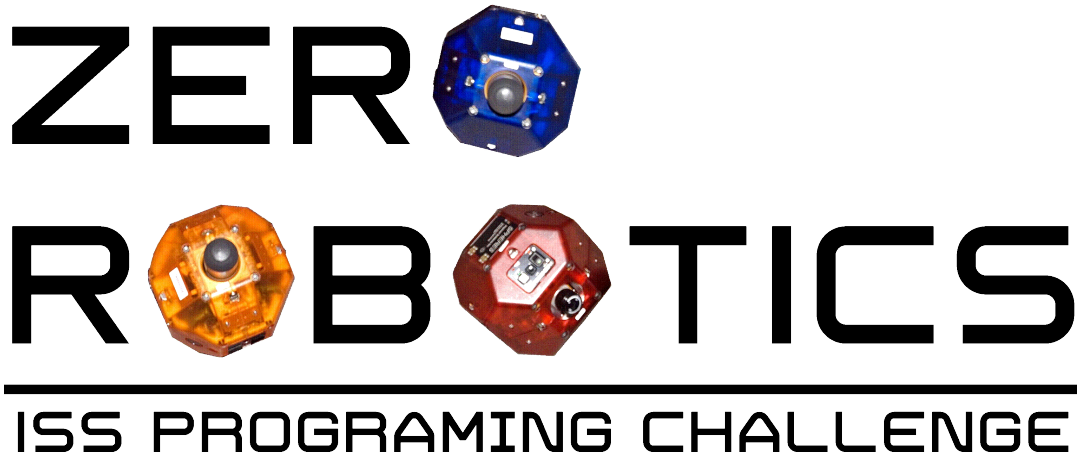
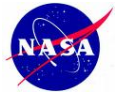
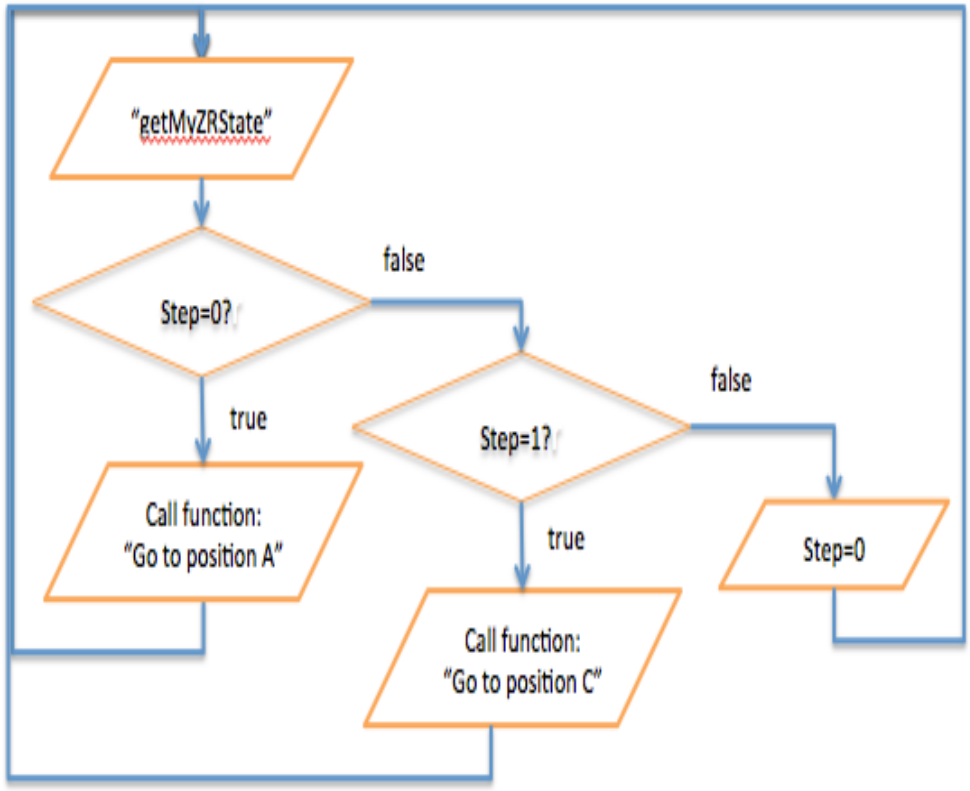


Functions and the Step Counter Model



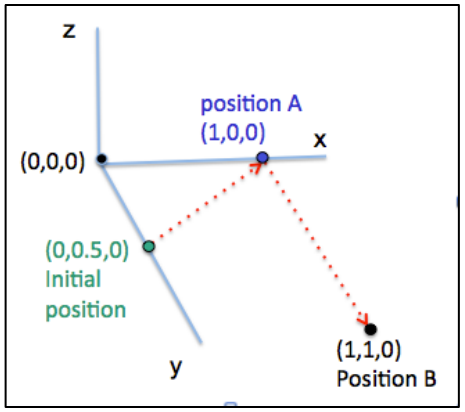

Goals

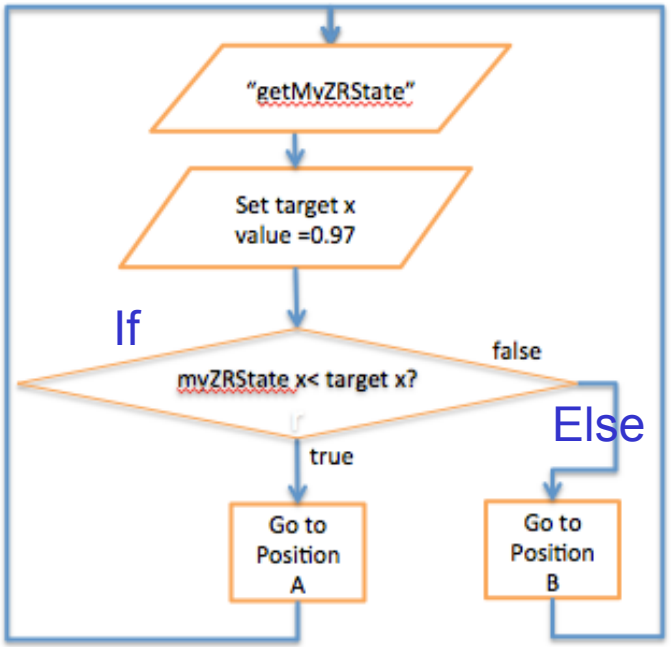
- In this tutorial you will:
  - Learn to use a step counter in your program
  - Practice creating functions
- Important note: There will be several slides at the beginning of this tutorial to read and understand before you begin to create your next program.



