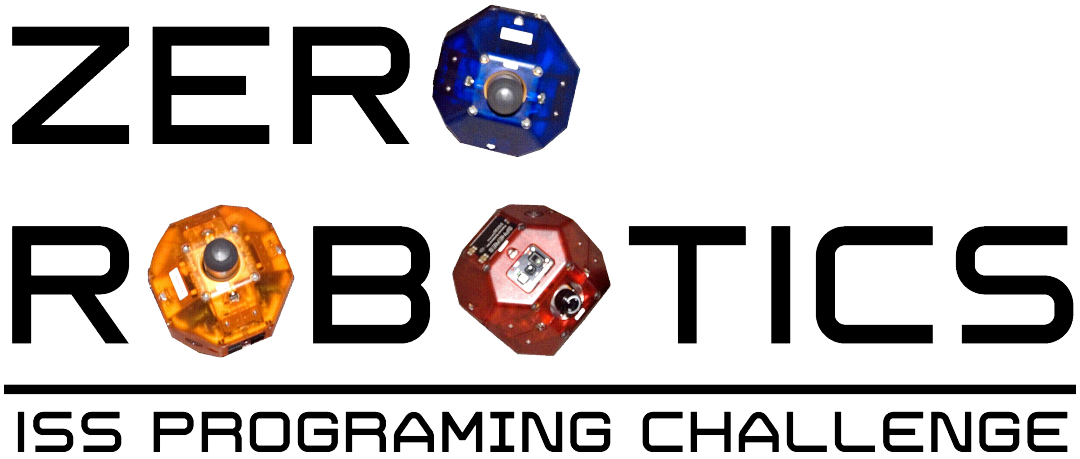


Programming Functions

Functions

- Can be more efficient for programmers to break up code into separate sections that perform different tasks (rather than repeating same text over and over again)
- Programmer puts set of instructions to do a particular task into a *function*, which can be called up repeatedly
- **Real-World Example:** Before you can brush your teeth you need to rinse your toothbrush, put toothpaste on it, and fill your water cup, so “PrepMaterials” might be a function containing all of these instructions that is called repeatedly in the “Brushing Your Teeth” program.

Functions (cont.)

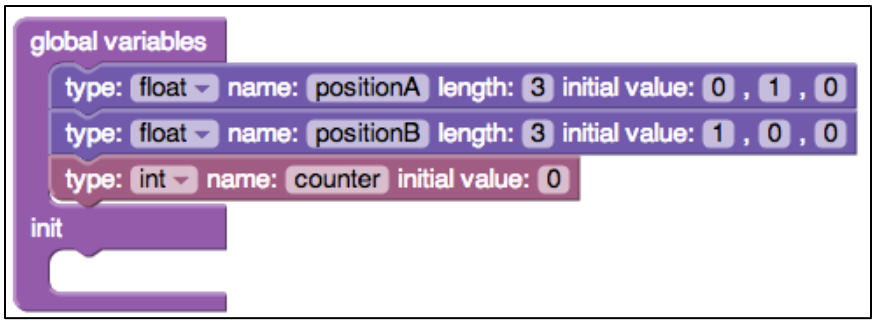
The ZR IDE contains several functions that you will use in your programs to maneuver the satellites:

- called “SPHERES Controls” because they let you control the satellites
- you have used two SPHERES Controls in ZR IDE tutorials:
  - setPositionTarget (which moves the satellite)
  - setAttitudeTarget (which rotates the satellite)

