stamp.peripheral.hid.mouse.serial

Class MS2ButtonSerial

```
java.lang.Object
  |
  +--stamp.peripheral.hid.mouse.serial.MS2ButtonSerial
```

public class **MS2ButtonSerial**
extends Object

Overview

This library tracks recent button and motion events by processing information sent by a Microsoft 2-button serial mouse.

See Also:

AppNote010 for circuit diagrams, examples, and instructions on how to use this class, the Javelin Stamp and a Microsoft serial 2-button mouse. This application note is available from www.javelinstamp.com.

Field Summary

`static int`	**`DTR`** I/O pin connected to the mouse's DTR line via an RS232 transceiver.
`static boolean`	**`leftButton`** Indicates the state of the left button.
`static boolean`	**`leftButtonOld`** Indicates the previous state of the left button.
`static boolean`	**`leftDrag`** Indicates whether or not a drag operation is in progress with the left mouse button pressed and held.
`static boolean`	**`leftDrop`** Indicates whether or not a left-drop operation has occurred, meaning that the left-mouse button has been released.
`protected static int[]`	**`packet`** 3-byte mouse information packet
`static boolean`	**`rightButton`** Indicates the state of the right button.

`static boolean`	**`rightButtonOld`** Indicates the previous state of the right button.
`static boolean`	**`rightDrag`** Indicates whether or not a drag operation is in progress with the right mouse button pressed and held.
`static boolean`	**`rightDrop`** Indicates whether or not a right-drop operation has occurred meaning that the right-mouse button has been released.
`static int`	**`RTS`** I/O pin connected to the mouse's RTS line via an RS232 transceiver.
`static int`	**`scale`** Stores a scale value that can be used to slow the mouse's travel on a given display.
`static int`	**`xDistance`** Stores cumulative `x-distance` traveled by the mouse.
`static int`	**`xDragEnd`** Stores the most recent x-coordinate recorded when a drop operation occurred.
`static int`	**`xDragStart`** Stores the most recent x-coordinate recorded at the beginning of a drag operation.
`static int`	**`xMax`** Stores the maximum allowable `xDistance` value.
`static int`	**`xMin`** Stores the minimum allowable `xDistance` value.
`static int`	**`yDistance`** Stores cumulative `y-distance` traveled by the mouse.
`static int`	**`yDragEnd`** Stores the most recent y-coordinate recorded when a drop operation occurred.
`static int`	**`yDragStart`** Stores the most recent y-coordinate recorded at the beginning of a drag operation.
`static int`	**`yMax`** Stores the maximum allowable `yDistance` value.
`static int`	**`yMin`** Stores the minimum allowable `yDistance` value.

Constructor Summary
MS2ButtonSerial(Uart rxUart, int DTR, int RTS) Creates a new mouse object with default settings: `scale = 1, xMin = yMin = -10000, xMax = yMax = +10000, xDistance = yDistance = 0`
MS2ButtonSerial(Uart rxUart, int DTR, int RTS, int scale, int xMin, int yMin, int xMax, int yMax, int xDistance, int yDistance) Creates a new mouse object user specified settings:

Method Summary	
`boolean`	**bootSequence**() Emulate PC boot sequence at the Mouse's serial connections.
`void`	**clearStatus**() Clears the status of all button triggered event flags.
`boolean`	**event**() Determines if data sent by the mouse is waiting in the `rxUart` buffer, indicating that an event has occurred since the last call to the `update` method.
`void`	**update**() Processes all information stored in the rxUart buffer and updates distance measurements and button triggered event flags.

Methods inherited from class java.lang.Object
equals

Field Detail

leftButton

`public static boolean leftButton`

Indicates the state of the left button. `true` if pressed, `false` if not pressed.

---

rightButton

`public static boolean rightButton`

Indicates the state of the right button. `true` if pressed, `false` if not pressed.

---

leftButtonOld

`public static boolean leftButtonOld`

Indicates the previous state of the left button. `true` if pressed, `false` if not pressed.

---

rightButtonOld

`public static boolean` **`rightButtonOld`**

Indicates the previous state of the right button. `true` if pressed, `false` if not pressed.

---

leftDrag

`public static boolean` **`leftDrag`**

Indicates whether or not a drag operation is in progress with the left mouse button pressed and held. `true` if pressed and held, `false` if a drag operation is not in progress.

---

rightDrag

`public static boolean` **`rightDrag`**

Indicates whether or not a drag operation is in progress with the right mouse button pressed and held. `true` if pressed and held, `false` if a drag operation is not in progress.

