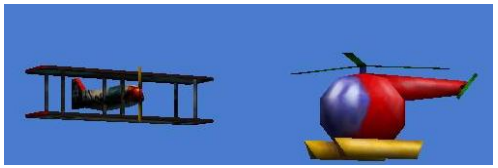
CompSci 4
Chap 6 Sec 2
Sep 28, 2006



Prof. Susan Rodger

Announcements

- Review for test next time.
  - Hand out Test 1 from last semester
    - Should try it before next class
  - Old Quizzes will be available on Blackboard
  - Study classwork and lecture notes
- Next assignment handed out after fall break
- Today – Chap 6, Sec 2
  - Execution control – if/else & Boolean functions
  - Relational operators
  - Logical Operators

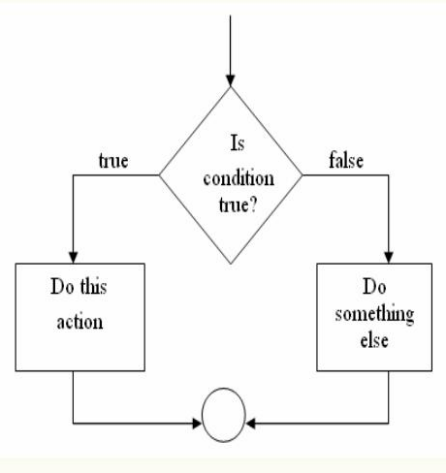
Thinking - More Advanced Worlds

- How do you build animations like simulations and video games?
- Need to write code that involves **decisions**
- Example car-race simulation
  - If the car stays on the road the score increases
  - If the car goes off the road into the stands, the car crashes
  - If the driver gets the car over the finish line, the time is posted and the driver wins!

Logical Expressions

- Decision is made based on current conditions.
- Condition is checked in a logical expression that evaluates to *true* or *false* (Boolean) value.
  - car on road → true
  - car over finish line → false

If/Else



- In Alice, a logical expression is used as the condition in an If/Else control structure
- Decisions (using If/Else) are used in
  - Functions
  - Methods

Example: Boolean Functions

- Suppose we build a simulation system used to train flight controllers
- One of the tasks of a flight controller is to be alert for possible collisions in flight space

Storyboard

- Two aircraft – biplane and helicopter
- As the biplane moves towards the helicopter we want to make sure they do not collide
- If they are too close, they need to adjust their altitude (height)

Storyboard (cont)

- Two factors in determining whether two aircraft are in danger of collision
  - distance between them
  - Vertical distance between them
- We can write functions to determine these
- Both functions return true if aircraft are too close, otherwise false

isTooCloseByDistance

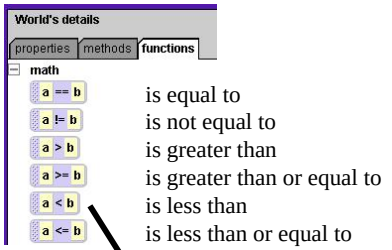
isTooCloseByDistance:

Parameters: aircraft1, aircraft2, minDistance

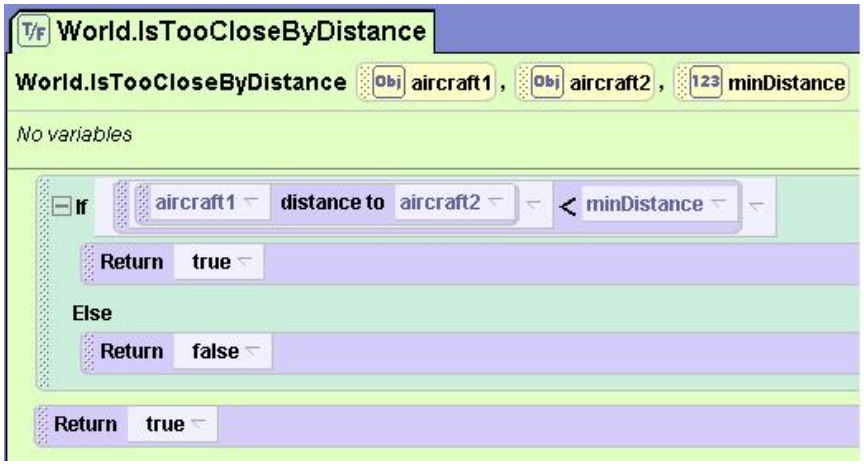
If distance between aircraft1 and aircraft2 is less than minDistance
    return true
Else
    return false

