By William Van Winkle

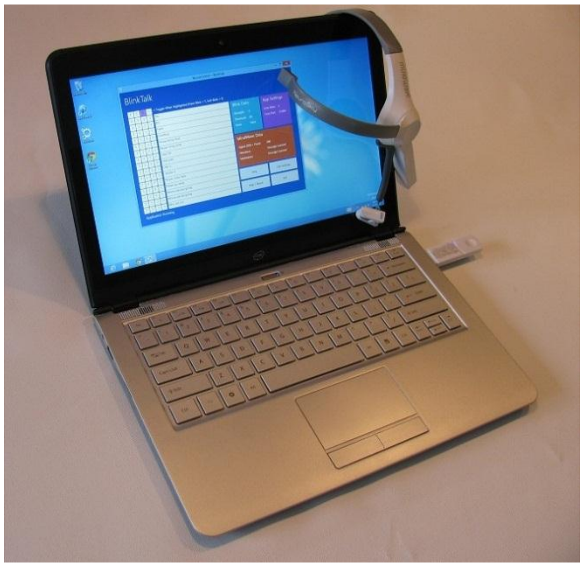
## App Innovation Winner Brings Speech to the Silent

BlinkTalk users never smile or even nod to indicate their joy with Bryan Brown's award-winning application. They can't. They have no muscle control below their eyes. But with this one app, an Ultrabook™ device, and the power of electromyography, many "locked-in" patients now have the ability to communicate verbally, perhaps for the first time in many years.

When Bryan Brown, a 20-year programming and development engineering veteran, ran across a posting for the Intel® App Innovation Contest on CodeProject, he saw a golden opportunity. This was a chance to take his knowledge of C# and Windows* Presentation Foundation (WPF) from a full-time day job writing software for chemical-processing instruments and apply it toward something new and amazing: helping paraplegics and others with no ability to control their bodies regain some ability to communicate. His journey from an idea to a contest-winning application able to change people's lives shows just how fluid and accessible the development process can be.

## How BlinkTalk Works

BlinkTalk performs binary encoding of the blink patterns detected by a NeuroSky MindWave* device and translates those codes into preset phrases that through speech synthesis can be selected and spoken by the host computer. The MindWave measures intentionally directed EMG activity (blink strength), which in turn become numeric values that ultimately manifest as communication from the user.



**Ultrabook with MindWave Sensor.** *Bryan Brown found that the Intel® reference Ultrabook™ device was the perfect platform for both developing the BlinkTalk app and using it in the field. In the photo, NeuroSky's MindWave* headset dangles on the right side of the screen.*

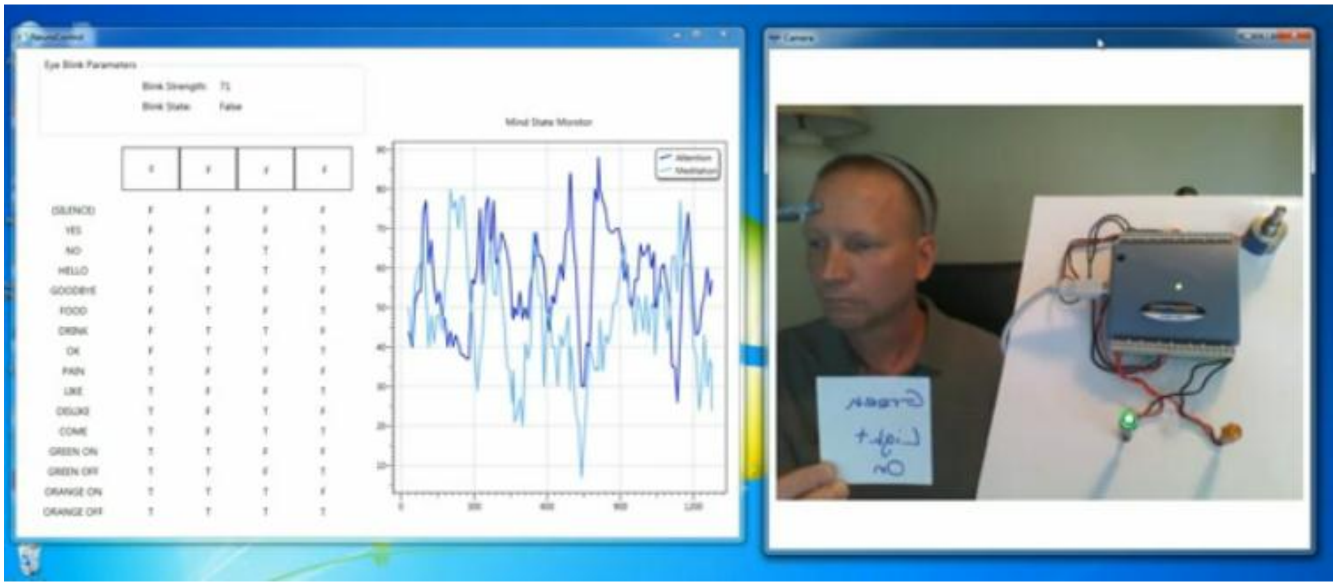
Brown began BlinkTalk's development by scouring the NeuroSky SDK and its supporting documentation. Much of his Microsoft Windows*-based development work, done in Microsoft Visual Studio* 2010 Pro, builds on the .NET framework, so he found a suitable DLL for the job (Brian Peek's ThinkGearNET). This driver essentially wraps itself around the NeuroSky DLL. With this in place, the C#/WPF-based BlinkTalk app fires an event handler whenever a monitored brainwave signal changes its state within a given range.

