Overview

- *The story continues:* After the conversation, we want skaterGirl to get on the skateboard, make the jump, and then skate around one of the cones

- What we will do:
  3. Properties
     - Changing properties during set up
     - Changing during animations
  4. Create new methods
     - Animating objects
     - Using As Seen By
     - Animating parts of objects
  5. Using As Seen By
     - To move around an object

Methods

- A *method* is a sequence of instructions that will be carried out when requested. Built in methods are used to create new methods so that the characters can learn to do more.

- The two types of methods are class-level and world-level. A class-level (or object-level) method defines the behavior for a single object. A world-level method has objects that interact with each other.

To Create your method

- Since our methods involve several objects, we will create world-level methods.

- Click on *world* in the object tree. Click on the *methods* tab in the *details* areas.

- Click *create new method*. Name it “*makeJump*”.

- Click *OK*. A new tab appears in the method editor
Introducing Another Tool

- **The Events Pane**
  - This is where you control when certain methods are called and the user interactions within your animation.

![Events pane](image)

When the world starts . . .

- As we write this new method, we don’t want to watch the entire conversation we wrote from part one every time we play the world.
- In the *Events pane*, click on the arrow and select *makeJump* from the drop down menu.

![Events pane](image)

Writing *makeJump* method

- In our story, we want the girl to go to the skateboard
- Click on the *makeJump* tab in the method editor. On top of the *Do Nothing* drag in *skaterGirl move to*, select *skateboard*, the entire skateboard
- Play your world.
  - See the next slide for screenshot of what went wrong

![SkaterGirl using skateboard](image)

- The girls moves into the skateboard because the *move to* method uses the *center* of each object.
- We would have to manually measure how far the *skaterGirl needs to move to the skateboard*
• To realistically make the skaterGirl move to the skateboard, we will put an invisible object on top of the skateboard and have skaterGirl move to that object.
• Click on Add objects. Go to the Shapes folder. Drag a box into your world.

Positioning the Box
• First, we want the skateboard and box oriented correctly.
• Right-click on the skateboard in the object tree. Select methods, turn to face, jump.
• Right-click on the box, select methods orient to, skateboard, the entire skateboard

Using Quad View
• Now, click on quad view. Use the move arrow to position the box on top of the skateboard. Remember to hold down shift as you drag in order to move up or down.

Step Three: Properties
Attaching objects together
• Now that we’ve positioned our box, we want it to stay next to the skateboard, wherever the skateboard moves.
  – In the properties tab, set the box’s vehicle property to skateboard, the entire skateboard
  – The vehicle property attaches an object to another object. They always move together
**Change makeJump**

- Click on the makeJump method in the move to instruction, click on the arrow beside the skateboard. Change it to box because we want skaterGirl to move to the box, not the skateboard. Play your world now.

**Understanding vehicle property**

- Click Done to exit the gallery.
- Drag Do together into the makeJump method.
- Drag the following into the Do together:
  - box = move forward 5 meters
  - skateboard = move forward 5 meters
  - jump
  - came
  - came2
  - came3
  - airport

- Play your world.
- See the next slide to see what went wrong.

**Properties continued: Making an object invisible**

- You can see the box move past the skateboard. We only need to move the object that is the vehicle, in this case the skateboard.
- Right click on the box instruction and select disable.
- Play your world to see the box move with the skateboard since the skateboard is the box's vehicle.
- Now that you see how vehicle property works, delete everything in this Do together by dragging it up to the trashcan.
- Now that the box is positioned and attached, we need to make it invisible.
- Click on the box in the object tree.
- Click on the properties tab in the details area.
- Set isShowing to false.
Finally, we can change the color of objects.

Click on the **cone** in the object tree. In the **properties** tab, change the **color** to orange. Do this to every cone.

— You can even change the color of default Alice objects. For example, click on **ground** in the object tree and change it's color to **green**.

**Continue makeJump method**

Now, when the skateboard turns to face the jump object, skaterGirl’s head will turn right.

Since we want this to happen at the same time, first, drag in the control statement, **Do together**

Drag in **skateboard turn to face**, select **jump**.

**skaterGirl Vehicle**

While skaterGirl is on the board, we want her to move with it, so we must change her vehicle property to skateboard.

Click on skaterGirl in the object tree. Click on the **properties** tab.

Since we want her vehicle property to be changed during the animation, drag her **vehicle** property into the **makeJump** method. Set it to **skateboard, the entire skateboard**.

**Animating Parts Of Objects**

To animate part of an object, expand the object in the object tree by the +

Expand **skaterGirl, upperBody, neck, head**

Drag **head** into the **Do together**. Select: **turn ,right, ¼ revolution**. Set duration to 0.5

Result:
• Next, (underneath the do together), drag in skateboard from the object tree. Select move toward, 1 meter, jump.
• Change the 1 meter so that the girl moves close to the board when you play your world.
  • I changed the amount to 7 meters.
  • Then, click on more in the instruction and change the style to abruptly.

Finish writing makeJump
• We change the style to abruptly so that there will not be a pause between each instruction.
• Drag in the rest of the instructions under the Do together. The complete method is on the next two slides.
  ➢ Notice the change in style and duration of each instruction. Play your world when you finish.

The makeJump method:
• First drag in all of the instructions.
• Then change the duration and style appropriately.
• See the next slide for the rest of the instructions in makeJump.

makeJump Continued
• First drag in all of the instructions.
• Then change the duration and style appropriately.
• Play your world.
Call MakeJump

• Now we need to call our new method. Click on world.my first method.
• Click on world.myfirstMethod tab
• Drag in makeJump
• Change the event “When the world starts” back to my first method
• Play your world

Writing SkateCircle

• Click on world in the object tree. Click create new method, name it skateCircle, click OK
• Drag in the following code. *except turn to face whichever cone is closest to the “jump” object

Step Five: As Seen By

• To make the skateboard turn around the cone, for the final instruction in skateCircle: Click on more, select asSeenBy, cone2.

Call skateCircle

• Drag skateCircle underneath the instructions in world.my first method
• Play your world
• You may have to change the move forward amount in skateCircle
Write skaterGuy.celebrate
• Click on skaterGuy. Create a new method, name it celebrate. Write your own short method to have him celebrate the jump.

Call skaterGuy.celebrate
• Click on the world.myfirstMethod tab.
• Drag skaterGuy.celebrate into the method above world.skateCircle

The End of Part Two
• Congratulations, this is the end of Part Two
• In Part Three, we will go over camera control and how to allow the user to interact with the animation