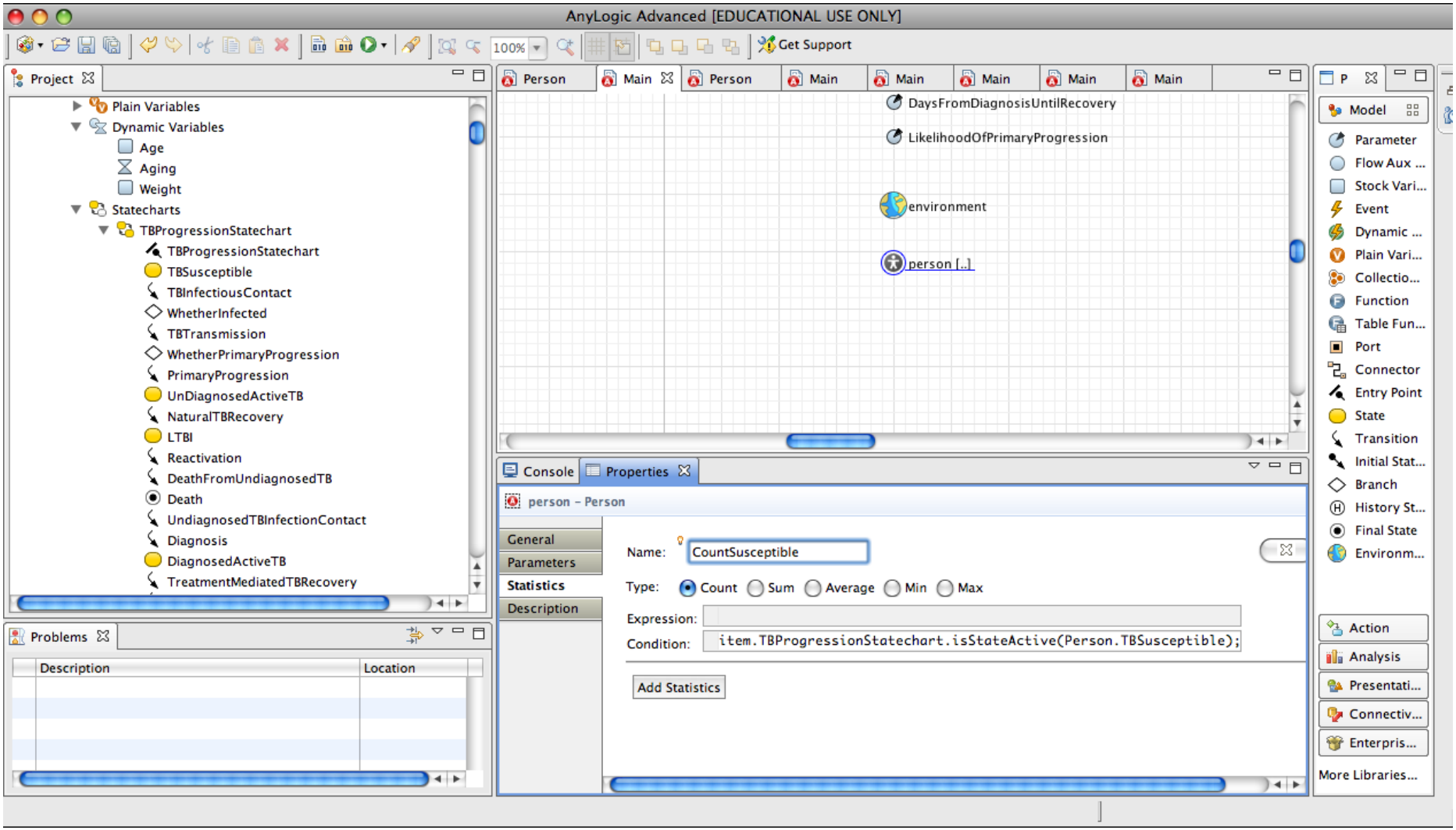
Statistics

- A population of agents can have associated statistics that calculate values
- Examples of things that can be computed with using AnyLogic's statistics
  - Count of agents in the population for which certain condition ("predicate") evaluates to true
  - Function of the values of some expressionover the population
    - Maximum value
    - Minimum value
    - Average value
    - Sum (total) over population
  - Statistics can be defined as properties of the population

