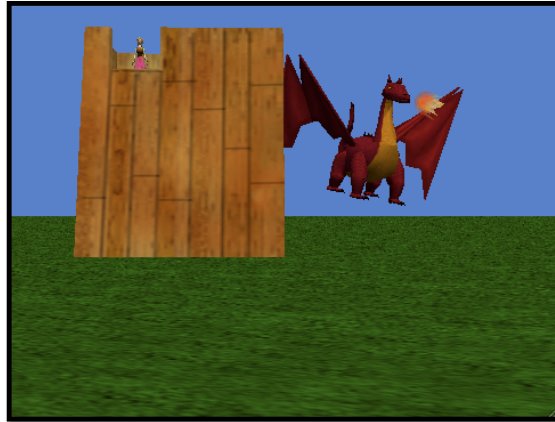


Princess & Dragon

Part 2: Teaching a Dragon to Fly—Methods & Properties



By Elizabeth Liang
under the direction of
Professor Susan Rodger
Duke University
June 2010

Introduction

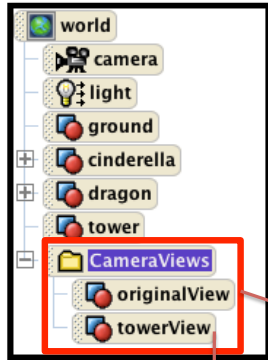
Welcome to Part 2 of the Princess & Dragon tutorial. In Part 1 we covered how to set up a world, add and position objects, and create a simple animation.

Part 1: Objects
Part 2: Methods & Properties
Part 3: Cameras & Events
Part 4: Billboards, Sound, & 3D-Text

In Part 2 we'll add more animations so that the dragon will kidnap the princess.

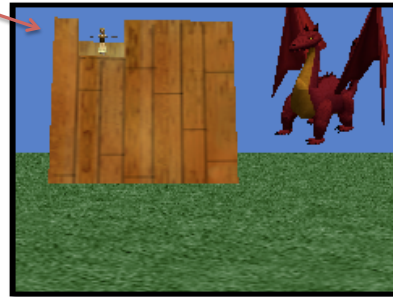
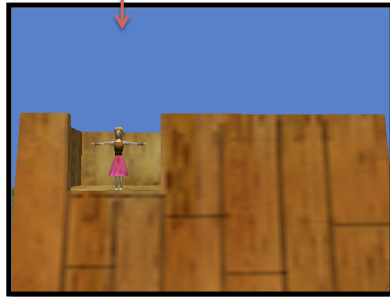
This will cover how to change camera views, create and edit methods, and change properties.

Step 1: Dummy Object Review



In Part 1 you learned how to drop a dummy object to save the view of the Camera. We dropped one named *originalView* (the location of the camera when Alice starts) and another dummy object called *towerView* (when we moved the camera to get a close up view of the tower).

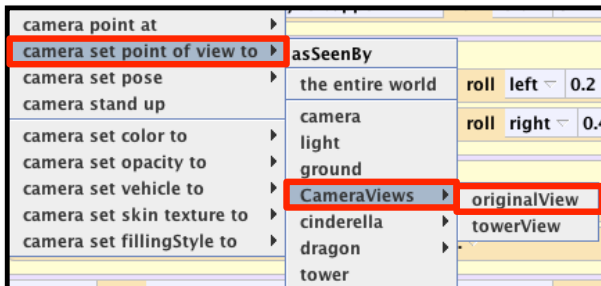
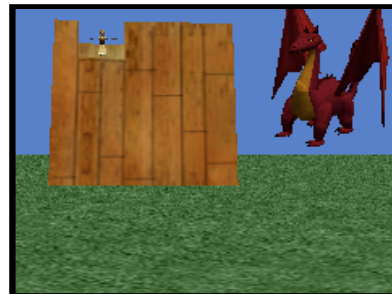
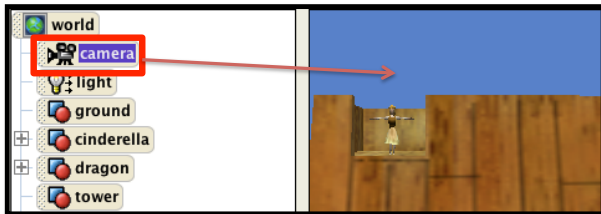
You can find these in the folder that we named *CameraViews*. We will now use these saved Camera views.



Step 1: Setting Camera View

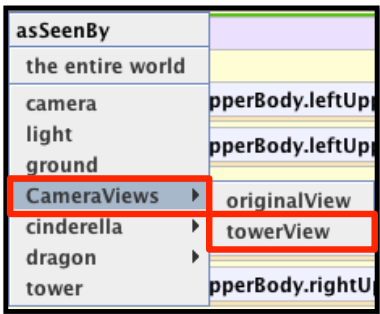
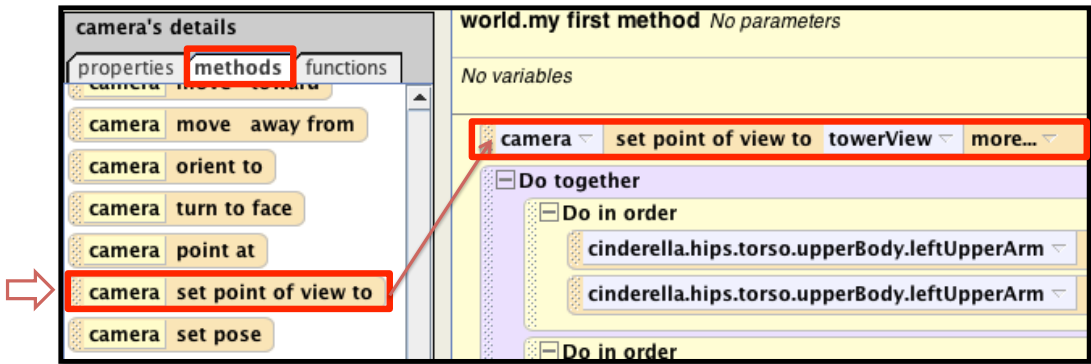
There are two ways to change the camera view. This first way changes the camera view before the animation.

Select the **camera** from the object tree and drag it into the world preview pane



You will see a menu pop up, go to **camera set point of view to** and select **CameraViews, originalView**. This will set the scene back to the original camera view.

