Borderless Windows Usage Guide:  
  
GML Functions:

**ATLA\_remap(Fn\_to\_call, keycode) – Remaps the keys used to invoke the DLL functions.**

Valid params for Fn\_to\_call:

* ”borderless”
* “bordered”
* “fullscreen”
* “restore”
* “dimensions”
* “reset\_window”
* “borderless\_fullscreen”
* “bordered\_fullscreen”

Valid params for keycode:

* Any keycode enum: e.g. vk\_left, vk\_right, etc.
* Any quoted character: e.g. “e”, “4”, “J”

**ATLA\_reset\_window\_pos() – Resets the window position to what it was at game start.**

**ATLA\_set\_corner\_drag\_icon(sprite) – Sets the sprite to be used for dragging a borderless window.**

**ATLA\_is\_sizing() – Returns whether the game is in a sizing operation.**

DLL Functions:

**attach(initial\_state) – Initializes the Borderless Windows DLL, and starts the game in bordered mode (initial\_state == 0) or borderless mode (initial\_state == 1)**

**setDimensions(x,y,cx,cy) – Sets the window position using screen relative coords x and y, with width cx and height cy.**

**drawBorderless() – Sets the bordered state to borderless. Function returns 1 to manage border state.**

**drawBordered() – Sets the bordered state to bordered. Function returns 0 to manage border state.**

**restore() – Restores the window to the last position and border state. Function returns the border state being enabled to manage within GM:S.**

**captionHeight() – Returns the size of the caption bar in pixels. The caption bar is the bar at the top of a window where the menu buttons exist, the title is displayed, and the close, minimize and maximize buttons exist.  
  
borderWidth() – Returns the width of the outlining border that typically surrounds the right and left edges of a window.  
  
borderHeight() – Returns the height of the outlining border that typically surrounds the top and bottom edges of a window (There is a border normally under the caption as well)**

**maxWidth() – Returns the system metric that determines the width of a maximized window.**

**maxHeight() – Returns the system metric that determines the height of a maximized window.**

**maximize() – Maximizes the window. Only works in bordered mode and can cause issues if called in borderless mode.**

**enterSizeMove() – Function that tells the DLL to save the current windows size information to ensure the restore() function restores to a state that didn’t occur during the sizing operation using the drag icons.**

**borderlessFullscreen() – Enables borderless mode and goes fullscreen. Returns 1 to manage border state.**

**borderedFullscreen() – Enables bordered mode and goes fullscreen. Returns 0 to manage border state.**

**GML Objects:**

**boot\_obj – Boot helper object to ensure the game loads everything properly prior to attaching the DLL. Variables are not shared nor should they be accessed or modified. This object should always exist in the top-most room that only contains this object. This object goes automatically to the next room after 30 frames**

**ATLA\_controller – Borderless Windowing control object. Performs keypress processing that calls DLL functions in the step event. The object also calculates whether the mouse is in a hotzone for the drag icons. Hotzones are not enabled when in bordered mode, and processing only occurs if a drag icon sprite has been assigned using ATLA\_set\_corner\_drag\_icon. The drawing of the icon occurs in the GUI event, and dragging is defaulted to left click. This object should be placed in the room following the boot room, and be above any instances that exist in the room that use mouse events. If your game has objects that contain mouse checking functions in a begin step event that shouldn’t process while the window is being resized, change the controllers step event to a begin step event. This object also manages itself so that accidental creation of additional controllers does not interfere with the main controller, and automatically deletes controllers created after the initial one. If no controller exists due to deleting it at runtime, a new one can be created normally without issue.**

**Important Instance Variables:**

* **brdMode – Keeps track of the current border state**
* **brdModeStart – Stores the initial border state for resetting to defaults.**
* **draw – Stores an enum value from the enum to\_draw that determines the hotzone where the mouse is located. Values able to be stored**
* **enum to\_draw – values are none, top\_left, top\_right, bottom\_left, and bottom\_right**
* **keymap – Map data structure that stores key codes used in the keyboard checks in the Step event.**
* **ch – Caption height**
* **bw – Border width**
* **bh – Border height**
* **mx, my – Mouse x and mouse y values that are window relative rather than room or view relative**
* **mxprev, myprev – Position of the mouse the previous step.**
* **app\_pos – Array that stores the coords where the application surface is drawn. This updates after a size-move event ends and is used to ensure that the GUI layer updates its draw params only when the array changes.**
* **reposition – true when a size-move operation is occurring in borderless mode, false otherwise. This variable can be used to ensure that mouse input performed in a step event is blocked in any instances created after it. Simply use ATLA\_is\_sizing() to get this value.**
* **rw, rh – Saves the last steps window width and height. Used to calculate the ratio used to scale the GUI layer.**
* **\_x, \_y, cx, cy – Windows x and y position relative to the screen, and the width and height respectively. These prefixed with orig\_\* are the values when the games boot room exits.**