

An Introduction to Alice



By Teddy Ward

Under the direction of Professor Susan Rodger
Duke University, May 2013

“Begin at the beginning,” the King said, very gravely, “and go on till you come to the end”

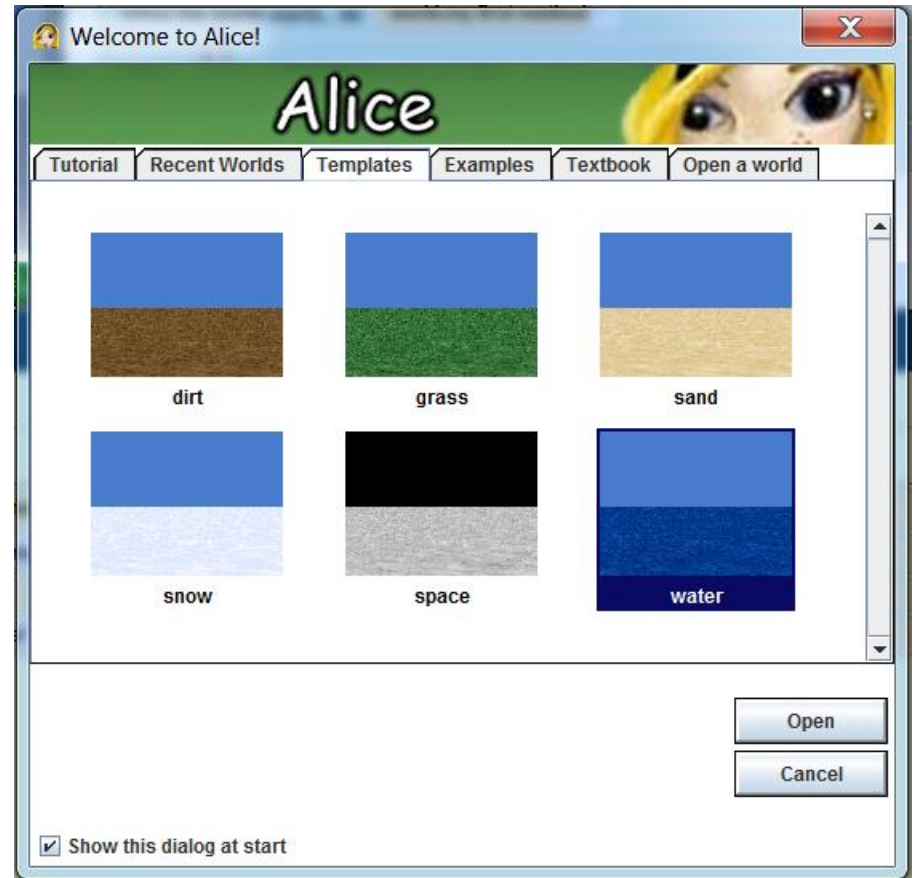
- Lewis Carroll, Alice in Wonderland

Hello! I'm Alice, and I'm going to teach you how to use the Alice program. With Alice, you can make your own animations, using tons of different characters.

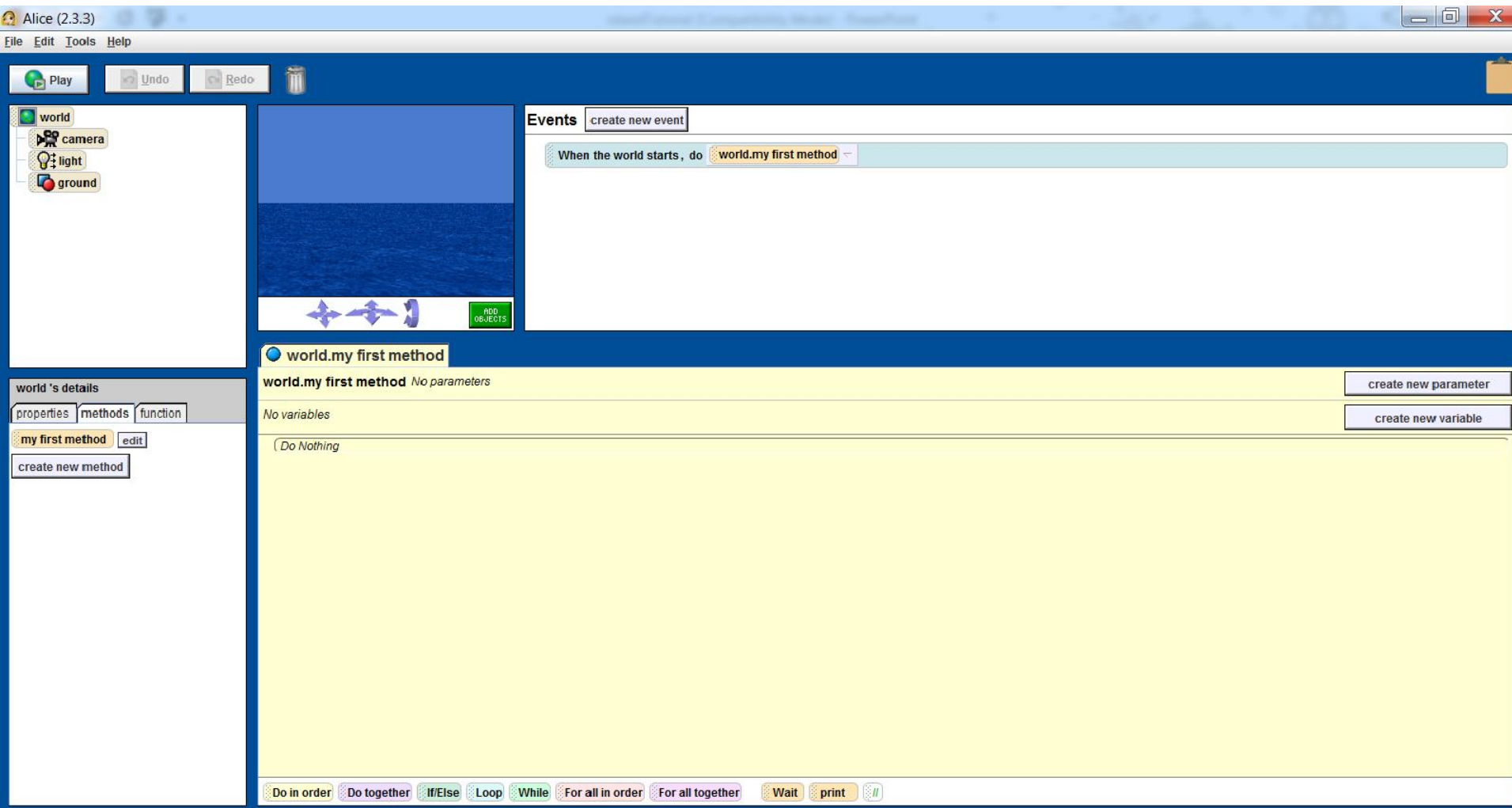


Part 1: Creating a Scene

- Our first step is to choose a background.
- When you open Alice, a box will pop up that has six different choices of background. It looks like the box to the right.
- Select the **water** background, because our world will be on an island.
- Click on **water** and then click **Open**.



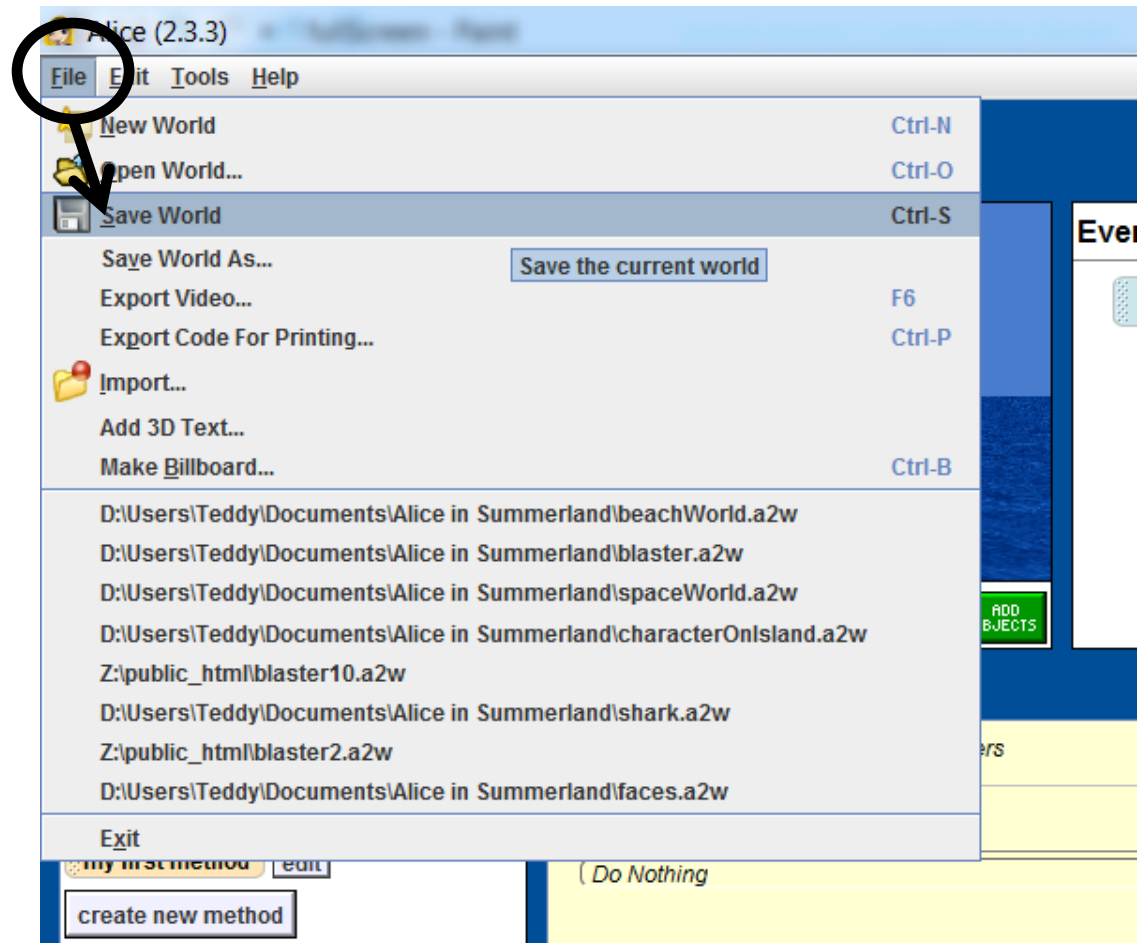
After you click **Open**, your screen will look like this:



Saving your world

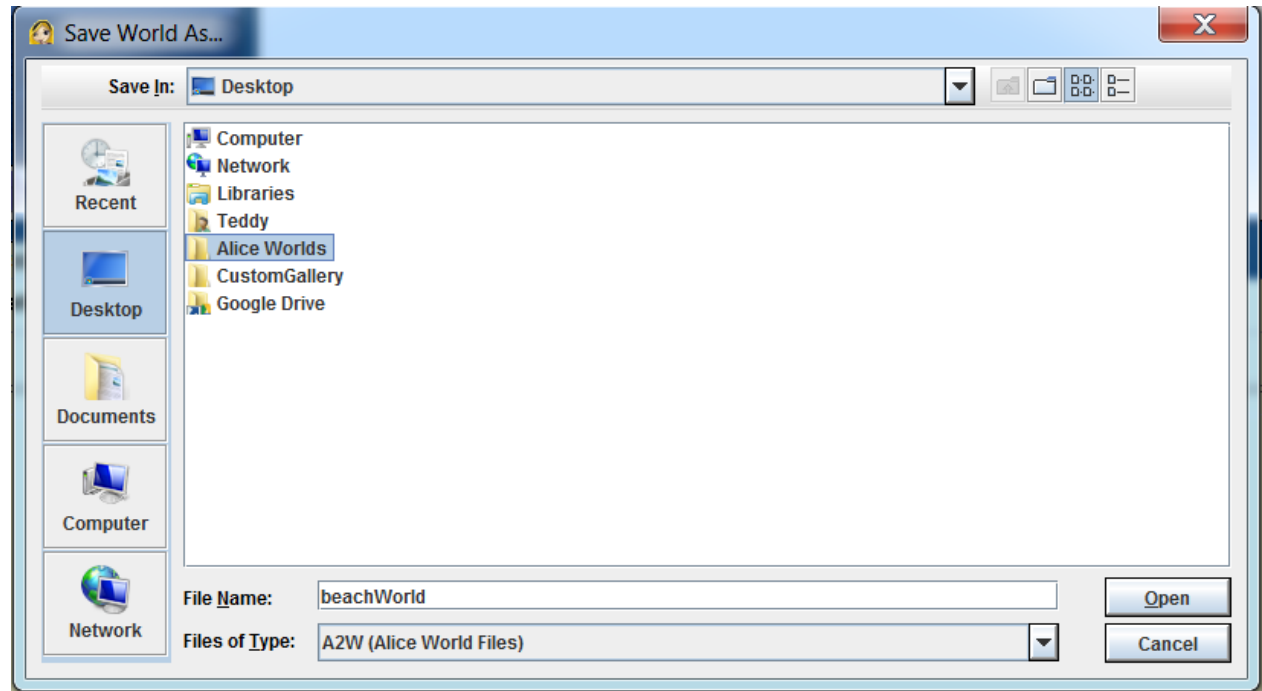
• Before we do anything else, let's save our world. You should also always do this before you close out of Alice.

• Click on **File** at the top left-hand corner of your screen, and then click on **Save World**.



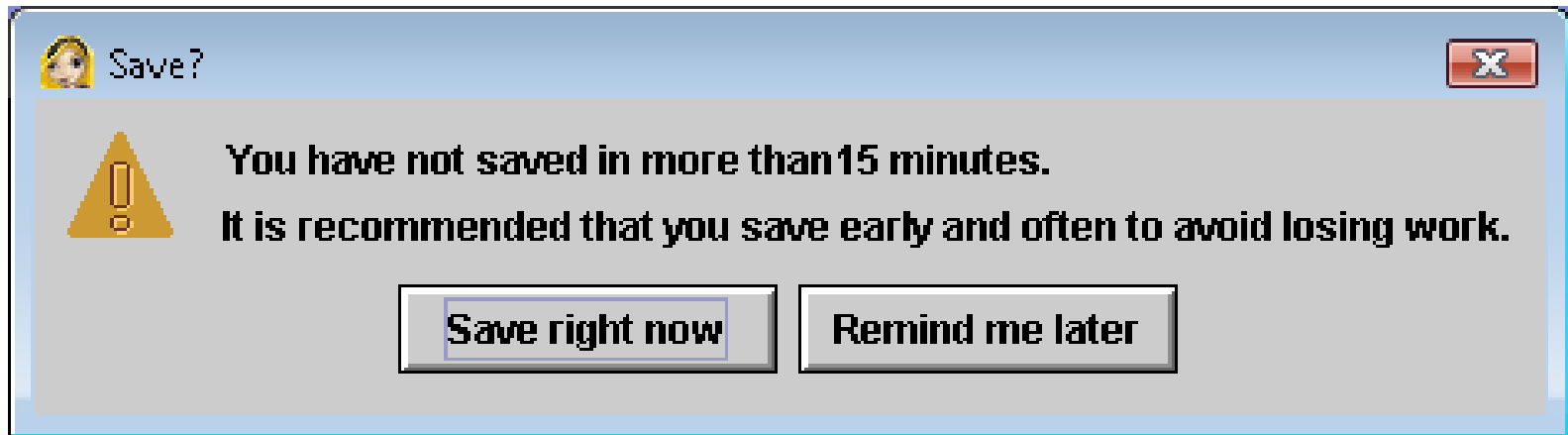
Saving your world

•In the box that pops up, name your world **island**, and save it in a place that you will be able to find again, such as in a folder on your Desktop.



Saving your world

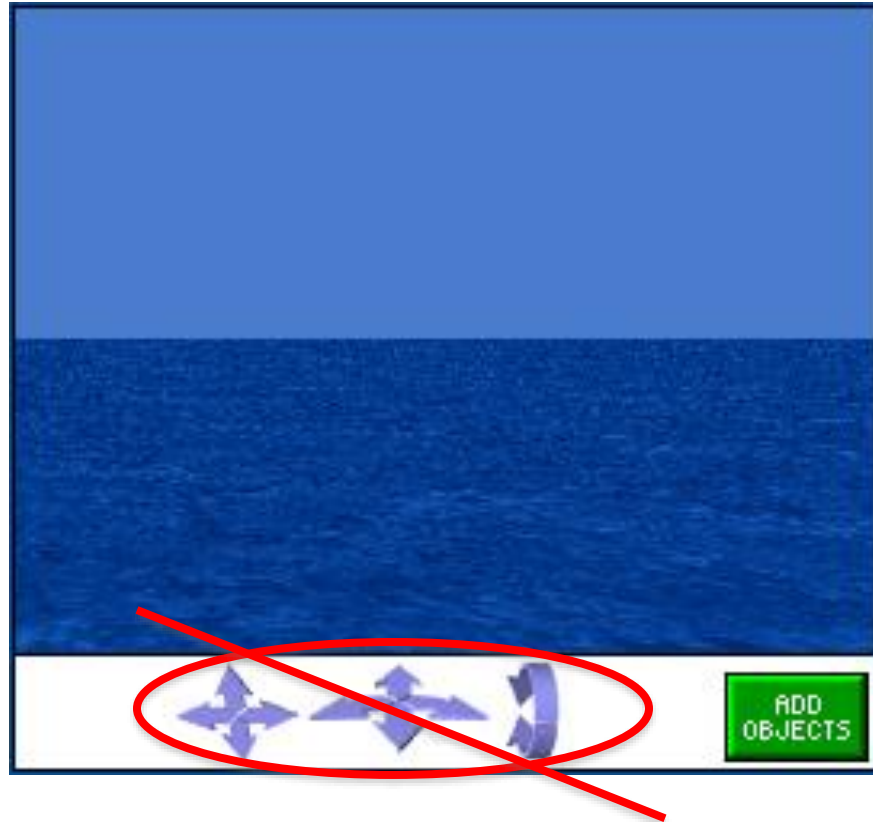
- Also, while you're working on your Alice world, this box will pop up about every 15 minutes.



- You should always click **Save right now**. This way, if Alice crashes, or if your computer crashes, you will have backups of your world and will not lose all of your work!

