PROGRAMMING IN ALICE

WORKSHOP: CREATING METHODS

LEARNING OBJECTIVES

The learning objectives for this workshop are:

- Students can define new methods for existing classes
- Students can define methods with parameters
- Students can use these methods in their programs

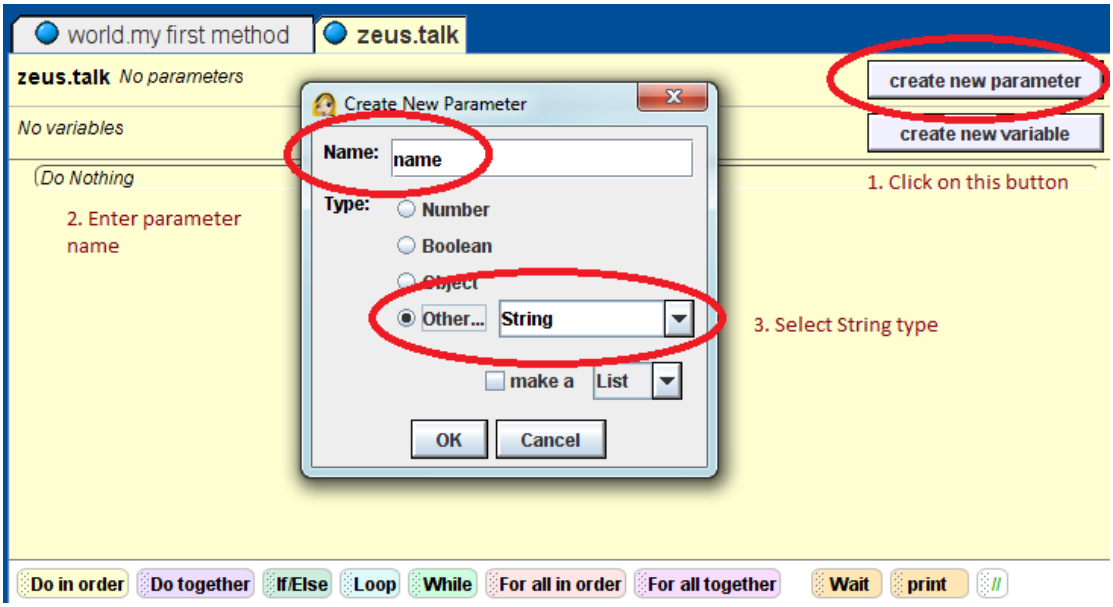
EXERCISE 1

Create a new world that includes the Zeus character (from the People gallery). In this exercise you will create a new method in which Zeus will tell us his name and age.

1. Create a new object level method called *talk* for the Zeus class. To do this click on the Zeus object, then click on the methods tab and finally click on the **create new method** button.
2. In the *talk* method program Zeus to say "Hello my name is Zeus", followed by "I am 99 years old".
3. Call the *talk* method in my first method and run the program.

You will now use parameters to vary the name and age of the character.

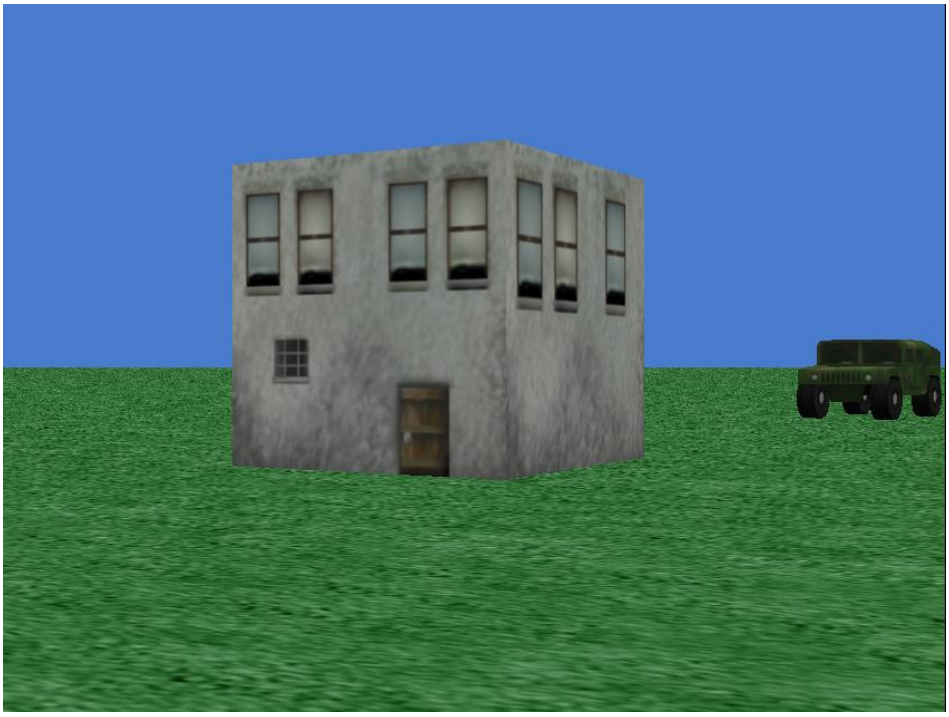
4. Create a new parameter for the talk method called *name*. This parameter should have type String.



5. Change the code inside of the talk method so that the Zeus object say "Hello my name is _*name*_" (where _*name*_ will be the value of the *name* parameter).
6. Update the call to the talk method in my first method by passing a value to the name parameter (you can pass it any string – perhaps try your own name).
7. Repeat the above steps, but this time adding a second parameter called *age*, representing the age of the character. You will need to use the string utility functions to combine the age with the other text. The Zeus object should say "I am _*age*_ years old".

EXERCISE 2

Create a new world that includes a Humvee object (from Vehicles gallery) and a building or other large object. Your objective is to move the Humvee around the building. To do this you will create methods to drivethe Humvee forward, turn left and turn right.



1. Create a new method called **drive.** This method should include a parameter called *distance*, with type Number. The method should move the Humvee forward by an amount given by the value of *distance*.
2. Call the drive method within **my first method** to ensure that it works correctly.
3. Create a new method called **turn_left**that includes a parameter called *revolutions*specifies the number of revolutions to turn, e.g. a value of 0.25 should turn the Humvee 90 degrees. The method should turn the Humvee left by the amount given by the *revolutions* variable.
4. Repeat the previous step to create a method called **turn_right**. It should the same as the previous method, except it will turn to the right.
5. Call **turn_left** and **turn_right** within **my first method** to ensure the work correctly.
6. Using a combination of calls to **drive**, **turn_left** and **turn_right**, get the Humvee to move around the building.