Review of previous tutorial

- The sketch and flow diagram on the right describe the program you wrote in a previous tutorial (*Applied Conditionals*)
- What would happen if the location of Position B were changed? Would this program always move the SPHERES satellite to Position B?
- On the next slide you will be given an example of a location where the program would not work and the reason why.





Example that doesn't work

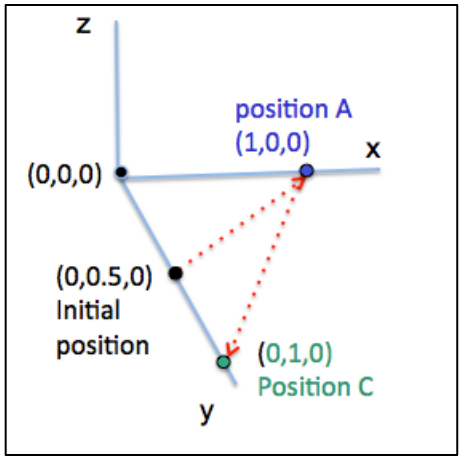
- Look at the new position called Position C.
  - Q: How does the *x*-coordinate of the satellite change as it moves from position A (1,0,0) to position C (0,1,0) in the picture?
  - A: The satellite starts with *x*-coordinate = 1 and moves towards *x* coordinate = 0
- The program sets target x = 0.97 then states:

  If myZRState[0] < target [0],

  (which means If SPHERES *x*-coordinate < 0.97)

  Then go to position A

  Else go to position C
- As the satellite moves from position A toward position C, its *x*-coordinate becomes < 0.97 and it will be sent back to position A.
  - The program will not allow the satellite to move to position C (which has *x*-coordinate < 0.97)

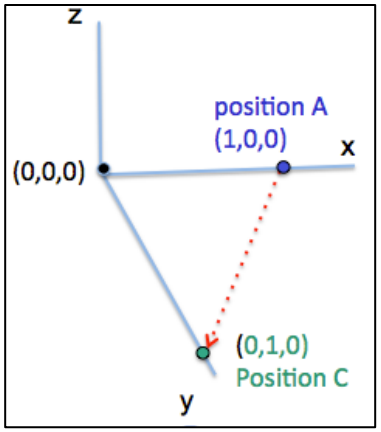


Picking a target value for Position C

- So what is the target value that you would choose to move the satellite from position A to position C?
  - Since the satellite starts with *x*-coordinate=1 and moves towards *x*-coordinate=0:
  - Pick a target close to zero.
  - Pick target[0]=0.03 to include margin for error
- In this case the satellite's *x*-coordinate is **greater than** 0.03 until it reaches the target
- So the conditional statement for this example would be:

"If myZRState[0] **>** target[0]

Then continue to position C"



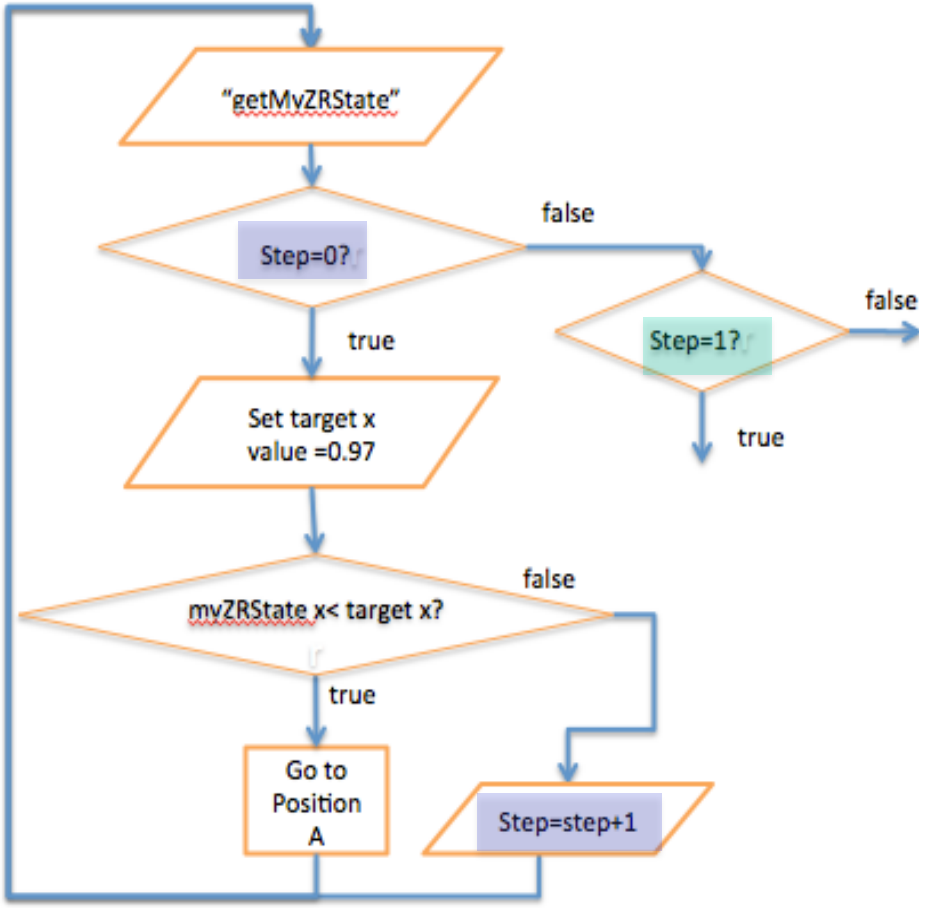
Solution to problem

- We need to write a program that allows us to pick different target values depending on where we want the satellite to move