The Viewer

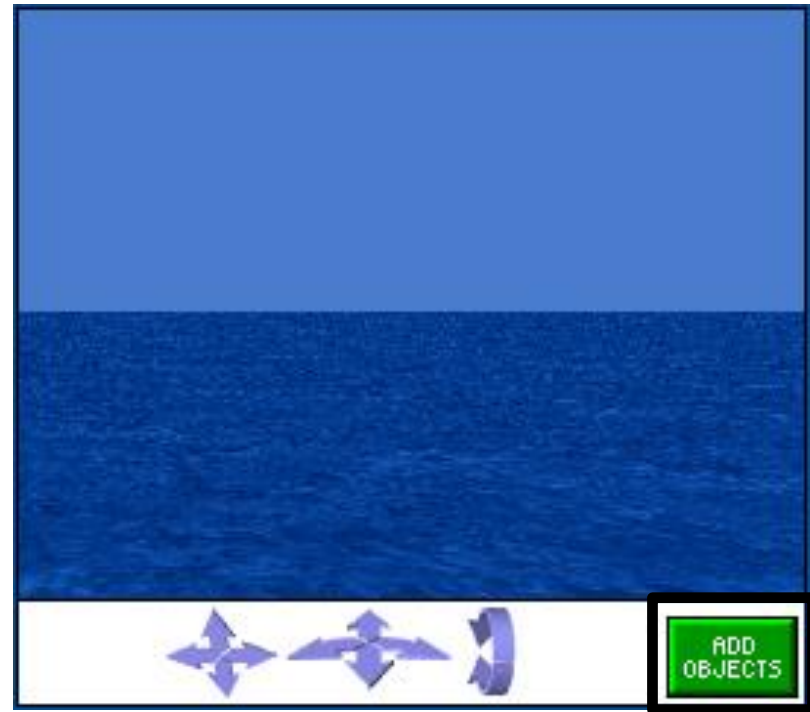
- The viewer lets you view what your world looks like:



- The arrows move the camera. **DO NOT touch them** for now as it's hard to get the camera back to the original camera view

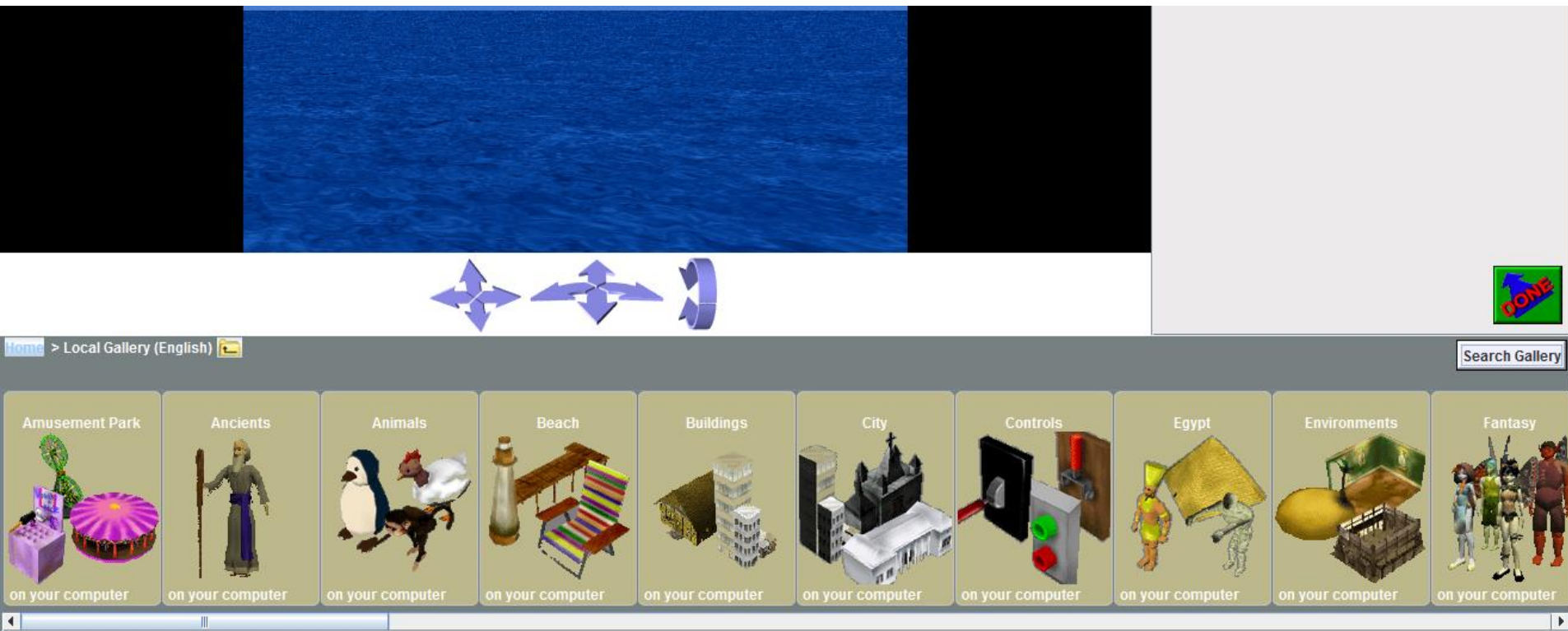
Adding objects to your world

- Now, we will add some objects to the world.
- Just below the picture of your empty space world, there is a small green button that says **Add Objects**.
- Click on this button.



Adding objects to your world

A new screen will appear, on which there is a large selection of objects below the space screen that you can add into your world. This is called the **Local Gallery**. Each folder of objects in the gallery has a different theme.



Adding objects to your world

- Find the **Environments** folder in the gallery – you may have to scroll to the right a bit.



- Scroll to the right again until you see the **Island**.
- Click on the **Island**.
- On the box that pops up, click **Add instance to world**.
- The island will appear in the center of the ocean screen.

Adding objects to your world

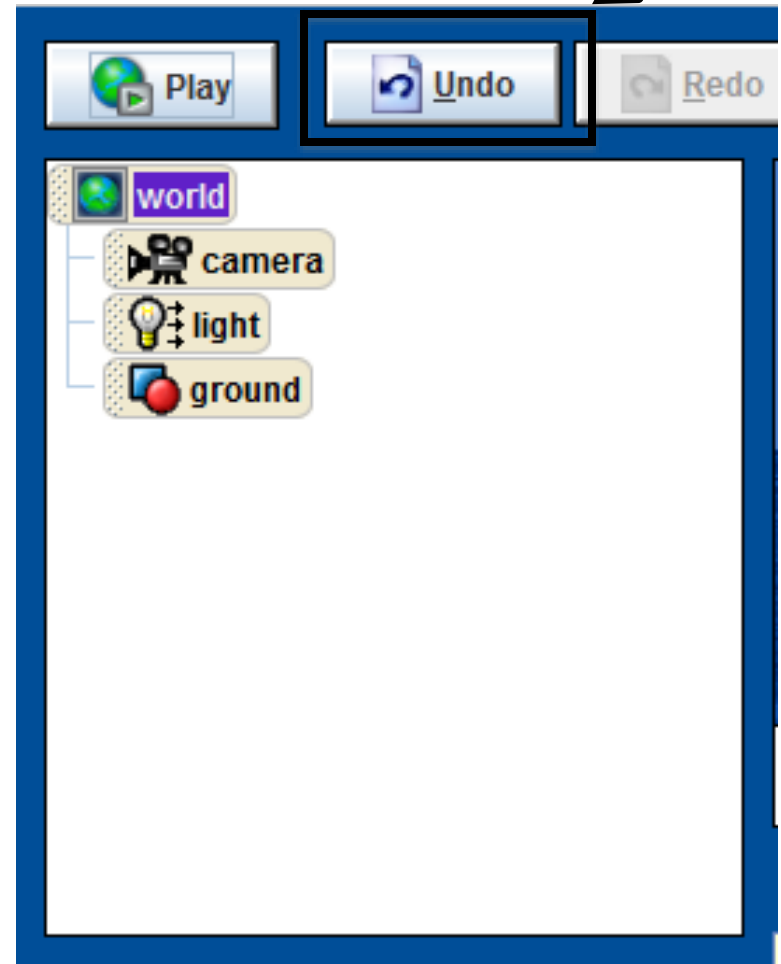
The island will take up most of your viewer, which will cause problems when we want to add more objects in a minute



To fix this, **click and drag the island** around the viewer until it looks like the picture on the **left**

The Undo button is your friend!

- What if you make a mistake, like accidentally clicking on the ocean and moving it? Or what if your island “jumps” offscreen?
- You can click on the **Undo** button above the object tree to undo the last thing you did.
- Use this button whenever you mess up, or want to get rid of something you just did.



Adding a character to your world

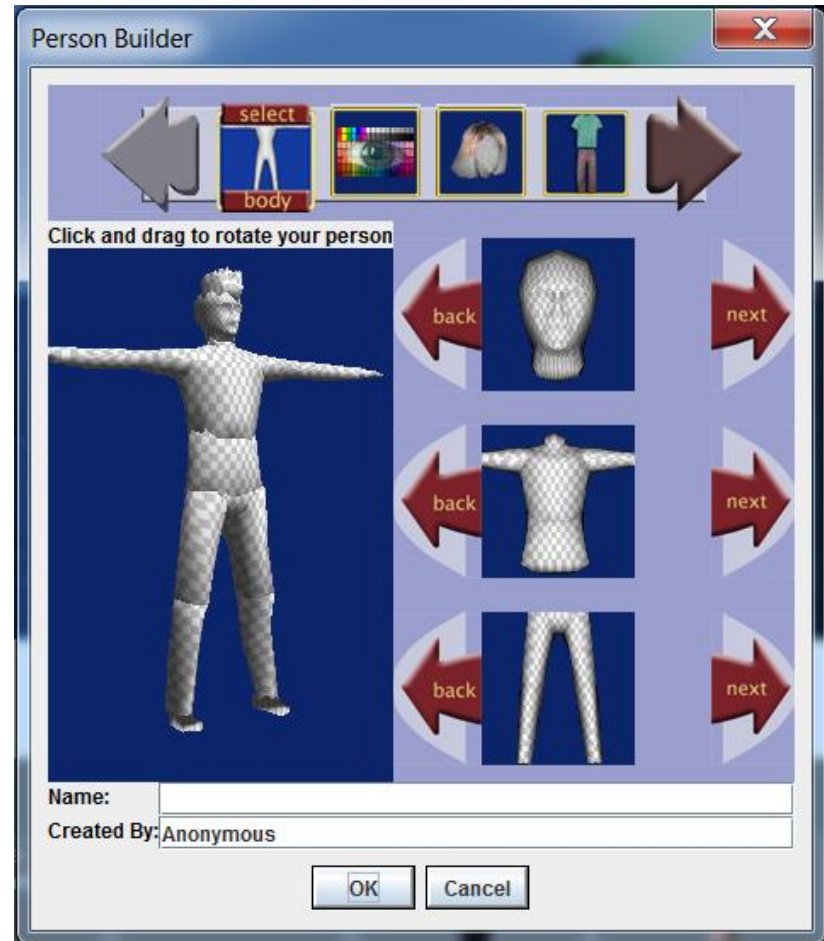
Anyway, we're going to add a person to our world now. Click **Local Gallery** above the pictures of objects to go back to the main gallery of objects.



Scroll to the right until you see a folder called **people**. Click on it.

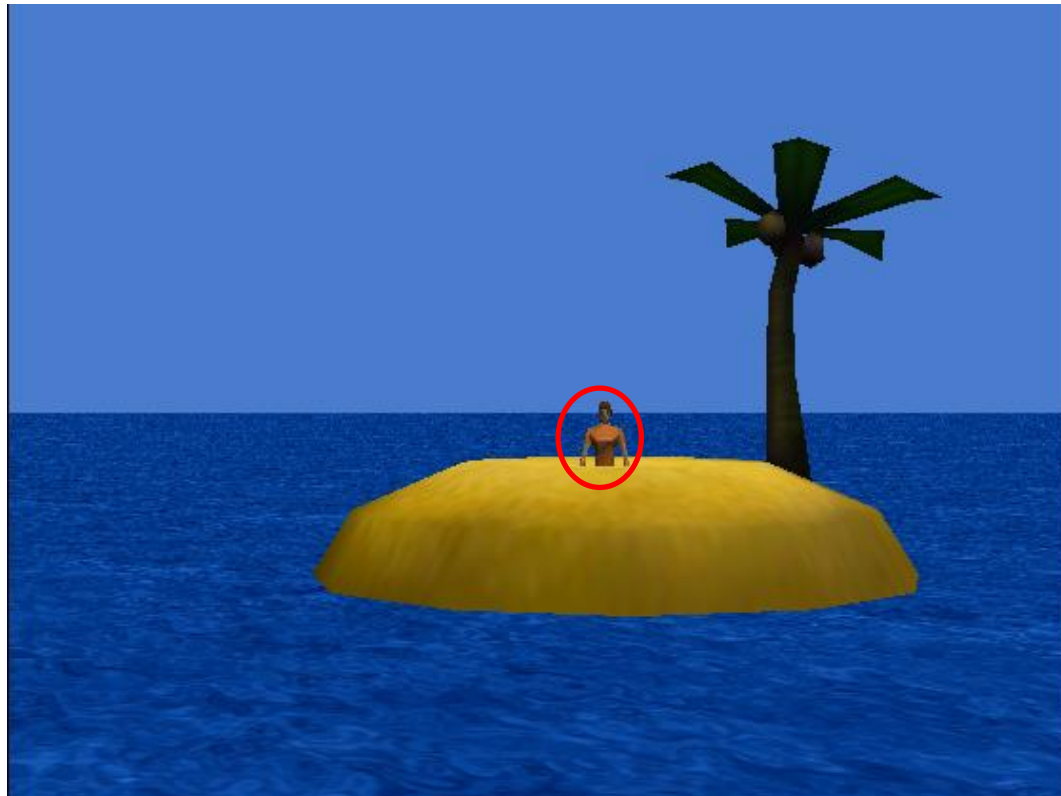
Adding a character to your world

- Click your favorite person from the gallery and then click **add instance to world**.
- You can also create your own character by clicking **hebuilder** (boys) or **shebuilder** (girls) on the far right
- Don't spend too long on this!



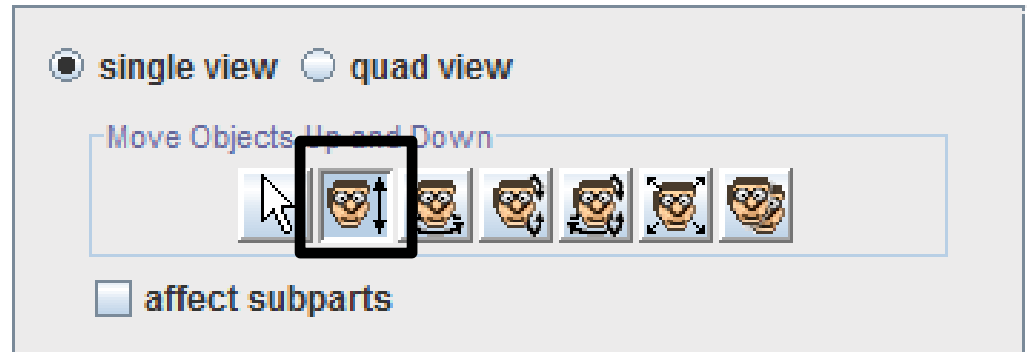
Positioning the objects

- **Click and drag** your character into the middle of the island as shown. We'll get him/her unstuck from the sand in a second.



Positioning the objects

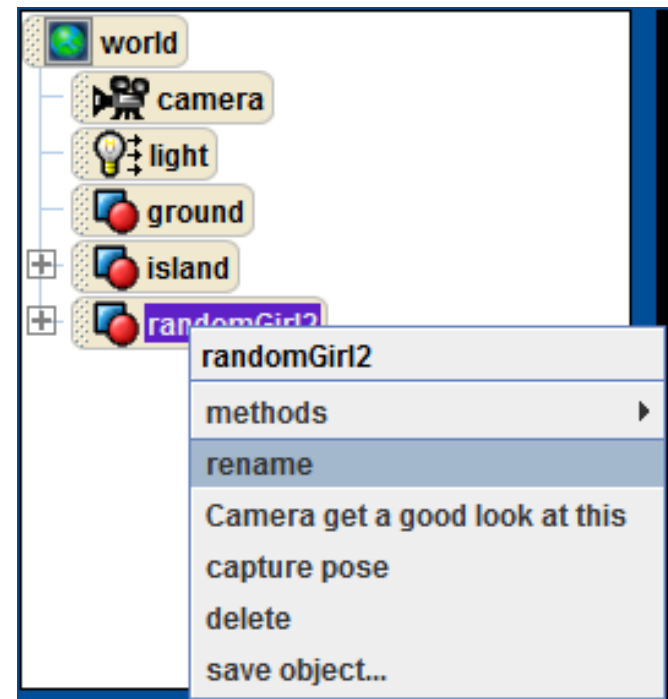
- Look at the right side of your screen.
- There is a group of buttons with faces on them that are used to position objects.
- We want to have our hero stand on the island, so click **Move objects up and down** (the 2nd one)



- Click on your character, hold, and move your mouse up until his/her feet just touch the sand.
- Bonus: move the island down into the water a little bit (move the character down again, too)

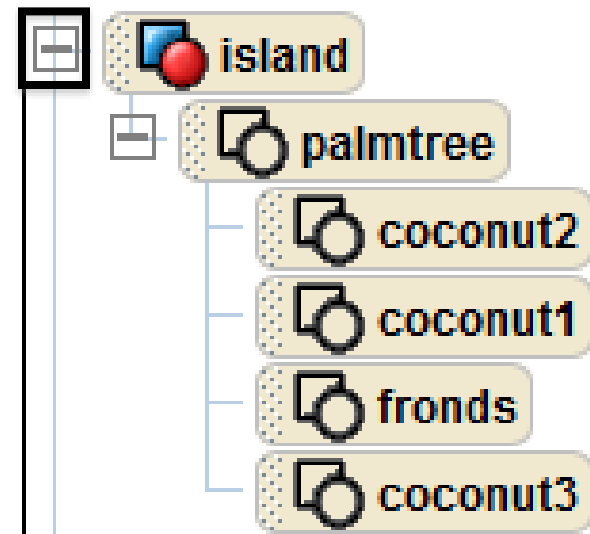
The Object Tree

- When you add objects to your world, they will appear in a list on the left of your screen, called the **Object Tree**.
- You should see both your island and your character there
- If you want, right click on your character's name, select **rename**, and type any name you want. I renamed mine "heroine."