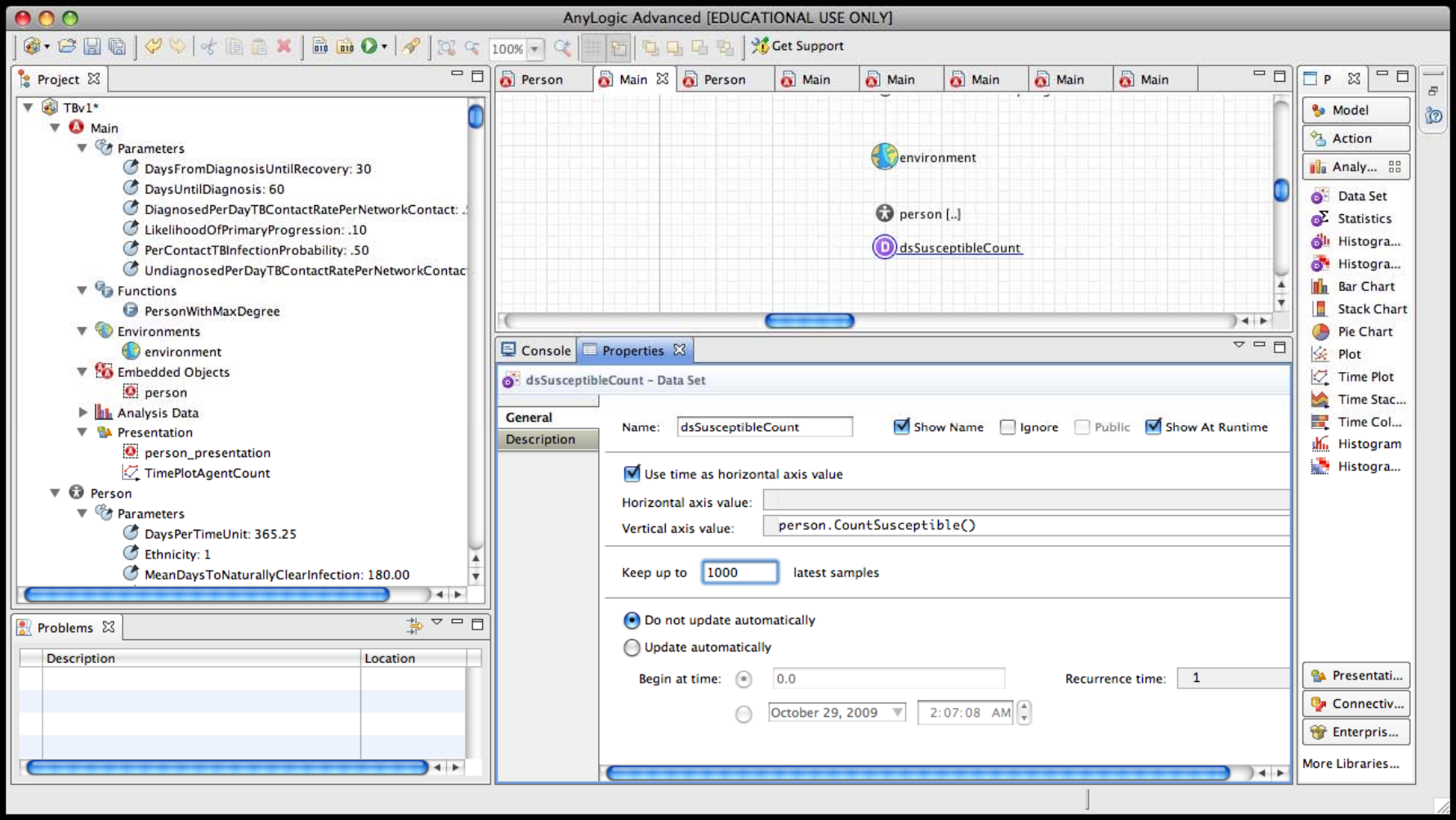
Statistics for Embedded People



Example Statistics



Output: Datasets



Datasets

- Datasets store recent values of some quantities from the model
- Datasets can be exported easily using custom code
  - This can simply call the dataset's to string method

