

Sample Turtle Programs

Sample 1: Drawing a square

This program draws a square. Default values are used for the turtle's pen color, pen width, body color, etc.

```
import galapagos.*;

/**
 * This sample program shows the most basic way
 * of using a Turtle.
 */
public class TurtleSample1
{
    public static void main( String args[] )
    {
        Turtle    myTurtle;

        int       size, turn;

        myTurtle = new Turtle( );

        size     = 100;           //logical units
        turn     = 90;           //in degree

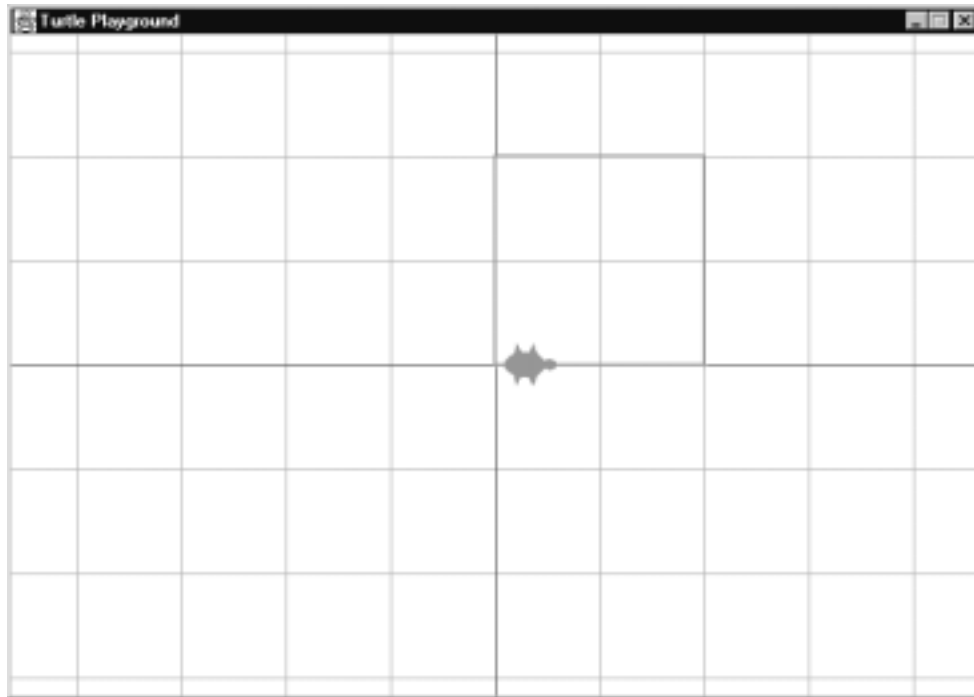
        //draw a square
        myTurtle.move( size );
        myTurtle.turn( turn );

        myTurtle.move( size );
        myTurtle.turn( turn );

        myTurtle.move( size );
        myTurtle.turn( turn );

        myTurtle.move( size );
        myTurtle.turn( turn );
    }
}
```

SampleTurtle1.java



Sample 2: Changing the Turtle's Properties

This program also draws a square, but this time several of the turtle's properties are changed.

```
import galapagos.*;
import java.awt.*; //for using Color

/**
 * This sample program shows several different of changing
 * the turtle's properties such as pen color, pen size,
 * body color, etc.
 */
public class TurtleSample2
{
    public static void main( String args[] )
    {
        Turtle    myTurtle;

        int       size, turn;

        myTurtle = new Turtle( );

        size     = 100;           //logical units
        turn     = 90;           //in degree
    }
}
```