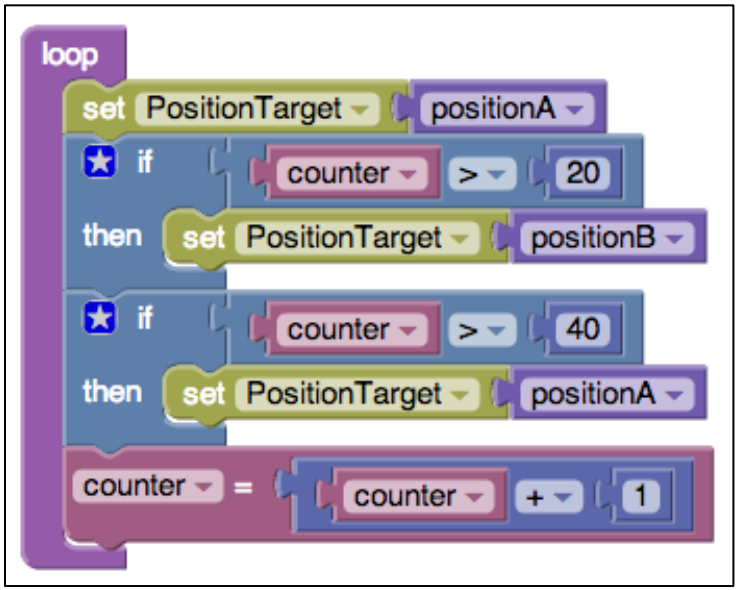
Code Review

- Look at the sample code on the next slide; provided in both text editor format (C code) and graphical editor format.
- Identify programming components you have learned about so far in both the C code and graphical editor code:
  - Arrays (data type)
  - Conditional statement
  - Logic Operator
  - Variable (data type)
  - Function

Code Review (cont.)

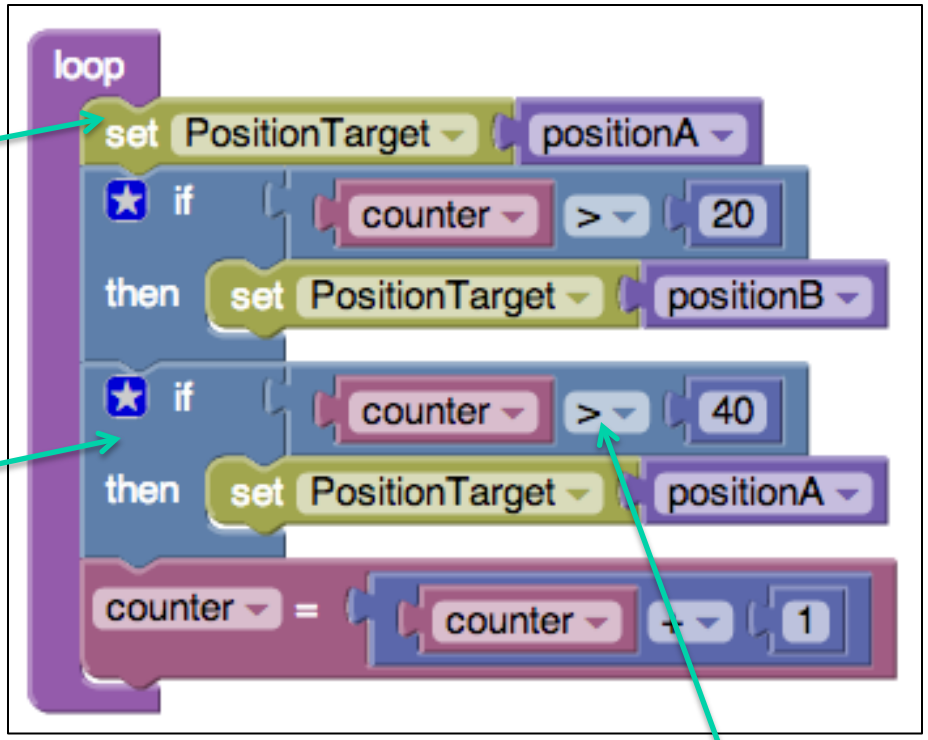



```
1  void init() {
2    positionA[0] = 0;
3    positionA[1] = 1;
4    positionA[2] = 0;
5    positionB[0] = 1;
6    positionB[1] = 0;
7    positionB[2] = 0;
8    counter = 0;
9  }
```