Example code

FileOutputStreamfos = **new FileOutputStream(strOutputFilename);**

PrintStreamp = **new PrintStream(fos);**

p.println(datasetName.toString()); // outputs comma delimited values

Dataset Properties

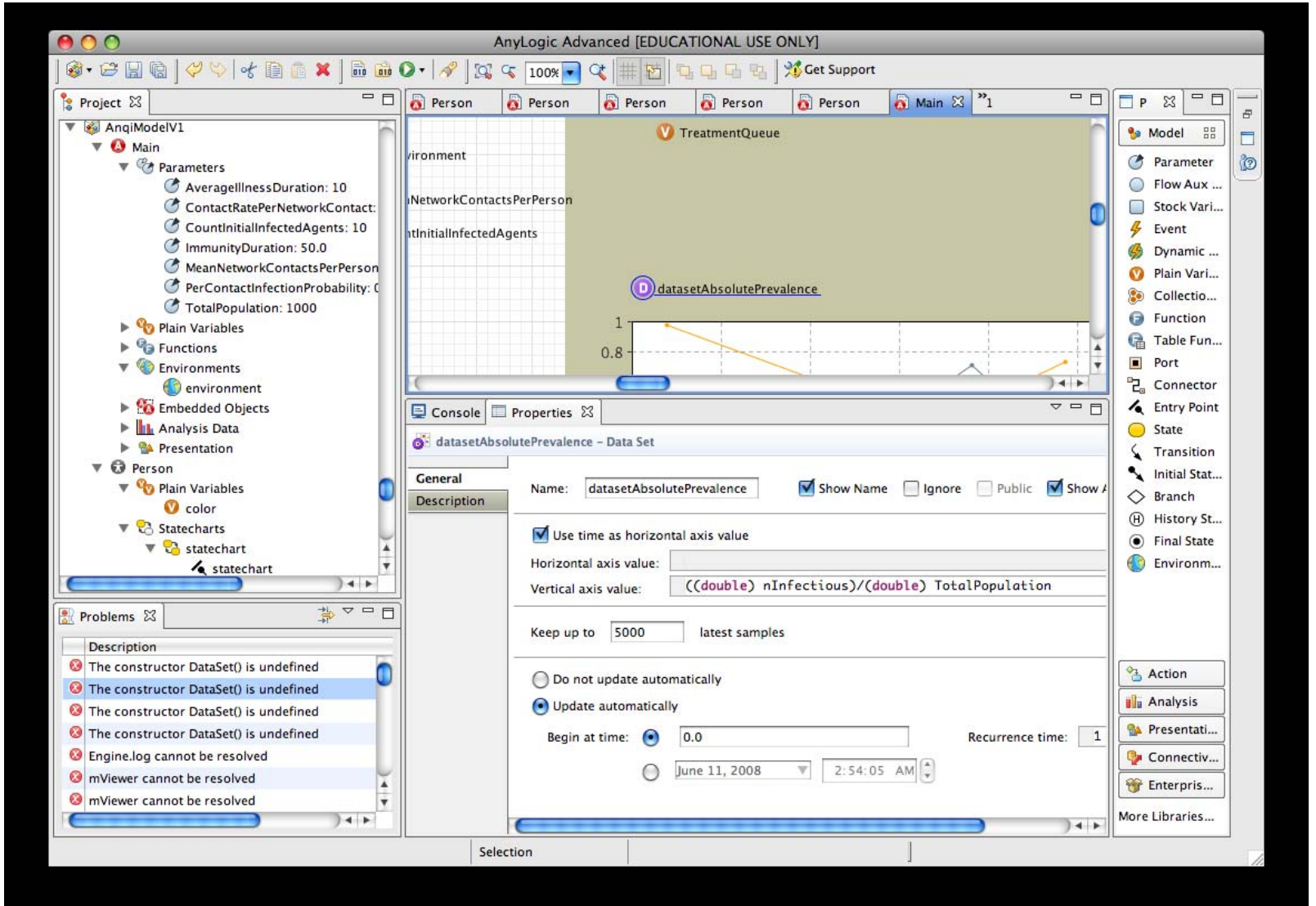


Chart Use of Datasets