SampleTurtle2.java

```

//set some properties
myTurtle.bodyColor( Color.black ); //body color is black

myTurtle.jumpTo( 50, 0 );           //set the starting position to (50,0)
                                     //jumping to it

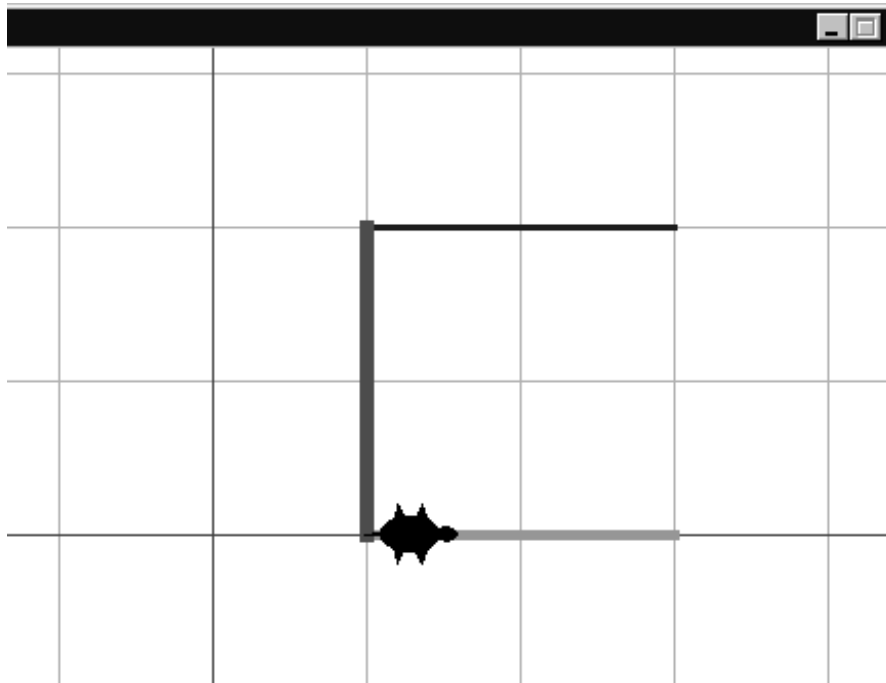
//draw a square
myTurtle.pensize( 5 );               //draw the bottom side
myTurtle.pencolor( Color.green );   //in green 5 units wide
myTurtle.move( size );
myTurtle.turn( turn );

myTurtle.penUp( );                  //just move along the right side
myTurtle.move( size );              //without drawing by placing the
myTurtle.turn( turn );              //pen in the up position

myTurtle.penDown( );               //draw again by putting the
myTurtle.pensize( 3 );              //pen down; set pen size to 3
myTurtle.pencolor( Color.blue );   //and color to blue
myTurtle.move( size );
myTurtle.turn( turn );

myTurtle.speed( 5 );
myTurtle.pensize( 7 );              //draw the final side in one-fourth
myTurtle.pencolor( Color.red );     //the default speed (which is 20) with
myTurtle.move( size );              //pen size 7 and color red
myTurtle.turn( turn );
}
}

```



Sample 3: Using TurtleDrawingWindow

In the previous two sample programs, the turtles used a default drawing window they created internally. This approach is adequate for a simple drawing, but not for others. In this sample program, the programmer explicitly creates a TurtleDrawingWindow, a window which the turtle will draw. Notice that the constructor used for the Turtle in this class is different from the previous programs. You must use this second constructor if you want to make the Turtle draw on the TurtleDrawingWindow you create. Errata: The sample source code on page 190 exercise 4.16 is wrong because the way turtles are created in the sample code is invalid.

```
import galapagos.*;
import java.awt.*; //for using Color

/**
 * This sample program shows how to use a TurtleDrawingWindow
 * explicitly. If TurtleDrawingWindow is not used, then a
 * Turtle will create one internally. If you want to set
 * the parameters such as moving the location of logical
 * origin point to places other than the center of window,
 * changing the window scale (zoom in or zoom out view), etc.
 * Also, you must use a TurtleDrawingWindow if you want
 * use multiple Turtle objects on a single window.
 */
public class TurtleSample3
{

    public static void main( String args[] )
    {
        Turtle          myTurtle;
        TurtleDrawingWindow  playground;

        int      size, turn;

        playground = new TurtleDrawingWindow( );

        /******* IMPORTANT *****/
        //You must pass an int argument so the turtle
        //will not create a default drawing window. You must
        //create a turtle in this manner if you want to
        //add a turtle to the drawing window you create within
        //your program. Sample code for exercise 4.16 on page
        //190 is therefore wrong. Sorry.

        myTurtle = new Turtle( Turtle.NO_DEFAULT_WINDOW );
    }
}
```

