Agent examples

Overview

This page contains several example agents. If you have not read the page on Agents, it's a good idea to do so now.

- The first "Hello, Web" agent demonstrates how an agent uses a Command to receive and respond to requests from a Web browser
- The second "Hello, Web" agent demonstrates how an agent can send a pre-composed Web page in response to a browser request
- The Suspend agent demonstrates how an agent suspends itself and schedules an activity to be performed at a later time.
- The Travel agent demonstrates how an agent receives input via a AgentWebForm object, and also how it moves from one place to another

A "Hello, Web" Agent

The HelloWeb agent demonstrates how an agent responds to a request from a Web browser. This agent that contains a single Command object, named "hello". When "hello" is queried by a Web browser, it sends back a web page that displays "Hello, World !" (of course). HelloWeb contains two classes, Hello and HelloAgent. The Hello class is derived from Guideware.Agent.Framework.Command. It overrides the Command.goTo() method, which is invoked when an object of this class is queried by a Web browser. Hello's implementation of onGet() creates a new Web page which contains "Hello, World" and sends it back to the browser.

```
import Guideware.Agent.Framework.*;
import java.io.*;

class Hello extends Command
{
        public void onCommand(int convId) // respond to a get request
        {
                try {
                        HelloAgent agent = (HelloAgent)getAgent();
                        AgentWebPage page = new AgentWebPage( "Hello agent", "Hello world");
                        page.send( getAgent(), convId);
                } catch ( Exception e) {
                        System.out.println( "getAgent() failed.");
                }
        }
}
```

The HelloAgent class overrides both the Agent.createType() method, and the Agent.initInstance() method. It creates a new Hello object and adds it to its persistent store. This object is recreated by the agent framework when the agent is restarted at a new place, or after being suspended. The override of Agent.initInstance() simply prints out an informative message.

```
import Guideware.Agent.Framework.*;
import java.io.*;

class HelloAgent extends Agent
{
        public boolean createType()
        {
                System.out.println( "Creating a new type: HelloWeb");
                Hello hello = new Hello();
                AgentWebPage helloPage = new AgentWebPage();
                try {
                        add( hello, null, "hello");
                        return true;
                }
                catch (Exception e) {
```


IPR2025-00714
DATABRICKS EX1014 Page 1

```
                        System.out.println( "Adding a persistent object failed");
                        return false;
                }
        }

        public boolean initInstance()
        {
                System.out.println("Creating a new instance: HelloWeb");
                return true;
        }
}
```

Another Hello Agent

This Hello agent is slightly different from the previous one. It demonstrates how an agent can carry a pre-designed Web page with it and send to a browser in response to a request.

In its createType() method, this agent creates a new AgentWebPage object and adds it to its persistent store. It also reads in and stores a pre-composed Web page from the file *hello.htm*.

```
import Guideware.Agent.Framework.*;
import java.io.*;

class HelloAgent extends Agent
{
        public boolean createType()
        {
                System.out.println( "Creating a new type: HelloAgent");
                Hello hello = new Hello();
                AgentWebPage helloPage = new AgentWebPage();
                try {
                        helloPage.fromFile( "\\Samples\\Hello\\hello.htm");
                        add( hello, null, "hello");
                        add( helloPage, null, "helloPage");
                        return true;
                }
                catch (Exception e) {
                        System.out.println( "Adding a persistent object failed");
                        return false;
                }
        }

        public boolean initInstance()
        {
                System.out.println("Creating a new instance: HelloAgent");
                return true;
        }
}
```

The Hello class contains a single method, onCommand() as in the previous example. In this case, the OnGet() method does not create a new Web page. It looks up the Web page which was created and added to the persistent store in the createType() method, and returns it to the browser.

```
import Guideware.Agent.Framework.*;
import java.io.*;

class Hello extends Command
{
        public void onCommand(int convId) // respond to a get request
        {
                try {
                        HelloAgent agent =  (HelloAgent)getAgent();
                        AgentWebPage page = (AgentWebPage)agent.lookup( "helloPage");
```


IPR2025-00714
DATABRICKS EX1014 Page 2

```
                        page.send( getAgent(), convId);
                } catch ( Exception e) {
                        System.out.println( "getAgent() failed.");
                }
        }
}
```

The Suspend Agent

The Suspend agent demonstrates how an agent suspends itself and schedules an activity to be performed at a later time. The suspend class is shown below. When its run() method is invoked, it composes a new status string that contains the current type and date, and updates the agent status with a call to agent.setState().

```
import Guideware.Agent.Framework.*;
import java.io.*;
import java.util.*;

class AfterSuspend extends AgentActivity
{
        public void run()
        {
                try {
                        SuspendAgent agent = (SuspendAgent)getAgent();
                        Date now = new Date();// get the date now
                        String string = "This agent has been running since ";
                        string += now.toString();
                        agent.setState( string);
                }
                catch (Exception e) { // not in the persistent store
                }
        }
}
```

The Suspend class suspends the agent, with a single activity, named "after" to be performed when the agent is re-started.

```
import Guideware.Agent.Framework.*;
import java.io.*;

class Suspend extends Command
{
        public void onCommand(int convId) // respond to a get request
        {
                AgentWebPage page = new AgentWebPage( "Suspend agent", "I'm suspending my self for now");
                try {
                        page.send( getAgent(), convId);
                        SuspendAgent agent = (SuspendAgent)getAgent();
                        agent.setState("This agent has been suspended at least once");
                        Activity activity = (Activity)agent.lookup( "after");
                        agent.suspend( activity);

                } catch (Exception e) {
                        System.out.println( "Suspend failed. ");
                }
        }
}
```