- We can solve this problem using a step counter (as described on the following slides)

Using a step counter

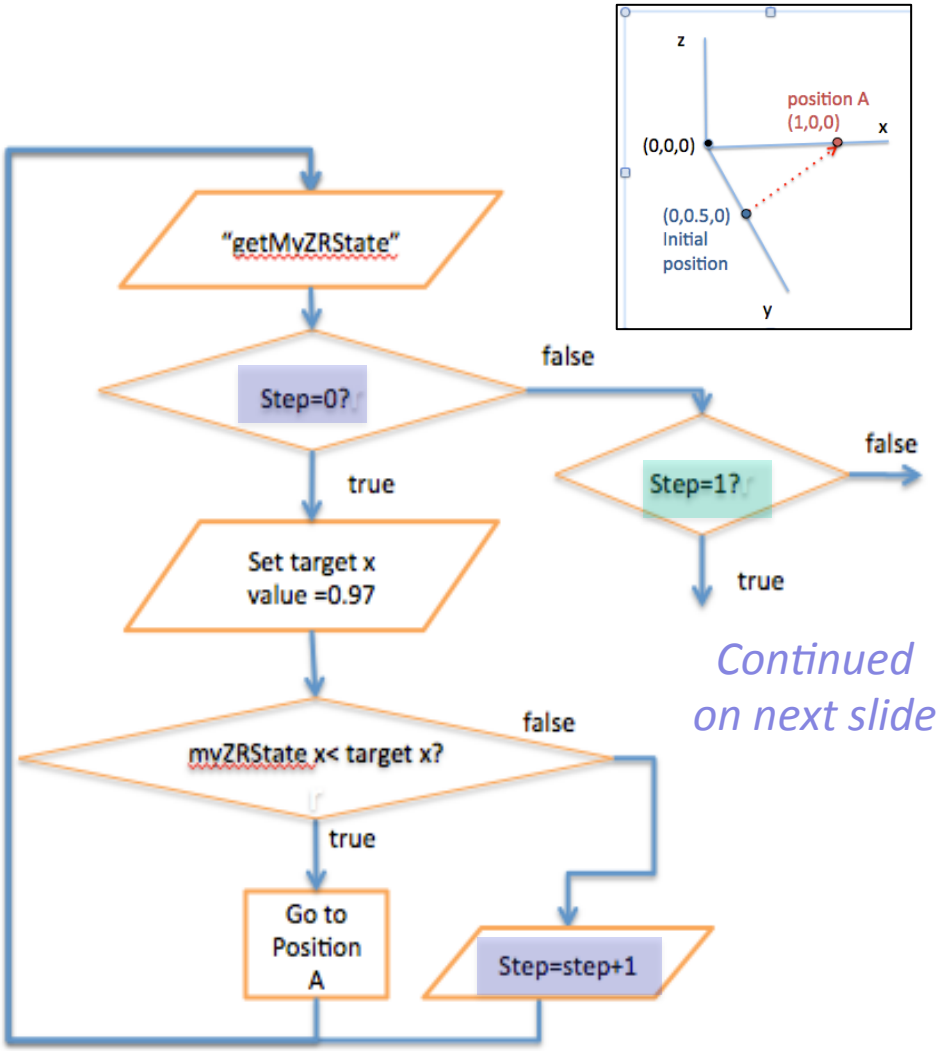
- A step counter is one way you can organize a program. It is a particularly useful example of a programming concept called a *finite state machine.*
- To use a step counter:
  - Break up the program into steps (for example, moving to a point is one step)
  - Use a variable to keep track of how many steps have been performed
  - Use conditionals to make sure you execute only the next step in the process
  - This process ensures that all steps happen in the right order
- An example of this process is provided on the next several slides.



Using a step counter (cont.)

Look more closely at the example: Let's say the first step in the program (step 0) is for the satellite to go to position A

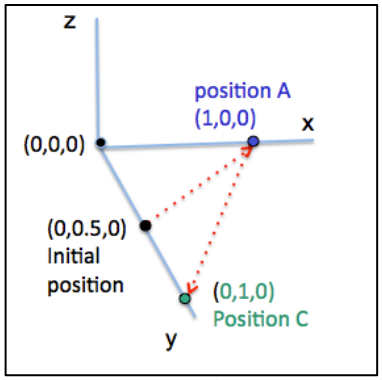
- The flow diagram to the right includes a step counter. Note that:
  - A conditional statement is added to the program to check if the program is in the first step (Step=0?)
  - The step is increased by 1 (Step=step+1) after the satellite completes the first step by reaching position A
  - Another conditional statement is added to check whether or not the program has moved to the next step (Step=1?).