Step 1: Changing Camera View

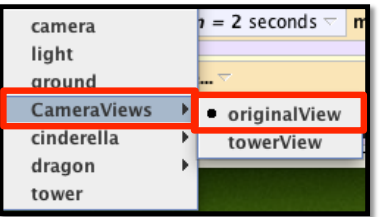
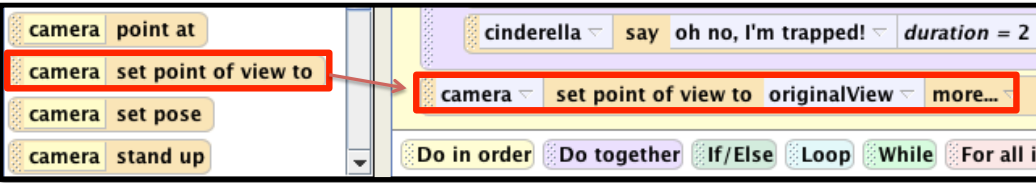


In order to change the camera view during animation, we must drag in a method.

Find camera set point of view to from the camera's list of methods (you will need to scroll down). Drag this in right above but outside of the Do together.

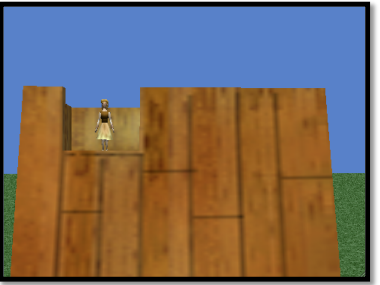
Select CameraViews, towerView.

Step 1: Changing Camera View



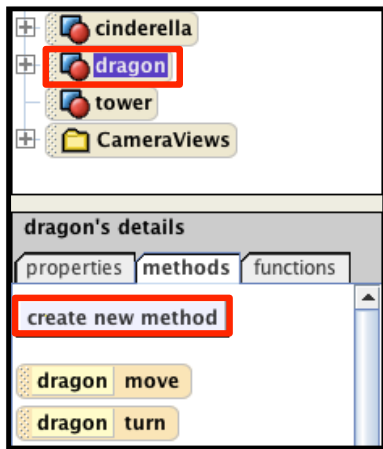
Drag in the same instruction at the end of the method, outside of the Do together.

This time set it to the originalView.



Play the world and you will see that the first instruction we added makes the camera zoom in on the tower and that the second instruction returns the camera to the first view.

Step 2: DragonFly Object Method

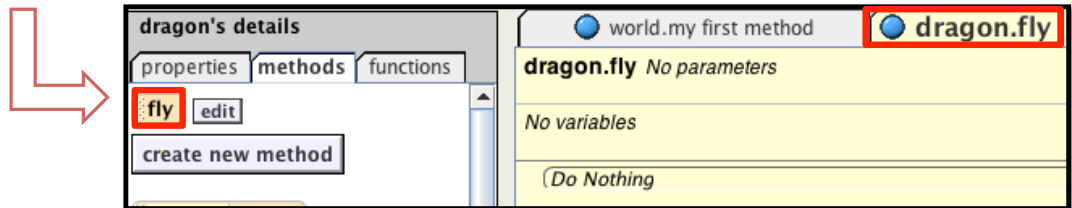


We create new methods that put together the simple methods Alice gives us for each object in order to create short animations that we can use over and over. We'll use this to teach the dragon to fly.

Click on **dragon** in the object tree and in the methods pane, click **create new method**. Name it **fly** and select **OK**.



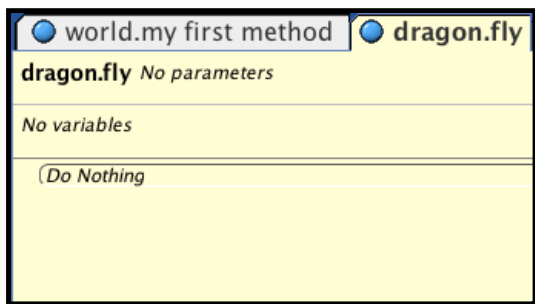
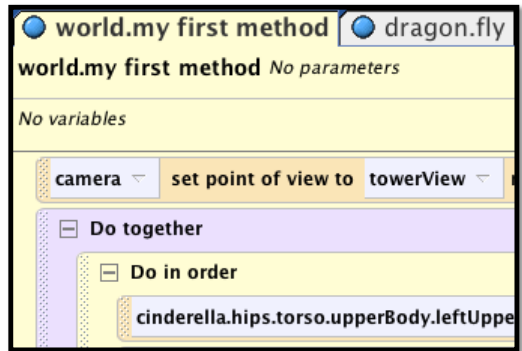
You will see a new tab pop up in the method editor labeled **dragon.fly** and also a new method called **fly** in the **dragon's methods**.



Step 2: Tabs

There are now two tabs in the method editor. Each tab represents a different method and the code for that method.

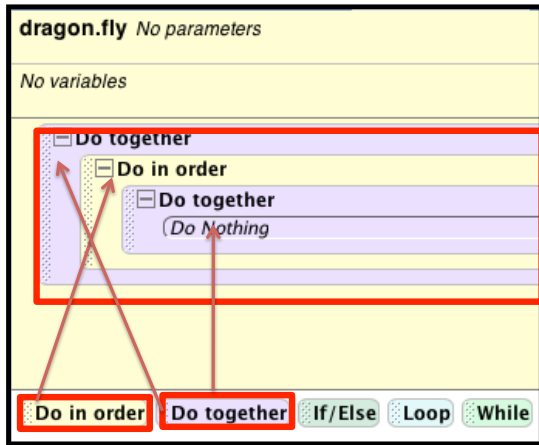
Click on **world.my first method** and you will see the code that we wrote before.



Click on the **dragon.fly** tab and you will see there is no code. The code that we will put in here will teach the dragon to fly.

CAUTION: When you have multiple tabs, always make sure the correct tab is up before dragging and dropping in code.

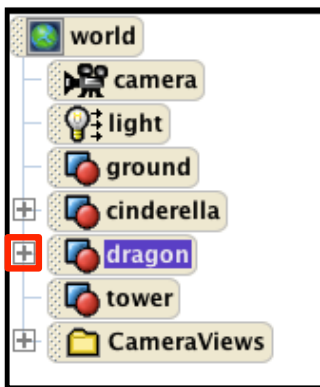
Step 2: DragonFly Object Method



There are two types of methods you can create, object level methods and world level methods. We have created an object level method because the method is inside the object dragon. In an object level method, all the animations must stay within the object and not use any other objects in the world.

First drag in a **Do together** then a **Do in order** inside and another **Do together**. These blocks set up the method that we will fill in.