Using a Relational Operator

- Use the < relational operator from the World's built-in functions to check the distance against the minimum



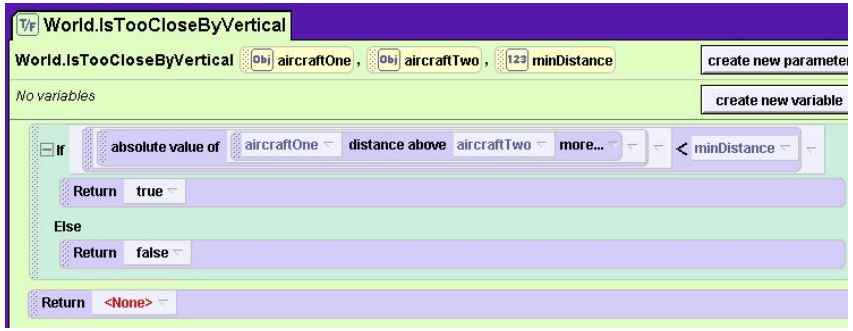
Implementing the Function



Vertical Distance Function

- To find the difference in altitude, use the built-in *distance above* function
  - Don't know which aircraft is above the other
  - To avoid a possible negative value, use *absolute value* of the distance

istooCloseByVertical



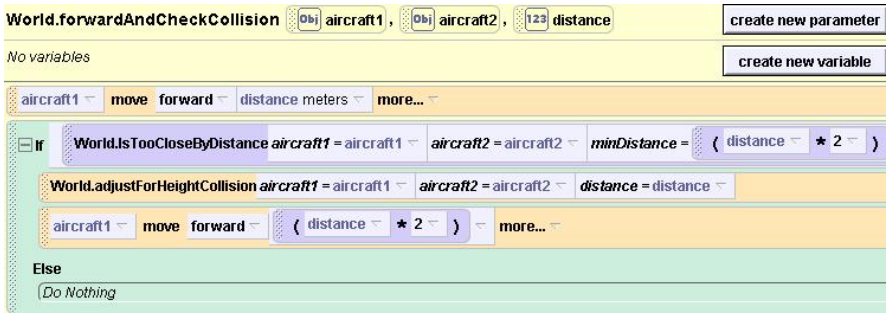
Storyboard

forwardAndCheckCollision

Parameters: *aircraft1*, *aircraft2*, *distance*

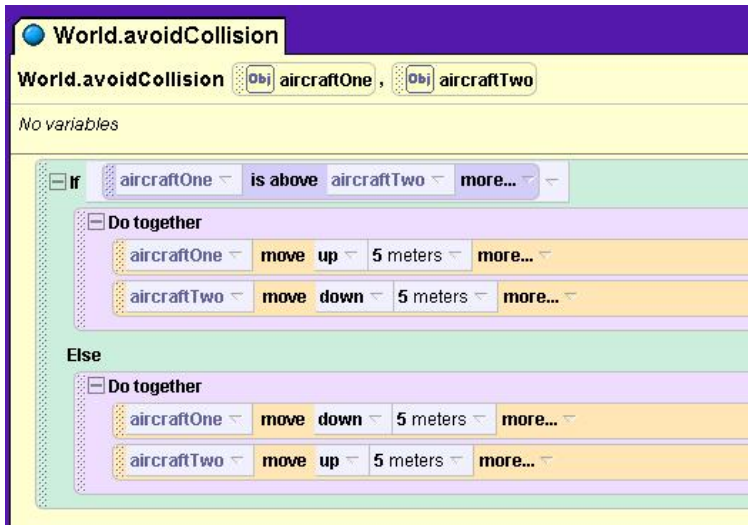
aircraft1 move forward *distance*
If *aircraft1* and *aircraft2* are closer than twice *distance*
avoid collision if they are too close heightwise
move *aircraft1* forward twice the *distance*