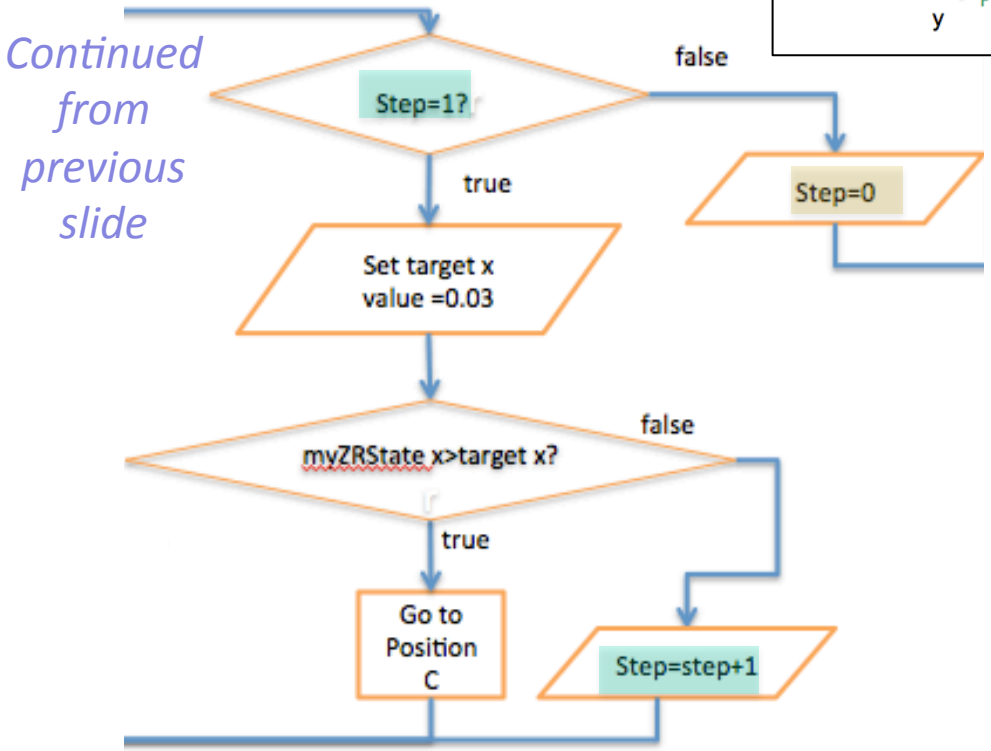


Continued on next slide

Using a step counter (cont.)

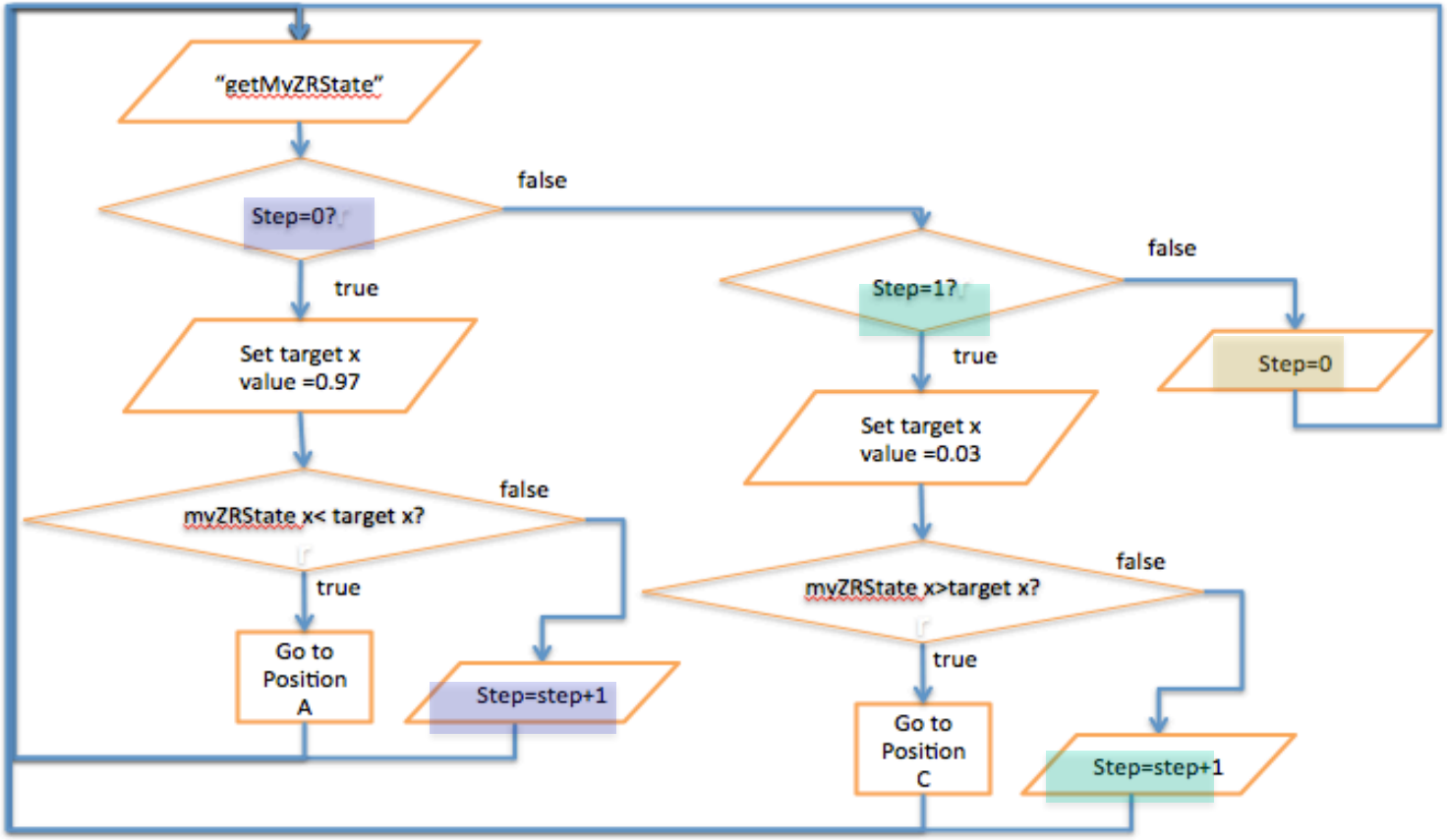
- Next we want the satellite to leave position A and go to position C (as shown in the sketch)
- Using a step counter, we don't have the same problem that we had before because the variable target x can be given a new value inside each step.
- For Step=0: target x was set to a value appropriate for position A (see previous slide)
- For Step=1? target x is set to a new value appropriate for position C (shown here)
- Next, Step=0 resets the step counter





