WAITER AGENT

Data

```
public enum AgentState
        {DoingNothing, SeatingCustomer, GetOrder, TakeOrderToCook, ServeFood, GetCheck,
TakeCheckToCustomer, CleanTable, Break};

        public AgentState state = AgentState.DoingNothing;//The start state

        public enum Event
        {HeLeft, GoToCashier, GotCheck, NewCustomerToSeat, DoneSeating, customerReady,
GotOrder, FoodReady, customerDone, OutOfFood, WantABreak, TakeABreak};

        private Event event = Event.DoneSeating;
        private Event currentEvent = Event.DoneSeating;

        public enum CustState {LeftEarly, other, NeedCheck, BringCheck, NewOrderNeeded,
Seating, Seated, ReadyToOrder, Ordered, WaitingForFood, OrderOut, Eating, Done, Leaving,
Gone};

        public List<MyCustomers> myCustomers
        = new ArrayList<MyCustomers>();

        private List<Event> allEvents = new ArrayList<Event>();

        public int tablenumber;
        private HostAgent host;

        private CashierAgent cashier;

        private CookAgent cook;

        public boolean busy = false;

        private MyCustomers CurrentCustomer = new MyCustomers(new
CustomerAgent("Frank"), -1, this);

        private Semaphore atTable = new Semaphore(0,true);
        private Semaphore atCook = new Semaphore(0,true);
        String name;
```

```
private Menu currentMenu = new Menu();

public WaiterGui waiterGui = null;

public class Menu {
	String optionOne = "Pizza";
	String optionTwo = "Steak";
	String optionThree = "Salad";
	String optionFour = "Chicken";
	double costOne = 8.99;
	double costTwo = 15.99;
	double costThree = 5.99;
	double costFour = 10.99;

	Menu() {

	}

	double GetCost(String choice) {
		if (choice == "Pizza") return costOne;
		else if (choice == "Steak") return costTwo;
		else if (choice == "Salad") return costThree;
		else return costFour;
	}

	String ChoosePizza() {
		return optionOne;
	}

	String ChooseSteak() {
		return optionTwo;
	}

	String ChooseSalad() {
		return optionThree;
	}

	String ChooseChicken() {
		return optionFour;
	}
}

public class MyCustomers {
```

```
CustomerAgent customer;
int tableNumber;
String order;
CustState currentState;
WaiterAgent waiter;
double bill;
Check CheckRepublic;
boolean owed;

MyCustomers(CustomerAgent customer, int tableNumber, WaiterAgent w) {
	this.tableNumber = tableNumber;
	this.customer = customer;
	this.waiter = w;
	owed = false;
}

void setOwed(boolean t) {
	owed = t;
}

boolean getOwed() {
	return owed;
}

void setCheck(Check k) {
	CheckRepublic = k;
}

Check getCheck() {
	return CheckRepublic;
}

void setState (CustState s) {
	currentState = s;
}

CustState getState () {
	return currentState;
}

double GetBill() {
	return bill;
}
```

```
        void setTableNumber (int n) {
                tableNumber = n;
        }

        int getTableNumber () {
                return tableNumber;
        }

        void setOrder (String o) {
                order = o;
                bill = currentMenu.GetCost(o);
        }

        String getOrder () {
                return order;
        }

        WaiterAgent getWaiter() {
                return waiter;
        }

        void setCustomer(CustomerAgent cust) {
                customer = cust;
        }

        CustomerAgent getCustomer() {
                return customer;
        }
}
```

