

Slidename	Text
Introduction	Welcome to the Getting Started guide for IBM Rational Rhapsody for Java.
Slide 3	This viewlet shows you how to open the Java HomeAlarm model that was installed with Rhapsody, examine some diagrams to understand the model, and then animate the model to validate its behavior.
Slide 4	The sample models are all in a directory called Samples below your Rhapsody installation directory, which is usually: C:\Program Files\IBM\Rational\Rhapsody\7.5
Slide 5	In this viewlet we open the HomeAlarm model directly where it was installed with Rhapsody, and you can do the same as long as you aren't going to modify the model.
Slide 6	BUT if you are going to modify the model at all, or save it, then you should copy the complete HomeAlarm directory to somewhere where you can work on it, and open it from there.
Open the HomeAlarm model	Now open the HomeAlarm model.
Open the HomeAlarm model	Click here
Slide 8	Double click here
Slide 9	Double click here
Slide 10	This is the directory you would copy if you wanted to modify the HomeAlarm model. We're going to open the model from here, being careful not to save any changes we make.
Slide 10	Double click here
Slide 11	Click here
Slide 12	Click the Open button
Slide 13	Click here
Slide 13	Drag to here
Open a use case Diagram	The model is open in Rhapsody. Now we can open some diagrams to see what is in the model. We'll start with the main use case diagram.
Open a use case Diagram	Click here
Slide 15	Double click here
Slide 16	The diagram shows that there are two actors involved with the HomeAlarm, the homeowner and the intruder. The homeowner can arm the alarm, change the code, and so on. The intruder interacts with the system when it detects intrusion.
Open the feature editor	There is additional information in the features (or properties) of the symbols on the diagram. To open the feature editor for anything on a diagram, double-click on it.
Open the feature editor	Double click here
Slide 18	The feature editor has tabs for different types of information.
Slide 18	Click here

Slide 19	The Description tab is used for comments or additional information.
Slide 19	Click here
Open an Object Model Diagram	Next we'll open an Object Model Diagram (OMD).OMDs show the static structure of the classes and instances in an object-oriented software system, and the relationships between them.
Open an Object Model Diagram	Click here
Slide 21	Double click here
Slide 22	This OMD shows that the HomeAlarm model has two packages hardware and alarm.Package hardware contains the class IHardware, an interface which can be called to interact with the hardware. There are further OMDs for this package below the Packages node in the model view.Package alarm contains AlarmController which controls the alarm system.
Open a Statechart Diagram	Now let's open the Statechart Diagram of the RadioController class.
Open a Statechart Diagram	Click here
Slide 24	Click here
Slide 25	Click here
Slide 26	Click here
Slide 27	Right click here
Slide 28	Click here
Slide 29	Click here
Slide 29	Because we can't see all of the diagram, click the Zoom to Fit button.
Slide 30	States are shown as boxes with a label at the top (for example armed) with specified Java code (in the features of the state) to execute on entry and exit. We'll look at this next.States can be nested in other states, like exiting inside armed, and entering inside active.
Slide 30	Transitions between states are shown as red lines with an arrow at the destination state.A transition is followed when the specified trigger occurs and an optional guard condition evaluates to true. Transitions can optionally execute code if needed.The trigger, guard and code are shown in blue on the red transition line.
Examine state entry and exit Java code	Let's have a look at the features of some of the states, so we can see the Java code that is executed on entry and exit.
Examine state entry and exit Java code	Double click here
Slide 32	Code is shown on the General tab.
Slide 32	Click here
Slide 33	Click here
Slide 33	The armed state has some code executed on exit.Let's look at

	the intrusion state.
Slide 34	Here is the entry and exit code for the intrusion state. On entry it turns the siren and red led on, and on exit it turns the siren off.
Slide 35	Let's look at the action code on a transition.
Slide 35	Click here
Slide 36	This transition has no action code, but when needed it can be entered directly into the feature editor, or using a larger text editor window.
Slide 36	Click here
Start animation	Next we're going to animate this model to visualise and maybe debug the behaviour of this model of an intruder alarm system. First, let's close the diagrams that we have open.
Start animation	Click here
Slide 38	Click here
Slide 39	Click here
Slide 40	To build the animation, which includes a Java GUI, first select the AlarmController class.
Slide 40	Click here
Slide 41	Click here
Slide 42	Click here
Slide 43	The Instances category has appeared because Rhapsody is ready to animate once we click the Go button.
Slide 44	Click here
Slide 45	Here's the GUI that allows you to control the model animation. The buttons in the GUI all create events that the animated model responds to by executing transitions and their associated code. You can input the four-digit control code then press On or Off to arm or disarm the alarm. The alarm security code is 1234. Remember that number because you will need it to arm and disarm the alarm during the animation! You can click Door to simulate a door opening. When arming the alarm, you can open a door before the alarm becomes active. Once the alarm is active, if you click Door you have a few seconds to enter the code and press Off otherwise the siren goes off. You can click Move to simulate an intruder being detected, which immediately triggers the alarm siren. There are also red and green 'LED' indicators for status indication.
Open the animated statechart	Now the model is being animated, there is an instance of the RadioController class in the model. Let's have a look at its statechart, with animation adding a highlight to indicate the current active state.
Open the animated statechart	Click here
Slide 47	Right click here
Slide 48	Click here
Slide 49	Click here