"Whenever the MindWave event handler fires," said Brown, "it updates four private data members: Attention, Meditation, BlinkStrength, and PoorSignal. These variables are used by two continuously running threads that are implemented in BlinkTalk—one to do the processing and the other for the user interface. The user interface thread creates a dispatcher that allows the processing thread and the user interface access to the same data members."

## Inside the Interface

BlinkTalk essentially consists of three pages: the main operational screen, an Edit Settings screen, and a Help/About page. This first version of BlinkTalk offers 16 possible blink-phrases, based on four data bits ($4^2$). Brown kept the number low in order to streamline the initial design for the Intel App Innovation Contest. The 16 blink-phrases were chosen by default, but all are customizable, as are the program parameters, such as blink threshold and scanning rate.

The operational screen presents a list of 16 possible statements: 15 phrases, such as "I feel OK" and "Where are we going," and silence. Each of these statements correlates to a string of four binary True/False (T/F) values. For example, TFFT can mean "I like it." BlinkTalk scrolls through each of the four letter positions, and the user must time his or her blink to coincide with the cursor resting on each position. The strength of the blink determines whether a value registers as T or F.



Eye Blink Parameters

Blink Strength: 71

Blink State: False

| | F | F | F | F |
|---|---|---|---|---|
| (SILENCE) | F | F | F | F |
| YES | F | F | F | T |
| NO | F | F | T | F |
| HELLO | F | F | T | T |
| GOODBYE | F | T | F | F |
| FOOD | F | T | F | T |
| DRINK | F | T | T | F |
| OK | F | T | T | T |
| PAIN | T | F | F | F |
| LIKE | T | F | F | T |
| DISLIKE | T | F | T | F |
| COME | T | F | T | T |
| GREEN ON | T | T | F | F |
| GREEN OFF | T | T | F | T |
| ORANGE ON | T | T | T | F |
| ORANGE OFF | T | T | T | T |

**Early Prototype with IO Control.** *The photo on the right shows an early prototype design of the BlinkTalk application, complete with the USB-based NeuroSky MindWave* controller. Real-time EMG graphs are shown on the left. Bryan Brown decided to remove the graphs from the final app interface in order to make it simpler and more streamlined.*

Operators controlling the app at the PC can fine-tune sensor thresholds to optimize hard and soft blink values. They can also change the scan rate, or the delay between the input cursor moving from one column to the next. Little additional configuration is necessary.

After the user selects a blink-phrase, the application plays the phrase through the computer's speakers. Although speech functionality is not necessary to BlinkTalk's core objective—providing immobilized people with the ability to have limited communication—the ability to have thoughts spoken aloud through the application can be gratifying.

To enable this functionality, BlinkTalk uses the System.Speech.Synthesis namespace, a quick yet robust way to add speech to .NET applications. This namespace holds the classes the application needs in order to interface with the OS's speech synthesis engine. In essence, the app generates a "speaker" object, and the object's Speak method gets called when the detected phrase string's parameter value is assessed.