The SuspendAgent is somewhat more complex than what we have seen before. Like the other classes derived from agent, the createType() method adds activities and commands to the persistent store. The SuspendAgent class also overrides the read() and write() methods of Agent, because it needs to save and retrieve the String data member, m_state, each time it is moved or suspended. (note on serialization).

```
import Guideware.Agent.Framework.*;
import Guideware.Agent.*;
```


IPR2025-00714
DATABRICKS EX1014 Page 3

```
import java.io.*;

public class SuspendAgent extends Agent
{
        public boolean createType()
        {
                System.out.println( "Creating a new type: Suspend");
                Suspend suspend = new Suspend();
                Hello hello = new Hello();
                AfterSuspend after = new AfterSuspend( );
                try {
                        add( suspend, null, "suspend"); // add to root
                        add( hello, null, "hello");
                        add( after, null, "after");
                        return true;
                }
                catch (Exception e) {
                        System.out.println( "Adding a persistent object failed");
                        return false;
                }
        }

        public boolean initInstance()
        {
                System.out.println("Creating a new instance of Suspend");
                return true;
        }

        public void write( GwDataOutputStream stream)
                throws IOException
        {
                super.write( stream);
                stream.writeString( m_state);
        }

        public void read( GwDataInputStream stream)
                throws IOException
        {
                super.read( stream);
                m_state = stream.readString();
        }

        public String myState() { return m_state; }
        public void setState( String state) { m_state = state; }
        private String m_state = "This agent has not been suspended.";

}
```

The Travel Agent

The Travel agent illustrates a few more uses of the Guideware Agent Framework Classes. This agent presents a form to the user when it is requested to "go" to another place. Once a user supplies the domain name of a computer and the name of an agent place on that computer, the agent goes to that place. Before leaving, it sends back a Web page confirming its destination. Lets examine the Go class first.When it is commanded, it returns a web form that the agent had previously stored.

```
import Guideware.Agent.Framework.*;
import java.io.*;

public class Go extends Command
{
        public void onCommand(int convId) // respond to a get request
        {
                try {
                        AgentWebForm placeform = (AgentWebForm)getAgent().lookup( "placeInfo");
```


IPR2025-00714
DATABRICKS EX1014 Page 4

```
                        placeform.send( getAgent(), convId);

                } catch (Exception e) {
                        System.out.println( "go failed. ");
                }
        }
}
```

The PlaceInfoForm class overrides the onPost() method of AgentWebForm. This method is invoked when a HTTP POST request is received by the agent. Before invoking this method, the agent framework updates the values of objects in the persistent store with data in the POST request. So, the onPost() method has access to the information provided by a user.

As you see below, the onPost method does several things. It uses the substitute() method to update a AgentWebPage that is contained in the persistent store and sends it to the application that sent the HTTP POST message. Then it goes to the agent place specified by the user.

```
import java.io.*;
import java.util.*;
import Guideware.Agent.*;
import Guideware.Agent.Framework.*;

public class PlaceInfoForm extends AgentWebForm
{
        public boolean onPost( )
        {
                try {
                        TravelAgent agent = (TravelAgent)getAgent();
                        PersistentString place = (PersistentString)agent.lookup( "GoPlace");
                        PersistentString host = (PersistentString)agent.lookup( "GoHost");
                        agent.setState("This agent has travelled atleast once");
                        AgentWebPage response = (AgentWebPage)agent.lookup("response");
                        response.substitute(); // put current values into the document
                        response.send( agent, conversationId());
                        agent.go( host.toString(), place.toString()); // go with no activities
                        return true;
                }
                catch ( Exception e) {
                        System.out.println( "Agent failed to go");
                        e.printStackTrace();
                        return false;
                }
        }

}
```

The TravelAgent class does not do anything much different from what you have seen before. The one difference is that it reads in a pre-designed document from response.htm, and a pre-designed form from placeform.htm

```
import Guideware.Agent.Framework.*;
import Guideware.Agent.*;
import java.io.*;

public class TravelAgent extends Agent
{
        public boolean createType()
        {
                System.out.println( "Creating a new type: Travel");
                Hello hello = new Hello();
                Go go = new Go();
                OnArrival onArrival = new OnArrival();
                PlaceInfoForm placeInfo = new PlaceInfoForm();
                PersistentString goHost = new PersistentString();
                PersistentString goPlace = new PersistentString();
```

```
                AgentWebPage response = new AgentWebPage();
                try {
                        placeInfo.fromFile( "\\Travel\\placeform.htm");
                        response.fromFile( "\\Travel\\response.htm");
                        add( hello, null, "hello");
                        add( go, null,"go");
                        add( onArrival, null, "onArrival");
                        add( response, null, "response");
                        add( placeInfo, null, "placeInfo");
                        add( goHost, placeInfo, "GoHost");
                        add( goPlace, placeInfo, "GoPlace");
                        return true;
                }
                catch (Exception e) {
                        System.out.println( "Adding a persistent object failed");
                        e.printStackTrace();
                        return false;
                }
        }

        public boolean initInstance()
        {
                System.out.println("Creating a new instance of Travel");
                return true;
        }

        public void write( GwDataOutputStream stream)
                throws IOException
        {
                super.write( stream);
                stream.writeString( m_state);
        }

        public void read( GwDataInputStream stream)
                throws IOException
        {
                super.read( stream);
                m_state = stream.readString();
        }

        public String myState() { return m_state; }

        public void setState( String state) { m_state = state; }

        private String m_state = "This agent has not gone anywhere else.";

}
```

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.

IPR2025-00714
DATABRICKS EX1014 Page 6