ZR User API- SpySPHERES MS 2D

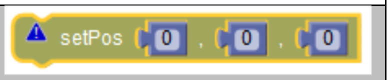
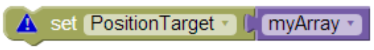
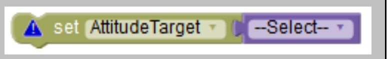
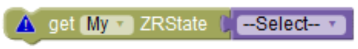
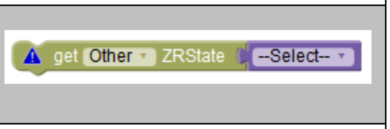
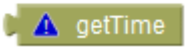
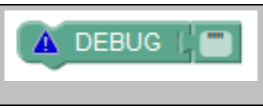
The following reference table explains how to use common api and game functions for the SpySPHERES MS 2D game.

SPHERES Controls API Functions*

These functions used to control a SPHERES satellite in Zero Robotics. These functions do not change from game to game.

Note for teams using the text editor: All SPHERES control functions except DEBUG are accessed as members of the api object. In order to use these functions, use the syntax api.function(arguments). For example:

```
api.setPositionTarget(mypos); //instructs the SPHERE to move to mypos
```

Name	Description	
NOTE: This function is only available for use with the graphical editor	Moves the player's satellite to the given x, y, and z coordinates.	
void setPositionTarget (float posTarget[3])	Moves the player's satellite to a point of your choice. You can select a point by creating a three element array, where each element represents an x, y, or z coordinate.	
void setAttitudeTarget(flo at attTarget[3])	Rotates the player's satellite to face along the x, y, or z axis. You can select the direction by creating a three element array, where each element represents the x, y, or z unit vector of the direction you want to face. For more information, see the *More Simple Arrays and setAttitudeTarget Function* tutorial on the ZeroRobotics website.	
void getMyZRState(float myState[12])	Retrieves the state of your SPHERE (location, velocity, attitude, and angular velocity). The state will be stored in a twelve element array that you create beforehand. After calling this function, the first three elements of your array will hold the x, y, and z coordinate of your SPHERE's location; the next three elements will hold the x, y, and z components of the velocity; the next three elements will hold the x, y, and z components of the attitude vector; and the final three elements will hold the x, y, and z components of the angular velocity.	
void getOtherZRState(flo at otherState[12])	Same as getMyZRState but gets the state of the opponent's satellite.	
unsigned int getTime()	Returns the time (in seconds) elapsed since the beginning of the game.	
DEBUG(("Some text!"))	Prints the supplied text to the console. If you are coding in the text editor, do not type api. before this function and make sure to use double parenthesis.	

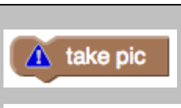
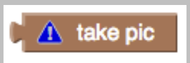
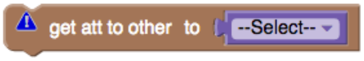
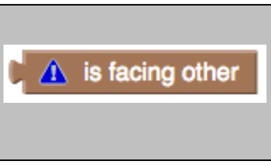
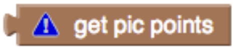
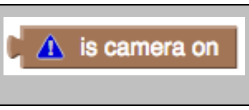
Game Specific Functions- SpySPHERES MS 2D

The functions in the table below are specific to the game SpySHERES MS 2D

Note for teams using the text editor: All game functions are accessed as members of the game object. In order to use these functions, use the syntax game.function(arguments). For example:

```
game.takePic(); //instructs the game to take a picture
```

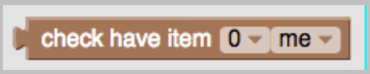
SpySPHERES MS - Pictures

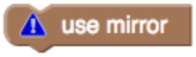
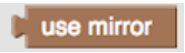
Name	Description	
float takePic()	Attempts to take a picture of the other satellite from its current position and disables the camera for 3 seconds. whether successful or not. Costs 1.0 energy. Returns the number of points that the picture taken is worth.	 take pic  take pic
void getAttToOther(float AttToOther[3])	Returns the x, y, z components of the attitude vector needed to point the players SPHERES camera toward the opponent satellite from its current position. The attitude vector is stored in a 3 element array of your choice.	 get att to other to --Select--
bool isFacingOther()	Check if the players SPHERES camera is facing the other satellite. Returns true if the players SPHERES camera is facing the other satellite within tolerances. Return false otherwise.	 is facing other
float getPicPoints()	Returns the amount of points a picture is worth if taken immediately when the camera is on, otherwise returns 0 when the camera is off. This does not take a picture. This costs 0.1 energy only when the camera is on.	 get pic points
bool isCameraOn()	Checks if the camera is on. Returns true if the camera is on and is usable. Returns false if the camera is off.	 is camera on

