Toggling the Visibility and Location of Digital Assets

This disclosure describes a method for computer user interfaces to adjust the visibility and location of digital assets using a toggling mechanism.

Problem / Opportunity:

Users of computer interfaces often require multiple “windows” or visible task interfaces open to complete their task. Frequently, users must refer back to another window to read, edit, or enter commands into as they work. For the purposes of this disclosure, we will refer to this reference window as the Target Window. The Target Window frequently needs to be referenced, then hidden quickly to enable user efficiency.

Description of Invention:

This disclosure describes a way of binding an event, including but not limited to a key press, a mouse action, or an action on another user input device such as a gamepad, to an action which will toggle the availability of the Target Window (the “Bound Event”). Triggering the Bound Event will alternately:

1. Turn visible and place the Target Window in a consistent, yet adjustable location on the computer screen or interface (“State One”)
2. Turn invisible and render the Target Window hidden until the next triggering of the Bound Event (“State Two”)

The Target Window could be any file or system or third-party application on a computer system. The Bound Event could be handled at an application level, where the application would listen for the Bound Event and appropriately render and hide itself. The Bound Event could be handled at the system level, where the computer operating system, firmware, or any other low or mid-level software would listen for the Bound Event and would subsequently issue a command to the application or system windowing software to toggle the state of the Target Window. This Bound Event could be configured through any sort of setup procedure in the application, Operating System, or other firmware. The Bound Event would be set and configured prior to use. Through this configuration method, the Bound Event could be changed at any time. An example of such a setup is shown in Figure 1. The user uses the Target Window’s interface to indicate that the user wishes for a specific Bound Event to toggle the Window’s state. The application listens, possibly by requesting that the System alert it upon specific global keypresses. When the application receives the Bound Event, it requests that the System re-draw the Target Window in the toggled fashion.

The Target Window could be a single digital asset or a cluster of digital assets. The Bound Event could toggle a single application or cycle through applications. Multiple Bound Events could be assigned, referring to different Target Windows or even progressing forward and backward through a series of possible Target Windows. This would be done through System

configuration, as shown in Figure 2.