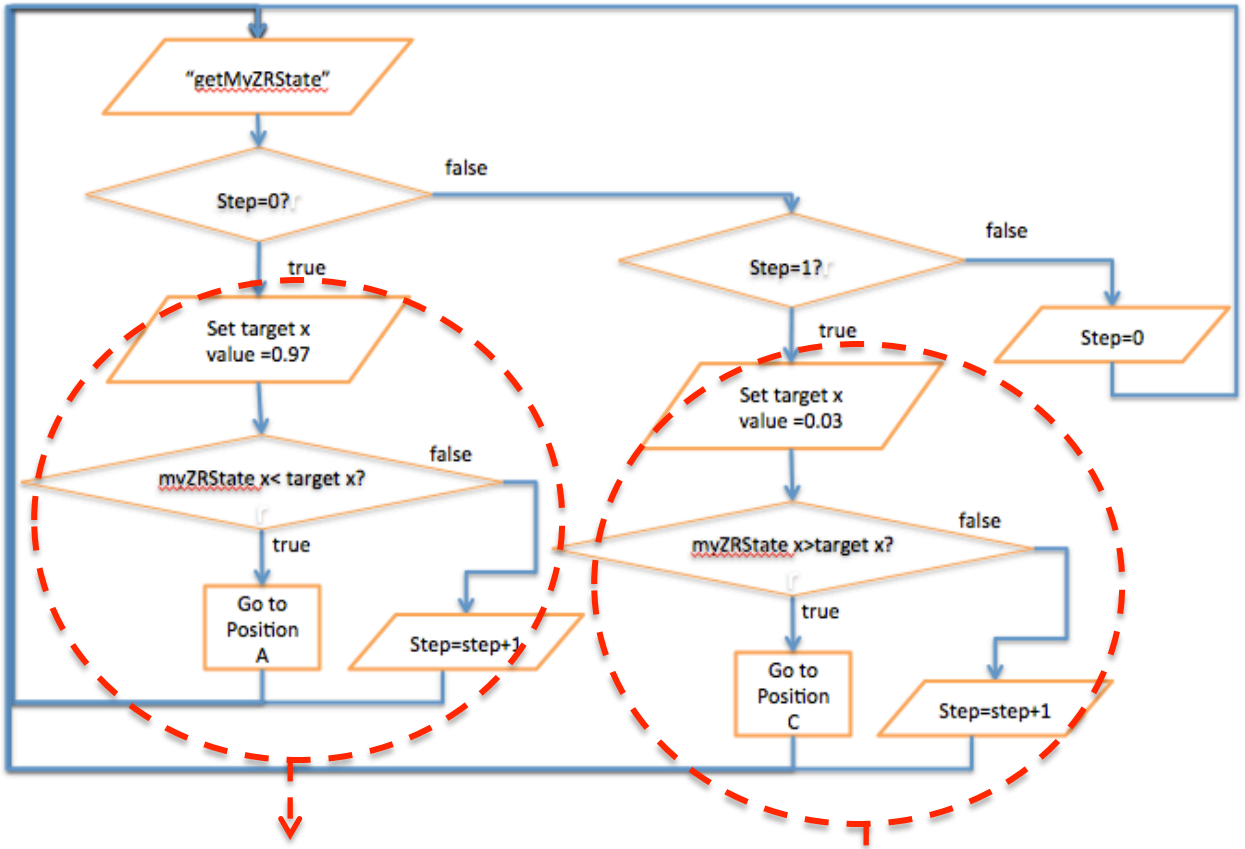
Using a step counter (cont.)

- This slide shows the complete flow chart



Functions

- We can make this program simpler to read by breaking it down into smaller pieces.
- This is done by creating procedural functions
- For example, we can create two functions
  - One that includes the parts of the program that sends the satellite to position A
  - One that includes the parts of the program that send the satellite to position C

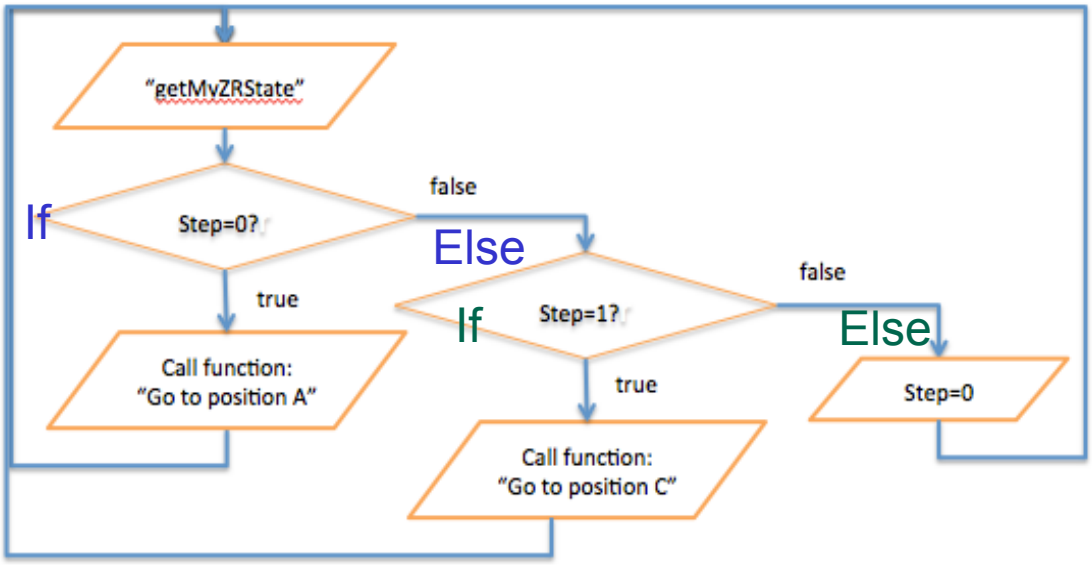


Create function called "Go to position A"

Create function called "Go to position C"

Functions (cont.)