Messages

```
        public void msgOutOfHere(CustomerAgent c) {
                for (int i=0;i<myCustomers.size();i++) {
                        if (myCustomers.get(i).getCustomer()==c) {
                                //myCustomers.remove(i);
                                myCustomers.get(i).setState(CustState.LeftEarly);
                                break;
                        }
                }
                state=AgentState.DoingNothing;
```

```
                event=Event.HeLeft;
                allEvents.add(event);
                stateChanged();
        }

        public void msgCheckPlease(CustomerAgent cust) {
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getCustomer() == cust)
myCustomer.setState(CustState.NeedCheck);
                }
                event = Event.GoToCashier;
                allEvents.add(event);
                stateChanged();
        }

        public void msgHereIsCheck(CustomerAgent cust, Check check) {
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getCustomer() == cust) myCustomer.setCheck(check);
                }
                event = Event.GotCheck;
                allEvents.add(event);
                stateChanged();
        }

        public void msgCantBreakNow() {
                print("I can't take a break now!");
        }

        public void msgGetNewOrder(CustomerAgent cust) {
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getCustomer() == cust)
myCustomer.setState(CustState.NewOrderNeeded);
                }
                event = Event.OutOfFood;
                allEvents.add(event);
                stateChanged();
        }

        public void msgNewCustomerToSeat(CustomerAgent cust, int table){
                event = Event.NewCustomerToSeat;
                allEvents.add(event);
                CurrentCustomer= new MyCustomers(cust,table, this);
                CurrentCustomer.setState(CustState.Seating);
```

```
                myCustomers.add(CurrentCustomer);
                tablenumber=table;
                stateChanged();
        }

        public void msgLeavingTable(CustomerAgent cust) {
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getCustomer() == cust) {
                                myCustomer.setState(CustState.Done);
                        }
                }
                event=Event.customerDone;
                allEvents.add(event);
                stateChanged();
        }

        public void msgAtTable() {//from animation
                atTable.release();// = true;
                stateChanged();
        }

        public void msgReadyToOrder(CustomerAgent cust) {
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getCustomer() == cust)
myCustomer.setState(CustState.ReadyToOrder);
                }

                event=Event.customerReady;
                allEvents.add(event);
                stateChanged();
        }

        public void msgOrderFood(CustomerAgent cust, String food) {
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getCustomer() == cust) {
                                myCustomer.setState(CustState.Ordered);
                                myCustomer.setOrder(food);
                        }
                }
                event=Event.GotOrder;
                allEvents.add(event);
                stateChanged();
         }
```

```
public void msgAtFront(){

	this.state = AgentState.DoingNothing;
	busy=false;
	host.msgImFree(this);
	stateChanged();
}

public void msgAtCook(){
	atCook.release();
	stateChanged();
}

public void msgOrderReady(String food, int table) {
	print ("Order Ready");
	for (MyCustomers myCustomer : myCustomers) {
		if (myCustomer.getTableNumber() == table) {
			myCustomer.setState(CustState.Eating);
			CurrentCustomer = myCustomer;
			}
	}

	event=Event.FoodReady;
	allEvents.add(event);
	stateChanged();
}

public void msgIWantToGoOnBreak() {
	event=Event.WantABreak;
	allEvents.add(event);
	stateChanged();
}

public void msgTakeABreak() {
	event=Event.TakeABreak;
	allEvents.add(event);
	stateChanged();
}

public void msgBreakOver() {
	this.Resume();
	host.msgBackFromBreak(this);
```

```
            stateChanged();
    }
```

Scheduler

1. if there exists an Event E within allEvents such that E == Event.TakeABreak
   ```
   Then: busy=true;
   currentEvent = E;
   allEvents.remove(currentEvent);
   TakeABreak();
   currentEvent=null;
   return true;
   ```

2. if there exists an Event E within allEvents such that E == Event.HeLeft
   ```
   Then: busy=true;
   currentEvent = E;
   allEvents.remove(currentEvent);
   PrepareTableEarly();
   currentEvent=null;
   return true;
   ```

3. if there exists an Event E within allEvents such that E == Event.WantABreak
   ```
   Then: busy=true;
   currentEvent = E;
   allEvents.remove(currentEvent);
   GiveMeABreak();
   currentEvent=null;
   return true;
   ```

4. if there exists an Event E within allEvents such that E == Event.GotOrder && state ==AgentState.GetOrder
   ```
   Then: busy = true;
   currentEvent = E;
   state = AgentState.TakeOrderToCook;
   TakeOrderToKitchen(CurrentCustomer);
   allEvents.remove(currentEvent);
   currentEvent=null;
   return true;
   ```