Object Parts

- Some objects have parts.
- For example, if you click on the plus mark next to the island in your object tree, you'll see that the palm tree and its fronds and coconuts are all objects in your world
- These are fixed in position, unless you move them with *methods*.



Object Parts

- Right click on **coconut1**, choose **methods, move, down, other...**, and then a calculator will pop up for input.

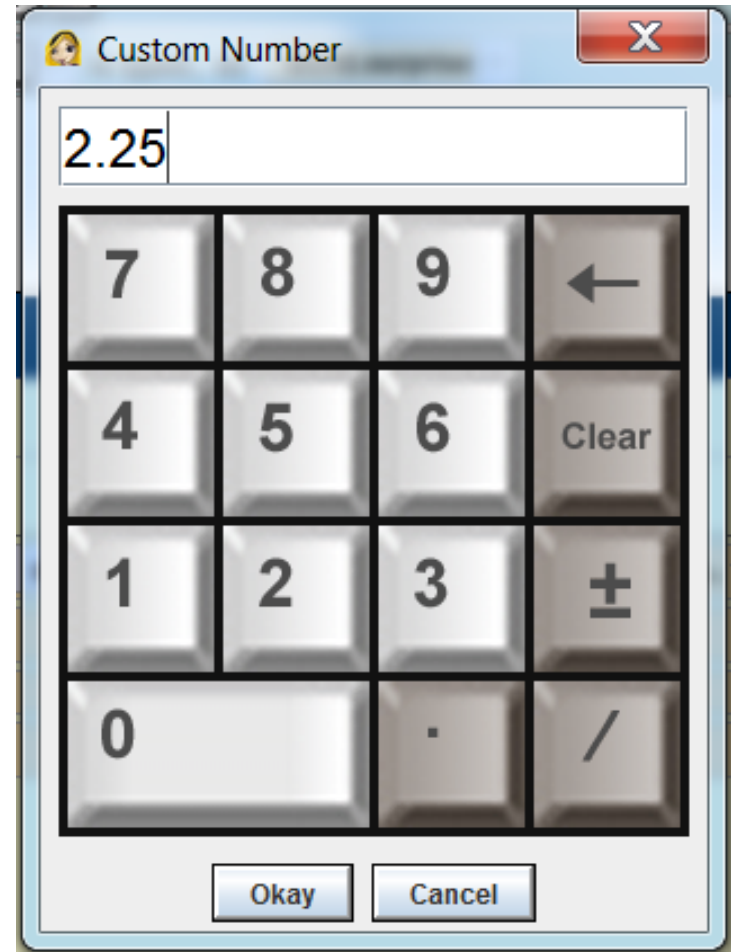
The screenshot displays a multi-level context menu for the object 'coconut1'. The menu structure is as follows:

- coconut1
 - methods
 - island.palmtree.coconut1 move
 - direction
 - up
 - down
 - amount
 - 1/2 meter
 - 1 meter
 - 5 meters
 - 10 meters
 - 2.2 meters
 - 2.25 meters
 - 2.5 meters
 - 2 meters
 - other...
 - left
 - right
 - forward
 - backward
 - more...
 - island.palmtree.coconut1 turn
 - island.palmtree.coconut1 roll
 - island.palmtree.coconut1 resize
 - island.palmtree.coconut1 say
 - island.palmtree.coconut1 think
 - island.palmtree.coconut1 play sound
 - island.palmtree.coconut1 move to
 - island.palmtree.coconut1 move toward
 - island.palmtree.coconut1 move away from
 - island.palmtree.coconut1 orient to
 - island.palmtree.coconut1 turn to face
 - rename
 - Camera get a good look at this
 - capture pose
 - delete
 - save object...

Below the menu, there are several buttons: 'thod', 'edit', 'edit', and 'method'.

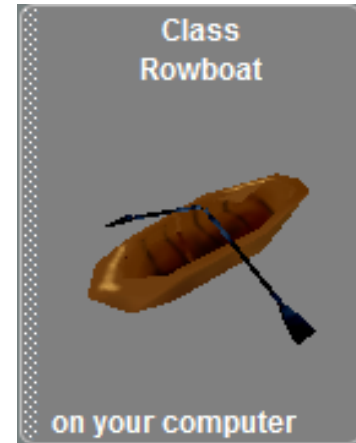
Object Parts

- Type **2.25** into the calculator, and when you press **okay** one of the coconuts will fall to the ground.
- Turning, rotating, and moving object parts is a good way to make more complex animations look more realistic!
- Note: “move” may detach a part from its object but “turn” and “roll” may move parts without detaching them



Adding objects to your world

- Now return to the **Local Gallery**.
- Scroll over to the **Vehicles** folder and click on it.
- Scroll over to **Rowboat** and add that to your world.

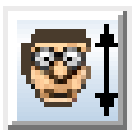
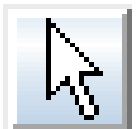


Your world should look like this:



Positioning the objects

- Now move the rowboat to the front left of the island using the first three buttons in that group on the top right
- Click **Move objects freely** (the 1st one) to move it around
- Click **Move objects up and down** (the 2nd one) to lower it into the water
- Click **Turn objects left and right** (the 3rd one) to align it with the shore



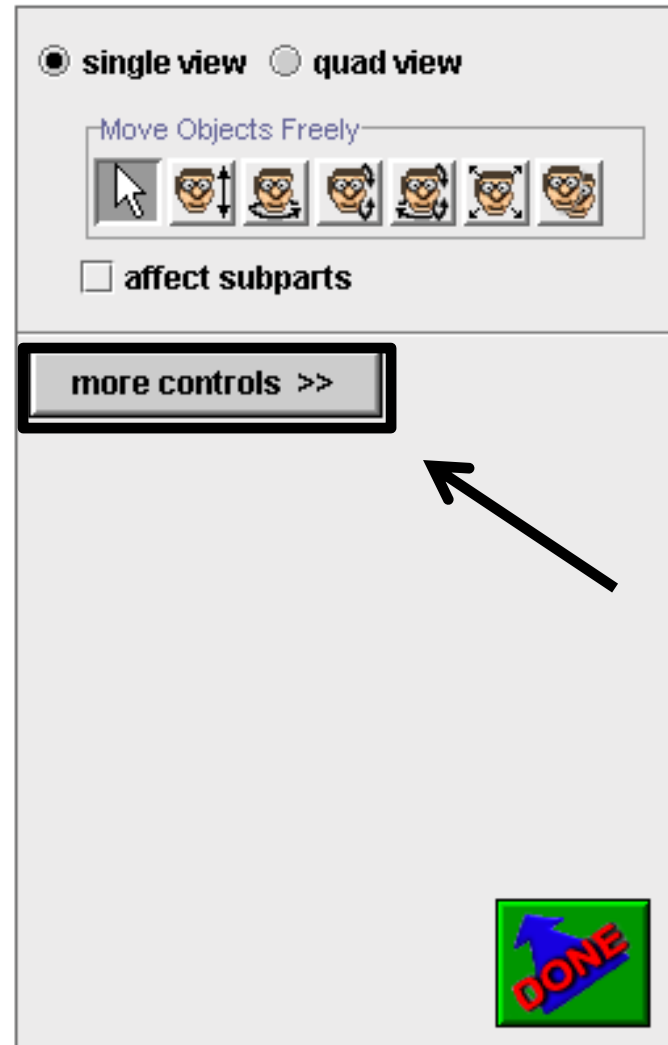
Nice job setting up
the scene! Now we're going to
learn about the camera!



(If you have extra time, add more scenery!
Use the **Ocean** folder in the local gallery to
add shells, fish, etc. Don't add a shark just
yet, though 😊)

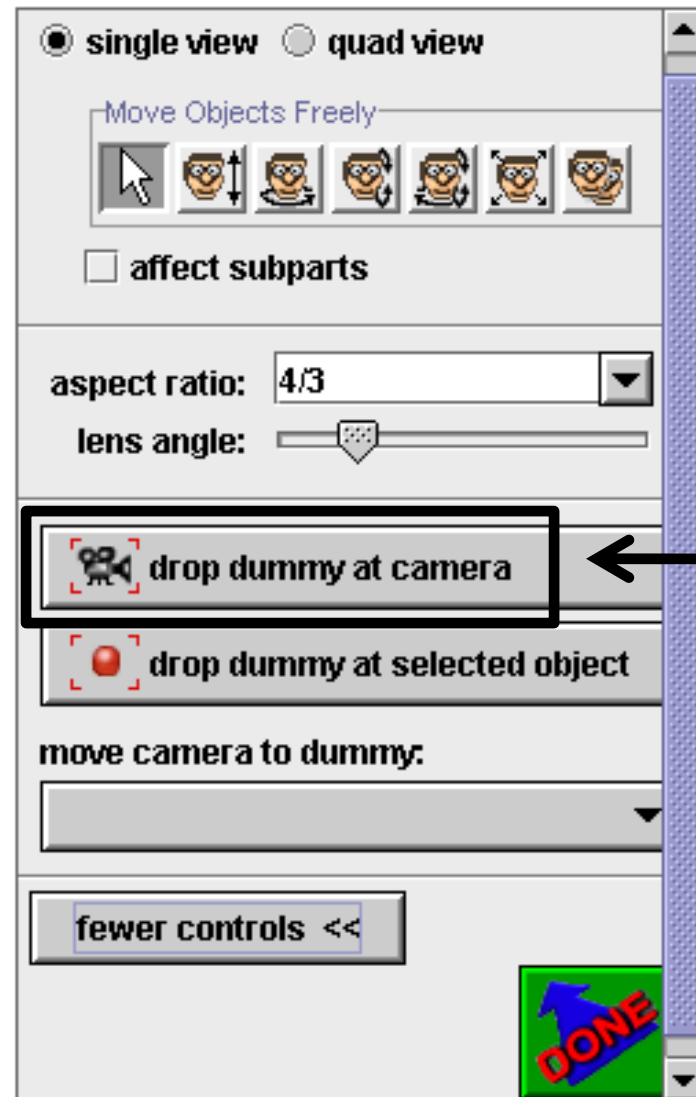
Part 2: The Camera

- A *Dummy camera* is like a camera tripod – it saves the location of your camera view. This way, if you move your camera around, you can always get back to a certain position by moving to a dummy camera location.
- Look to the right side of your screen, and find a gray button under your object positioning buttons labeled **more controls**



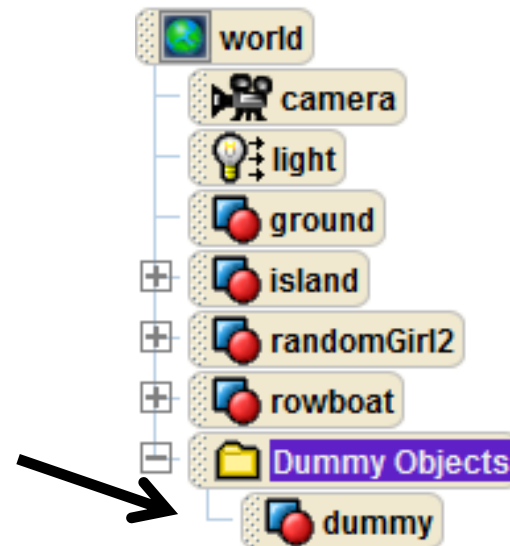
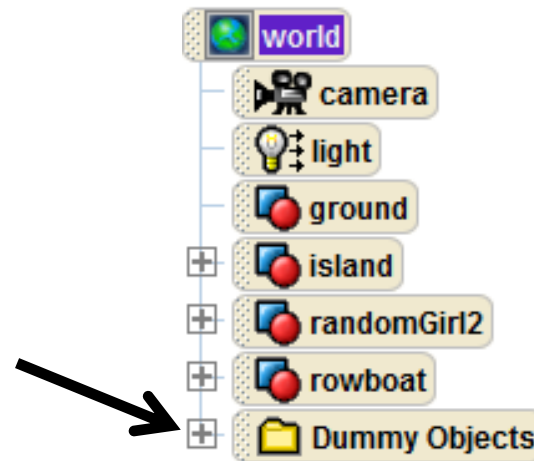
The Dummy Camera

- More buttons will appear after you click **more controls**.
- Click on the button that says **drop dummy at camera**. It will seem like nothing happens, but don't worry, and only click the button one time.



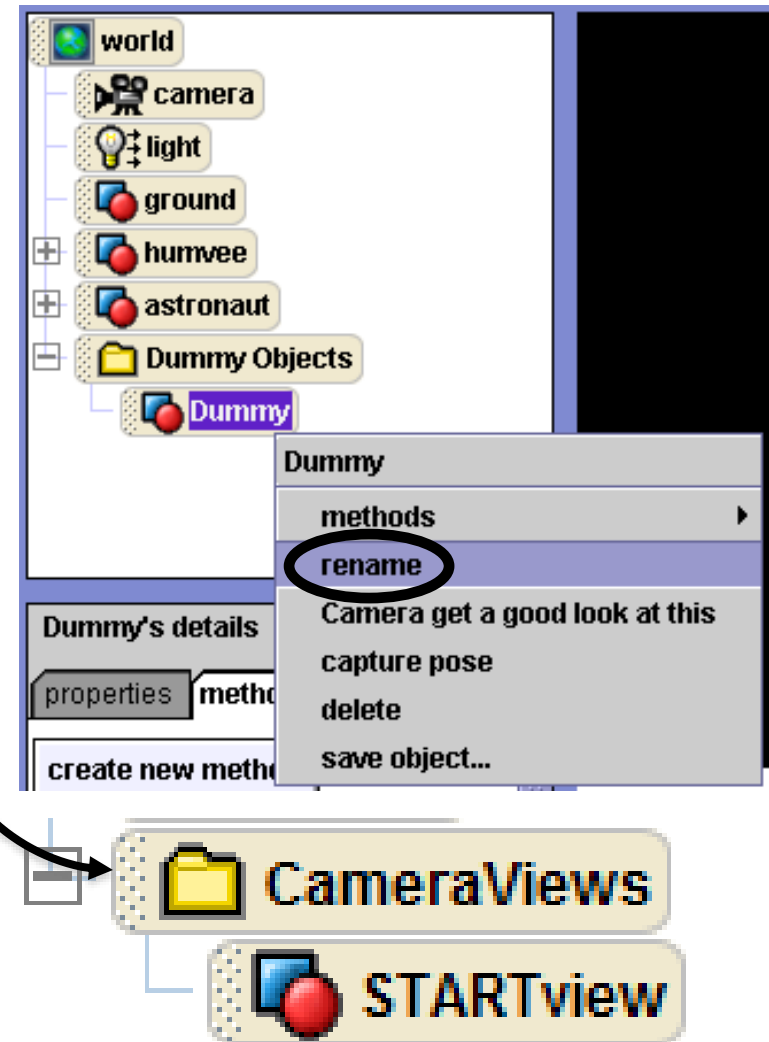
The Dummy Camera

- Once you have clicked this button, a folder will appear on your object tree labeled **Dummy Objects**.
- If you click on the **plus sign** next to the **Dummy Objects** folder, a list of your dummy camera positions will appear.
- Right now, there is only one position, called **dummy**.

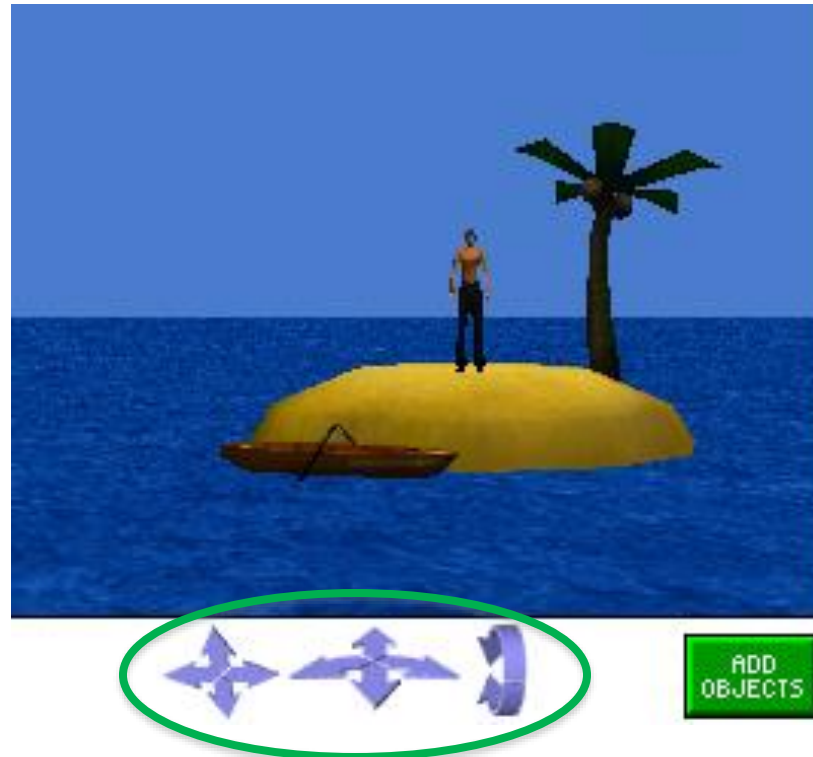


The Dummy Camera

- Whenever you add a dummy camera position, you should rename it so that you know which position it is.
- Right click on **dummy** in the object tree, and then choose **rename**. Type in **STARTview**.
- Similarly, change the folder name **Dummy Objects** to **CameraViews**
- You should add a dummy at your starting camera position is whenever you start a new Alice world.



