Agents- the inside story

- What is an Agent
- Agent Ports
- Agent Types, Instances and Threads
- Agent URLs: Communicating with agents
- Named objects and the Persistent Store
- How Agents Work
- The Agent class
- Examples

What is an Agent?

An agent consists of Java code, which describes what the agent does, and a *information store* which contains several items of information. Some of these items are created when the agent starts work. Other items may be created and added to the information store as the agent travels from place to place while performing its tasks.

For developers...

An agent is a collection of Java classes combined with a collection of persistent objects. It contains at least one class derived from Guideware.Agent.Framework.Agent. The Guideware Agent Framework provides other useful classes, some of which are described later on this page. Each agent also contains a collection of named objects, called the *persistent store*. This collection of objects is transported with the agent as it moves from computer to computer. Objects can be added and deleted from the persistent store at any time.

Agent Ports

Each computer that can send and receive agents maintains one or more agent ports. Each port manages several agents. The Agent Monitor takes care of tracking agents and sending and receiving them from one port to another.

Agent types, instances and threads

Agents are grouped into agent types. The Agent Registrar packages a collection of Java classes into a new agent type and registers it at an agent port. Then, the Agent Monitor can create and track many instances of that agent type.

Each agent instance may create several threads. Each agent thread executes autonomously, and may perform activities on different computers. An agent thread can exchange information with other agent threads. It can also synchronize its work with other threads.

Agent URLs: Communicating with Agents

Each agent instance and thread is identified by a unique handle that is comprised of its registration port, its

IPR2025-00715
DATABRICKS EX1013 Page 1

type, its instance identifier and its thread name. This handle is represented as a unique URL. Therefore, any agent can be identified and contacted by a Web browser, a Java applet, an ActiveX control, or any other Internet-aware application. A URL for an agent looks like:

/Agent/<Registration Port>/<Type>/A<Instance Id>/<Thread name>

The strings in angle brackets (<>) will be replaced by actual information pertinent to the agent being referenced in the URL. The URL for an agent of type "CustomerSupportRep", registered at a server named "guideware.com" at an agent port named "Support", with an Id of 55 and a thread name of "main" would be:

/Agent/guideware.com/Support/CustomerSupportRep/A55/main

Agent monitors service HTTP requests. When an agent monitor receives a GET request for a URL of the type described above, it locates the agent and forwards the request to it.

Named Objects and the Persistent Store

Named objects provide a way for an agent to expose its functionality to a Web browser, or an application. Agents can accept commands, or collect information from a user. To access an object named "hello" in an agent, write a URL that looks like:

/Agent/guideware.com/Support/CustomerSupportRep/A55/main/hello

When an agent receives a request for a particular object it responds in different ways, depending on the class of the requested object. If an AgentWebForm or an AgentWebPage object is requested, that web page or form is returned back to the requester. If any other object is requested, its toString() method is invoked, and the resulting text is returned to the requesting application. If an object derived from Command is requested, its onCommand() method is invoked. The onCommand() method can perform any processing and then return a Web page or a text string to the requester.

Objects may be added to the persistent store at any time. When an agent is created, it usually adds a number of AgentWebPage, AgentWebForm and Command objects to the persistent store, so that it can communicate with users. An agent may also insert objects into its persistent store simply to transport them back to its home port. For example, an agent may collect comments on a document, an electronic signature, or the results of a web search, and transport them back.

How agents work

The process an agent performs is described by one or more Activities. Classes derived from the Activity class are used to specify what an agent does after it reaches a new place, after it restarts, or after another agent thread completes. Activity objects are also inserted into the persistent store, so that the activity can be automatically started by the agent framework.

Agents present information to a human user via classes derived from AgentWebPage. An AgentWebPage can compose a Web page dynamically, or it can carry a Web page that was composed previously in a Web design tool. AgentWebPages can contain fields which allow dynamic updates. Each time the web page is sent, these fields are replaced by current information generated by the agent.

IPR2025-00715
DATABRICKS EX1013 Page 2

Agents collect information from users via AgentWebForms. An AgentWebForm is composed in a Web design tool and stored by the agent when it is created. When users submit information via an form, the agent updates the information in the persistent store, and calls the *onPost()* method of the AgentWebForm object to process that information.

Agents accept commands from users via Commands. A class derived from Command overrides the *onCommand()* method. When a Web browser requests an object derived from Command, the onCommand() method is called. It can do any desired processing, and then return a Web page to the requesting browser.

Java applets and applications can communicate with agents and Agent Monitors via the two proxy classes, AgentServerProxy and AgentProxy.

The Agent class

Each agent must contain a class derived from Guideware.Agent.Framework.Agent. Two key methods should be overridden by the derived class, namely *Agent.createType()* and *Agent.initInstance()*. When an agent type is registered by the Agent Registrar, the *createType()* method is called. This is the appropriate place to:

- Create objects and insert them into the persistent store. AgentWebPages, AgentWebForms, Commands and Activities must be inserted into the persistent store at this time.
- Read in pre-composed web pages and forms. The Agent Registrar uses a Security Manager which permits agents to access the file system, so you can read files at this time. The Agent Monitor uses a more restrictive Security Manager which will not permit an agent class to access the file system.

The *initInstance()* method is called when a new instance of an agent is started. In this method, you specify what each agent instance should do when it starts. For example, if you wish the agent to move to a new place as soon as starts, specify it here.

Examples

All of this will be clearer by looking at the examples . Check them out.

Copyright) 1996, 1997 Guideware Corporation. All rights reserved.
Information in this document is subject to change without notice.
Other products and companies referred to herein are trademarks or registered trademarks of their respective companies or mark holders.

IPR2025-00715
DATABRICKS EX1013 Page 3