5. if there exists an Event E within allEvents such that E == Event.FoodReady && state ==AgentState.DoingNothing
    Then: busy = true;
    currentEvent = E;
    state = AgentState.ServeFood;
    DeliverFoodToTable(CurrentCustomer);
    allEvents.remove(currentEvent);
    currentEvent=null;
    return true;

6. if there exists an Event E within allEvents such that E == Event.GotCheck && state ==AgentState.DoingNothing
    Then: busy = true;
    currentEvent = E;
    state = AgentState.TakeCheckToCustomer;
    BringCheckTo(CurrentCustomer);
    allEvents.remove(currentEvent);
    currentEvent=null;
    return true;

7. if there exists an Event E within allEvents such that E == Event.GoToCashier && state ==AgentState.DoingNothing
    Then: busy = true;
    currentEvent = E;
    state = AgentState.GetCheck;
    GetCheckFromCashier();
    allEvents.remove(currentEvent);
    currentEvent=null;
    return true;

8. if there exists an Event E within allEvents such that E == Event.customerDone&& state ==AgentState.DoingNothing
    Then: busy = true;
    currentEvent = E;
    state = AgentState.CleanTable;
    PrepareTable();
    allEvents.remove(currentEvent);
    currentEvent=null;
    return true;

9. if there exists an Event E within allEvents such that E == Event.NewCustomerToSeat && state ==AgentState.DoingNothing
    Then: busy = true;

currentEvent = E;
state = AgentState.SeatingCustomer;
seatCustomer();
allEvents.remove(currentEvent);
currentEvent=null;
return true;

10. if there exists an Event E within allEvents such that E == Event.OutOfFood && state ==AgentState.DoingNothing
    Then: busy = true;
    currentEvent = E;
    state = AgentState.GetOrder;
    GetNewCustomerOrder();
    allEvents.remove(currentEvent);
    currentEvent=null;
    return true;

11. if there exists an Event E within allEvents such that E == Event.customerReady && state ==AgentState.DoingNothing
    Then: busy = true;
    currentEvent = E;
    state = AgentState.GetOrder;
    GetCustomerOrder();
    allEvents.remove(currentEvent);
    currentEvent=null;
    return true;
12. else return false;