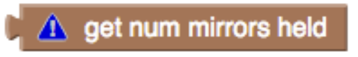
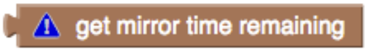
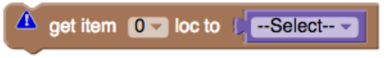
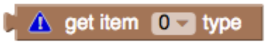
SpySPHERES MS - Items

NOTE: **Item Id** is the number assigned to a particular Item (see Game Manual for details)

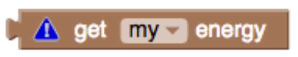
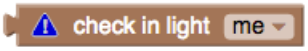
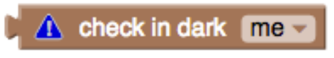
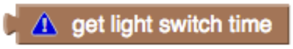
Name	Description	

bool checkHaveItem(itemID);
bool checkHaveItemOther(itemID);
bool checkHaveItemNoOne(int itemID)
	Checks who (me/other/no-one) has a specified item (items 0-8). Returns true if me/other/no-one has picked up the specified item. Returns false otherwise.	 check have item 0 me

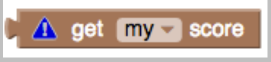
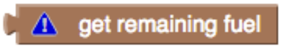
bool useMirror ()	Uses a held mirror item. Returns true if the item was held and was used and false otherwise.	 use mirror  use mirror

int getNumMirrorsHeld()	Returns the number of mirrors currently held by the player that are available for use	 get num mirrors held
int getMirrorTimeRemaining ()	Returns the amount of time left on your current mirror. Returns remaining time in seconds is the mirror is active. Returns zero if the mirror is not active.	 get mirror time remaining
void getItemLoc(float pos[3], int itemID)	Copies the location of a given items into the given array. Stores the location of an Item with the id number **id** in a three element array of your choice. After calling this function, each index in your array will hold the x, y, or z coordinate of the Item's location.	 get item 0 loc to --Select--
int getItemType(int itemID)	Returns what a given item does Score item returns 0, energy item returns 1, mirror item returns 2.	 get item 0 type

SpySPHERES MS - Light/Dark

Name	Description	
float getEnergy() float getOtherEnergy()	Tells how much energy the player satellite currently has.	 get my energy
bool checkInLight(); bool checkInLightOther()	Returns true if the player is in the light zone. False if not.	 check in light me
bool checkInDark(); bool checkInDarkOther()	Returns true if the player is in the dark zone. False if not.	 check in dark me
Int getLightSwitchTime ()	Determines how long until the light and dark zone next switch. Returns the number of seconds until the light switches.	 get light switch time

SpySPHERES MS - Other

Name	Description	
float getScore() float getOtherScore()	Returns player's current score if "my" (Player =0) is selected. Returns opponent's current score if "other's" (Player =1) is selected.	 get my score
float getFuelRemaining()	Tells the player how much fuel remains. Returns a float indicating how many seconds of fuel remain.	 get remaining fuel

API Note Index

API Note 1: *The function* getEnergy() *returns the amount of energy remaining. Once your energy drops below 1.0 your will not be able to activate thrusters .Hint: If you call this function often, you can you can allow* *yourself enough time to make some changes before you run out of energy, for example move toward the light zone where you can recharge, stop the SPHERES so it won't drift out of bounds or rotate to point the SPHERE towards an opponent so you can attempt to take pictures.*

Revision History

Revision	Date	Changes Made
1.0 DRAFT	5/11/16	Initial release
1.1	6/3/16	Updates shown in blue font.