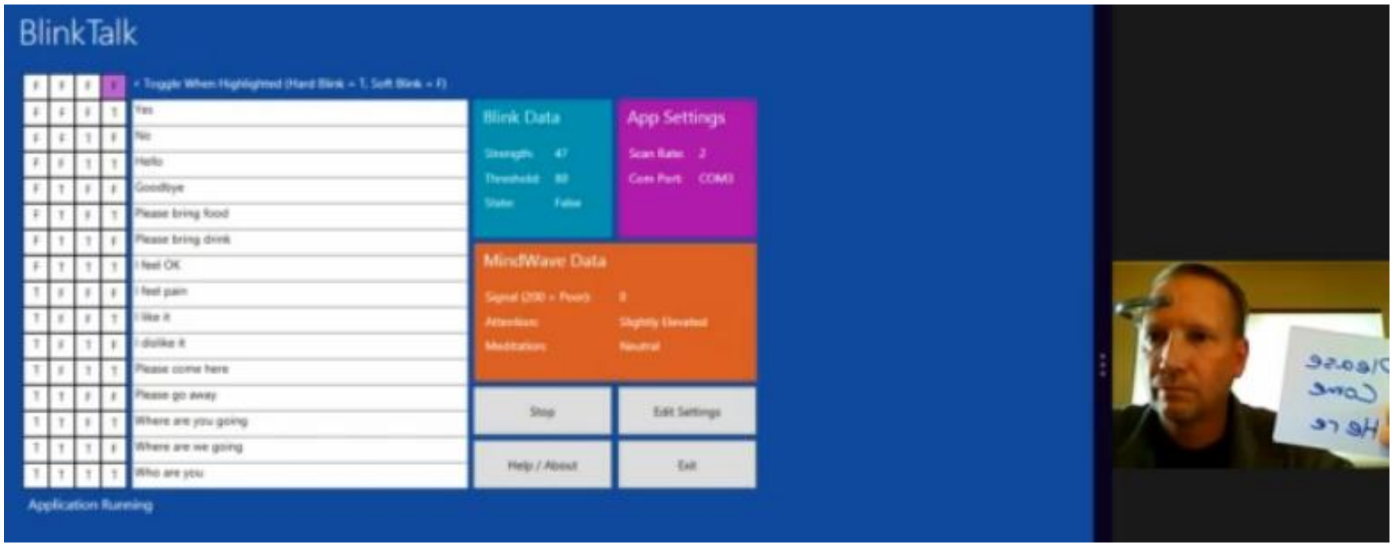
"Currently, BlinkTalk uses a string array to hold spoken phrases," said Brown. "A static AppData class was implemented with methods for saving and retrieving this data along with other application configuration settings, such as blink strength threshold and scan rate. Right now, the number of available phrases is rather limited, but ultimately we want to implement a context-sensitive menu approach to allow blink-navigating to a larger selection of phrases. A database-driven architecture may prove to be a better approach to expanding BlinkTalk's capabilities."

## Final Design Decisions

After reaching the second round of the contest, Brown procured a Code Signing Certificate from Comodo (a prerequisite for publication on the Intel AppUp® center) and received a touch-screen-enabled Ultrabook device from Intel. According to Brown, the addition of touch screen functionality helped to significantly improve BlinkTalk's usability and ease operators' learning curves.

"The ability for a therapist to use touch during patient training sessions, as opposed to keyboard and mouse manipulation, can greatly simplify the interactive learning process," said Brown. "Furthermore, patients with limited hand mobility might be able to augment their interactions with the software and achieve even greater flexibility and depth in how they can communicate."

There is a direct link between touch functionality and UI design, and this extends even to the choice of desktop interface. With a Windows 8-based Ultrabook device in hand, Brown had to make the difficult choice between developing for the conventional Windows desktop or for Microsoft's newer Windows 8-style interface. He opted for the former, even though BlinkTalk has a Windows 8 panel type of appearance. Buttons are oversized and cued for single-touch response for easier operator navigation. Brown noted that "the color coordination and the layout of 'the Windows 8' UI lends itself to making the experience as simple and straightforward as possible."



BlinkTalk

| | | | | |
|---|---|---|---|---|
| F | F | F | T | < Toggle When Highlighted (Hard Blink = T, Soft Blink = F) |
| F | F | F | T | Yes |
| F | F | T | F | No |
| F | F | T | T | Hello |
| F | T | F | F | Goodbye |
| F | T | F | T | Please bring food |
| F | T | T | F | Please bring drink |
| F | T | T | T | I feel OK |
| T | F | F | F | I feel pain |
| T | F | F | T | I like it |
| T | F | T | F | I dislike it |
| T | F | T | T | Please come here |
| T | T | F | F | Please go away |
| T | T | F | T | Where are you going |
| T | T | T | F | Where are we going |
| T | T | T | T | Who are you |