Moving the Camera



- Now that we have a dummy camera set up, we can move the camera freely without losing our place.
- There are three sets of arrows beneath the scene that **move the camera.**

Moving the Camera

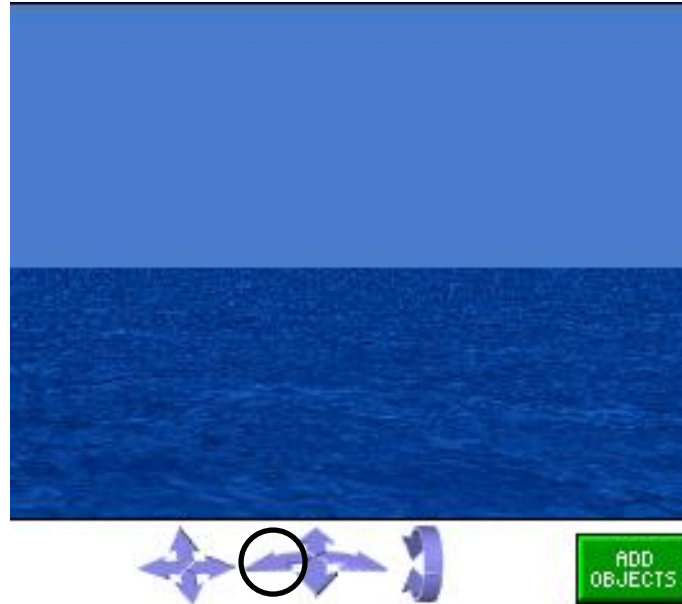


The first set *moves* the camera up, down, left, and right.

The second set moves it forward and backward and *pans* it from left to right.

The last set *rotates* the camera up and down.

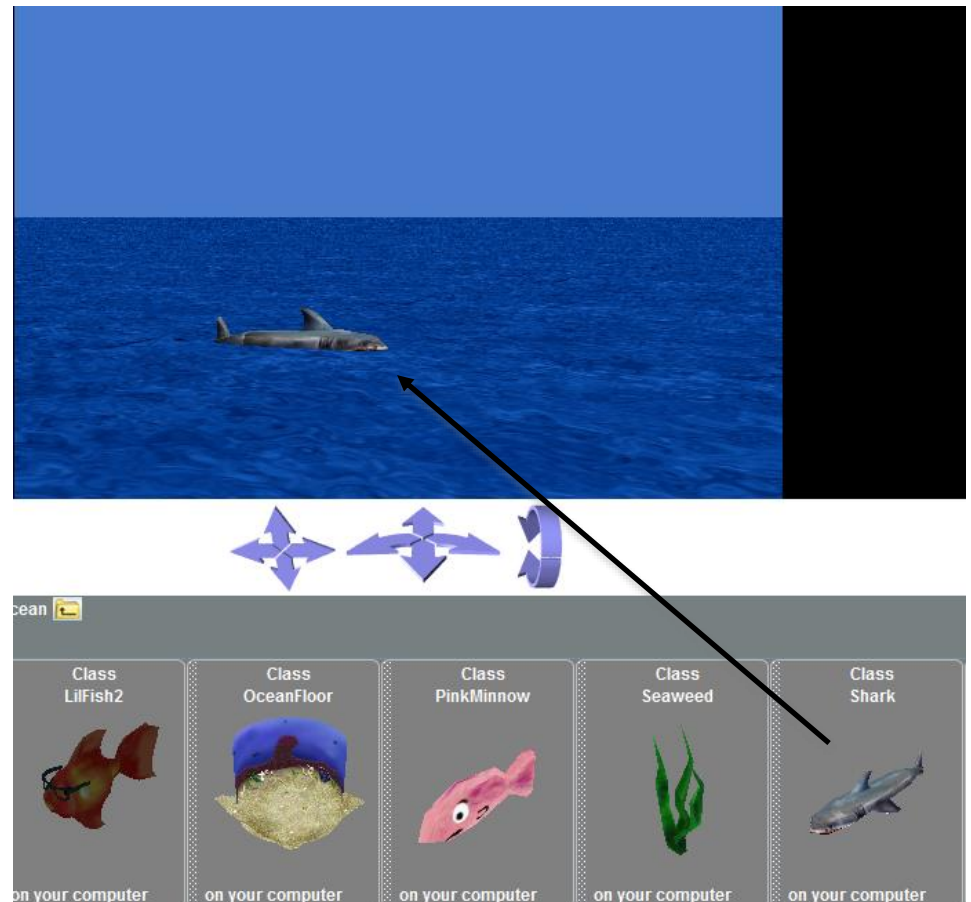
Moving the Camera



- Click and hold the **left arrow in the second group** to pan the camera left until the island is out of the picture
- Dragging your mouse farther from the arrows moves the camera faster
- Be sure not to move the camera too far up or down

The plot thickens...

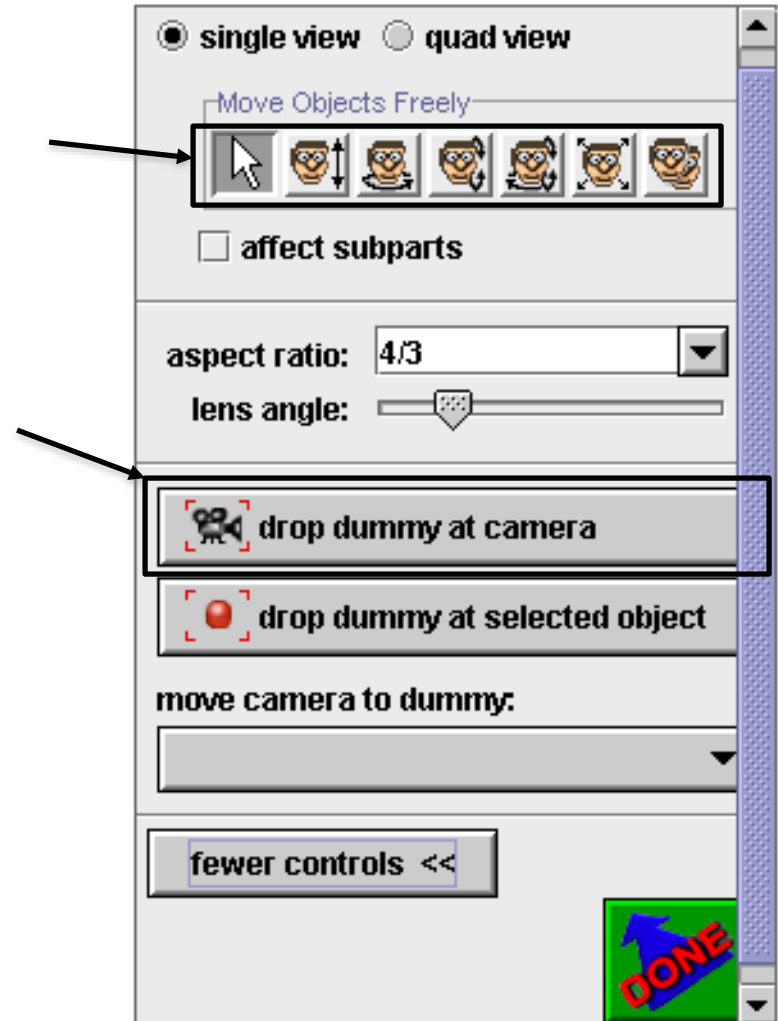
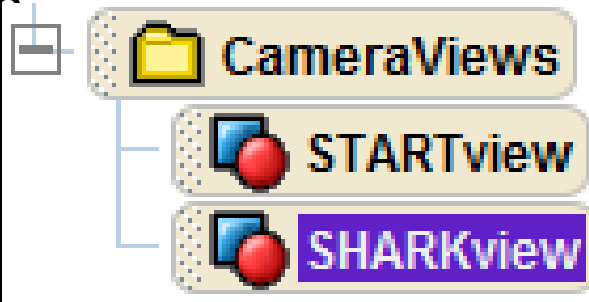
- Click **Add Objects**, and navigate to the **“Ocean”** gallery
- Click and **drag a shark** up to the window to add it to this part of the world



The plot thickens...

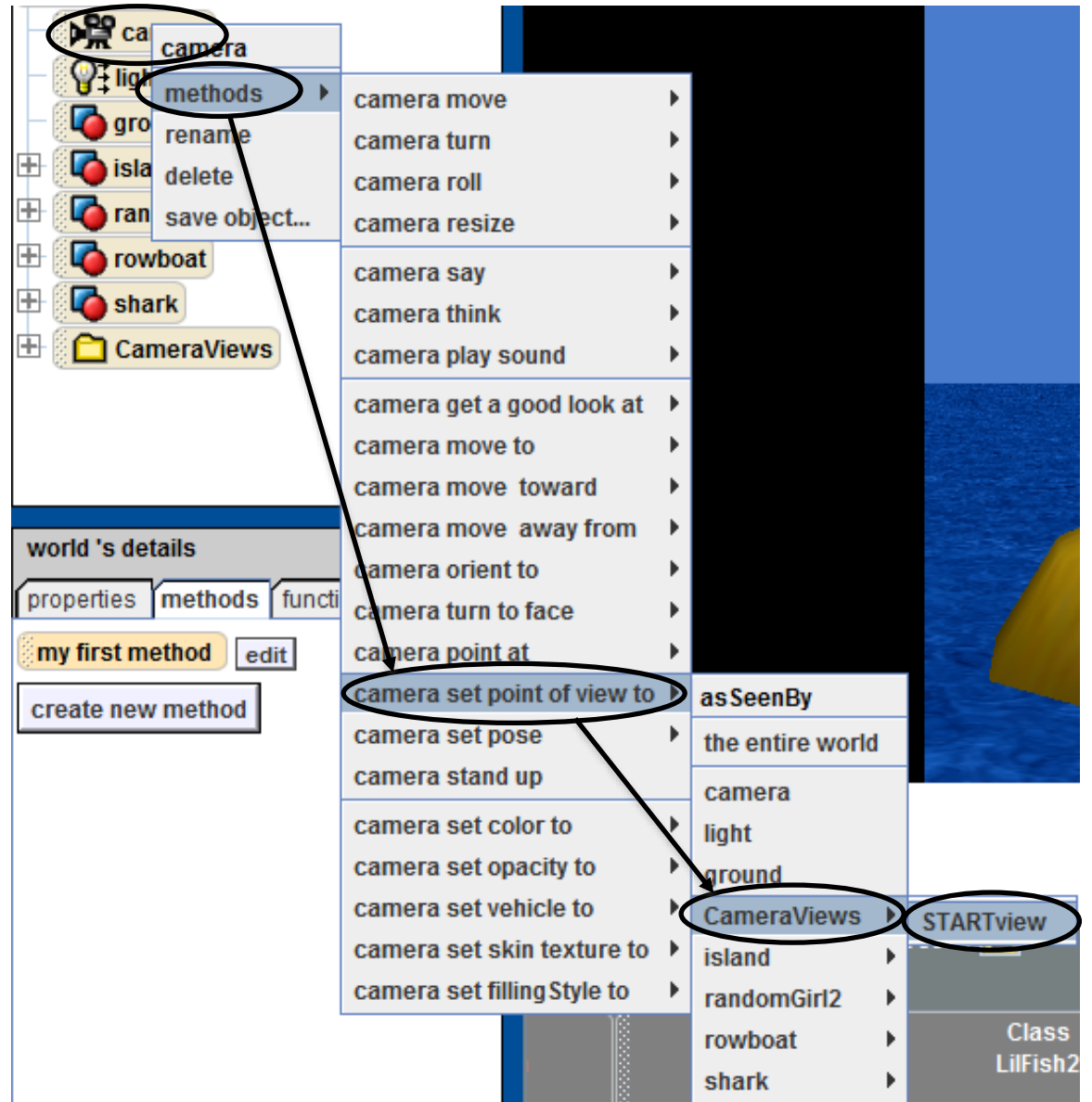
- Move the shark using the positional buttons so that it's half in the water and roughly facing the island
- Click **more controls**, then **drop dummy at camera** so that we have this view saved
- **Rename** this dummy camera **"SHARKview"** in the object tree

tree



Moving the Camera

- To restore the camera, right click on **camera** in the object tree
- On the menu that pops up, choose **methods**, then **camera set point of view to**, then **CameraViews**, then **STARTview**.



Finishing Setup

- Now just click **Done** (in green) towards the bottom right!



Now that we
are done setting up our camera,
we can start to animate the characters
in the world!



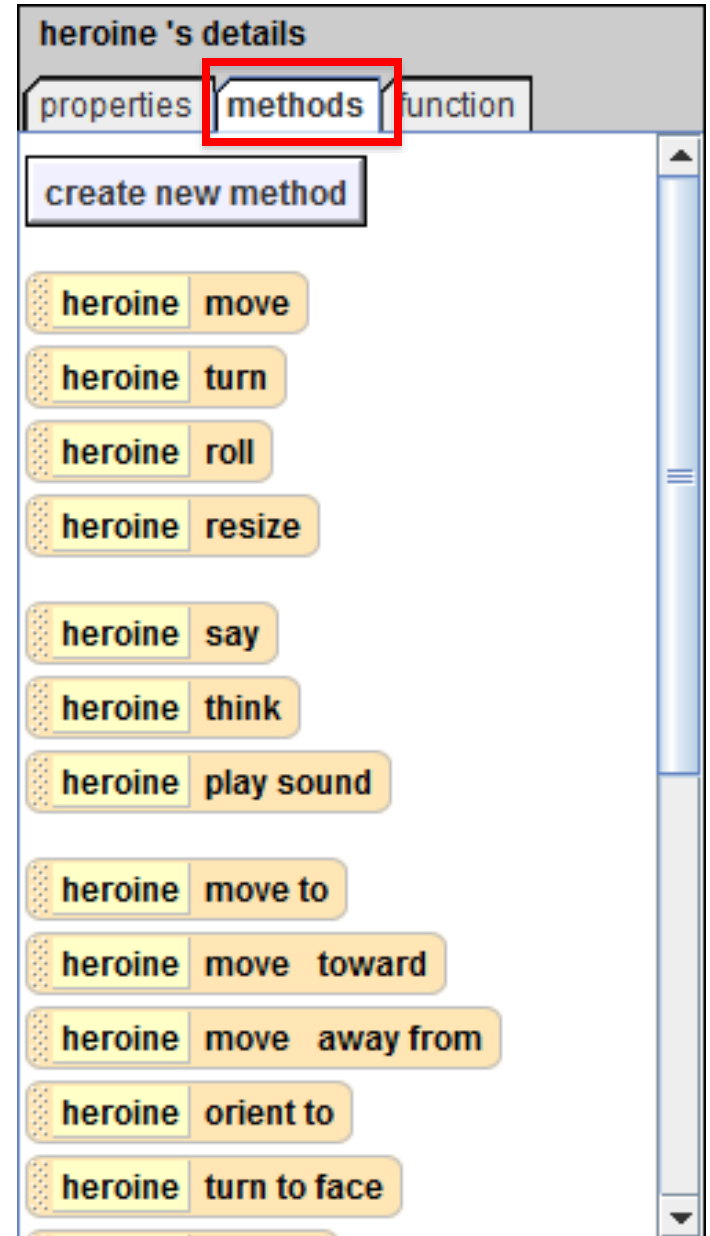
Part 3: Methods

- The large tan rectangle in the center of your screen is called the **Method Editor**. Right now, it is blank.

The screenshot shows the Method Editor interface. At the top left, there is a tab labeled "world.my first method" with a blue circular icon. Below the tab, the text "world.my first method" is displayed, followed by "No parameters" in a smaller font. To the right of this text are two buttons: "create new parameter" and "create new variable". Below this header area, the text "No variables" is visible on the left, and another "create new variable" button is on the right. The main area of the editor is a large, empty tan rectangle. At the top left of this area, the text "Do Nothing" is visible. At the bottom of the editor, there is a toolbar with several buttons: "Do in order", "Do together", "If/Else", "Loop", "While", "For all in order", "For all together", "Wait", "print", and a comment icon (//).

Methods

- The method editor is where you can make your characters do things.
- Your characters already know how to do certain things.
- These are some of the things that your character already knows how to do. To find this list, click on **your character's name** in the object tree. Then look below the object tree at the box that says **details**, and click on the **methods** tab. This list will appear.



Methods

- To program your character to do something, click on one of these methods, hold down your mouse, and drag and drop it into your method editor. Let's try dragging in **say** to start. Select **other...** to be able to choose what we want he/she says freely. Then type something like "Hey! Welcome to my island!" and press **okay**.

The screenshot shows a programming environment with a method editor on the right and a list of methods on the left. The method editor is titled "world.my first method" and shows "No parameters" and "No variables". The current method content is "(Do Nothing)". A dropdown menu is open, showing options: "what", "Hello", "Goodbye", and "other...". An arrow points from the "heroine say" method in the list to the "other..." option in the dropdown. Below the method editor, the text "Becomes..." is visible. At the bottom, a command bar shows "heroine" selected, followed by "say", the text "Hey! Welcome to my island!", and "more..." with a dropdown arrow.

heroine 's details
properties methods function
create new method
heroine move
heroine turn
heroine roll
heroine resize
heroine say
heroine think

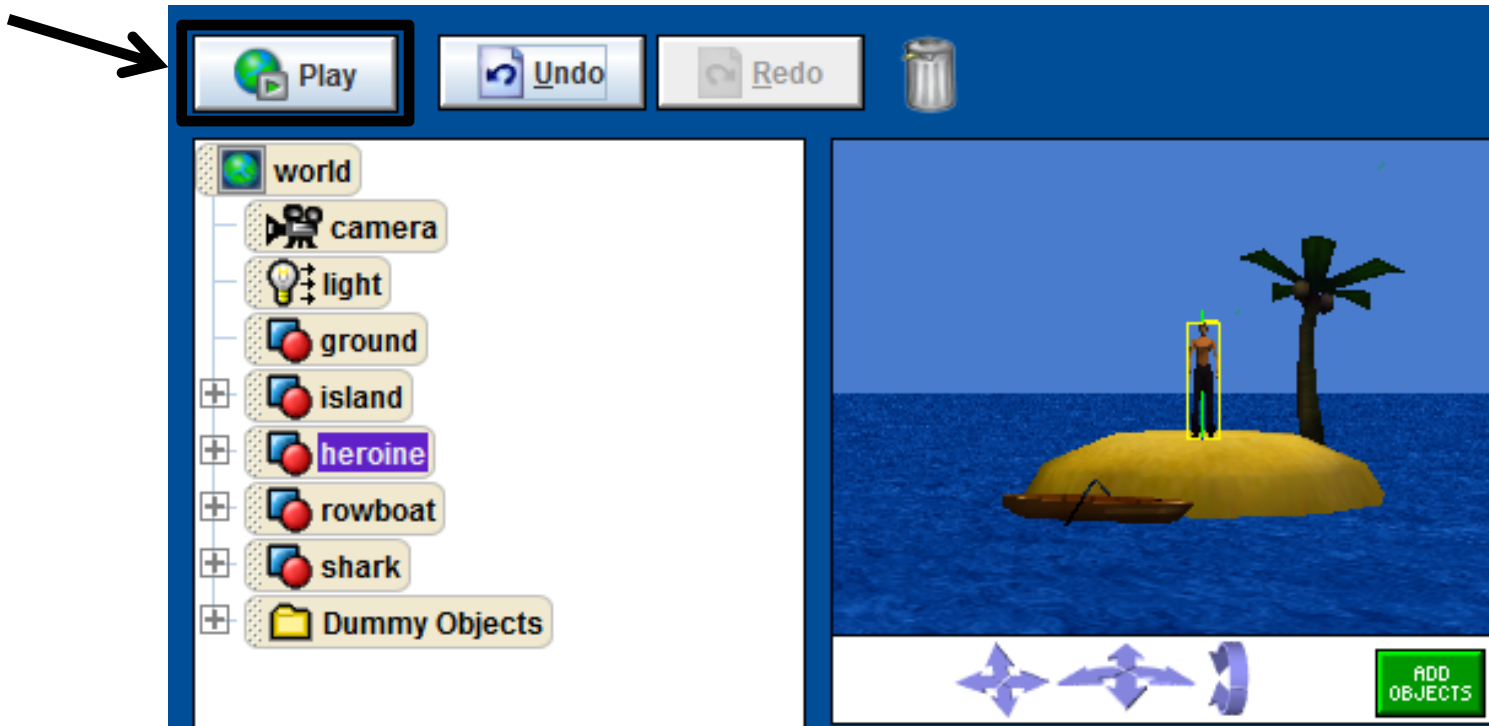
world.my first method
world.my first method No parameters
No variables
(Do Nothing
what
Hello
Goodbye
other...