Slide 50	The AlarmController instance is in its initial state (denoted by the transition from the red dot) which is the off state, so that state is highlighted.The highlight will move as the instance changes state in response to events, so that you can understand and debug the behavior of the model.
Create an animated sequence diagram	As well as animating a statechart, Rhapsody can animate a sequence diagram, which is a really good way to understand the interactions in the model as it executes.An animated sequence diagram (ASD) shows the messages passed between the lifelines on the diagram as the animation runs.There is a prepared empty ASD in the model that already has the lifelines we're interested in watching, so we'll open that diagram now.
Create an animated sequence diagram	Click here
Slide 52	Click here
Slide 53	Click here
Slide 54	Click here
Slide 55	Click the Open button
Slide 56	Now that the ASD is open, it will show the interactions that happen, and we'll come back and look at it later after we've watched the animated statechart as we arm and trigger the alarm.
Slide 56	Click here
Arm the alarm	Now use alt-tab to bring the GUI to the front
Slide 58	To arm the alarm, enter the four-digit code then click On.In case you forgot, the code is 1234.
Slide 58	Click here
Slide 59	Click here
Slide 60	Click here
Slide 61	Click here
Slide 62	Click here
Slide 64	The alarm is now armed so the armed state is highlighted.Within the armed state there are nested states.The nested statemachine is in the exiting state so that is highlighted too, but only until the EXIT_TIME timer expires when the transition to the active state is followed and the alarm is active. In the active state, if movement is detected the nested state will change from detecting to intrusion and the alarm siren will sound.If you're going to use the door to leave, you'd better click Door quickly!
Slide 64	Click here
Simulate intruder detection	We're going to simulate an intruder being detected.You might want to turn your PC sound volume up now, so you can hear the alarm siren!Take a deep breath and click Move.
Slide 67	Luckily for you (and your co-workers!) this model only sounds the siren for a few seconds.Phew, that was loud!

Simulate intruder detection	Click here
Slide 68	Let's look at the animated sequence diagram to see what the trace looks like.
Slide 68	Click here
Slide 69	Click here
Slide 70	Here's where the model sounded the siren because it detected intruder movement.
Disarm the alarm	Let's disarm the alarm while watching the animated sequence diagram.If you need to, use alt-tab to bring the GUI to the front again.Click Door to simulate getting into the area protected by the alarm.
Disarm the alarm	Click here
Slide 72	Use the same code 1234 then click Off.
Slide 72	Click here
Slide 73	Click here
Slide 74	Click here
Slide 75	Oh no! we entered the code too slowly because the timer for code entry expired and the siren went off again!Did you turn your PC volume down? We'd better finish the sequence to turn this pesky alarm off once and for all!
Slide 75	Click here
Slide 76	Click here
Exit Rhapsody	That's the end of this viewlet, so it's time to stop the animation and exit Rhapsody.Don't forget that you mustn't save the model if you opened it from the Rhapsody installation!
Slide 78	Click here
Slide 79	Click here
Slide 80	Click the No button
Slide 81	Click here
Slide 82	Click here
Slide 83	Click the No button
Conclusion	You have completed this viewlet! We hope that it has given you an insight into how to get started with IBM Rational Rhapsody for Java.Rhapsody has allowed you to observe the behavior of the HomeAlarm model by animating it, and you have seen how the animation works in real-time in this example.Now you can follow the steps in the viewlet to use Rhapsody to animate the HomeAlarm model yourself.The HomeAlarm is a small model, but Rhapsody works in just the same way with much larger models where understanding the behavior without animation would be extremely difficult or impossible. With Rhapsody it is simple to animate a model to learn or test its behavior.Please note that this viewlet has not shown you many of the features of Rhapsody that help to make it the leading Model-Driven Development tool for systems and software development.For

	more information on Rhapsody, Rational software, and IBM please see the resources and links on the next slide.
--	---