Step 2: DragonFly Object Method



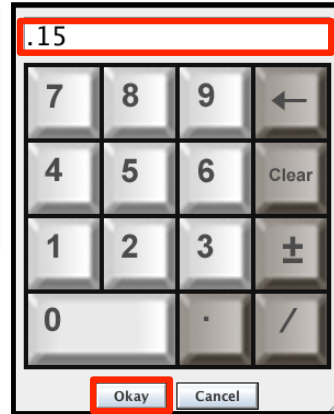
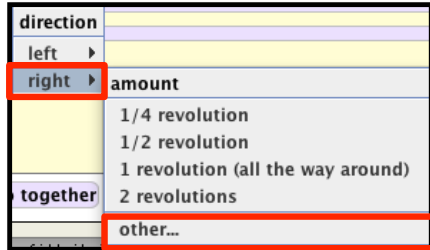
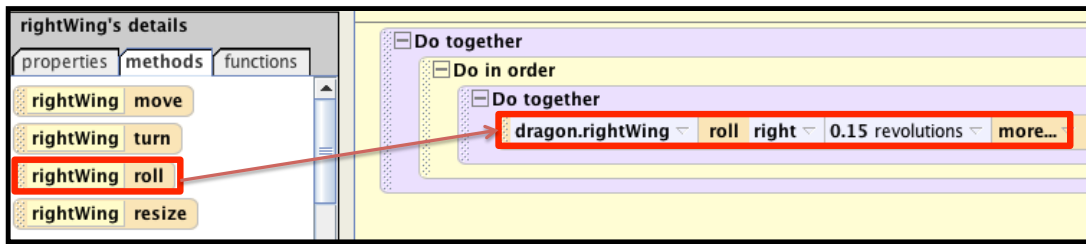
Since this is an object method in the object dragon, we will only use objects that are part of the dragon. The dragon has a lot of parts but we will focus on his wings which are further divided into flaps.

Click on the + next to the **dragon** in the object tree and scroll down until you find the **rightWing** which we will animate.



Note: Your dragon may have a **rightWingClose** instead of **rightWing** depending on your version of the Alice object. These are the same but with different names. Use **rightWingClose** instead of **rightWing** and instead of **flap** use **rightWingFar**.

Step 2: Animation

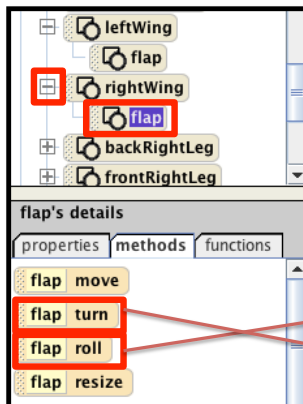


Drag in a **roll** command from the **rightWing**'s list of **methods** into the second **Do together**.

Select **right**, **other**, and punch in **.15**.

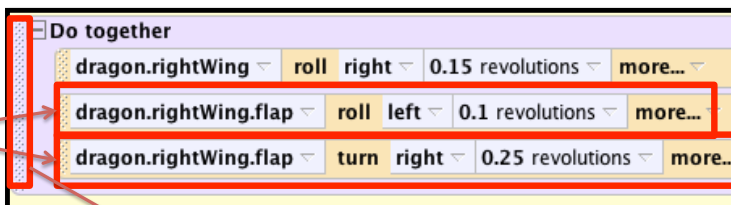
Step 2: Animation Continued

Click on the **+** next to the **dragon's rightWing** in the object tree and then click on **flap**.



Drag in the **flap roll** method into the Do together, select **left**, **other**. Punch in **0.1**.

Drag in the **flap turn** method into the Do together, select **right** and then **1/4 revolution**.

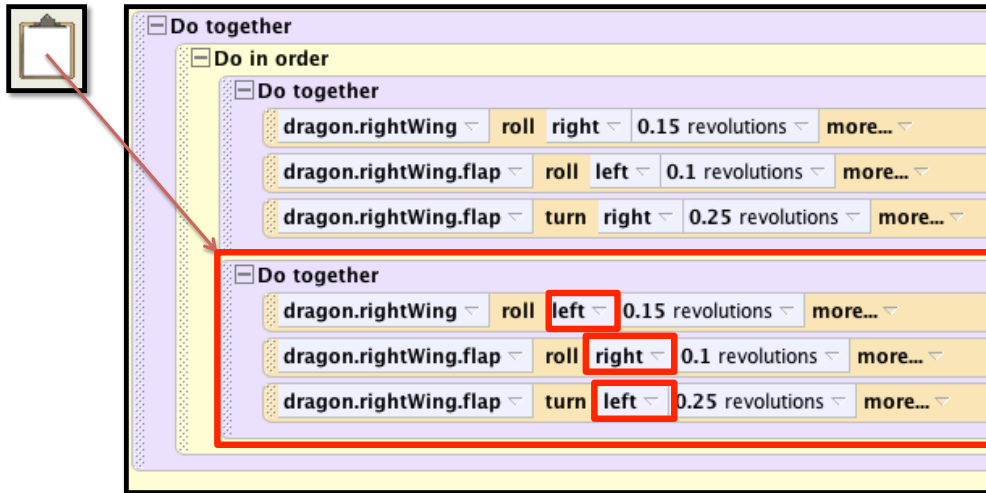


Most of the time when you are creating your own animation, you will need to play around with the numbers and methods until you get it just right. In this tutorial we give you the exact amounts but it took a lot of trial and error to find the right number!



We need to make a copy and do the reverse so drag the **Do together** block onto the **clipboard**.

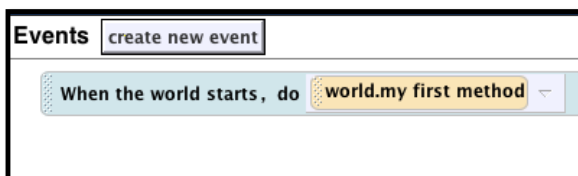
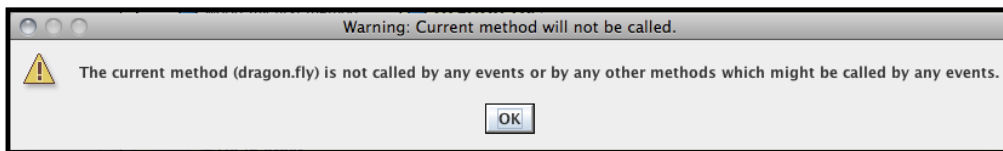
Step 2: Animation Continued



Drag the block of code from the clipboard underneath the **Do together** inside the **Do in order** block. Reverse the directions from **right** to **left** and **left** to **right** on each line.

Step 2: Testing DragonFly Method

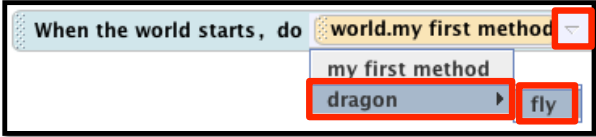
Press **Play**. When you try to test the method we've been writing a warning box will pop up.



*In Alice there has to be a starting point for running your code. In the event editor, you can see **When the world starts, do world.my first method**.*

*The default is to run the code in the first method you wrote, **my first method**. For testing purposes, we can change the method that is run. In this case we would like to test out the **dragon.fly** method to make sure it works before we integrate it into the program*

Step 2: Testing DragonFly Method



In the **event editor**, change the method that plays **When the world starts to dragon fly**.