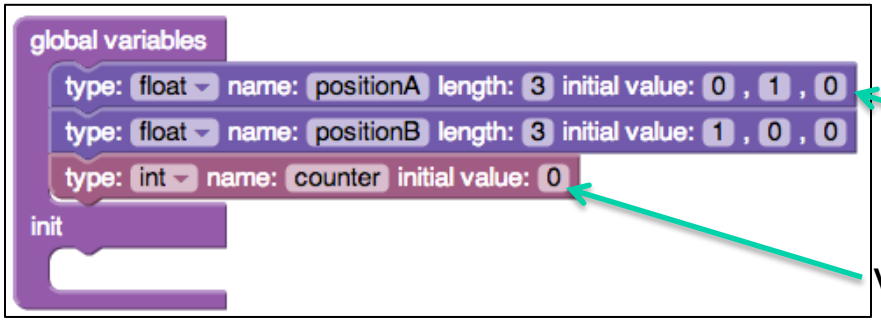

```
1  void loop() {
2    api.setPositionTarget(positionA);
3    if (counter > 20) {
4      api.setPositionTarget(positionB);
5    }
6    if (counter > 40) {
7      api.setPositionTarget(positionA);
8    }
9    counter = counter + 1;
10 }
```

Code Review—Answer



function

conditional statement



array

variable

logic operator

Code Review—Answer (cont.)

function

conditional statement

```
1  void loop() {
2    api.setPositionTarget(positionA);
3    if (counter > 20) {
4      api.setPositionTarget(positionB);
5    }
6    if (counter > 40) {
7      api.setPositionTarget(positionA);
8    }
9    counter = counter + 1;
10 }
```

```
1  void init() {
2    positionA[0] = 0;
3    positionA[1] = 1;
4    positionA[2] = 0;
5    positionB[0] = 1;
6    positionB[1] = 0;
7    positionB[2] = 0;
8    counter = 0;
9  }
```

array

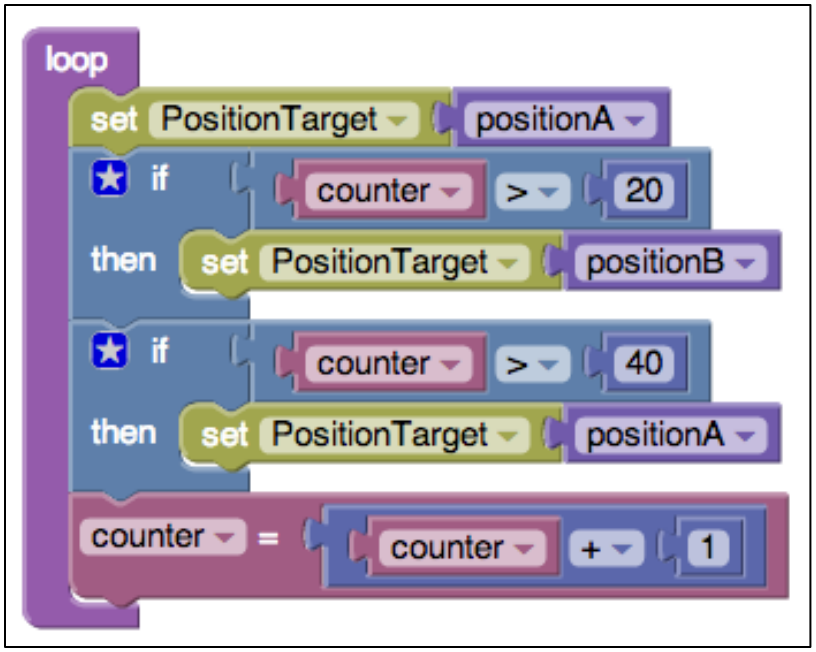
logic operator

variable

Code Review—What Does It Do?