Becomes...

heroine say Hey! Welcome to my island! more...

Methods

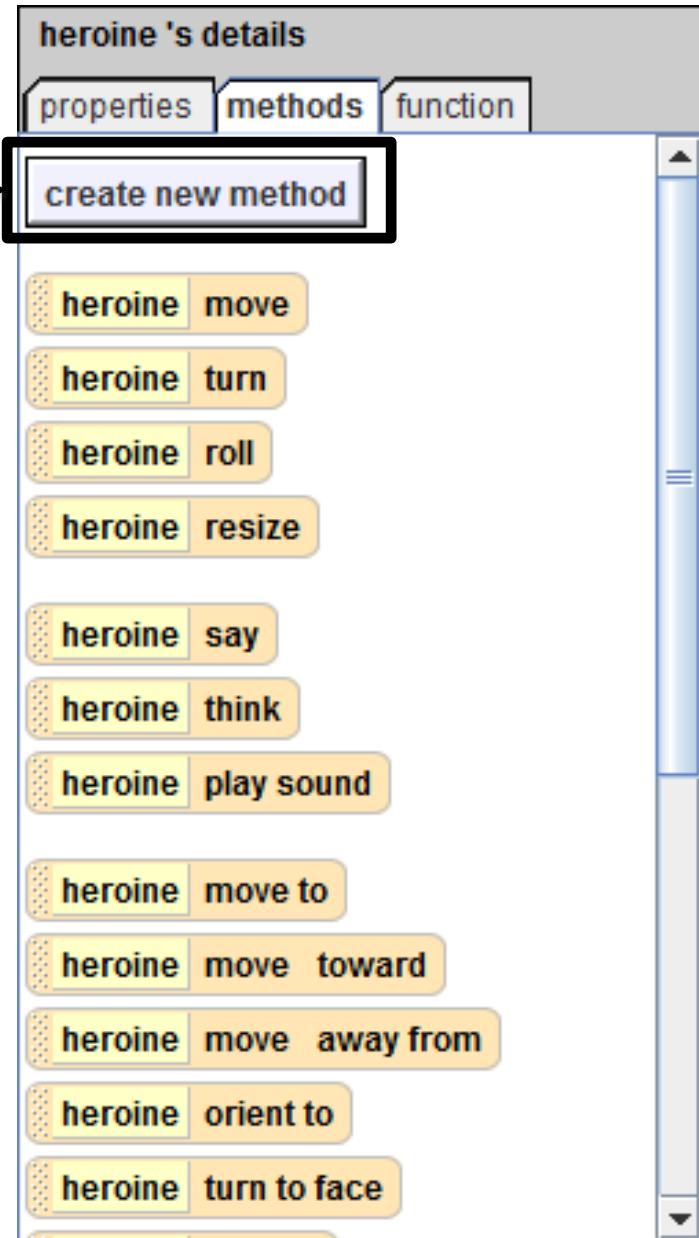
- Now press the **Play** button in the upper left-hand corner of the screen to watch your AMAZING program in action!!!!



- Okay. That was pretty boring. Let's spice things up by teaching our character to do a backflip

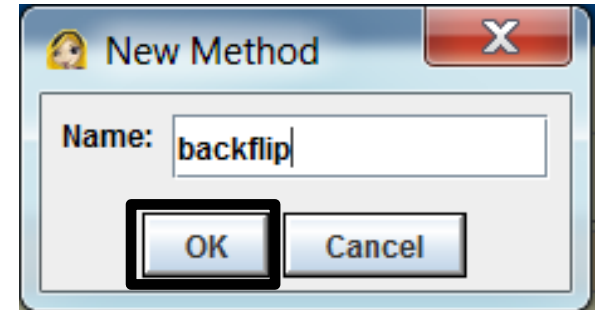
Methods

- To teach your character new things, you can combine methods that he/she already knows into new methods.
- Make sure you have clicked on your character in the object tree. Then, go to the methods for your character and click **create new method**.

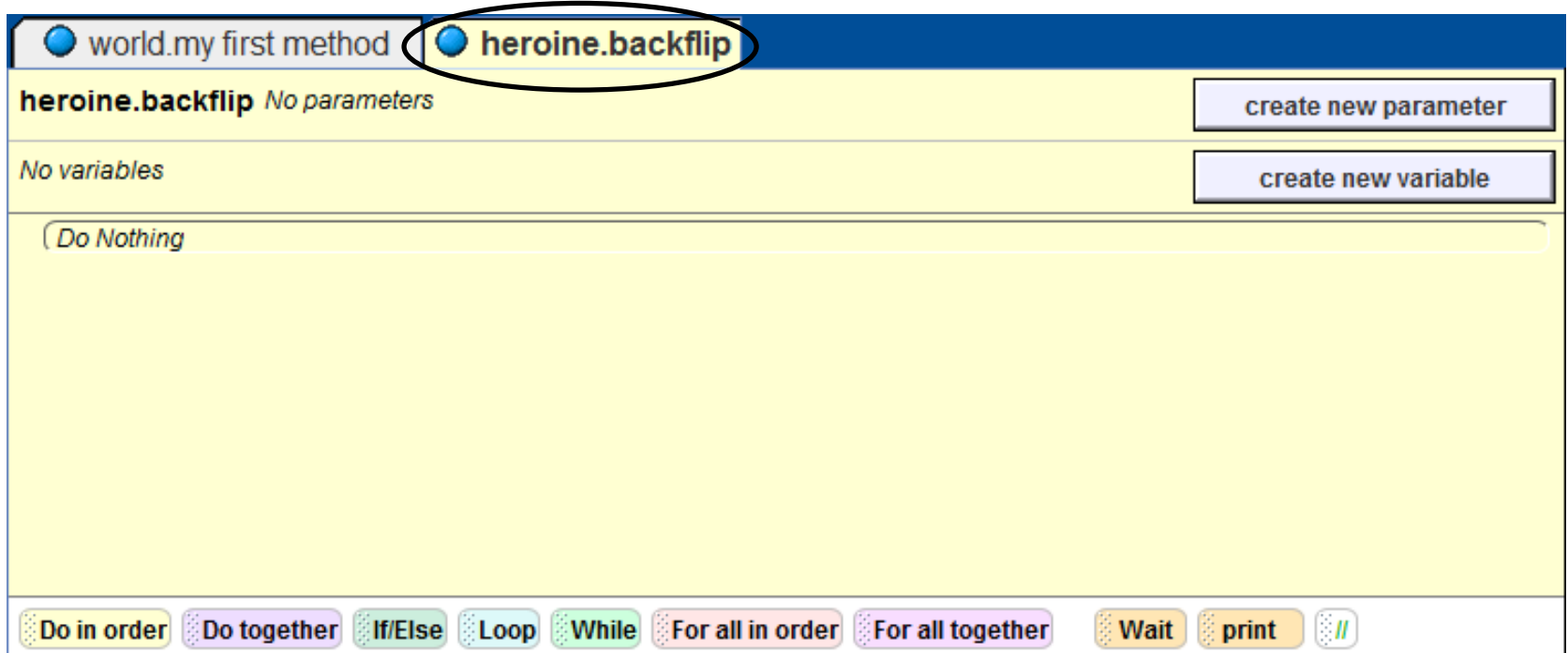


Backflip

- In the box that pops up, type **backflip**, then click **OK**.



- You should see a new tab appear in your method editor called **heroine.backflip** (heroine will be replaced by your character's name). This is the space where you will program the backflip.



Backflip

- Drag **heroine move** into the method editor
- Select **up**, then **1 meter**
- Drag another **move** below this, but this time select **down** > **1 meter**.

The screenshot shows a Scratch-like environment with two panes. The left pane, titled 'heroine's details', has tabs for 'properties', 'methods', and 'function'. Under 'methods', there is a 'backflip' block with an 'edit' button and a 'create new method' button. Below this, there are several 'heroine' blocks: 'move', 'turn', 'roll', 'resize', and 'say'. The right pane, titled 'world.backflip', shows the method editor for 'world.backflip' with 'No parameters' and 'No variables'. It contains a 'Do Nothing' block and a dropdown menu for 'direction' and 'amount'. The 'direction' dropdown is open, showing options: 'up', 'down', 'left', 'right', 'forward', 'backward'. The 'amount' dropdown is also open, showing options: '1/2 meter', '1 meter', '5 meters', '10 meters', 'other...'. The '1 meter' option is selected in both dropdowns.

direction	amount
up	1/2 meter
down	1 meter
left	5 meters
right	10 meters
forward	other...
backward	

Backflip

- To finish the flip, drag in **turn** between the two movements (a green line should appear when you're between them)
- Choose backward, then 1 revolution (all the way around)
- Now your character will move up, flip, and then move down

The screenshot shows a programming environment with a sidebar on the left and a script editor on the right. The sidebar, titled 'heroine's details', has tabs for 'properties', 'methods', and 'function'. Under 'methods', there is a 'backflip' block with an 'edit' button, and a 'create new method' button. Below that, several 'heroine' blocks are listed: 'move', 'turn', 'roll', 'resize', 'say', 'think', and 'play sound'. The 'turn' block is highlighted with a black box. An arrow points from this 'turn' block to the script editor. The script editor shows a script for 'world.backflip' with 'No parameters' and 'No variables'. It contains two 'heroine' blocks: 'move up 1 meter' and 'move down'. A green line is visible between the two 'move' blocks. A dropdown menu for the 'turn' block is open, showing 'direction' with options: 'left', 'right', 'forward', and 'backward'. The 'backward' option is selected and highlighted with a black box. A sub-menu for 'backward' is open, showing 'amount' with options: '1/4 revolution', '1/2 revolution', '1 revolution (all the way around)', '2 revolutions', and 'other...'. The '1 revolution (all the way around)' option is selected and highlighted with a black box.

Backflip

- Your code should look like this:

The image shows a Scratch code editor interface. At the top, there are two tabs: 'world.my first method' and 'heroine.backflip'. The 'heroine.backflip' tab is selected and highlighted in yellow. Below the tabs, the main area is also yellow and contains the text 'heroine.backflip' followed by 'No parameters'. Below this, there is a section labeled 'No variables'. At the bottom, there are three script blocks for the 'heroine' sprite. The first block is 'move up 1 meter more...'. The second block is 'turn backward 1 revolution more...'. The third block is 'move down 1 meter more...'.

world.my first method heroine.backflip

heroine.backflip *No parameters*

No variables

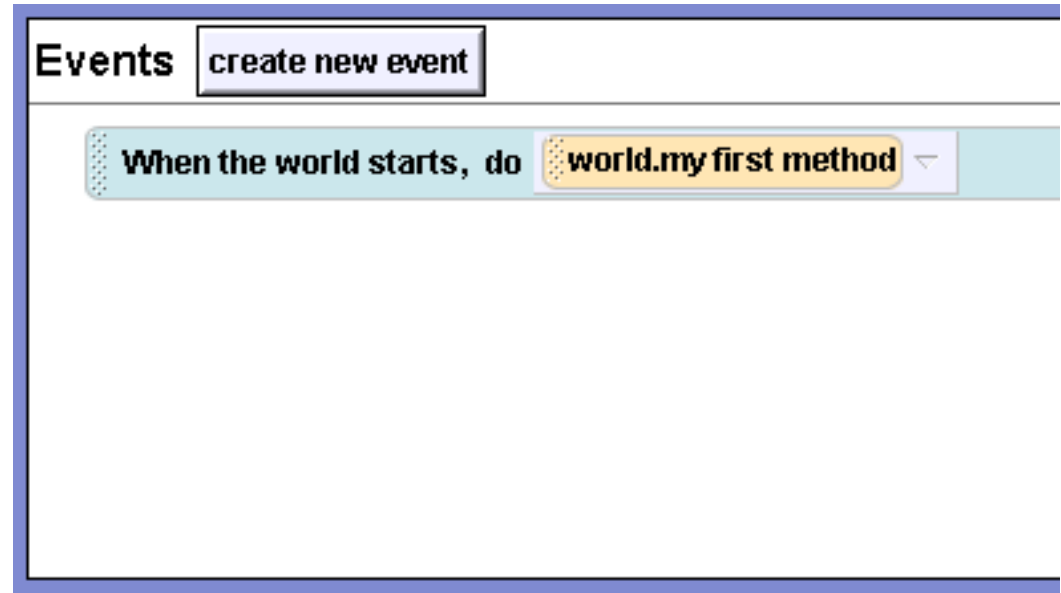
heroine ▾ move up ▾ 1 meter ▾ more... ▾

heroine ▾ turn backward ▾ 1 revolution ▾ more... ▾

heroine ▾ move down ▾ 1 meter ▾ more... ▾

Events

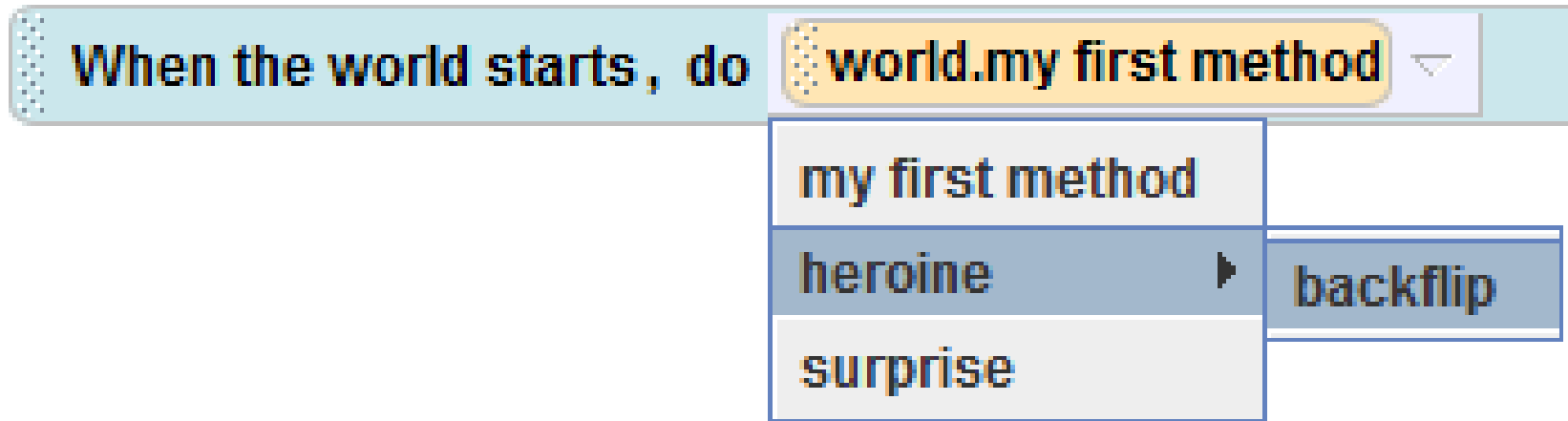
- Now that we have written a method, we'll use an **event** to see it in action.
- The **event editor** is found in the top right-hand corner of your screen.
- Events are used to tell our program when it should call our methods.




This is the event editor

Events

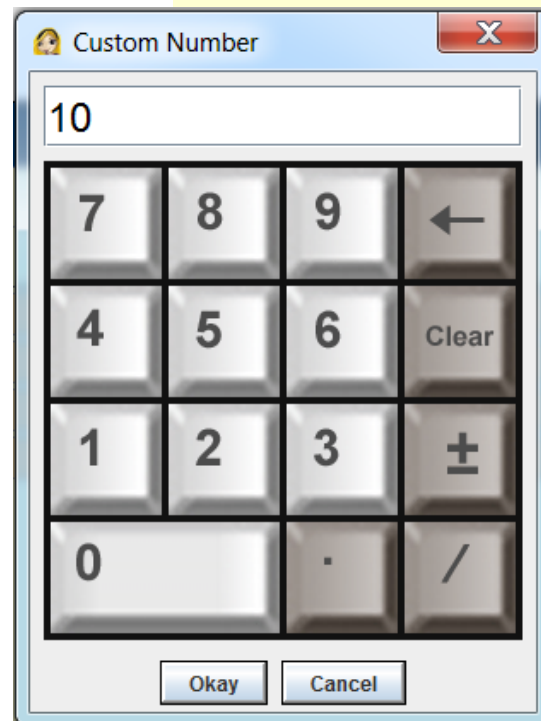
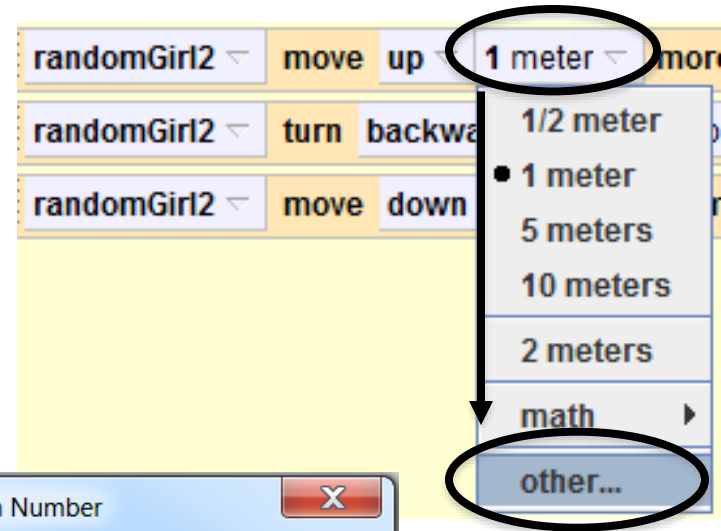
- You'll see that when the world starts (when you press Play), your world does world.my first method.
- Click where it says world.my first method, and change it to **your character > backflip**.