Now press **play** and you will see the dragon's right wing flap once. Let's repeat for the left wing!

Step 2: Animation Continued

Copy the entire **Do in order** block onto the clip board

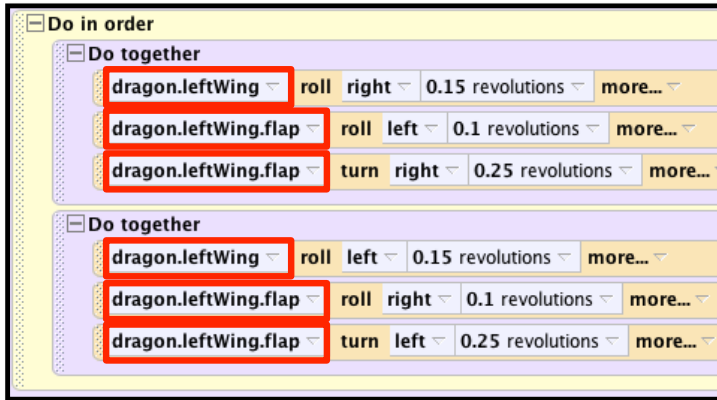
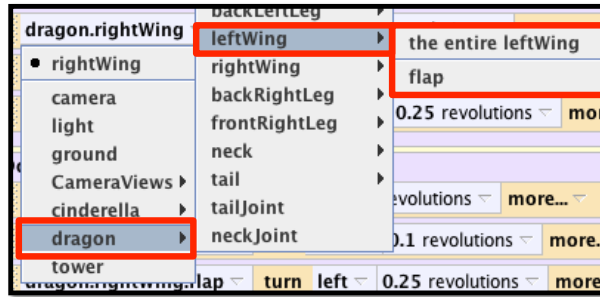


Drag from the clipboard into the big **Do together**, underneath the previous **Do in order**.

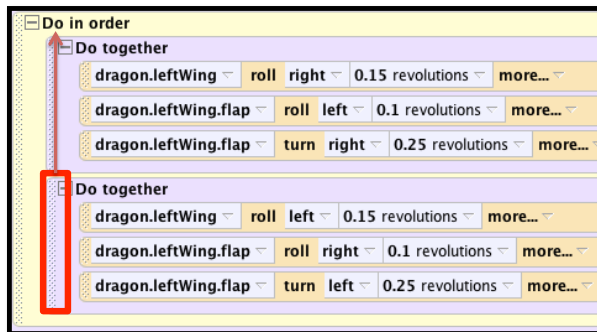


Step 2: Animation Continued

Change all the **rightWing** references to **leftWing** by clicking each of the lines next to **dragon.rightWing**. Select **dragon**, **leftWing**, the entire **leftWing**. Do the same for the **flap** references.

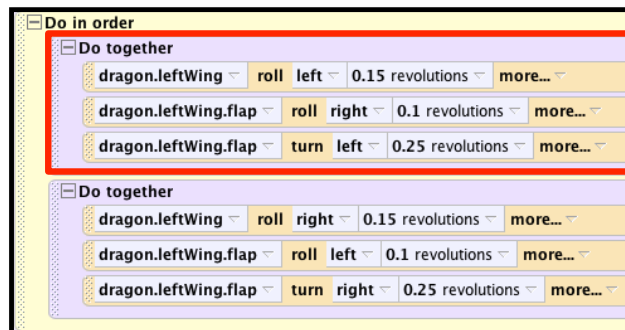


Step 2: Animation Continued



Flip the order of the **two Do together** blocks by dragging the second one above the first one.

This is what it should look like when you're done. Go to the next slide to see the final code for the method.



Step 2: Animation Continued

The screenshot shows a timeline with a 'Do together' container. Inside, there are three 'Do in order' blocks. Each 'Do in order' block contains three 'Do together' sub-blocks. The first sub-block in each 'Do in order' block contains: 'dragon.rightWing' roll right 0.15 revolutions, 'dragon.rightWing.flap' roll left 0.1 revolutions, and 'dragon.rightWing.flap' turn right 0.25 revolutions. The second sub-block contains: 'dragon.rightWing' roll left 0.15 revolutions, 'dragon.rightWing.flap' roll right 0.1 revolutions, and 'dragon.rightWing.flap' turn left 0.25 revolutions. The third sub-block contains: 'dragon.leftWing' roll left 0.15 revolutions, 'dragon.leftWing.flap' roll right 0.1 revolutions, and 'dragon.leftWing.flap' turn left 0.25 revolutions. The fourth sub-block contains: 'dragon.leftWing' roll right 0.15 revolutions, 'dragon.leftWing.flap' roll left 0.1 revolutions, and 'dragon.leftWing.flap' turn right 0.25 revolutions.



This is the final code for `dragon.fly`. Press `play` to see the dragon flap his wings!

Step 3: CapturePrincess World Method

The screenshot shows the 'world' object selected in the object tree. The details panel shows the 'methods' tab with 'my first method' and 'create new method' (highlighted with a red box).

Now we need to create a method that uses `dragon.fly`.

To create a world level method, click on the `world` in the `object tree` and in the `methods` pane, click on `create new method`. Name it `capturePrincess` since that is what it will do. You will see a new tab pop up in the method editor and a new method in the world's methods list.

A dialog box titled 'New Method' with a text field containing 'capturePrincess' (highlighted with a red box) and 'OK' and 'Cancel' buttons.

The screenshot shows the 'world's details' panel with the 'methods' tab. The 'capturePrincess' method is now listed and highlighted with a red box.

The screenshot shows the method editor with a new tab for 'world.capturePrincess'. The method name is 'world.capturePrincess' (highlighted with a red box) and it has 'No parameters'. The body is empty, showing '(Do Nothing)'.

Step 3: World vs. Object Methods

The left screenshot shows a method named 'dragon.fly' with no parameters and no variables. It contains a 'Do together' block with three nested 'Do in order' blocks. The first 'Do in order' block contains three actions: 'dragon.rightWing roll right', 'dragon.rightWing.flap roll left', and 'dragon.rightWing.flap turn right'. The second 'Do in order' block contains three actions: 'dragon.rightWing roll left 0.5', 'dragon.rightWing.flap roll right', and 'dragon.rightWing.flap turn left'. The right screenshot shows a method named 'world.my first method' with no parameters and no variables. It starts with a 'camera' object, followed by 'set point of view to towerView', and a 'Do together' block. Inside the 'Do together' block, there are two 'Do in order' blocks. The first 'Do in order' block contains two actions: 'cinderella.hips.torso.upperBody.leftUp' and 'cinderella.hips.torso.upperBody.leftUp'. The second 'Do in order' block contains two actions: 'cinderella.hips.torso.upperBody.rightUp' and 'cinderella.hips.torso.upperBody.rightUp'.