SampleTurtle3.java

```

//You must connect the turtle to a TurtleDrawingWindow
//by 'adding' it to the window.

playground.add( myTurtle );

//*****//

//Set some properties for the playground.
//Experiment with different values.
playground.setUnit( 2.5 ); //one logical unit is equal to 3 pixels
                           //default is 1, i.e. one unit equals one pixel

playground.setOrigin( 75, 50 ); //logical point (100,100) is at the
                                //center of the window

playground.setGrid( true ); //Displays the grid line, which is a default
                             //change the argument to false and see
                             //what happens

playground.setVisible( true ); //Don't forget to make the window visible.
                                //You can make the
                                //window visible first and then set properties
                                //using setUnit, setOrigin, etc., but in this
                                //case you may see the display in the original
                                //property and suddenly change to the properties
                                //you set.

//Now draw a square

size      = 100;           //logical units
turn      = 90;           //in degree

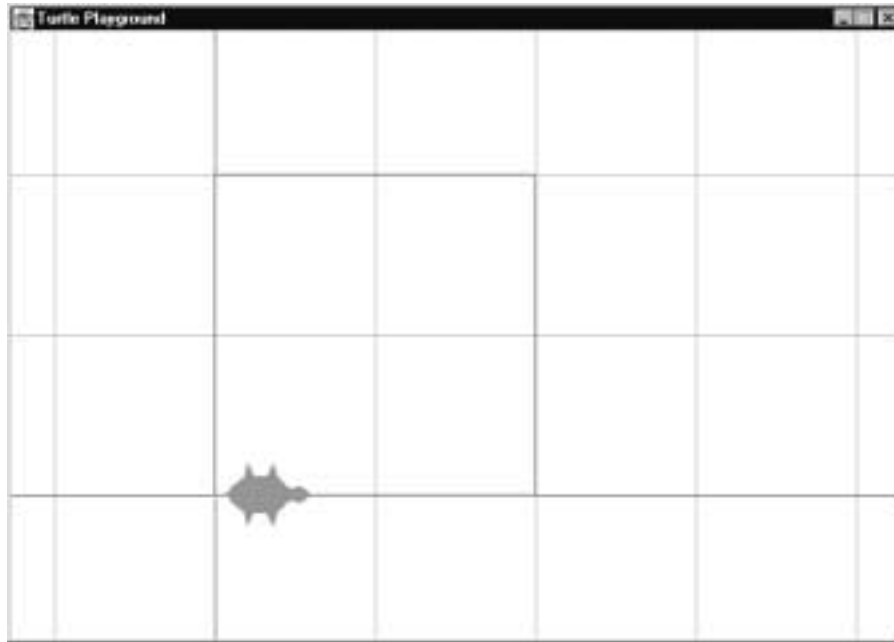
//draw a square
myTurtle.move( size );
myTurtle.turn( turn );

myTurtle.move( size );
myTurtle.turn( turn );

myTurtle.move( size );
myTurtle.turn( turn );

myTurtle.move( size );
myTurtle.turn( turn );
}
}

```



Sample Program 4: Using Multiple Turtles

This program creates two turtles and let them draw a square simultaneously.

```
import galapagos.*;
import java.awt.*; //for using Color

/**
 * This sample program shows how to use a TurtleDrawingWindow
 * explicitly with two turtles.
 */
public class TurtleSample4
{
    public static void main( String args[] )
    {
        Turtle          betsy, emily;
        TurtleDrawingWindow  playground;

        int            size, turn;

        playground = new TurtleDrawingWindow( );
        playground.setVisible( true );
```