---

leftDrop

`public static boolean` **`leftDrop`**

Indicates whether or not a left-drop operation has occurred, meaning that the left-mouse button has been released. `true` if pressed, `false` if not pressed.

---

rightDrop

`public static boolean` **`rightDrop`**

Indicates whether or not a right-drop operation has occurred meaning that the right-mouse button has been released. `true` if pressed, `false` if not pressed.

---

xDragStart

`public static int` **`xDragStart`**

Stores the most recent x-coordinate recorded at the beginning of a drag operation.

---

xDragEnd

```
public static int xDragEnd
```

Stores the most recent x-coordinate recorded when a drop operation occurred.

---

yDragStart

```
public static int yDragStart
```

Stores the most recent y-coordinate recorded at the beginning of a drag operation.

---

yDragEnd

```
public static int yDragEnd
```

Stores the most recent y-coordinate recorded when a drop operation occurred.

---

xDistance

```
public static int xDistance
```

Stores cumulative `x-distance` traveled by the mouse. Distances traveled to the right are added to this value, and distances traveled to the left are subtracted from this value.

---

yDistance

```
public static int yDistance
```

Stores cumulative `y-distance` traveled by the mouse. Distances traveled downward are added to this value, and distances traveled upward are subtracted from this value.

---

xMax

```
public static int xMax
```

Stores the maximum allowable `xDistance` value.

---

xMin

```
public static int xMin
```

Stores the minimum allowable `xDistance` value.

---

yMax

```
public static int yMax
```

Stores the maximum allowable `yDistance` value.

---

yMin

```
public static int yMin
```

Stores the minimum allowable `yDistance` value.

---

scale

```
public static int scale
```

Stores a scale value that can be used to slow the mouse's travel on a given display.

DTR

```
public static int DTR
```

I/O pin connected to the mouse's DTR line via an RS232 transceiver.

RTS

```
public static int RTS
```

I/O pin connected to the mouse's RTS line via an RS232 transceiver.

---

packet

```
protected static int[] packet
```

3-byte mouse information packet

Constructor Detail

MS2ButtonSerial

```
public MS2ButtonSerial(Uart rxUart,
                       int DTR,
                       int RTS)
```

Creates a new mouse object with default settings: `scale = 1, xMin = yMin = -10000, xMax = yMax = +10000, xDistance = yDistance = 0`

Parameters:

`rxUart` - Uart object setup for serial mouse communication.
`DTR` - I/O pin connected to the DTR line through an RS232 transceiver.
`RTS` - I/O pin connected to the RTS line through an RS232 transceiver.

MS2ButtonSerial

```
public MS2ButtonSerial(Uart rxUart,
                       int DTR,
                       int RTS,
                       int scale,
                       int xMin,
                       int yMin,
                       int xMax,
                       int yMax,
                       int xDistance,
                       int yDistance)
```

Creates a new mouse object user specified settings:

Parameters:

`rxUart` - Uart object setup for serial mouse communication.
`DTR` - I/O pin connected to the DTR line through an RS232 transceiver.
`RTS` - I/O pin connected to the RTS line through an RS232 transceiver.
`scale` - value slows apparent pointer motion on the display.
`xMin` - sets the minimum value that `xDistance` will not be allowed to go below.
`yMin` - sets the minimum value that `yDistance` will not be allowed to go below.
`xMax` - sets the maximum value that `xDistance` will not be allowed to go above.
`yMax` - sets the maximum value that `yDistance` will not be allowed to go above.
`xDistance` - initializes the value of `xDistance` for initial cursor value.
`yDistance` - initializes the value of `yDistance` for initial cursor value.

Method Detail

bootSequence

```
public boolean bootSequence()
```

Emulate PC boot sequence at the Mouse's serial connections. IMPORTANT: This method must be called before attempting to use other methods in this library.

Returns:

`true` if a Microsoft serial mouse is detected, `false` if the mouse is not detected.

---

event

```
public boolean event()
```

Determines if data sent by the mouse is waiting in the `rxUart` buffer, indicating that an event has occurred since the last call to the `update` method.

Returns:

`true` if an event has occurred or `false` if an event has not occurred.

---

clearStatus

```
public void clearStatus()
```

Clears the status of all button triggered event flags.

---

update

```
public void update()
```

Processes all information stored in the rxUart buffer and updates distance measurements and button triggered event flags.

Javelin Stamp is a trademark or registered trademark of Parallax, Inc. in the US and other countries.
Copyright 2000-2002 Parallax, Inc. 599 Menlo Drive,
Rocklin, California, 95765, U.S.A. All Rights Reserved.