The `dragon.fly` method is an object method because all of the code refers to the `dragon` and no other object. My first method is a world method because multiple objects are referred to: the `camera`, the `towerView` object, and `Cinderella`.

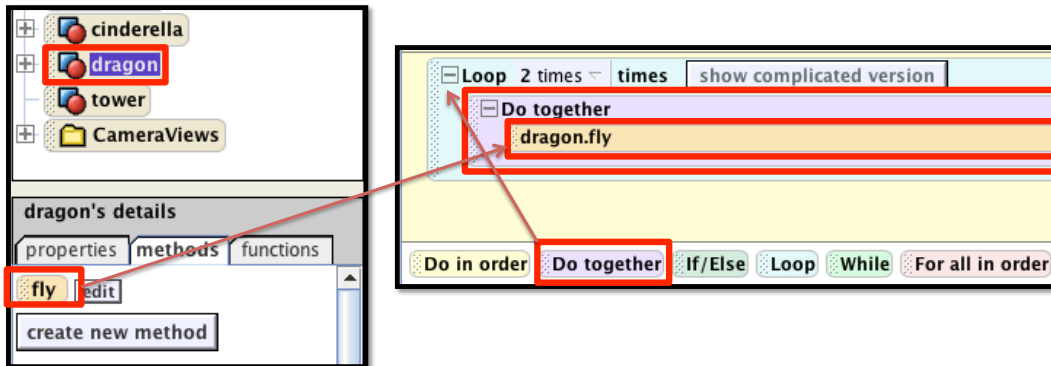
Step 3: Loop

The screenshot shows a method named 'world.capturePrincess' with no parameters and no variables. A 'Loop 2 times' block is added, containing a 'Do Nothing' block. The 'Loop' button in the bottom toolbar is highlighted. A dropdown menu is open, showing options: end, 1 time, 2 times, 5 times, 10 times, infinity times, other... The '2 times' option is selected.

The first thing we want to do is make the dragon fly around the tower twice while flapping his wings. Because we do not want to repeat the code we will use a loop. A loop repeats the code inside the block the specified number of times.

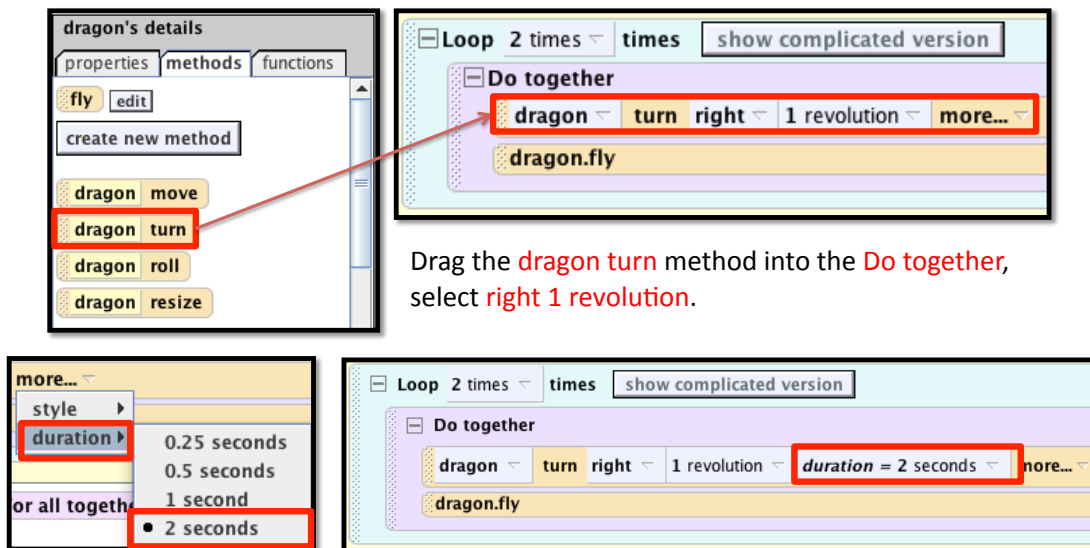
Drag a loop into the `world.capturePrincess` method. Select **2 times**.

Step 3: Loop Continued



Into the loop, drag in a **Do together**. Then click on **dragon** in the **object tree** and under **methods**, drag in the **fly** method we just finished writing. Put it inside the **Do together**.

Step 3: Turn



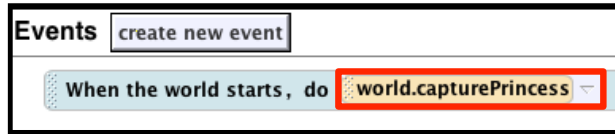
Drag the **dragon turn** method into the **Do together**, select **right 1 revolution**.

Click on **more...** and select **duration 2 seconds**.

This is to make sure the dragon finishes one turn after he flaps his wings once.

Step 3: Testing CapturePrincess

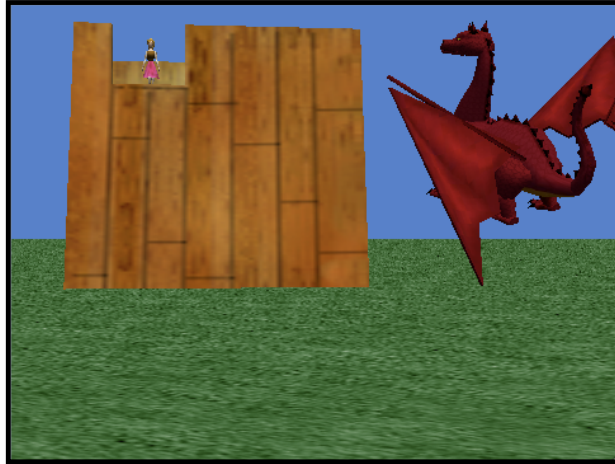
In the **Events editor**, change the **When the world starts** event from **dragon.fly** to **capturePrincess**.



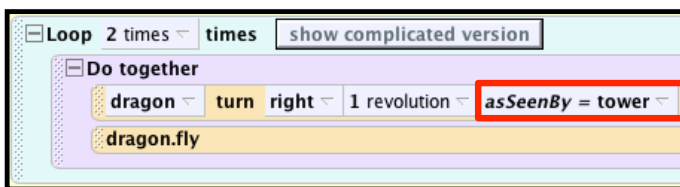
Press **Play** to test **capturePrincess**.