- This is what the example program's flow diagram and text editor program would look like if written using two functions: one called "Go to position A" and one called "Go to position C"




```
15
16  void loop(){
17    api.getMyZRState(myZRState);
18    if (step==0){
19  Go_to_postionA();
20  }
21    else if (step==1){
22      Go_to_positionC();
23    }
24    else{
25      step=0;
26    }
27  }
```

- You don't see Step=step+1 here because it is included inside each function.

Create a New Project Using "Save As"

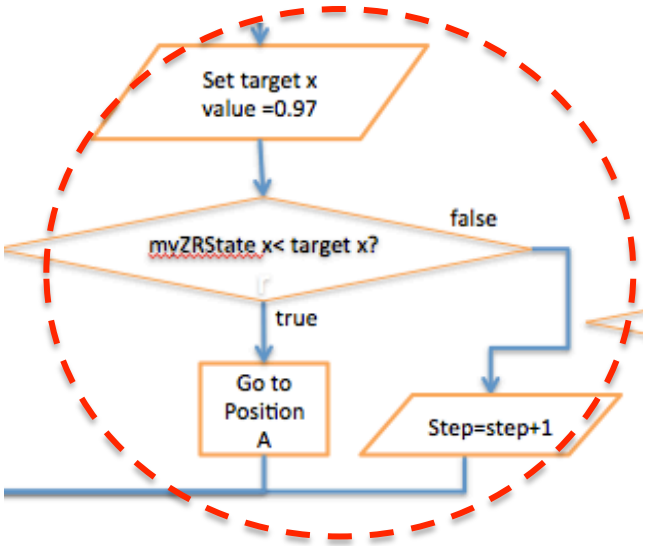
- To create this program with a step counter and functions, you will start from the program you created in the previous tutorial
  - Open the ZR IDE
  - Open Project 10
  - On the menu bar select "File" and then "Save As" from the drop down menu.
  - Type in **Project 11** and select **Free Mode**
- You will need to create two new variables
  - `float positionC [3]` : Set initial value to 0,1,0
  - `Int step :` Leave initial value blank
- Verify the remaining variables as follows:
  - `float positionA[3] :` Verify initial value is set to 1,0,0
  - `float myZRstate[12] :` Leave initial value blank
  - `float target[3] :` Leave initial value blank

main | Rename | Update | Revert | Remove

```
1  float myZRState[12];
2  float positionA[3];
3  float target[3];
4  float positionC[3];
5  int step;
6
7  void init(){
8
9  positionA[0] = 1;
10 positionA[1] = 0;
11 positionA[2] = 0;
12 positionC[0] = 0;
13 positionC[1] = 1;
14 positionC[2] = 0;
15
16 }
```

Create Go_to_positionA Function

- Your program already has a function to send the SPHERES to position A that is very similar to the one shown in the flow diagram
- Click on the page Go_To_positionA
- Can you see the difference between your program and the part of the flow diagram that we want to make into a function called **go_to_PositionA**?
- (Hint:
  - Compare your program's **`else`** statement with the flow diagram
    - your old program continues to set the SPHERES position to positionA
    - the new program increments the step counter (step=step+1)



Go_To_poistionA | Rename | Update | Revert | Remove

```
1  void go_to_positionA(){
2
3    target[0] = 0.97;
4
5    if (myZRState[0] < target[0]){
6      api.setPositionTarget(positionA);
7    }
8
9    else{
10     api.setPoistionTarget(positionA);
11   }
12 }
```

Create "go_to_positionA" Function (cont.)

- Delete **api.setPositionTarget(positionA)** block from the else statement in your function **go_to_positionA**.

- Increment the step counter in the "else" slot
  - It should read **step++;**

- Your function "go_to_PositionA" is complete!

delete

Go_to_positionA | Rename | Update | Revert | Remove

```
1  void go_to_positionA(){
2    target[0] = 0.97;
3    if (myZRState[0]< target[0]){
4      api.setPositionTarget(positionA);
5    }
6    else{
7      api.setPositionTarget(positionA);
8    }
9  }
```