- Now when you press  in the upper left, your character will do a backflip!

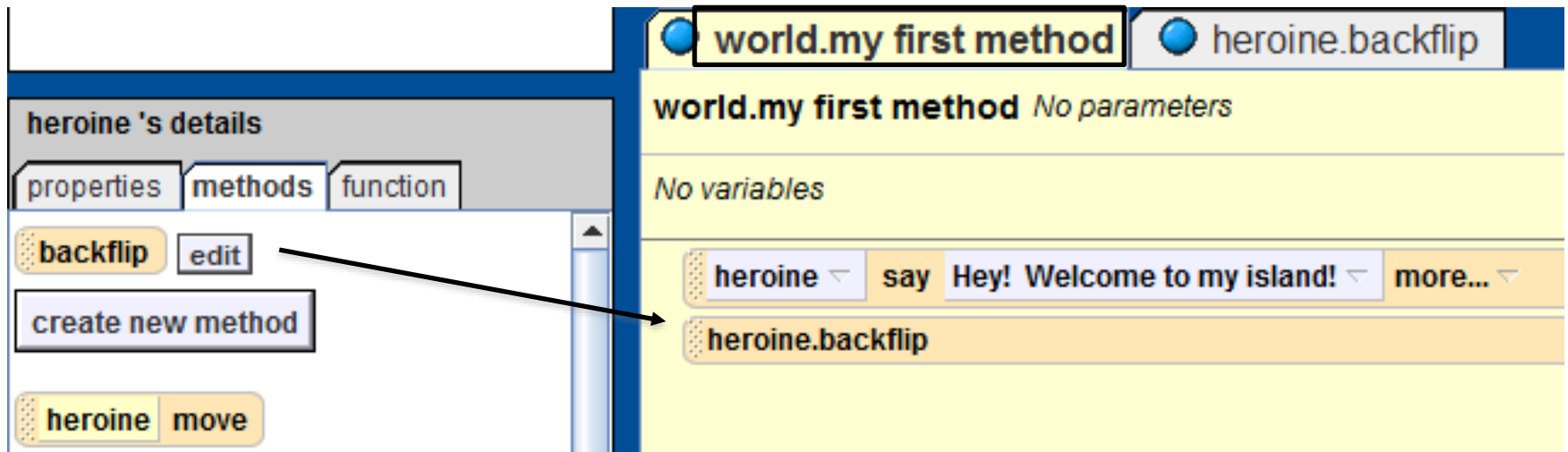
Backflip

- Depending on how tall your character is, he/she might have hit his/her head on the ground. To fix this, click **1 meter** next in the **move up** command, and choose **other...** Try typing **2** in the calculator that comes up to start.
- Be sure to adjust the **move down** command, too.



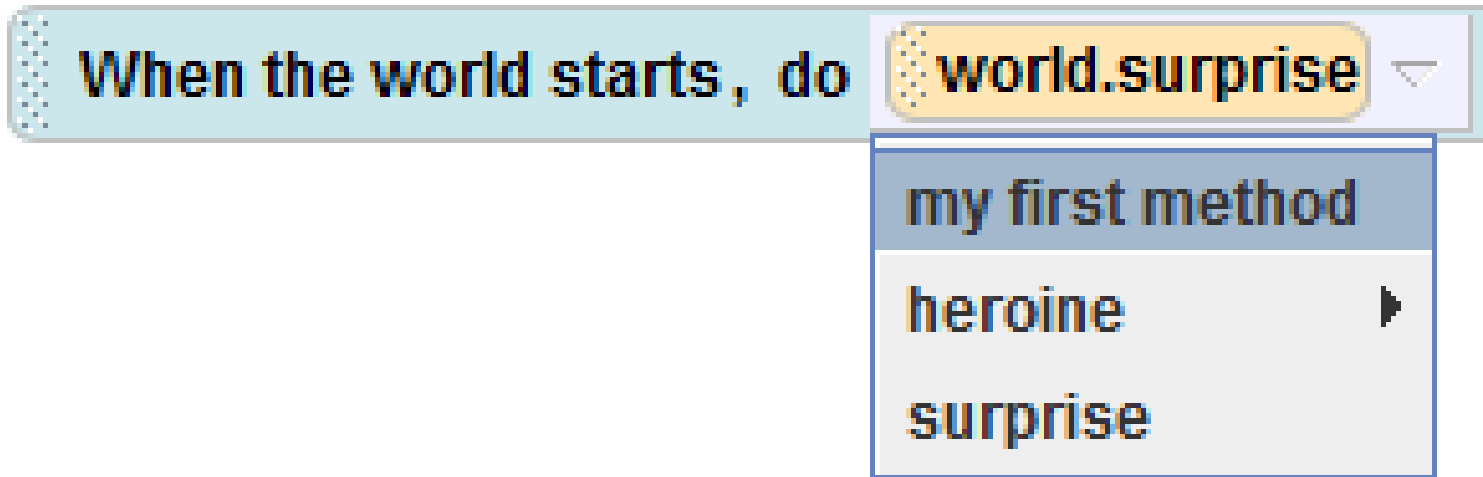
Backflip


- Finally, to make have our character say “hey,” and then do the backflip, select the **world.my first method** tab again
- Drag in **backflip** from your character’ s list of methods.



Testing!

- Now change your “When the world starts” event in the event editor *back* to world.my first method.



- We do all this because testing one method at a time will be a really good time-saving habit later.
- Now just press  to run the whole thing!

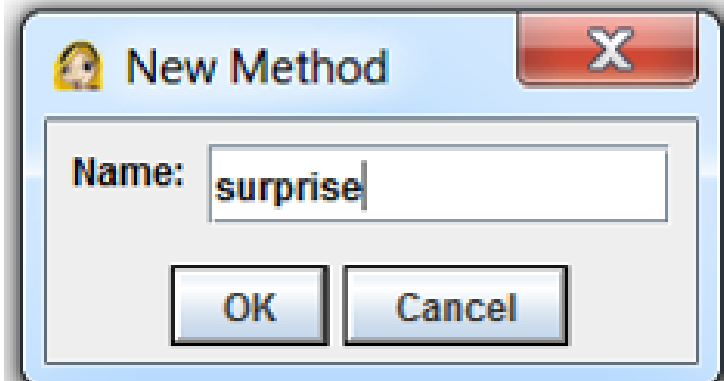
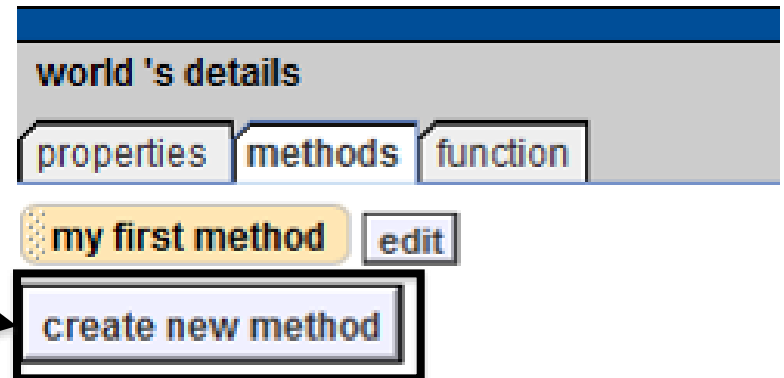
Part 4: A Surprise Method

- Now we're going to write a more complex method that will activate when we press a button
- First, drag in another **heroine say** to the bottom of World.my first method. Select other, then type **“Press S for a Surprise”** in the box that comes up
- Your method will look like this:

The image shows a screenshot of a Scratch-style method editor. At the top, there are two tabs: "world.my first method" (selected) and "heroine.backflip". Below the tabs, the "world.my first method" section is highlighted in yellow and contains the text "world.my first method" and "No parameters". Below this, it says "No variables". The main area shows a list of blocks. The first block is "heroine say Hey! Welcome to my island! more...". The second block is "randomGuy1.backflip". The third block is "heroine say Press S for a Surprise! more...".

Surprise

- Select **world** at the top of the object tree, then hit **create new method** in the details pane, and type **surprise** in the box
- We're creating our surprise method in world because we're going to have several different objects (your character, the shark, the camera) do things in one method!



Surprise

- S also stands for “Shark,” so we’ll have the camera get a look at the shark.
- To get the camera to change position *while your program runs*, we have to use its methods.
- Select **camera** in the object tree, then drag in its **set point of view to** method.
- Select **CameraViews** > **SHARKview** on the menu that comes up.

The screenshot displays a programming environment with three main panels:

- Object Tree (Top Left):** A list of objects including 'world', 'camera', 'light', 'ground', 'island', 'teddy', 'rowboat', 'shark', and 'cameraViews'. The 'camera' object is highlighted with a black box.
- Scene View (Top Right):** A 3D rendering of a scene with a blue sky, blue water, a yellow island with a palm tree, a person standing on the island, and a small boat. Navigation arrows and an 'ADD OBJECTS' button are visible at the bottom.
- Camera Details (Bottom):** A panel titled 'camera's details' with tabs for 'properties', 'methods', and 'function'. The 'methods' tab is active, showing a list of methods for the 'camera' object: 'turn to race', 'point at', 'set point of view to', 'set pose', 'stand up', 'move at speed', 'turn at speed', and 'roll at speed'. The 'set point of view to' method is highlighted with a black box. A dropdown menu is open below it, showing a list of objects: 'the entire world', 'camera', 'light', 'ground', 'island', 'teddy', 'rowboat', 'cameraViews', and 'shark'. The 'cameraViews' option is highlighted with a black box.

Surprise

- Your method editor should look like this:

The screenshot shows a method editor interface. At the top, there are three tabs: 'world.my first method', 'heroine.backflip', and 'world.surprise'. The 'world.surprise' tab is selected and highlighted in yellow. Below the tabs, the text 'world.surprise No parameters' is displayed. Underneath, it says 'No variables'. At the bottom, there is a control bar with a 'camera' dropdown menu, the text 'set point of view to', a 'SHARKview' dropdown menu, and a 'more...' dropdown menu.

world.my first method heroine.backflip **world.surprise**

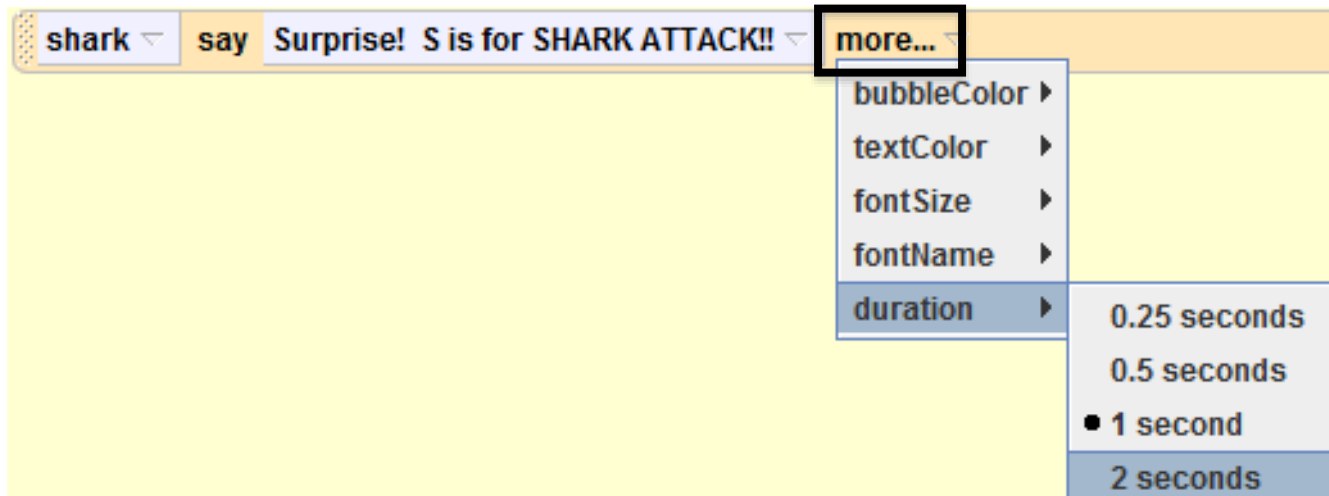
world.surprise *No parameters*

No variables

camera ▾ set point of view to SHARKview ▾ more... ▾

Surprise

- Now select **shark** in the object tree, and have it **say** “Surprise! ‘S’ is for SHARK ATTACK!!” or something similar (use a method)
- Click **more...** next to this in your method editor, and a menu of properties will appear. Select **duration**, and change it to **2 seconds** so people have more time to read the message
- **Play around** with the other properties if you have time, but **don’ t use fontName** (or your program will break).



Surprise

- Your method editor should look like this:

The screenshot shows a method editor interface with three tabs at the top: 'world.my first method', 'heroine.backflip', and 'world.surprise'. The 'world.surprise' tab is selected and highlighted in blue. Below the tabs, the method name 'world.surprise' is displayed in bold black text, followed by the text 'No parameters' in a lighter font. Below this, the text 'No variables' is displayed in a lighter font. The main area of the editor contains two rows of code blocks. The first row has a 'camera' dropdown menu, followed by the text 'set point of view to', a 'SHARKview' dropdown menu, and a 'more...' dropdown menu. The second row has a 'shark' dropdown menu, followed by the text 'say', the text 'Surprise! S is for Shark attack!', a dropdown menu, the text 'duration = 2 seconds', a dropdown menu, and a 'more...' dropdown menu.

world.my first method heroine.backflip **world.surprise**

world.surprise *No parameters*

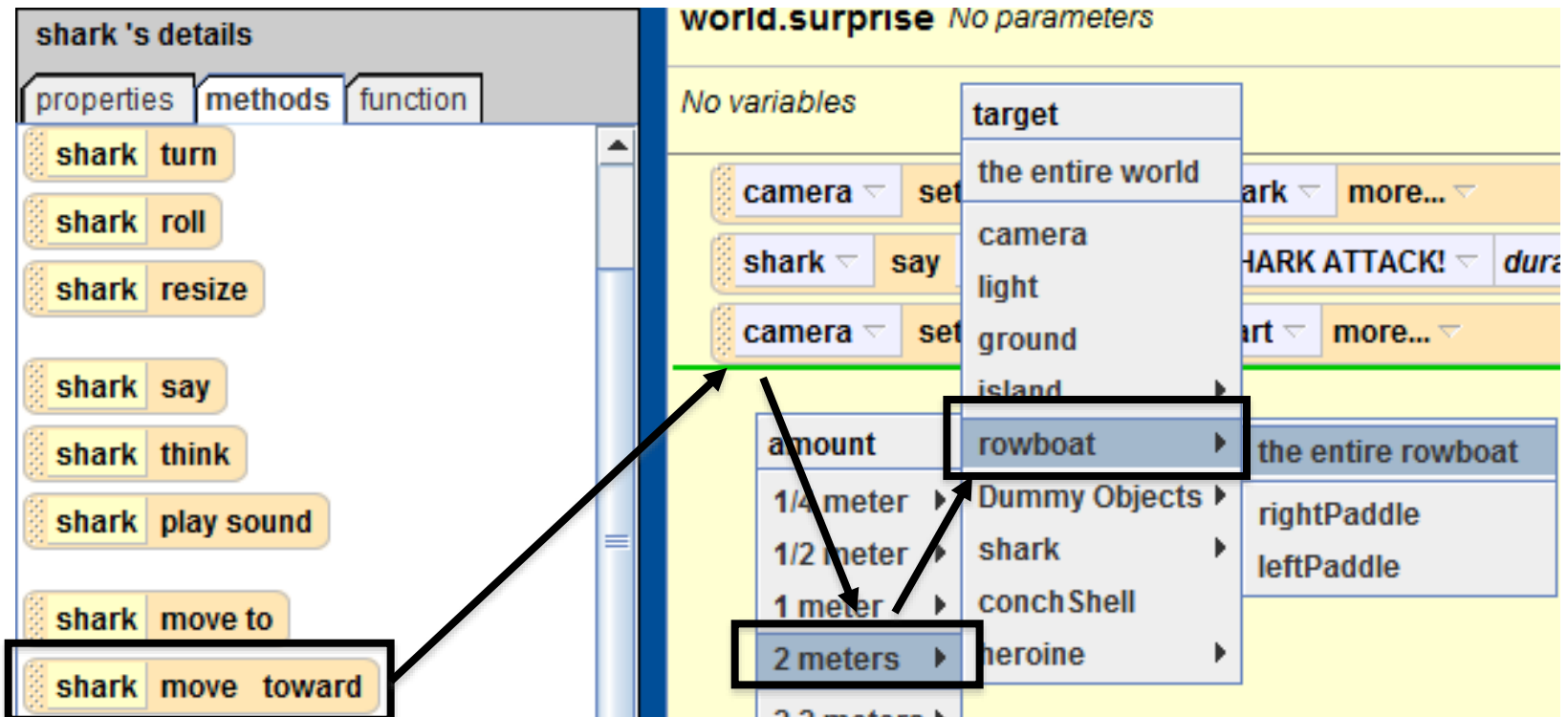
No variables

camera ▾ set point of view to SHARKview ▾ more... ▾

shark ▾ say Surprise! S is for Shark attack! ▾ *duration = 2 seconds* ▾ more... ▾

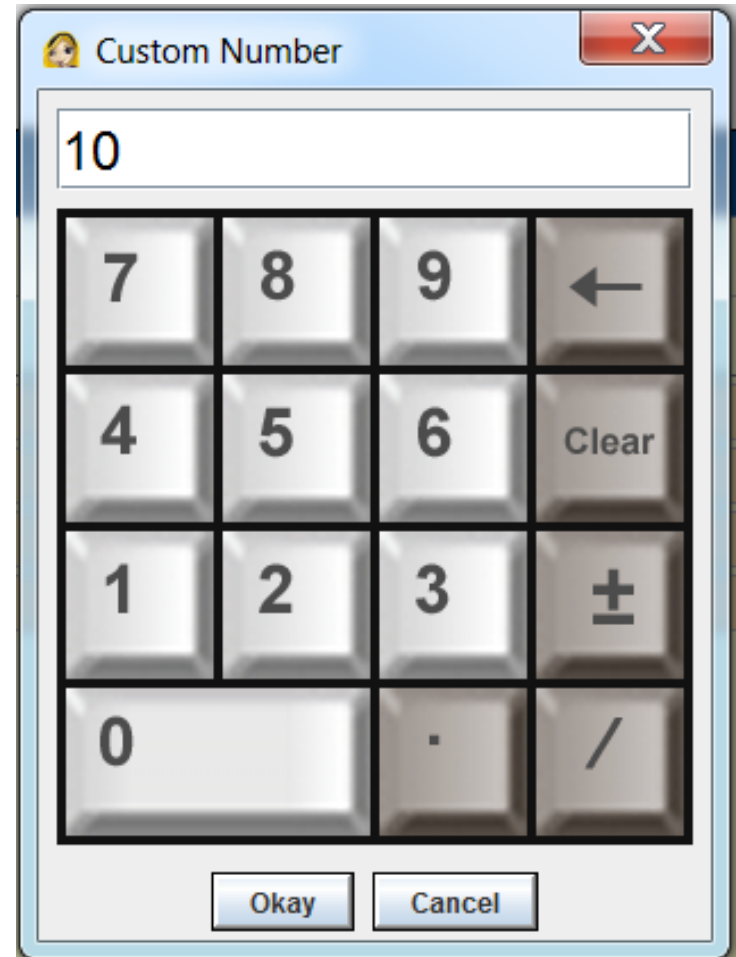
Surprise

- Now drag in a method to have the **camera set point of view to** the start position
- Then select **shark** in the object tree, drag in a **move towards** method, and choose **2 meters > rowboat > the entire rowboat**
- This will have the shark move 2 meters towards your boat.