*It looks like the dragon is just turning in place here, but we want to dragon to go around the tower. That can be done using **asSeenBy**.*

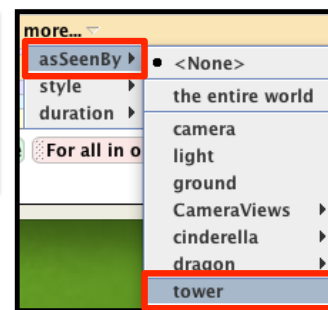
***AsSeenBy** can be used to make an object go around another object.*



Step 3: As Seen By

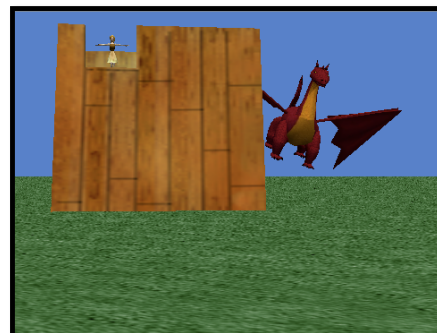


In order to make him go around the tower we will use **asSeenBy**.
Click on **more** and select **asSeenBy tower**.

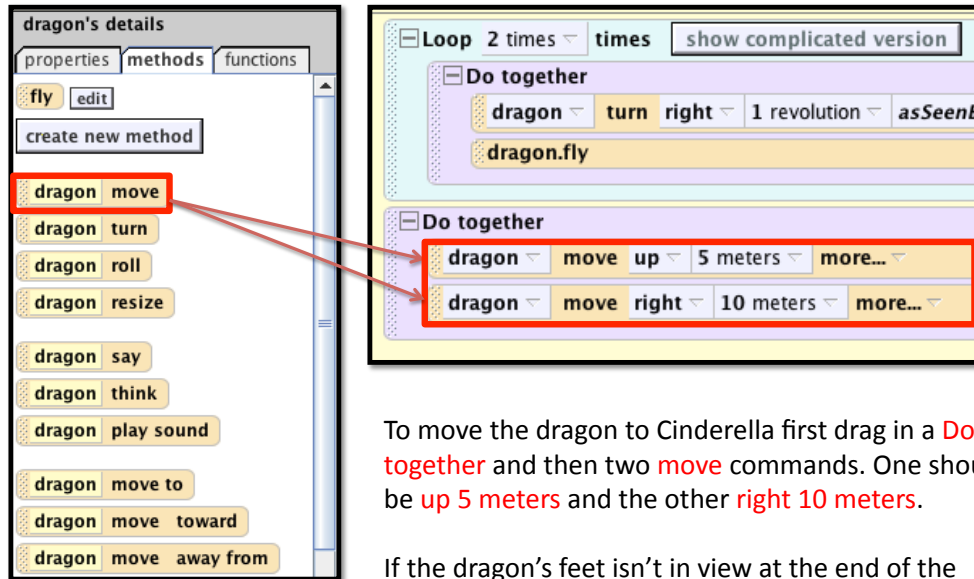


Press **Play** to test **capturePrincess** once more.

You will see that the dragon goes around the tower!



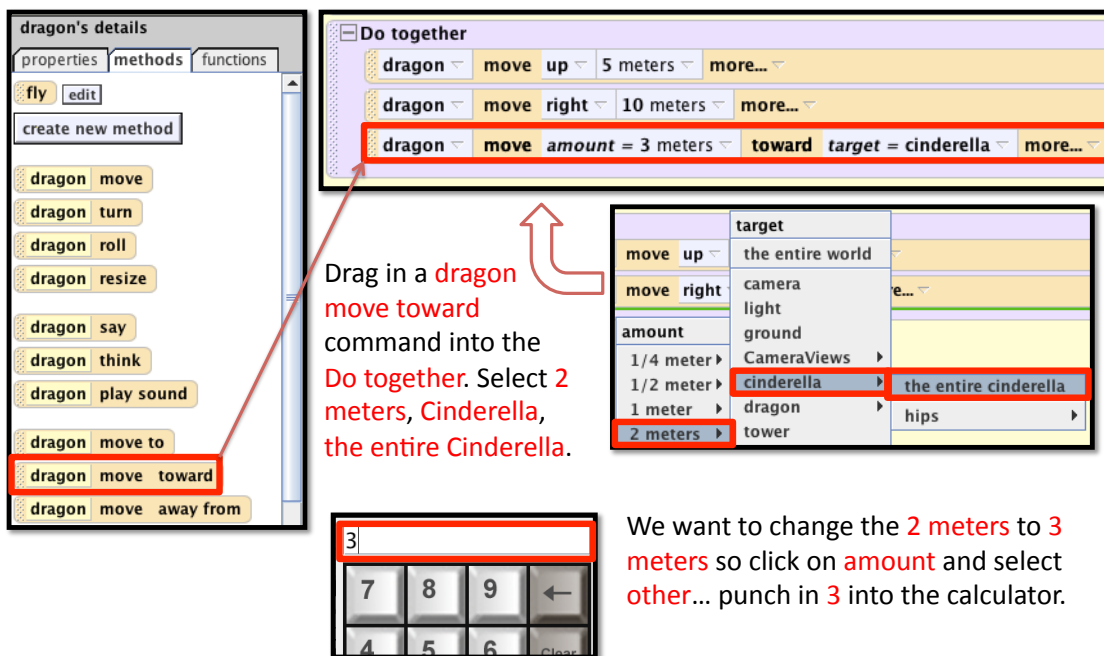
Step 3: Animation



To move the dragon to Cinderella first drag in a **Do together** and then two **move** commands. One should be **up 5 meters** and the other **right 10 meters**.

If the dragon's feet isn't in view at the end of the animation when you test this, decrease the amount he moves up from **5 meters** to **4 meters**.

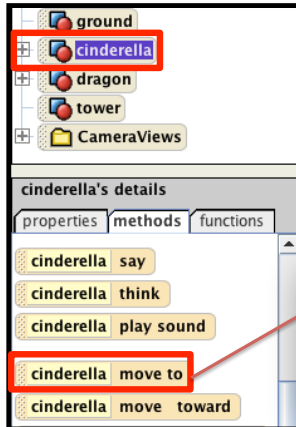
Step 3: Animation Continued



Drag in a **dragon move toward** command into the **Do together**. Select **2 meters**, **Cinderella**, **the entire Cinderella**.

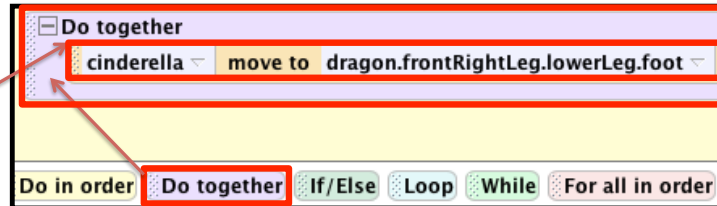
We want to change the **2 meters** to **3 meters** so click on **amount** and select **other...** punch in **3** into the calculator.