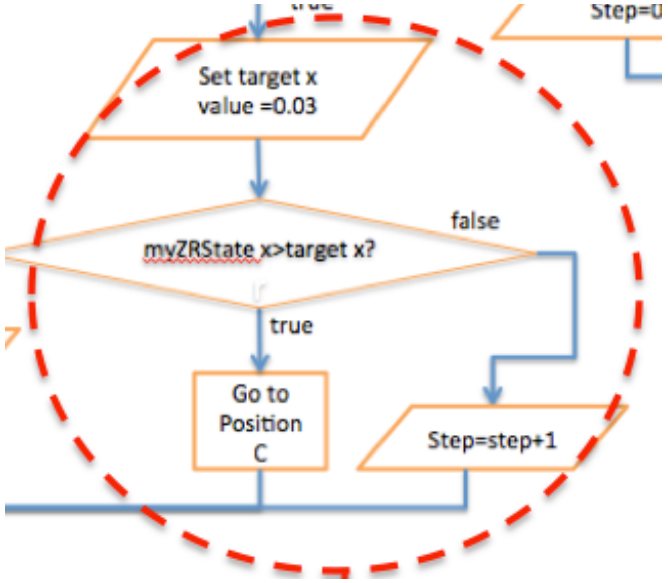
add

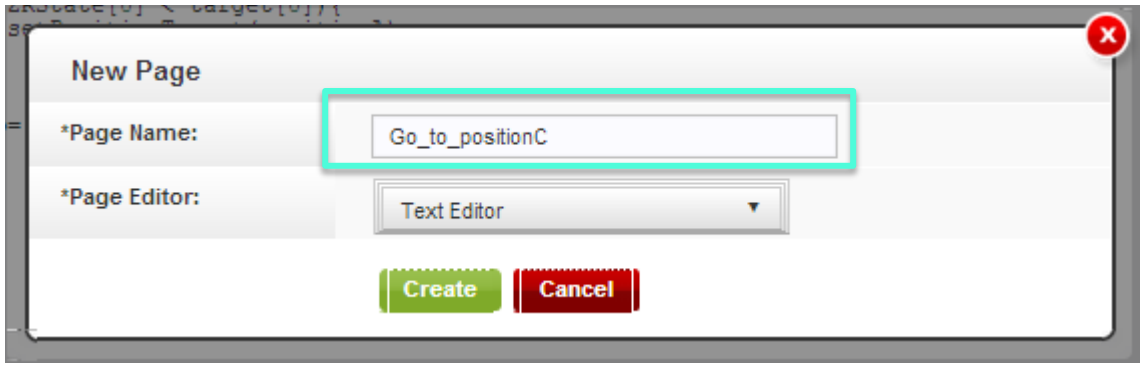
Go_to_positionA | Rename | Update | Revert | Remove

```
1  void go_to_positionA(){
2    target[0] = 0.97;
3    if (myZRState[0]< target[0]){
4      api.setPositionTarget(positionA);
5    }
6    else{
7      step++;
8    }
9  }
```

Create go_to_positionC Function

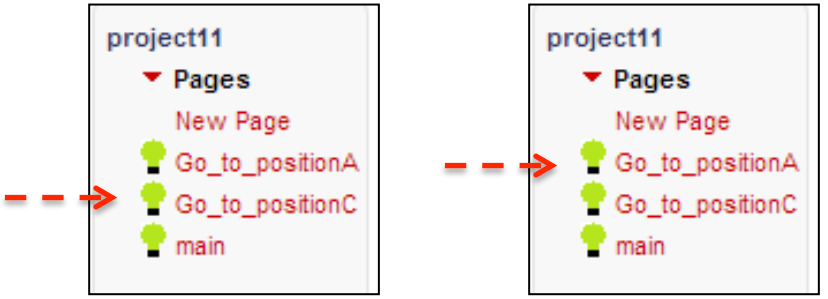
- Now let's create the second function in the flow diagram: **go_to_positionC**
- The first step is to create a new page called **go_to_positionC**
- Click on New Page to create a new function
  - For Page Name type: go_to_positionC
    - This will be the name of your function
  - Select Text Editor
  - Click the green "Create" button





Create go_to_positionC Function (cont.)

- Your new page will show up in the list of pages.
- Click on the **Go_to_positionA** page.
- Because the two functions are similar, you will copy and paste the code from the **Go_to_positionA** page into the **go_to_positionC** page and then edit.
- This code now needs to be edited to send the satellite to positionC (instead of positionA) as described on the next slide.



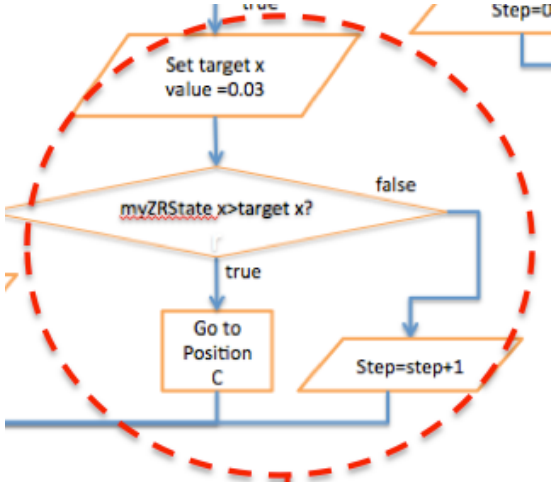
Go_to_positionC | Rename | Update | Revert | Remove

```
1  void go_to_positionA(){
2    target[0] = 0.97;
3    if (myZRState[0]< target[0]){
4      api.setPositionTarget(positionA);
5    }
6    else{
7      step++
8    }
9  }
```

Create "go_to_positionC" Function (cont.)

- The portion of the flow diagram that sends the satellite to position C is shown to the right as a reference
- Change the first line to

  void go_to_positionC()
- Change "target [0] = 0.97" to: "target [0] = **0.03**"
- Change the conditional statement from myZRState[0]<target[0] to myZRState[0]>target[0]
- Change the api.setPositionTarget(positionA) to api.setPositionTarget(positionC)
- **Step++;** should already be there
- Your function is complete!