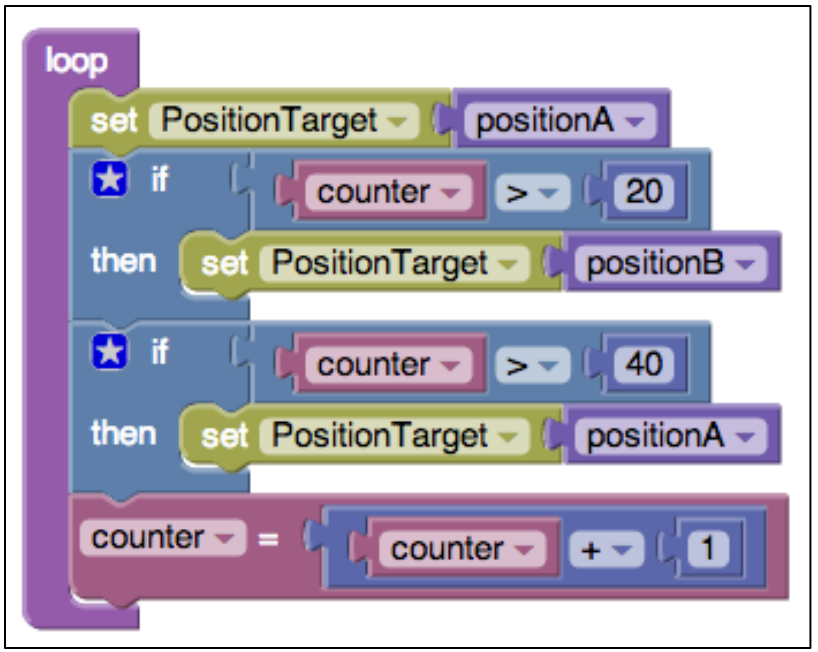
- Now look at the code again
- Can you describe what this code tells the SPHERES to do?

```
1  void loop() {
2    api.setPositionTarget(positionA);
3    if (counter > 20) {
4      api.setPositionTarget(positionB);
5    }
6    if (counter > 40) {
7      api.setPositionTarget(positionA);
8    }
9    counter = counter + 1;
10 }
```



Code Review—What Does It Do? (cont.)

- If you said the program tells the SPHERES to move to
  - Position A
  - Then to position B
  - Then back to Position A
- You are correct!




```
1  void loop() {
2    api.setPositionTarget(positionA);
3    if (counter > 20) {
4      api.setPositionTarget(positionB);
5    }
6    if (counter > 40) {
7      api.setPositionTarget(positionA);
8    }
9    counter = counter + 1;
10 }
```

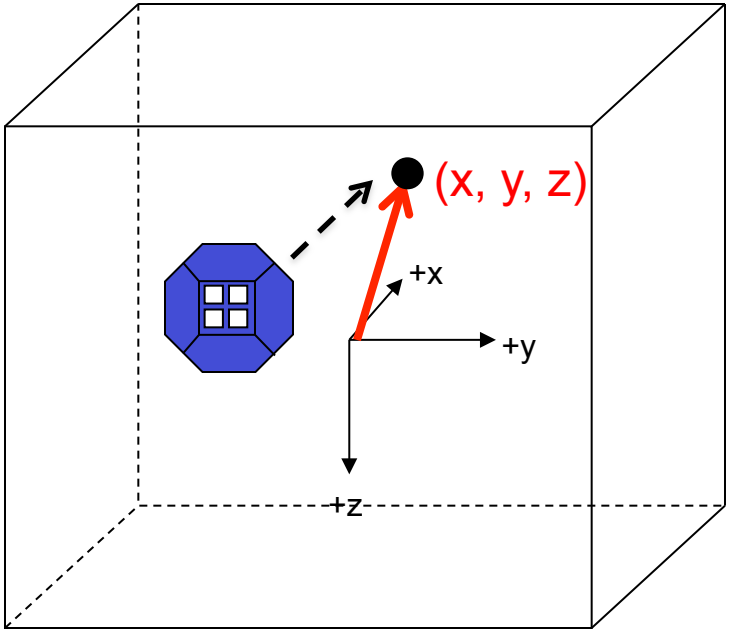
SPHERES Controls Functions

Four functions can be used to control the SPHERES in the ZR Middle School Program:

- **setPositionTarget:** sets x, y, and z position target
- **setAttitudeTarget:** specifies a unit vector for the satellite to point toward (i.e., rotates satellite)
- **getMyZRState:** retrieves ZR state for current satellite
- **getOtherZRState:** retrieves ZR state for second satellite