Step 3: Animation Continued

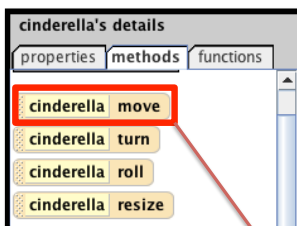


The next step is to have Cinderella "picked up" by the dragon.

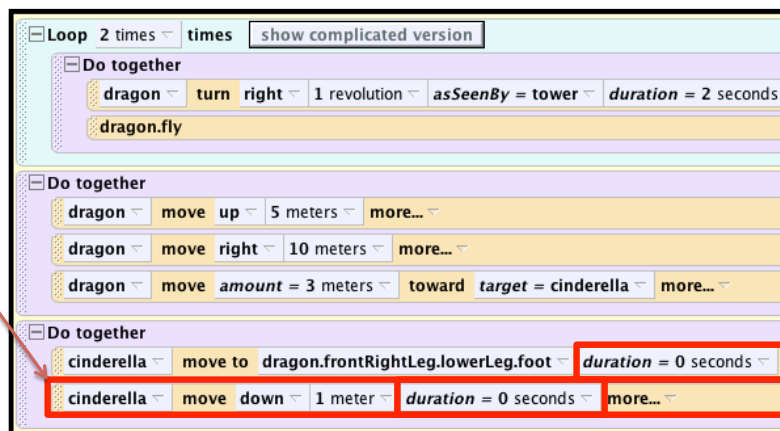
Drag in a **Do together** and then click on **Cinderella** in the object tree. Drag in a **move to** command and select the dragon's **frontRightLeg**, **lowerLeg**, **foot**, the entire **foot**.



Step 3: Animation Continued



Drag in a **Cinderella move** command into the **Do together**. Have her move **down 1 meter**.

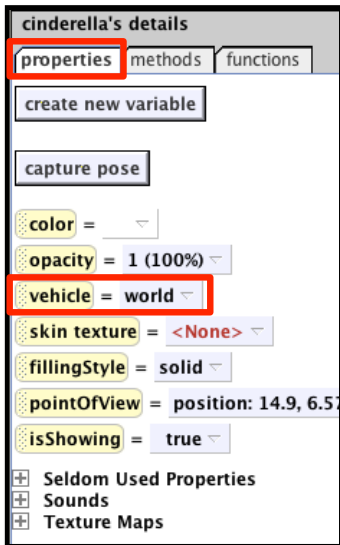


We want this to happen immediately so set the **duration** of both commands inside the **Do together** to **0 seconds** (click on **more** and enter in **0** into the calculator).

Watch the animation and you will see that Cinderella is instantaneously picked up by the dragon.



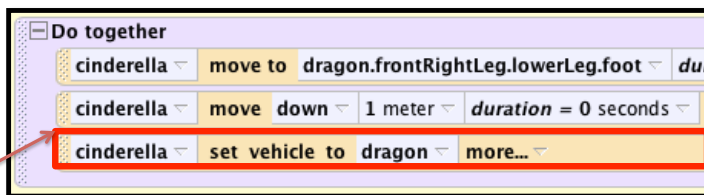
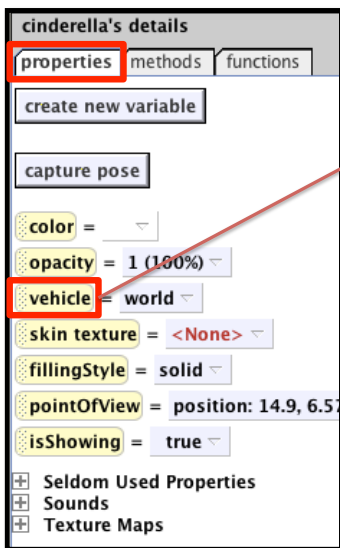
Step 3: Vehicle Property



In order for the dragon to fly off with Cinderella we need to 'glue' Cinderella to the leg of the dragon. To do this we will use the vehicle property. Properties are information about an object. You can change them in a method just like you can animate the parts but they reflect the current state of the object.

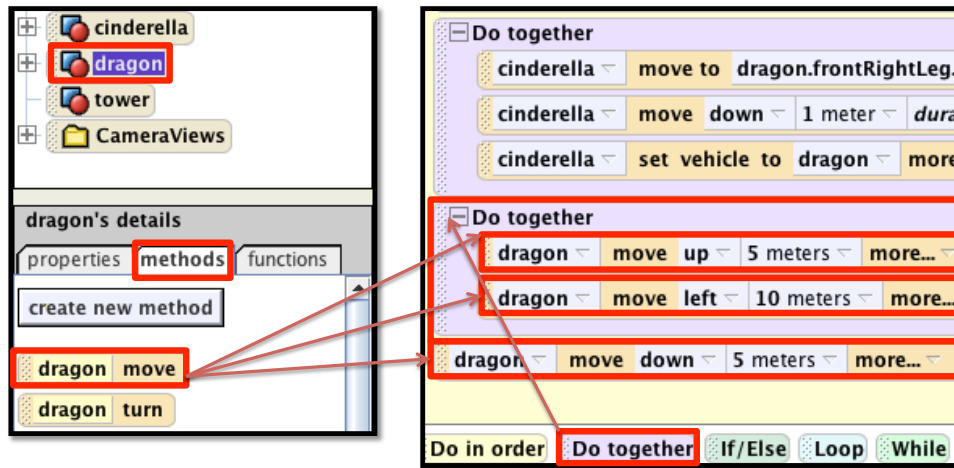
The vehicle property is set to world by default. When you change the vehicle property it means that whenever the vehicle moves, the object also moves with it. So when the dragon moves, Cinderella will move with it.

Step 3: Gluing the Princess



To change the vehicle of an object during an animation, click on the **properties** tab and drag **vehicle** into the code. Put it inside the **Do together** and select **dragon**, the entire dragon.