Blink Data
Strength 47
Threshold 80
State False

App Settings
Scan Rate 2
Com Port COM3

MindWave Data
Signal (200 = Poor) 0
Attention Slightly Elevated
Meditation Neutral

Application Running

**Post-publication Video of BlinkTalk in Action.** *This video screen capture shows BlinkTalk as it finally looked when published to the Intel AppUp® center. Note the Microsoft Windows* 8-themed, touch-sensitive tiles and color scheme. At right, Bryan Brown holds up one of the 16 blink-phrases while silently selecting that phrase through the BlinkTalk UI on the left.*

Brown took the Windows desktop path in part because of his own experience and familiarity with multi-threaded UI design for desktop apps in WPF XAML. Nevertheless, Brown appreciates that the market and need for more mobile-centric apps is growing. He is continuing his studies of Windows 8 development and focusing on HTML5- and JavaScript*-based approaches to app development.

## The Port Problem

The first challenge Brown encountered was similar to the one every developer working on his or her own faces: He had to do everything single-handedly, from coding and debugging to video production and website design.

The other human hurdle Brown faced was end-user training. He assumed that the ability to select T/F bit values by blink strength as the app scrolled across the bit positions on-screen would be straightforward and intuitive. Yet trials with able-bodied test subjects revealed that the average user needs about two hours of blink training to become proficient. Fortunately, NeuroSky offers games that can help make the training process more entertaining, and developers are free to create their own training titles.

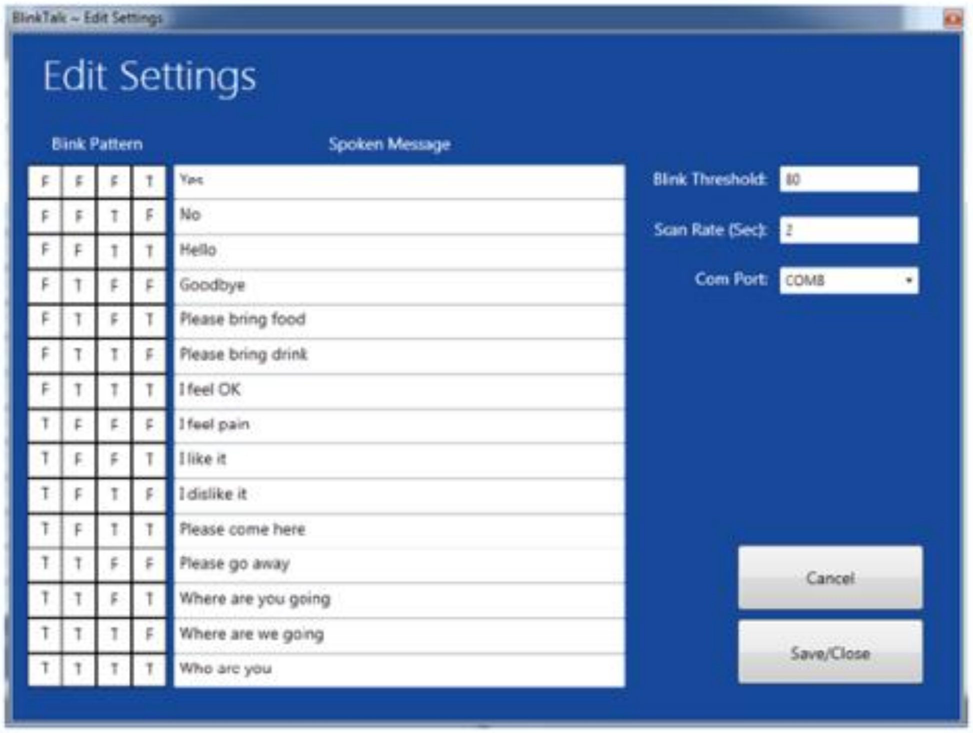
One of the chief technical obstacles Brown faced focused on drivers and port allocation. The original MindWave device used a Bluetooth* connection from the headset to a USB dongle plugged into the Ultrabook device. Every time a user plugged in the dongle, Windows 8 overrode the MindWave driver with a generic USB driver and created a virtual COM port (COM1, COM2, and so on) associated with the MindWave. Occasionally, though, after unplugging and replugging the dongle, the OS swapped not only the assigned COM port but also the associated device driver, requiring the operator to access the Windows Device Manager and manually change the settings in order to get the MindWave functioning again.