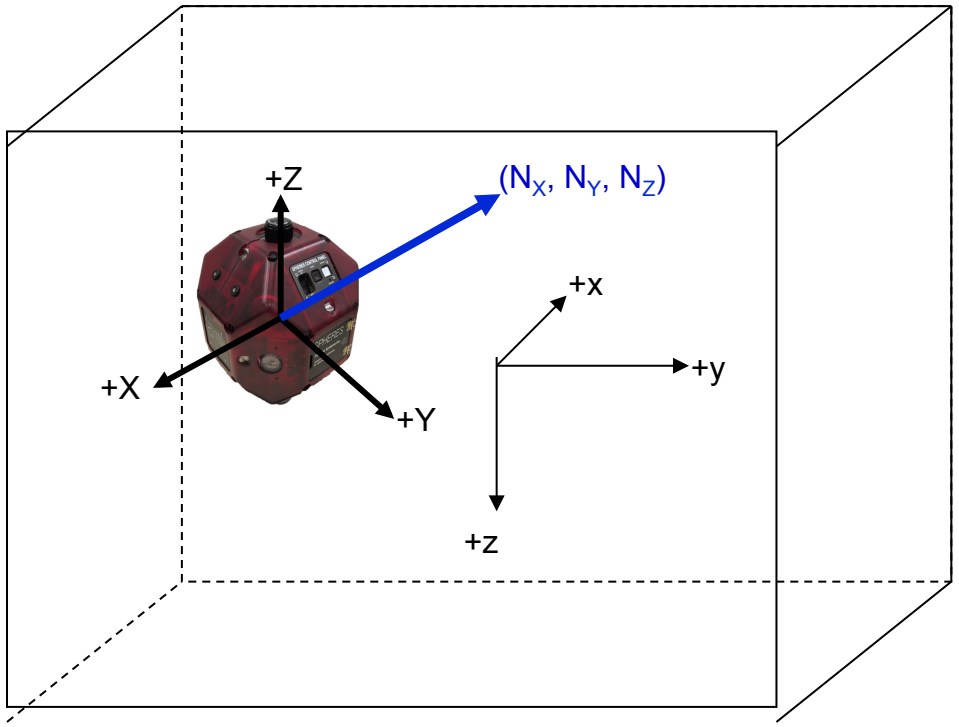
setPositionTarget

- **"setPositionTarget"** allows you to move satellite to a target position
- Input target as an array of three floats (representing its *x*, *y*, *z* coordinates, in meters)
- When position is commanded, satellite will fire thrusters to move to target point, then stop



setAttitudeTarget

- **"setAttitudeTarget"** allows you to set direction for satellite to point its Velcro* face
- Specifies a pointing **direction** (N_x, N_y, N_z) , not a pointing location
- Commanding an attitude target makes satellite fire thrusters to rotate to target direction, then stop



***Note:** Remember that this is the –X face of the satellite and has Velcro on it

getMyZRState

"getMyZRState" retrieves ZR state information (position, velocity, pointing vector, rates) for current satellite

	My_ZR_State		
Position	X: 0.0	Y: 0.0	Z: 0.0
Velocity	Vx: 0.0	Vy: 0.0	Vz: 0.0
Pointing vector	Nx: 0.0	Ny: 0.0	Nz: 0.0
Rotation rates	ωx: 0.0	ωy: 0.0	ωz: 0.0

Note: It is helpful know the 'ZR state' of your own satellite and use that information as you manipulate/move it. *You will learn more about "getMyZR State" in Week 3.*

getOtherZRState

"getOtherZRState" retrieves ZR state information (position, velocity, pointing vector, rotation rates) for second satellite

	Other_ZR_State		
Position	X: 0.0	Y: 0.0	Z: 0.0
Velocity	Vx: 0.0	Vy: 0.0	Vz: 0.0
Pointing vector	Nx: 0.0	Ny: 0.0	Nz: 0.0
Rotation rates	ωx: 0.0	ωy: 0.0	ωz: 0.0

Note: It is sometimes helpful know the 'ZR state' of the other satellite and use that information as you manipulate/move your own satellite. *You will learn more about "getOtherZRState" in Week 3.*