AIWHEELEDVEHICLE: BASIC CREATION

Exercise Files

Starter – "Kit/gpgt/server/scripts/gpgt/chapter7/exercise101.cs"
Answers – "Kit/gpgt/server/scripts/gpgt/chapter7/answers/exercise101_f.cs"

Exercise Mission

Chapter 7: "101 AIWheeledVehicle: Basic creation"

Synopsis

In this exercise, we will familiarize ourselves with some of the supplied kit code that will allow us to do the following AIWheeledVehicle exercises. Additionally, we will do a simple exercise using some of this code to create an AIWheeledVehicle instance.

Prerequisites

1. *ch1_001.pdf "Using The Kit"*

Exercises

AIWheeledVehicle: Basic Creation

1 AIWheeledVehicle::Spawn()

You can find a function named "AIWheeledVehicle::spawn()" in the file "Kit/gpgt/server/scripts/gpgt/common/AI/AIWheeledVehicle.cs". This function simplifies the creation and placement of AIWheeledVehicle objects. A slightly modified version of this function is listed below.

```
// 1
function AIWheeledVehicle::spawn( %spawnTransform, %DBName )
{
	// 2
	%theBot = new AIWheeledVehicle()
	{
		dataBlock = %DBName;
	};

	// 3
	MissionGroup.add(%theBot);

	// 4
	%theBot.setTransform(%spawnTransform);

	// 5
	return %theBot;
}
```

It has the following characteristics.

1. Takes two arguments.
   - %spawnTransform – A standard seven-element Torque transform. The new AIWheeledVehicle instance will be assigned this transform.
   - %DBName – The name (or ID) of the datablock to use to initialize the new AIWheeledVehicle instance (during creation).
2. This function creates an unnamed instance of AIWheeledVehicle, using the "new" keyword and standard object creation syntax.
3. The function also places this wheeled bot in the SimGroup MissionGroup to ensure that when the mission is exited, this wheeled bot will be deleted.
4. As I noted above, bot's transform is set to the value passed in %spawnTransform. Please note that a position (first three elements of a transform) is OK too.
5. Lastly, this function returns the ID of the newly created bot.

2 Paths

This kit comes with a number of features that allow us to easily create a round path and to place it in a mission. The features that allow this work are:

- A path creation and maintenance toolkit located in the file "Kit/gpgt/server/scripts/gpgt/common/AI/PathUtils.cs"
- A path assignment method called "AIWheeledVehicle::assignPath()" which can be located in the file "Kit/gpgt/server/scripts/gpgt/common/AI/AIWheeledVehicle.cs".
- A special marker (placed in every mission at the same place) named exerciseCenter. This object exists in the center of the area that I wanted our exercises to take place at.

These features, combined with the spawning function above, make it as easy as this to create a wheeled bot, create a path, and then to assign that path to the wheeled bot.

```
// Make the AIWheeledVehicle
//
%bot = AIWheeledVehicle::spawn( "0 0 0" , DefaultCar );

// Make the Path (15 world-unit radius, centered at exerciseCenter)
//
exerciseCenter.createSimplePath( "testPath" , 15 );

// Assign the path
//
%bot.assignPath( testPath );
```

After this code is executed, the bot will have the following dynamic variables assigned to it:

- myPath – This will contain the ID of the path we just assigned to this bot.
- currentPathNode – This will be set to 0, where valid numbers are between 0 and 7. When we call createSimplePath(), it makes a round path with eight navigation nodes. We will be using this variable (in our exercises) to track the node that the bot is currently moving towards.
- destination – This is the second navigation variable we will use in our exercises. We will use it to store the transform of the path node our bot will use as its destination.

In order to fill in the destination field (above) we write code like this.

```
%pathNode = %bot.myPath.getObject( %bot.currentPathNodeNum );

%bot.destination = %pathNode.getTransform();
```

This code retrieves the ID of the current target path node and then gets that object's transform, finally assigning that value to the destination.

AIWheeledVehicle: Basic Creation

Visible Markers

The supplied path utilities create a visible marker for each path node. These markers have the special capability of being re-skinned. I don't ask you to write any code using this feature, but in the answers I supply, I do add extra code to change the color of a marker depending on whether it is the current destination (green marker) or not (red marker). So, don't be confused when you read this code. I merely added it to provide the paths and the exercises better visual feedback.

3 Create a Wheeled Bot

Goal: Using what we have just discussed, place a bot in an exercise mission.

Starter Code: You are provided with one function body to start this exercise.

```
function startexercise101()
{

  // 1

}
```

Steps:

1. In the exercises starter function (shown above) write the code needed to create a single AIWheeledVehicle, where the code you write satisfies these rules:

   Asasigns new bot to local variable %myBot.

   Uses the DefaultCar datablock to initialize the new bot.

   Places the bot in the center of the exercise area.

Output Goal:

When you run the completed exercise, you will see a cool race car bot standing in front of you.

Hints:

1. The method getTransform() retrieves an object's transform.