With only NeuroSky's own driver and the third-party .NET wrapper in play, Brown was unable to find a direct fix for this problem. Fortunately, the Ultrabook device provided a workaround. Because the

Ultrabook device integrates Bluetooth functionality, Brown was able to connect directly to the "mobile" version of the MindWave. This model dispenses with the USB dongle because it was designed primarily for Android* and iOS* devices, although it also works under Windows. Without a USB device in the configuration, the problem vanished.

## Keeping Threads Straight

Brown also encountered a synchronization problem with the threads, which proved more troublesome than the driver and port allocation issue. In the first version of BlinkTalk, the application constantly fired data along one thread to incrementally advance an array. A second thread used the resulting value from the first thread to access array elements. Unfortunately, this thread implementation was flawed and lacked the necessary exception handling. With only one month between initial work and contest submission, Brown had overlooked this flawed synchronization, which sometimes yielded random runtime errors and crashed the application. The mistake became evident only after publication and only when running Windows 8, not Windows 7.



| Blink Pattern | | | | Spoken Message |
|---|---|---|---|---|
| F | F | F | T | Yes |
| F | F | T | F | No |
| F | F | T | T | Hello |
| F | T | F | F | Goodbye |
| F | T | F | T | Please bring food |
| F | T | T | F | Please bring drink |
| F | T | T | T | I feel OK |
| T | F | F | F | I feel pain |
| T | F | F | T | I like it |
| T | F | T | F | I dislike it |
| T | F | T | T | Please come here |
| T | T | F | F | Please go away |
| T | T | F | T | Where are you going |
| T | T | T | F | Where are we going |
| T | T | T | T | Who are you |

Blink Threshold: 80
Scan Rate (Sec): 2
Com Port: COM8

**Phrase Editing Screen.** *BlinkTalk operators can customize any of the 16 possible spoken phrases as well as adjust the app's blink strength threshold, how quickly the app scans from one input column to the next, and which COM port should be used for the MindWave* device.*

Brown managed to find the root of the problem through a long debugging process in Visual Studio 2010 followed by extensive endurance testing. Intel support representatives worked with Brown throughout the remediation process, even going so far as to purchase a MindWave device for their own BlinkTalk testing. This allowed Intel to confirm the problem and help Brown resolve it. This level of assistance was a significant benefit to Brown because it not only uncovered an operational issue but also revealed some of the complexities of using the software from a user experience perspective. For instance, Brown found that what seems obvious to a developer may sometimes be confusing to an end user. Intel's support allowed Brown to release a BlinkTalk version 1.1 update on the Intel AppUp center in short order, which included more extensive instructions on the application's use and troubleshooting.

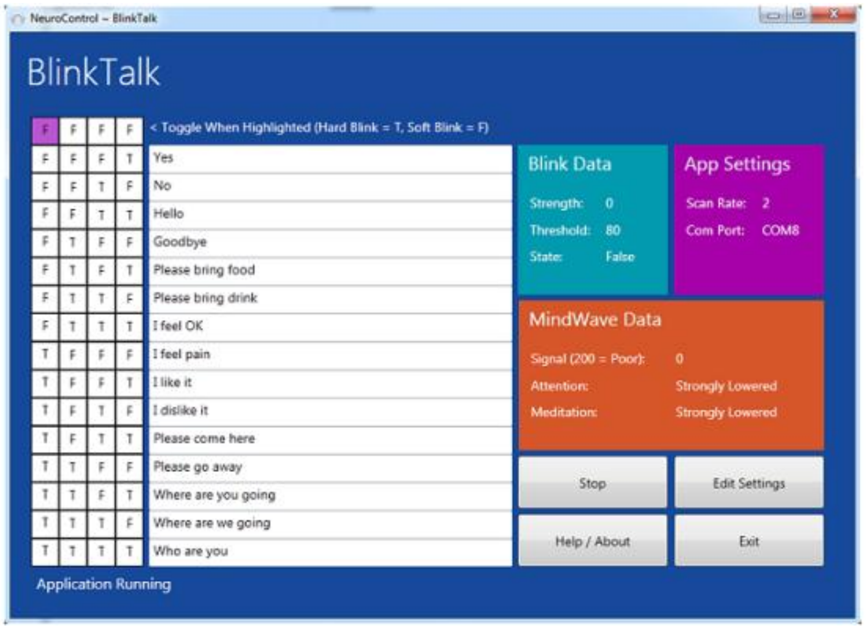
"Any time you develop software that interacts with external devices, you have timing issues, timing constraints, and timing concerns," Brown said. "There are issues associated with streaming data over an Ethernet connection, with latency and devices reacting to what you're doing. This approach to programming is much more dynamic compared to the type of programming that some developers are accustomed to, where they rely on responding to user input in displaying data and that type of thing."