SampleTurtle4.java

```

//Create two turtles and add them to the playground

betsy  = new Turtle( Turtle.NO_DEFAULT_WINDOW );
emily  = new Turtle( Turtle.NO_DEFAULT_WINDOW );

playground.add( betsy );
playground.add( emily );

//Don't show the turtle bodies
betsy.hide( );           //if you want to show turtle bodies
emily.hide( );           //then set their body color to distinguish
                           //them. Use the bodyColor method.
                           //  betsy.bodyColor( Color.blue );
                           //  emily.bodyColor( Color.yellow );

//Set pen colors
betsy.penColor( Color.blue );
emily.penColor( Color.yellow );

//Now draw a square

size    = 100;           //logical units
turn    = 90;            //in degree

//betsy draw counter clockwise square
betsy.pause( );
betsy.move( size );
betsy.turn( turn );

betsy.move( size );
betsy.turn( turn );

betsy.move( size );
betsy.turn( turn );

betsy.move( size );
betsy.turn( turn );

//emily draws clockwise square of the same size
emily.pause( );
emily.move( -size );     //If you make emily visible, you will
emily.turn( -turn );     //notice that emily is moving backward
                           //If you want emily to face forward
emily.move( -size );     //then you must change emily's heading
emily.turn( -turn );     //to 180 deg first and make both
                           //size and turn positive, exactly like
emily.move( -size );     //betsy's code
emily.turn( -turn );

emily.move( -size );
emily.turn( -turn );

//now start both turtles together

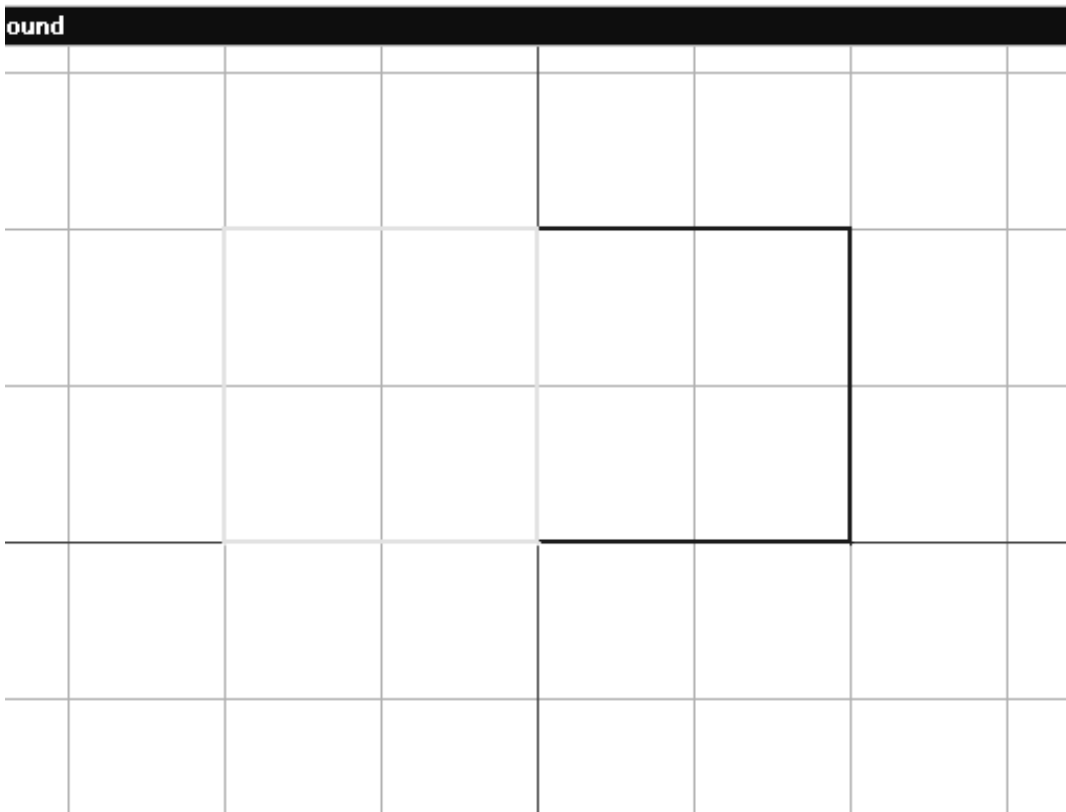
```

```

betsy.start( );
emily.start( );

//NOTE:
// Theoretically, the turtles move in their own thread, so if you don't
// pause them, they would move immediately after receiving the first command.
// This makes betsy start moving while emily is still waiting for the commands
// to arrive.
// In reality you will not see the delay even if you don't put the pause and
// and start commands, because the sequence of commands in this program
// is very short. The time to set up the drawing window will take much
// longer time and executing the statements in this program.
}
}

```



Sample Program 5:

In this program, the programmer uses the javabook class `InputBox` along with the `Turtle`. Since `TurtleDrawingWindow` is a `Frame` object, it can serve as an owning window for the `InputBox` class (and other javabook dialogs).

SampleTurtle5.java

```
import galapagos.*;
import java.awt.*; //for using Color
import javabook.*;

/**
 * This sample program shows how to use a TurtleDrawingWindow
 * and the InputBox class from the javabook package.
 */
public class TurtleSample5
{

    public static void main( String args[] )
    {
        Turtle          myTurtle;
        TurtleDrawingWindow playground;

        InputBox        inputBox;

        int             size, turn;

        playground = new TurtleDrawingWindow( );

        inputBox    = new InputBox( playground );

        myTurtle    = new Turtle( Turtle.NO_DEFAULT_WINDOW );
        myTurtle.bodyColor( Color.magenta );

        playground.add( myTurtle );

        playground.setVisible( true );

        //Get input
        size = inputBox.getInteger( "Enter the size of a square:" );

        turn = 90;

        //draw a square
        myTurtle.move( size );
        myTurtle.turn( turn );

        myTurtle.move( size );
        myTurtle.turn( turn );

        myTurtle.move( size );
        myTurtle.turn( turn );

        myTurtle.move( size );
        myTurtle.turn( turn );
    }
}
```