Surprise

- Change the distance of your move towards method to how far away you think you estimate your shark is from the rowboat.
- To do so, click **amount=2 meters, other**, then type in **your guess** on the calculator
- Set the **duration** (under more...) of your **move towards** method to **5 seconds** so the user has time to react to the shark



Surprise

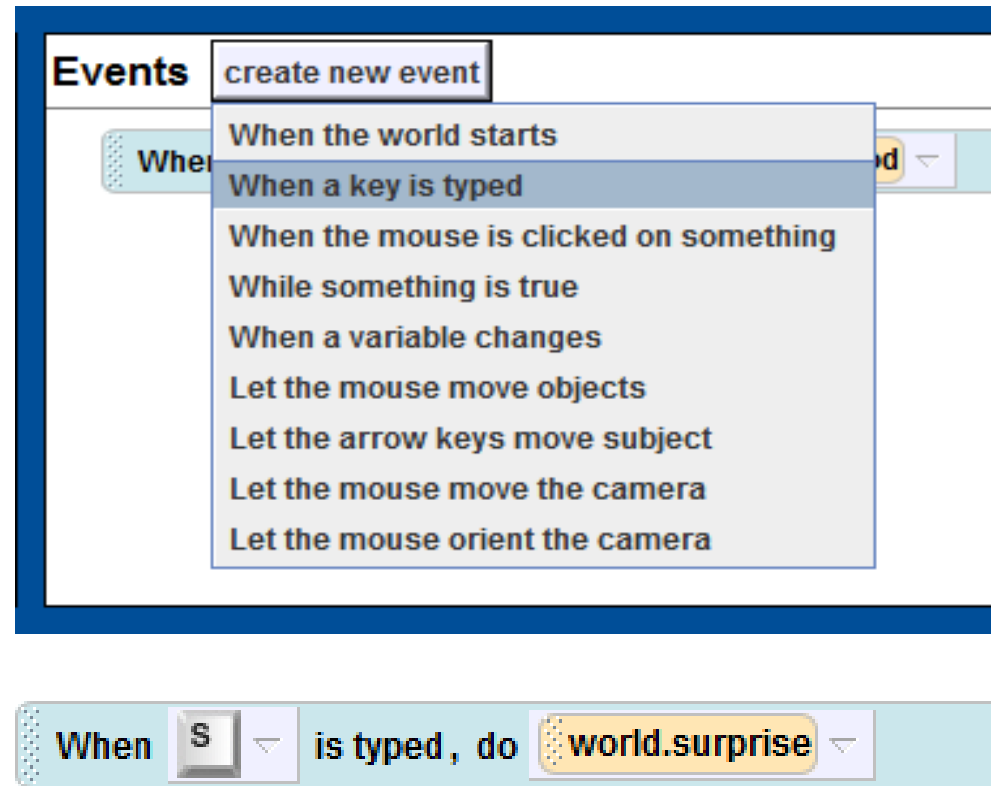
- Your method editor should look like this:

The screenshot shows a method editor interface. At the top, there are three tabs: 'world.my first method', 'heroine.backflip', and 'world.surprise'. The 'world.surprise' tab is selected and highlighted in blue. Below the tabs, the text 'world.surprise No parameters' is displayed. Underneath, it says 'No variables'. The main area contains four rows of code blocks, each with a dropdown menu on the left and a 'more...' dropdown on the right:

- camera ▾ set point of view to SHARKview ▾ more... ▾
- shark ▾ say Surprise! S is for Shark attack! ▾ duration = 2 seconds ▾ more... ▾
- camera ▾ set point of view to STARTview ▾ more... ▾
- shark ▾ move amount = 10 meters ▾ toward target = rowboat ▾ duration = 5 seconds ▾ more... ▾

Events

- We want to make S actually prompt the surprise method.
- Select **create new event** in the event editor (top right), then choose **When a key is typed**.
- A new event will appear!
- Change **any key** to **letters > S** and **Nothing** to **surprise**.
- Note that this call surprise ANYTIME you press S, but your users will only know to do so when you instruct them.



Test

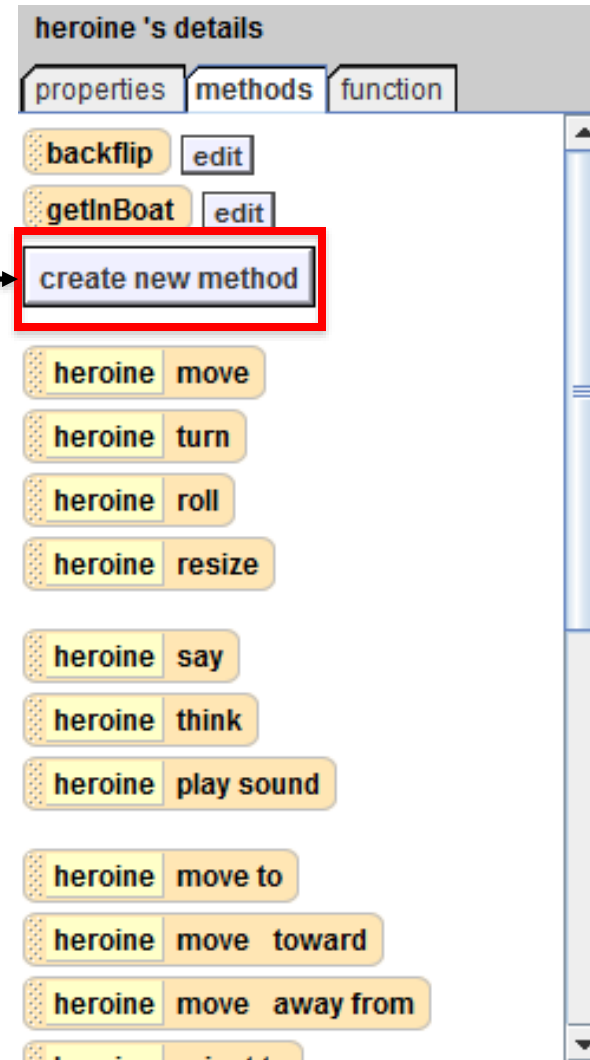
- Now press **Play** to test your world!
- Your shark should stop just in front of the boat, but because we just guessed his distance to the boat, he probably won't. Play with the **distance** in your **move toward** call until your shark ends up like this:



- For me, the distance was 4 meters.
- *Those seeking a challenge can try to figure out how to use the `distanceTo` **function** (and it's **math** parameter) here to get it exact*

Part 5: Vehicles

- While the shark can't exactly walk on land and attack our character, we're going to have him/her escape in the rowboat anyway
- Click on your **character** in the object tree, then create a new method. Call it getInBoat



Vehicles

- Now click on the **properties** tab in the details pane (where the methods are)
- The properties pane lets you change various things about your object.
- Later, you'll be able to make your character red, radioactive, or invisible
- For now, look at the **vehicle** property

heroine's details

properties methods function

create new variable

capture pose

color =

opacity = 1 (100%) ▾

vehicle = world ▾

skin texture = heroine.texture ▾

fillingStyle = solid ▾

pointOfView = position: 1.46, 0.89, -3.9

isShowing = true ▾

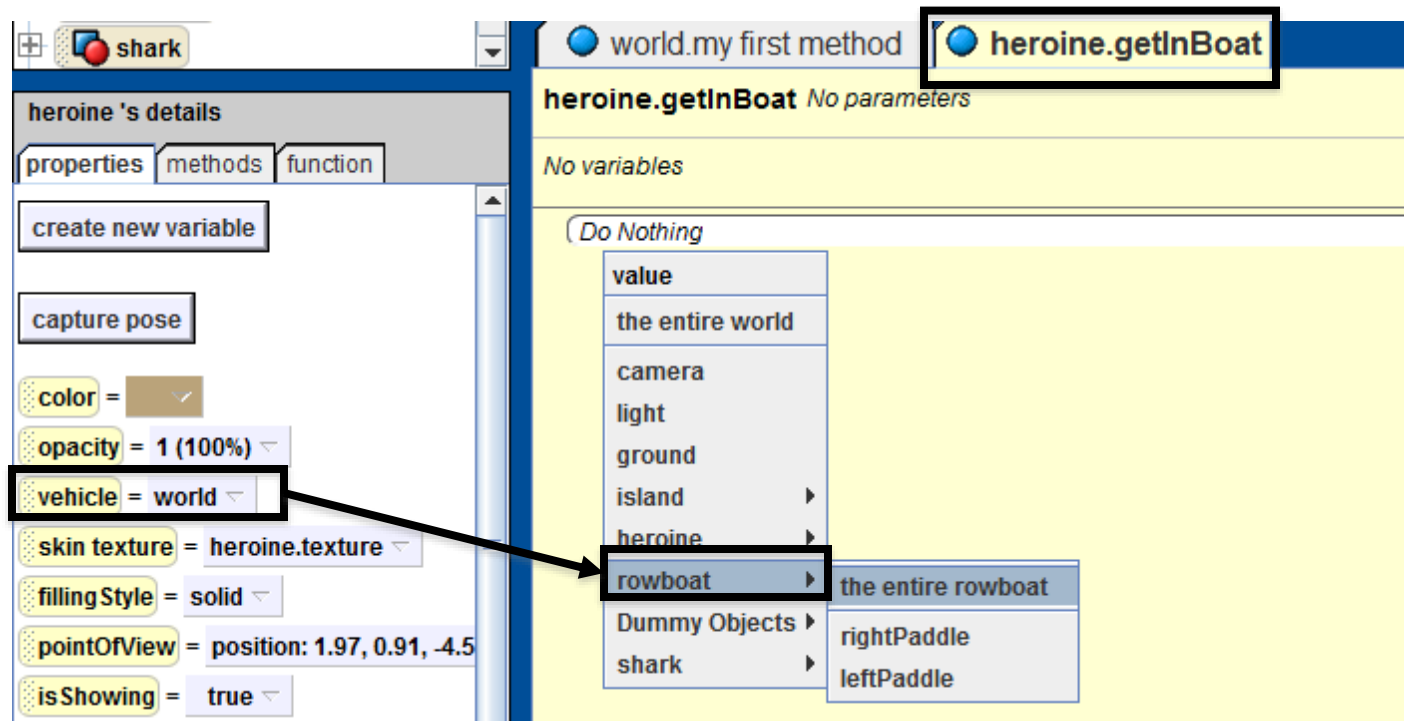
+ Seldom Used Properties

+ Sounds

+ Texture Maps

modeled by: Zach Beard
painted by: Sylvia Chen
depth: 0.263

Vehicles



- Drag the **vehicle** property into the method editor (make sure the new **getInBoat** tab is selected) and select **rowboat > the entire rowboat**.
- Now when the rowboat moves, your character will move with it; your character is riding the boat.

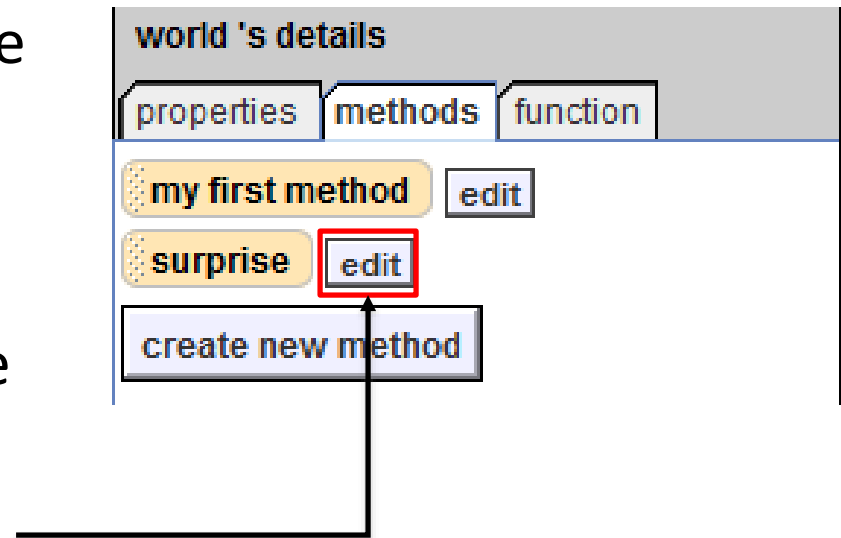
Vehicles

- Repeat the steps from the last two slides but this time set your **camera's** vehicle to the rowboat
- Now your *camera* will follow the boat when you ride around
- Your method should look like this:

The screenshot shows the Scratch IDE interface for editing the `heroine.getInBoat` method. The top bar displays two tabs: `world.my first method` and `heroine.getInBoat`. Below the tabs, the method name `heroine.getInBoat` is shown with the note *No parameters*. To the right of this is a button labeled `create new parameter`. Below the method name, the text *No variables* is displayed, with a `create new variable` button to its right. The main workspace contains two code blocks. The first block is for the `heroine` object, with the text `set vehicle to rowboat` followed by a `more...` dropdown. The second block is for the `camera` object, also with the text `set vehicle to rowboat` followed by a `more...` dropdown.

Vehicles

- We need to do two more things for this method to work
- First, we need to *call* (use) the method that we just made
- Go to your **world.surprise** method. If there isn't a tab above the method editor, then click on **world** in the object tree, then click **edit** next to **surprise**.



Vehicles

- Select your character in the object tree, and drag your new method, **getInBoat** into your **surprise** method.

The screenshot shows a coding environment for a 'world.surprise' method. At the top, it says 'world.surprise No parameters' with a 'create new parameter' button. Below that, it says 'No variables' with a 'create new variable' button. The main area contains five steps in a list:

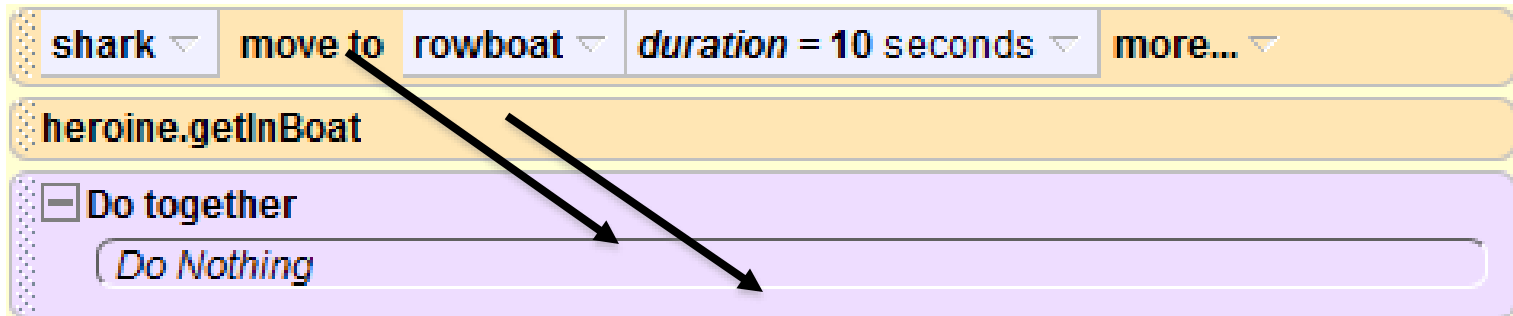
- camera ▾ set point of view to Shark ▾ more... ▾
- shark ▾ say Surprise! S is for SHARK ATTACK!! ▾ more... ▾
- camera ▾ set point of view to start ▾ more... ▾
- shark ▾ move to rowboat ▾ duration = 10 seconds ▾ more... ▾
- heroine.getInBoat

At the bottom, there is a palette of coding constructs: 'Do in order', 'Do together', 'If/Else', 'Loop', 'While', 'For all in order', 'For all together', 'Wait', 'print', and '//'. The 'Do together' block is highlighted with a black box, and an arrow points from it to the 'heroine.getInBoat' step in the list above.

- Now, from the bottom of the screen, where there are several advanced (but common) coding constructs, drag in a **do together**.