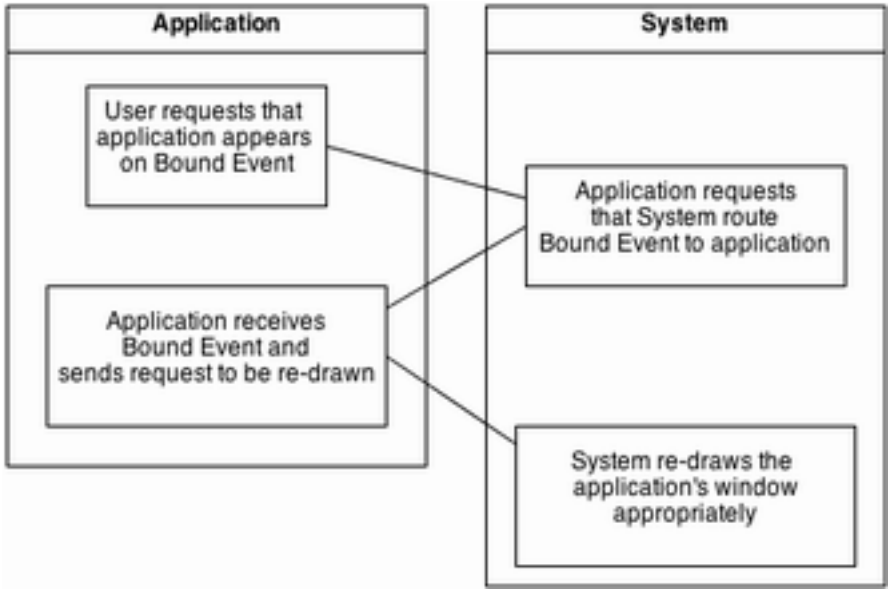

Figure 1: An application-based setup for Bound Events

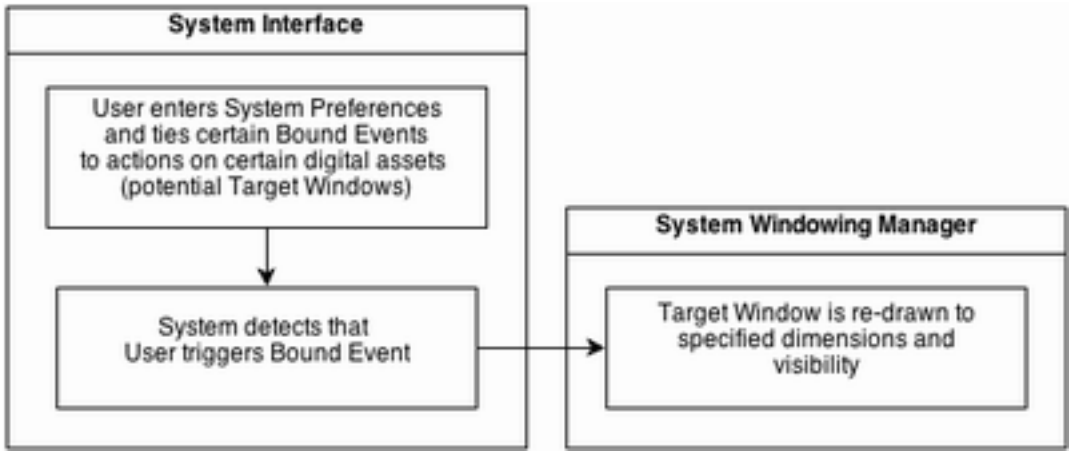

Figure 2: A System-based setup for Bound Events. The Target application is not involved with recognizing or reacting to the Bound Event. Note that this process allows multiple Bound Events to interact with any number of potential Target Windows.

Example:

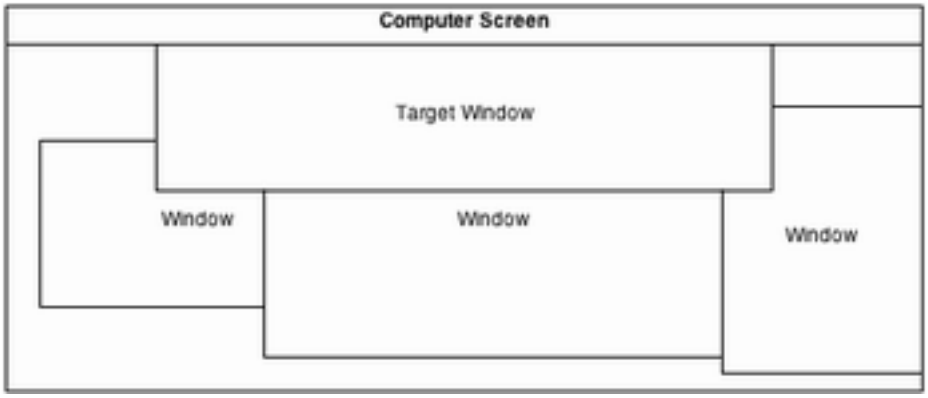

Figure 3: “State One”

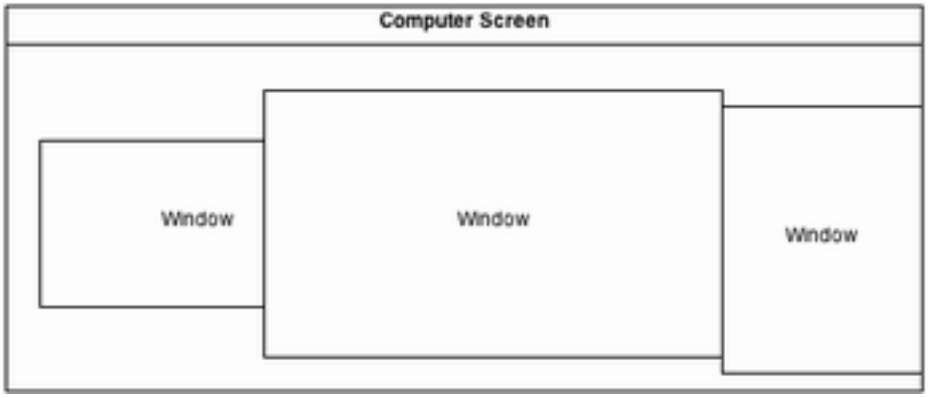

Figure 4: “State Two”

In this example, two potential states are shown. The Target Window is visible and available for use in State One. Then, the Bound Event is triggered. The results are visible in Figure 4: the Target Window is hidden and unusable until the Bound Event is once again triggered. The Bound Event will continue to cause these states to toggle.

An example where this invention may be of use could be for terminals. Developers of computer software and computer power users often make use of Command Line Interfaces for executing text-based commands. Configuring a terminal to appear and disappear on a Bound Event such as a keypress such as the tilde (~) button would be a convenient way to ensure that the terminal would be readily available on command, yet would not take up screen real estate when not needed.