Implementation and Calling Function

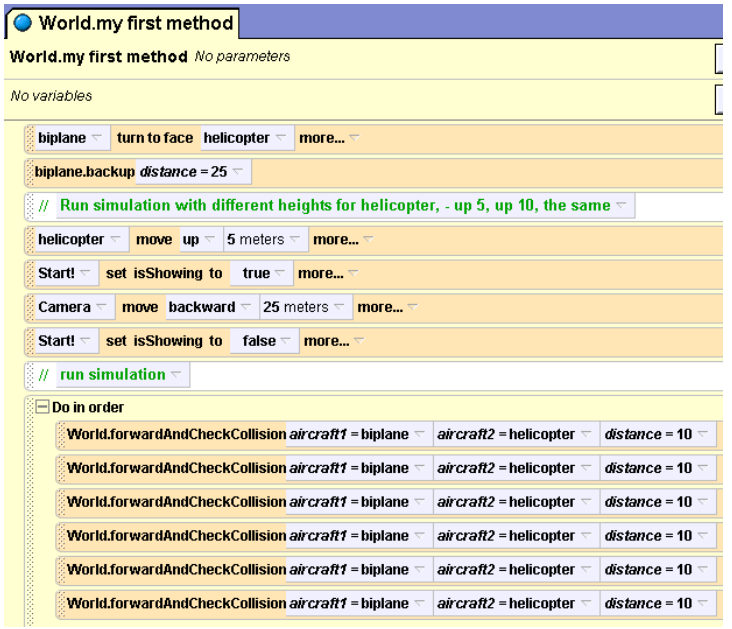


adjustForHeightCollision

Avoid Collision



Putting it All Together - Demo

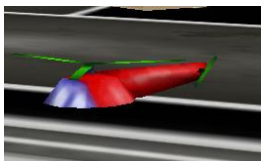


Demo and Testing

- Try helicopter at different heights
  - Move up 5 meters
  - Move up 10 meters
  - Stay the same
  - Down 5 meters

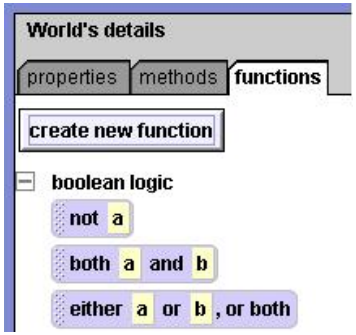
Problem

- The helicopter may go below the ground!



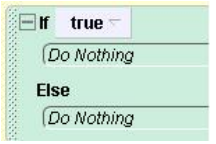
- How do we fix this?
  - Only move down if above a certain distance!
  - Use nested if's to check more than one condition

Another Way - Logical Operators



- Use Boolean logic operators to check more than one condition

Check



- Where do you get the if?
- Do you have to fill all the parts of the if?
- Where do you find the relational operators?
- Where do you find the logical operators?

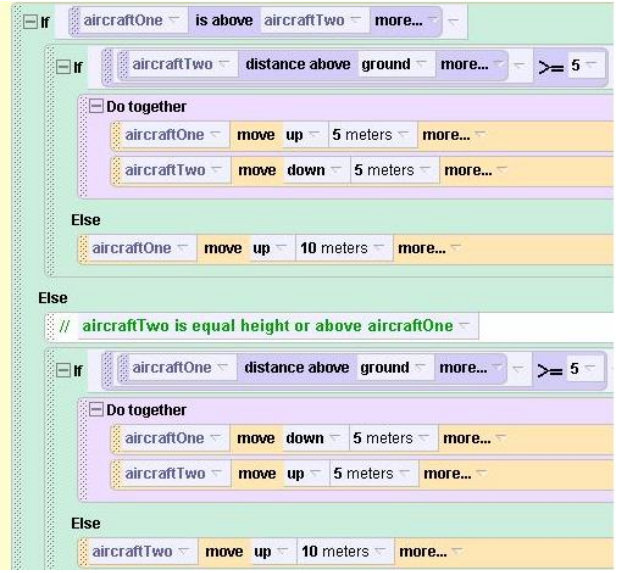
Random Numbers

- We will cover this later in more detail

Classwork today

- Write functions and methods with if/else

avoidCollisionGroundCheck1



avoidCollisionGroundCheck2