Actions

```
private void GetCheckFromCashier() {
        for (MyCustomers myCustomer : myCustomers) {
            if (myCustomer.getState() == CustState.NeedCheck) {
                myCustomer.setState(CustState.BringCheck);
                CurrentCustomer = myCustomer;
                break;
                }
        }
        print("Getting check for " + CurrentCustomer.getCustomer());

        cashier.msgCheckPlease(CurrentCustomer);
        stateChanged();
```

```
	}

	private void BringCheckTo(MyCustomers c) {
		print("Taking check to " + c.getCustomer());
		waiterGui.DoGoToTable(c.getCustomer(), c.getTableNumber());
		try {
			atTable.acquire();
		} catch (InterruptedException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		busy=true;
		this.state=AgentState.TakeCheckToCustomer;

		c.getCustomer().msgHereIsYourBill(c.getCheck());
		waiterGui.DoLeaveCustomer();
	}

	private void TakeABreak() {
		this.Pause();
		stateChanged();
	}

	private void GiveMeABreak() {
		host.msgIWantABreak(this);
		stateChanged();
	}

	private void seatCustomer() {
		for (MyCustomers myCustomer : myCustomers) {
			if (myCustomer.getState()==CustState.Seating) {
				myCustomer.setState(CustState.Seated);
				CurrentCustomer = myCustomer;
				break;
			}
		}

		CurrentCustomer.getCustomer().msgFollowMeToTable(this, currentMenu);
		DoSeatCustomer(CurrentCustomer.getCustomer(),
CurrentCustomer.getTableNumber());
		try {
			atTable.acquire();
		} catch (InterruptedException e) {
```

```
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
                busy=true;
                this.state=AgentState.SeatingCustomer;
                waiterGui.DoLeaveCustomer();
        }

        private void DoSeatCustomer(CustomerAgent customer, int table) {
                print("Seating " + customer + " at table " + table + ".  Here is a menu.");
                waiterGui.DoBringToTable(customer, table);

        }

        public void GetNewCustomerOrder() {
                print("Going to get a new order");
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getState()==CustState.NewOrderNeeded) {
                                CurrentCustomer = myCustomer;
                                break;
                        }
                }
                waiterGui.DoGoToTable(CurrentCustomer.getCustomer(),
CurrentCustomer.getTableNumber());

                try {
                        atTable.acquire();
                } catch (InterruptedException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                busy=true;
                this.state=AgentState.GetOrder;

                CurrentCustomer.setState(CustState.Ordered);
                CurrentCustomer.getCustomer().msgHereForNewOrder();
                print("Here for a new order");
        }

        public void GetCustomerOrder() {
                print("Going to get order");
                for (MyCustomers myCustomer : myCustomers) {
                        if (myCustomer.getState()==CustState.ReadyToOrder) {
```

```
                        CurrentCustomer = myCustomer;
                        break;
                    }
                }
                waiterGui.DoGoToTable(CurrentCustomer.getCustomer(),
CurrentCustomer.getTableNumber());

                try {
                    atTable.acquire();
                } catch (InterruptedException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
                busy=true;
                this.state=AgentState.GetOrder;

                CurrentCustomer.setState(CustState.Ordered);

                CurrentCustomer.getCustomer().msgHereForOrder();
                print("Here for order");
        }

        public void TakeOrderToKitchen(MyCustomers current) {
                print("Taking order to chef");
                waiterGui.BringOrderToCook(current.getOrder());
                try {
                    atCook.acquire();
                } catch (InterruptedException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
                busy=true;
                this.state=AgentState.TakeOrderToCook;
                current.setState(CustState.WaitingForFood);
                cook.msgNewOrder(this,current.getTableNumber(), current.getOrder());
                waiterGui.DoLeaveCustomer();
                print("Chef has order");
        }


    public void DeliverFoodToTable(MyCustomers current) {
        waiterGui.DoGoToCook();
        try {
```

```
                atCook.acquire();
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        waiterGui.BringFoodToCustomer(current.getCustomer(),
current.getTableNumber(),current.getOrder());

        try {
                atTable.acquire();
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }

            busy=true;
            this.state=AgentState.ServeFood;
            current.setState(CustState.Eating);
            current.getCustomer().msgDeliveredFood();
        waiterGui.DoLeaveCustomer();
        print("Delivered Food");
    }

        public void PrepareTable() {
            print("Table empty");
            for (int i=0;i<myCustomers.size();i++) {
                if (myCustomers.get(i).getState()==CustState.Done) {
                    CurrentCustomer = myCustomers.get(i);
                    myCustomers.remove(i);
                    break;
                }
            }
            CurrentCustomer.setState(CustState.Leaving);
            waiterGui.DoGoToTable(CurrentCustomer.getCustomer(),
CurrentCustomer.getTableNumber());
            try {
                atTable.acquire();
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            busy=true;
            this.state=AgentState.CleanTable;
```

```
                host.msgLeavingTable(CurrentCustomer.getCustomer(), this);
                CurrentCustomer.setState(CustState.Gone);

                myCustomers.remove(CurrentCustomer);

                print("Table clean");
                waiterGui.DoLeaveCustomer();
        }

        public void PrepareTableEarly() {
                print("Table empty");
                for (int i=0;i<myCustomers.size();i++) {
                        if (myCustomers.get(i).getState()==CustState.LeftEarly) {
                                CurrentCustomer = myCustomers.get(i);
                                myCustomers.remove(i);
                                break;
                        }
                }
                CurrentCustomer.setState(CustState.Leaving);
                waiterGui.DoGoToTable(CurrentCustomer.getCustomer(),
CurrentCustomer.getTableNumber());
                try {
                        atTable.acquire();
                } catch (InterruptedException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                busy=true;
                this.state=AgentState.CleanTable;
                host.msgLeavingTable(CurrentCustomer.getCustomer(), this);
                CurrentCustomer.setState(CustState.Gone);

                myCustomers.remove(CurrentCustomer);

                print("Table clean");
                waiterGui.DoLeaveCustomer();
        }
```