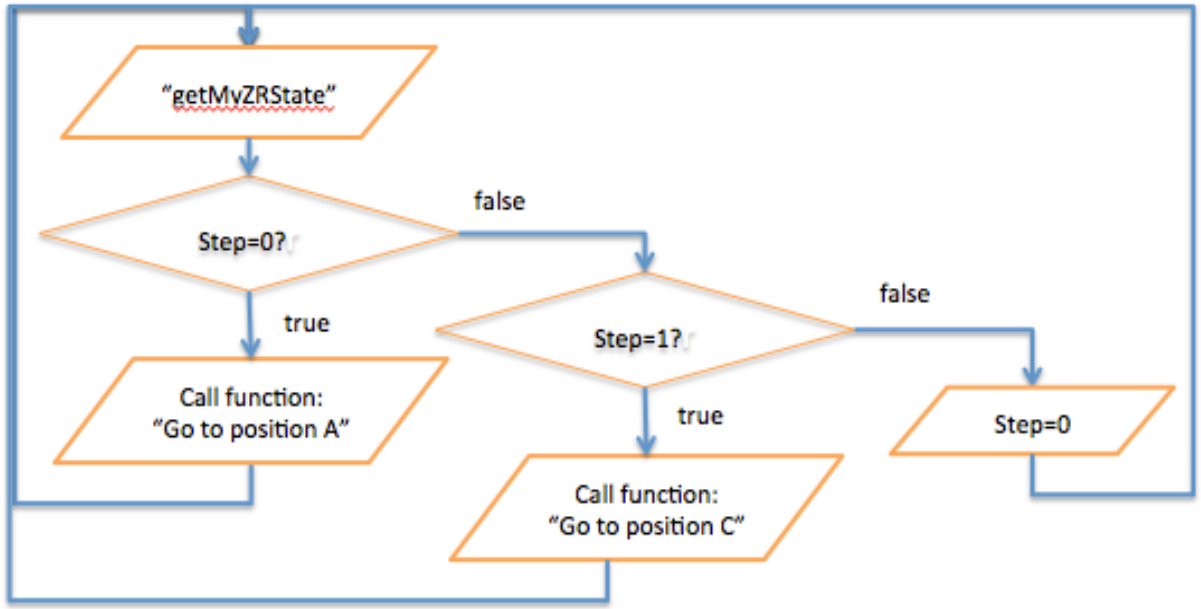


Go_to_positionC | Rename | Update | Revert | Remove

```
1  void go_to_positionC(){
2    target[0] = 0.03;
3    if (myZRState[0]< target[0]){
4      api.setPositionTarget(positionC);
5    }
6    else{
7      step++
8    }
9  }
```

Using the Step Counter Model

- The next step is to go back to the main loop and create the program shown below
- This program uses a step counter and “calls” the functions
- Try creating the program on your own, and use the next slide to check your work.

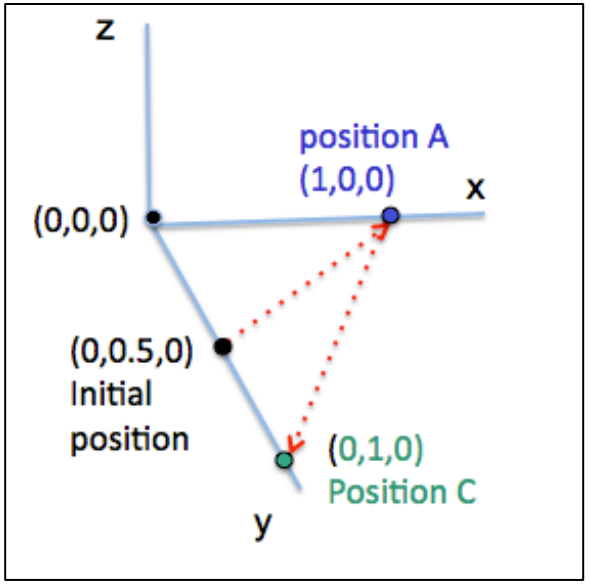


Using the Step Counter Model (cont.)


```
void loop(){
   api.getMyZRState(myZRState);
   if (step==0){
Go_to_postionA();
}
   else if (step==1){
      Go_to_positionC();
   }
   else{
      step=0;
   }
}
```

View Simulation

- Your code is complete!
- Compile, Simulate
  - Load settings: Tutorial _180
  - View simulation



Your Code

The code for the pages **main**, **Go_to_positionA**, and **Go_to_positionC** is shown below:

main | Rename | Update | Revert | Remove

```
1  float myZRState[12];
2  float positionA[3];
3  float target[3];
4  int step
5
6  void init(){
7    postionA[0] = 1;
8    postionA[1] = 0;
9    postionA[2] = 0;
10   postionC[0] = 0;
11   postionC[1] = 1;
12   postionC[2] = 0;
13 }
14
15 void loop(){
16   api.getMyZRState(myZRState);
17   if (step==0){
18 Go_to_postionA();
19 }
20   else if (step==1){
21     Go_to_positionC();
22   }
23   else{
24     step=0;
25   }
26 }
27
```

Go_to_positionA | Rename | Update | Revert | Remove

```
1 void go_to_positionA(){
2   target[0] = 0.97;
3   if (myZRState[0]< target[0]){
4     api.setPositionTarget(positionA);
5   }
6   else{
7     step++;
8   }
9 }
```

Go_to_positionC | Rename | Update | Revert | Remove

```
1 void go_to_positionC(){
2   target[0] = 0.03;
3   if (myZRState[0]< target[0]){
4     api.setPositionTarget(positionC);
5   }
6   else{
7     step++
8   }
9 }
```

Using Functions

- The program you just created resets the step counter to zero.
- If you wanted to program the satellite to go to another position after going to positionC, can you see how this would be done?
  - You would:
    - Create a new function
    - Replace **step==0** with an **else if** statement for **step==2**
    - Call the new function
- When you program your SPHERES for the game, you will probably use a series of nested **else if** statements with multiple steps
- Using functions will also make it easier for you to figure out which parts of your program need debugging.

Review

- Congratulations!
  - You have learned how to:
    - Use a step counter
    - Create multiple functions
  - You are almost ready to start programming for the game!