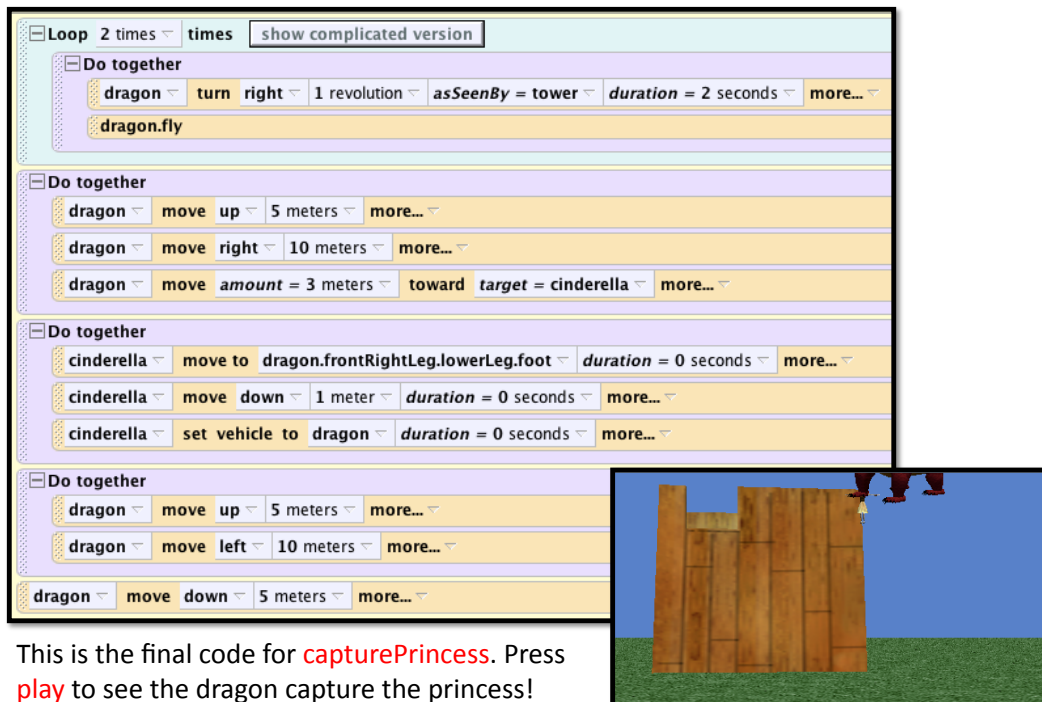
Step 3: Animation Continued



To finish up the code drag in another **Do together**. We will move the dragon away from the tower. Drag in two **dragon move** commands, one for moving **up 5 meters** and another for **left 10 meters**. Put these inside the **Do together**.

Outside of the **Do together**, drag in another **move** command for **down 5 meters**. The method is now finished! See the next slide for a copy of the final code.

Step 3: Animation Continued



This is the final code for **capturePrincess**. Press **play** to see the dragon capture the princess!

Step 4: Calling Methods

Now let's put all the code that we've written together.



Click on the **world. my first method** tab in the **method editor**

Click on **world** in the object tree and find the **capturePrincess** method under the **methods** tab.

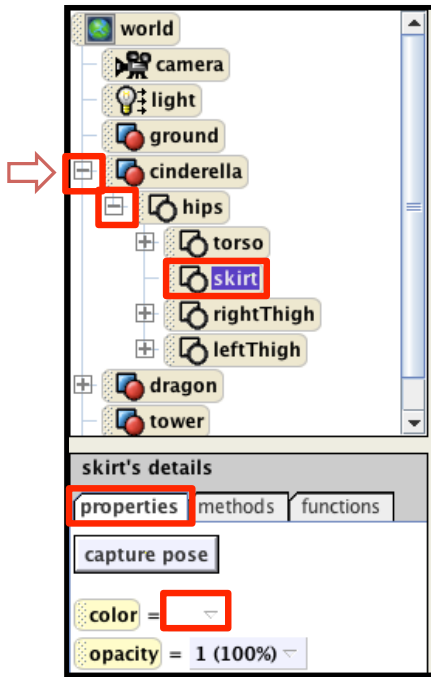
Scroll down to the end of **my first method** and drag the **capture Princess** method into the very end of the method.

Step 4: Comments

Comments are notes for people and are not code, so when your code is executed, comments are ignored.

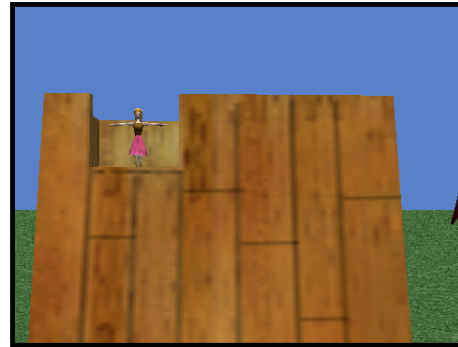
Drag in two **comment** lines into the very top of **world.my first method**. Double click on them and enter in your **name** and today's **date**.

Step 4: Color Property

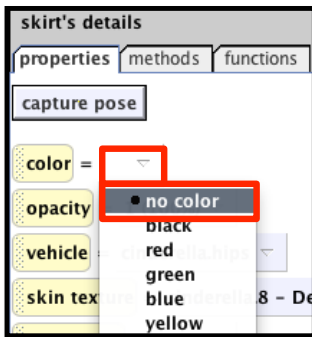


You can also change the properties of subparts of an object. Properties can be changed either before animation to set up a world or during animation. One property that can be fun to change is the color property. Not every object can be colored but do try it out!

Click on **Cinderella's skirt** in the **object tree** and click on the box next to the **color property**. Select **magenta** and watch her dress change color!



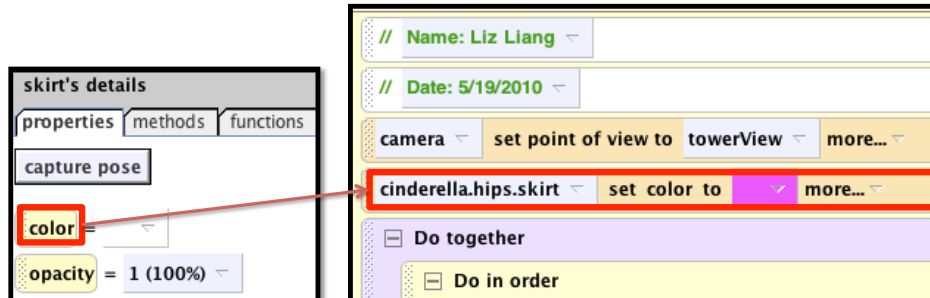
Step 4: Color Property Continued



To change the color of an object back, click on the box of color and select **no color**. This will return the original color back to the object.

Another way to change color is during an animation. In order to change properties during in animation we need to turn it into a line of code.

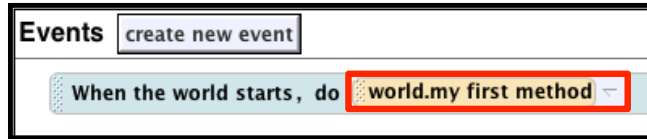
Click on **color** and drag it into **my first method**. Release it right before the **Do together**. Select **Magenta**.



Congratulations!



To play the whole animation, change the method called by the **When the world starts** event back to **world.my first method**



Play the world to see the skirt change color and everything come together!

Congratulations on finishing Part 2! Part 3 will teach you more about events and different uses for the camera.

