

## WedgeMagic™ API – Palm OS 5.2

WedgeMagic™ supplies the interface to SerialMagic as an API for 3<sup>rd</sup> party developers who wish to interface with SerialMagic. Only SerialMagic Professional supports the WedgeMagic API.

Is the SerialMagic API available at all?

The API is only available when SerialMagic has been started by manually pressing the "Start" button. At that time it connects to the port (serial port or Bluetooth device) and "owns" the port.

Prior to SerialMagic being manually started, all the APIs will return 'false'

```
Boolean SerialMagic_IsAPIAvailable (void);
```

Stop SerialMagic. SerialMagic will release the port (serial port or Bluetooth)

When SerialMagic "owns" the port, SerialMagic\_Stop() will release it and place SerialMagic in a "disconnected" state. SerialMagic is still listening for API calls, but will not reconnect to the port until SerialMagic\_Start() is called.

```
Void SerialMagic_Stop (void);
```

Give the port back to SerialMagic. This will cause SerialMagic to open the port again (and "own" it) and listen for data received from the port.

```
Void SerialMagic_Start (void);
```

When the port is started (either by manually pressing the "start" button or by calling SerialMagic\_Start() to re-open it) there is a delay before the port is fully open and configured. SerialMagic\_IsStarted() returns true once the port is fully open and configured.

```
Boolean SerialMagic_IsStarted (void);
```

When SerialMagic\_Stop() is called, it triggers shutting down the port. Under some circumstances, this may not occur instantly; it may take several seconds for a Bluetooth connection to be torn down for example. SerialMagic\_IsStopped() returns true when the port has completely been stopped. At this time another app may open it.

```
Boolean SerialMagic_IsStopped (void);
```

Returns true if the port is active and open. Returns false if the port is off for any reason SerialMagic\_Stop() has been called Bluetooth device powered off etc.

```
Boolean SerialMagic_IsConnected (void);
```

These routines are called as you'd expect:

```
if (SerialMagic_APIAvailable() != true) {  
  FrmCustomAlert(AlertID, "oh, dear! SerialMagic does not seem to  
  be alive!", "", "")  
}
```

WedgeMagic™ supplies the interface to SerialMagic as an API for 3<sup>rd</sup> party developers who wish to interface with SerialMagic. Only SerialMagic Professional supports the WedgeMagic API.

Here is example code on how an application can hook data from the barcode scanner if the application wishes to handle the barcode scans directly instead of having the data place in the application at the cursor location.

```
Boolean HandleBarcodeReaderKeyDown(WChar key)
{
    switch (key)
    {
        case 2:
            gInBarcodeSequence = true;
            return true;

        case 13:
            if (gInBarcodeSequence)
            {
                gInBarcodeSequence = false;
                gBarcodeBuffer[gBarcodeBufferLen] = 0;
                // now post a bar code input event
                PostPPEvent(kMsgBarcodeInput);
            }
            return true;

        case 10: // just ignore it
            return true;

        default:
            if (gInBarcodeSequence)
            {
                gBarcodeBuffer[gBarcodeBufferLen++] = key;
                return true;
            }
            break;
    }
    return false; // key event was not handled
}
```