Across the board, Brown found Intel's resources and support very helpful in his development process. For instance, a tutorial video on the Intel® Developer Zone allowed him to construct the silent installer necessary for MSI image creation so he could submit BlinkTalk to the Intel AppUp center. The video was so effective that Brown was able to get the routine right on the first try. He also found that the questions he posted on CodeProject forums were answered "almost immediately." Thanks to these available resources, Brown encountered no serious problems during the creation and submission of BlinkTalk.

## Lessons and Looking Ahead

The Ultrabook device that Intel provided proved essential in ways Brown hadn't anticipated. From a strict Windows-based x86 processing standpoint, any PC could have served for the BlinkTalk project's needs. However, Brown was accustomed to working with touch screen interactivity from all of the analysis instruments at his day job. He laughed about there being "times when I would sit down at a conventional laptop and actually start touching the screen, expecting something to happen."

The Ultrabook device bridged these two input paradigms—conventional desktop and Windows 8-style touch interface—plus it had the power and flexibility to become his main programming platform for other projects he is working on through Human-Machine Technologies. In the field, Brown finds that the compact size and light weight of the Ultrabook device are ideal for fitting into tight bedside patient environments while still providing a convenient keyboard for operators to use.


NeuroControl – BlinkTalk

BlinkTalk

| F | F | F | F | < Toggle When Highlighted (Hard Blink = T, Soft Blink = F) |
|---|---|---|---|---|
| F | F | F | T | Yes |
| F | F | T | F | No |
| F | F | T | T | Hello |
| F | T | F | F | Goodbye |
| F | T | F | T | Please bring food |
| F | T | T | F | Please bring drink |
| F | T | T | T | I feel OK |
| T | F | F | F | I feel pain |
| T | F | F | T | I like it |
| T | F | T | F | I dislike it |
| T | F | T | T | Please come here |
| T | T | F | F | Please go away |
| T | T | F | T | Where are you going |
| T | T | T | F | Where are we going |
| T | T | T | T | Who are you |

Blink Data

Strength: 0

Threshold: 80

State: False

App Settings

Scan Rate: 2

Com Port: COM8

MindWave Data

Signal (200 = Poor): 0

Attention: Strongly Lowered

Meditation: Strongly Lowered

Application Running

**BlinkTalk Main Screen.** *BlinkTalk's main interface screen divides into four basic areas: blink input values (T/F), a list of possible blink-phrases, settings and input data readouts, and a set of four buttons primarily meant for accessing settings options and stopping or exiting the app.*

"You really want to design with larger controls and think in terms of how people will interact with the screen, navigating," Brown said. "Depending on the computer, you can have parallax problems, and, depending on the viewing angle, it can be difficult to have the accuracy to touch small things. Approach your UI design with those factors in mind. Windows 8 panels can facilitate ease-of-use for a touch application."

BlinkTalk won the Healthcare category of the 2012 Intel App Innovation Contest. Before the contest, Brown had no interest in putting his applications into app stores. However, going through the process with the Intel AppUp center showed him that bringing a new app to market can be quick and straightforward, inspiring him to pursue such avenues in the future.

Today Brown is not only focusing on more contest participation and pursuing technology grants, but he is also helping others to explore them.

"I want to expand my website to include open-source projects and more information for developers who are interested in learning how to do this type of programming," he said. "I think there's a bit of a void in some developers' toolboxes. Many lack the experience to be able to interact with physical devices. I want to use my site as a vehicle to help people bridge that gap."

Achieving success in app development doesn't require extensive financial or even time resources. Sometimes, all that's needed is a little push. Bryan Brown illustrates the impact that grassroots efforts such as vendor contests and coding forums can have on developers and the world at large.

## Helpful Resources

Bryan Brown reviewed several information sources while writing BlinkTalk, beginning with background reading on brainwave technology and a thorough review of the input device's technical documentation at developer.neurosky.com. After exhausting the vendor's resources, he turned to ThinkGearNET's third-party library, which provided easy use of the NeuroSky headset through .NET. Bryan used the Intel AppUp® Developer certification tool for creating simple MSI setup files and Comodo for procuring a security certificate. Check out the detailed BlinkTalk article at CodeProject and a live use video at http://youtu.be/YFa9b01u6lo. Find additional information on Brown and his projects at Human-Machine Technologies at http://human-machinetechnologies.com.