Do Together

- Normally, actions in Alice take place *in order*. One line of code will run, then the next one.
- By putting multiple methods inside of a *do together*, those methods can happen at the same time
- Drag our last two lines into the do together



- Otherwise, our character would wait until AFTER the shark attacked to get in the boat

Vehicles

Your surprise method should look like this:

world.surprise *No parameters* create new parameter

No variables create new variable

camera ▾ set point of view to Shark ▾ more... ▾

shark ▾ say Surprise! S is for SHARK ATTACK!! ▾ more... ▾

camera ▾ set point of view to start ▾ more... ▾

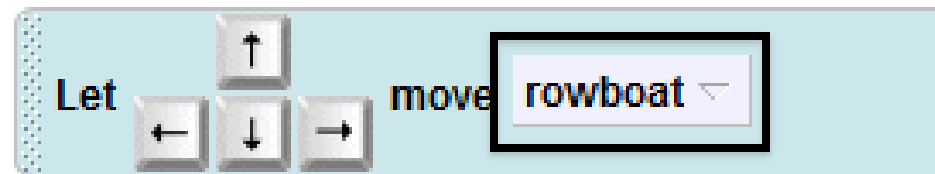
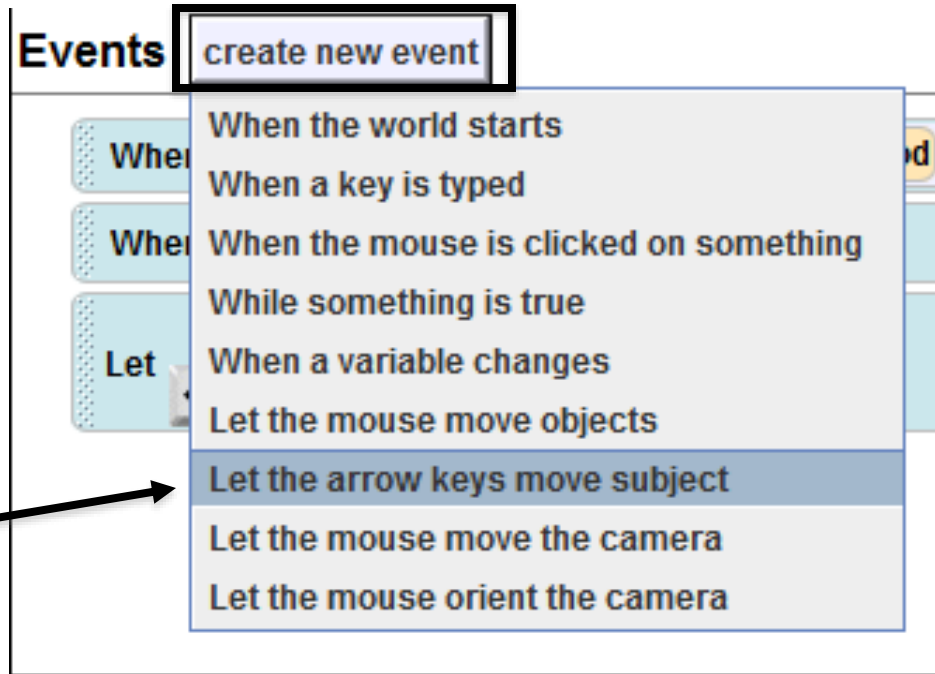
Do together

shark ▾ move to rowboat ▾ duration = 10 seconds ▾ more... ▾

heroine.getInBoat

Vehicles

- Finally, lets make the arrow keys move the rowboat
- Click **create new event** in the event editor (top right)
- Choose **Let the arrow keys move subject**
- Change **camera** to **rowboat > the entire rowboat**



Vehicles

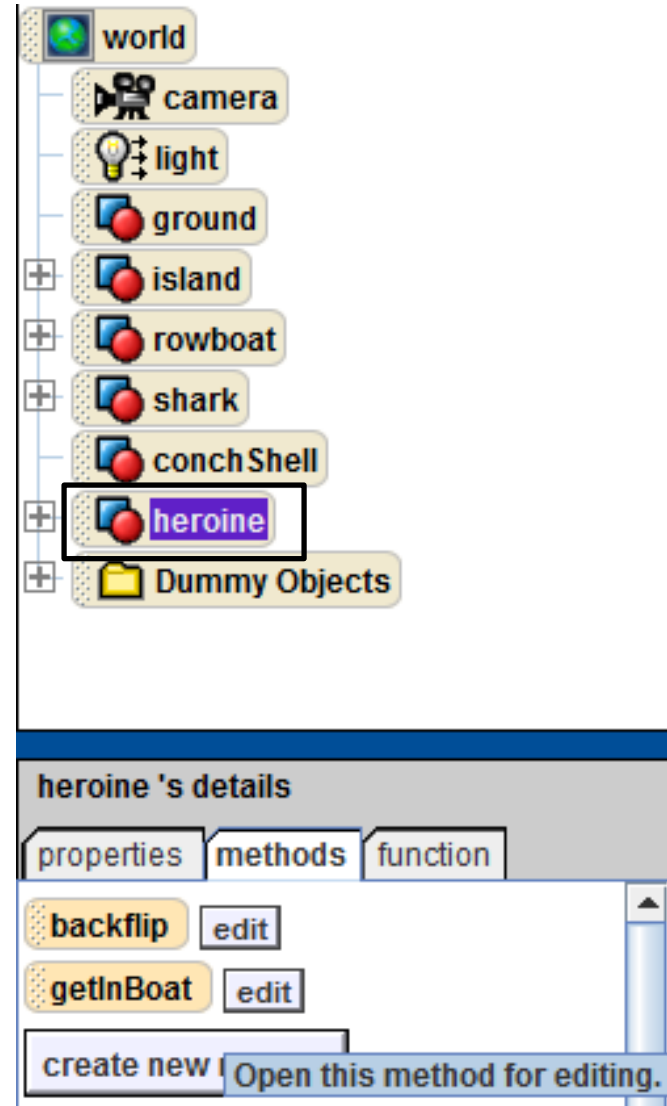
- **Play** your world! You should notice a problem:



- Your character *moves with* the boat, but he/she isn't *in* the boat

Vehicles

- We'll fix this problem in the `character.getInBoat` method.
- Open this method by selecting your **character** from the object tree and clicking **edit** next to **`getInBoat`**.
- A tab will come up as selected in your method editor for the **`getInBoat`** method



Vehicles

- Drag in a **move to**, and select **rowboat > the entire rowboat** (to move your character to the boat)
- Drag in an **orient to** (scroll down), and select **rowboat > the entire rowboat** (to face the front of the boat)
- Now drag in a **do together**, and move everything inside

The screenshot shows a Scratch script editor interface. At the top, a yellow box contains the text **heroine.getInBoat** with *No parameters* to its right and a **create new parameter** button. Below this, another yellow box contains *No variables* and a **create new variable** button. The main area is a purple box labeled **Do together** with a minus sign icon on the left. Inside this box, four yellow action blocks are stacked vertically: 1. **heroine** (with a dropdown arrow) **set vehicle to** **rowboat** (with a dropdown arrow) **more...** (with a dropdown arrow). 2. **camera** (with a dropdown arrow) **set vehicle to** **rowboat** (with a dropdown arrow) **more...** (with a dropdown arrow). 3. **heroine** (with a dropdown arrow) **move to** **rowboat** (with a dropdown arrow) **more...** (with a dropdown arrow). 4. **heroine** (with a dropdown arrow) **orient to** **rowboat** (with a dropdown arrow) **more...** (with a dropdown arrow).

Vehicles

- Drag in a **say** above the do together, and have your character say something like “Oh no! Use the arrow keys to help me escape!” so that the person using your world knows what to do.

The screenshot shows a Scratch script editor with a yellow background. At the top, a script block for `heroine.getInBoat` is highlighted with a black border. Below it, there are two buttons: "create new parameter" and "create new variable". The script block contains a "say" block with the text "Oh no! Use the arrow keys to help me escape!" and a "do together" block. The "do together" block contains four sub-blocks: "heroine set vehicle to rowboat", "camera set vehicle to rowboat", "heroine move to rowboat", and "heroine orient to rowboat".

heroine.getInBoat *No parameters* [create new parameter](#)

No variables [create new variable](#)

heroine ▾ say Oh no! Use the arrow keys to help me escape! ▾ more... ▾

[-] Do together

- heroine ▾ set vehicle to rowboat ▾ more... ▾
- camera ▾ set vehicle to rowboat ▾ more... ▾
- heroine ▾ move to rowboat ▾ more... ▾
- heroine ▾ orient to rowboat ▾ more... ▾

Vehicles

- Finally, drag in a **move** below the do together, select **down > ½ meter**.
- This will make your character sit more realistically in the boat. The exact distance (or even direction!) may be different for different characters – experiment!

The screenshot shows a Scratch script editor with the following elements:

- A yellow header bar containing the text `heroine.getInBoat` (highlighted with a black box) and *No parameters*, and a button labeled "create new parameter".
- Below the header, the text *No variables* is displayed, along with a button labeled "create new variable".
- The script area contains a sequence of actions:
 - A "say" block: `heroine` (dropdown), `say` (action), `Oh no! Use the arrow keys to help me escape!` (text), and `more...` (dropdown).
 - A "Do together" block (purple background) containing four sub-actions:
 - `heroine` (dropdown), `set vehicle to` (action), `rowboat` (dropdown), and `more...` (dropdown).
 - `camera` (dropdown), `set vehicle to` (action), `rowboat` (dropdown), and `more...` (dropdown).
 - `heroine` (dropdown), `move to` (action), `rowboat` (dropdown), and `more...` (dropdown).
 - `heroine` (dropdown), `orient to` (action), `rowboat` (dropdown), and `more...` (dropdown).
 - A "move" block: `heroine` (dropdown), `move` (action), `down` (direction dropdown), `0.5 meters` (distance dropdown), and `more...` (dropdown).

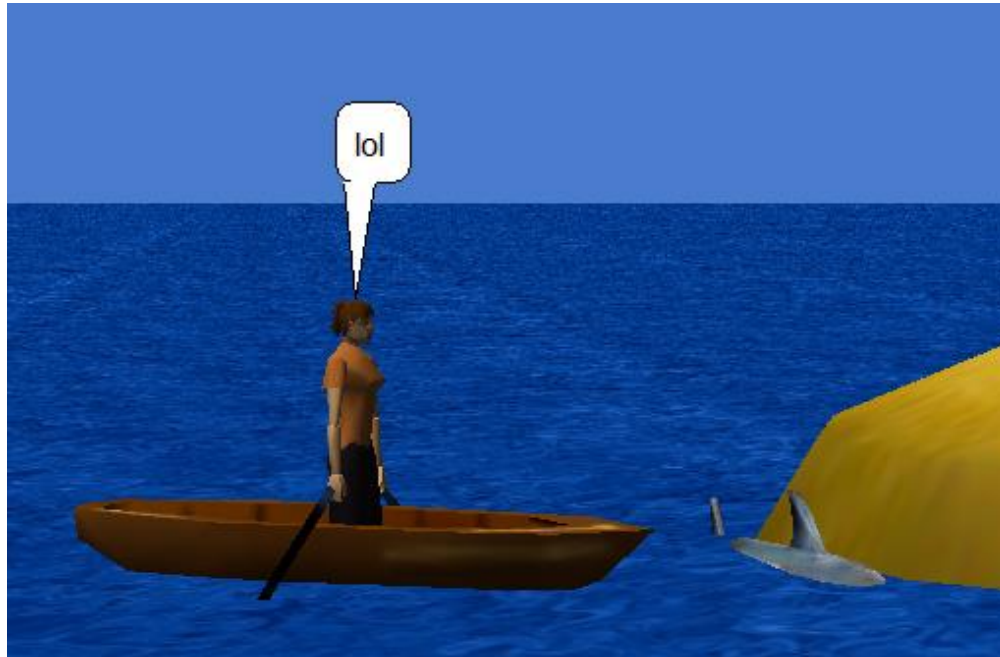
Test

Play your world! The problem should be fixed.



Circling

- Right now, the shark just sits by the beach like a dork after you ride your boat away



- We' ll have it swim circles around the island instead to keep things exciting

Circling

- Open up the **world.surprise** method via the object tree.
- Click on **shark** in the object tree, and drag **shark think** into the method editor. Select **other**, and type in something like “Rats. I’ ll just have to wait here.”

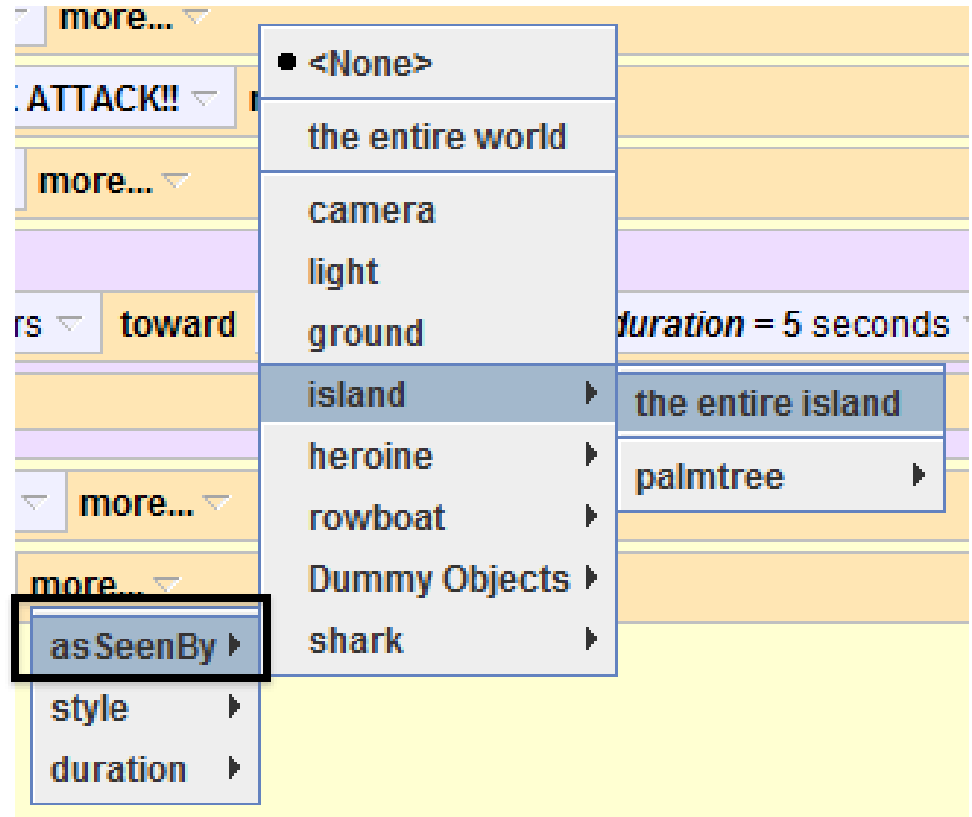
The screenshot shows a method editor for the **world.surprise** method. The editor has a yellow background and contains several blocks. At the top, there are two buttons: "create new parameter" and "create new variable". Below these are several blocks:

- A block for "camera" with "set point of view to" set to "Shark" and "more..."
- A block for "shark" with "say" set to "Surprise! S is for SHARK ATTACK!!" and "more..."
- A block for "camera" with "set point of view to" set to "start" and "more..."
- A purple "Do together" block containing:
 - A block for "shark" with "move" set to "amount = 10 meters", "toward" set to "target = rowboat", "duration = 5 seconds", and "more..."
 - A block for "heroine.getInBoat"
- A block for "shark" with "think" set to "Rats. I'll just wait here." and "more..."

The "shark think" block at the bottom is highlighted with a black border. The "world.surprise" label at the top left is also highlighted with a black border.

Circling

- Drag in **shark turn**, then select **left > other > 10 revolutions**.
- Under **more...** choose **asSeenBy > island > the entire island**.
- This will make the shark turn around the island instead of just spinning in circles
- Change the **duration** to **50 seconds**



Circling

- Play your world.
- If the shark circles very close to the island, move your boat offshore a bit in the viewer (why?)
- If the shark doesn't circle the island, copy the code below (note your mistakes, though!)

camera ▾ set point of view to SHARKview ▾ more... ▾

shark ▾ say Surprise! S is for Shark attack! ▾ duration = 2 seconds ▾ more... ▾

camera ▾ set point of view to STARTview ▾ more... ▾

Do together

shark ▾ move amount = 4 meters ▾ toward target = rowboat ▾ duration = 5 seconds ▾ more... ▾

heroine.getInBoat

shark ▾ think Rats. I'll just wait here. ▾ more... ▾

shark ▾ turn left ▾ 10 revolutions ▾ asSeenBy = island ▾ duration = 50 seconds ▾ more... ▾

Comments

- Now note the **double slash** on the far right side of the line of coding constructs at the bottom of your editor.
- This icon lets you add *comments* to your code that won't affect the program when it actually runs.
- Drag and drop one anywhere in your code and write an explanation for the line below it. An example:

The screenshot shows a code editor interface with a yellow background. At the top, a comment is added to a code block: `// You escaped the shark attack, so he'll just swim around the island`. Below this, there are two code blocks. The first block contains: `shark think Rats. I'll just wait here. more...`. The second block contains: `shark turn left 10 revolutions asSeenBy = island duration = 50 seconds style = abruptly`. At the bottom of the editor, there is a toolbar with various icons for code constructs: `Do in order`, `Do together`, `If/Else`, `Loop`, `While`, `For all in order`, `For all together`, `Wait`, `print`, and a comment icon (two slashes). A black arrow points from the comment icon in the toolbar to the comment text in the code block above.

Notes for later

- If you later want to unglue your character from the rowboat, set the **vehicle** property of your character from rowboat back to **world** (be sure to do the same thing for **camera**!)

The image shows a software interface with two main panels. The left panel, titled "heroine 's details", has tabs for "properties", "methods", and "function". Under the "properties" tab, there are several settings: "create new variable", "capture pose", "color" (set to a brown swatch), "opacity" (set to 1 (100%)), "vehicle" (set to "world" and highlighted with a black box), "skin texture" (set to "heroine.texture"), "filling Style" (set to "solid"), "pointOfView" (set to "position: 1.97, 0.91, -4.5"), and "is Showing" (set to "true"). There are also expandable sections for "Seldom Used Properties" and "Sounds". The right panel shows a script titled "heroine.getInBoat No parameters" with "No variables" listed below. The script contains a "say" block with the text "Oh no! Use the arrow", followed by a "Do together" block containing four actions: "heroine set vehicle to rowboat", "camera set vehicle to rowboat", "heroine move to rowboat", and "heroine orient to rowboat". Below the "Do together" block is a "move" block: "heroine move down 0.5 meters". At the bottom of the right panel, a "value" dropdown menu is open, showing "the entire world" (highlighted with a black box), "camera", and "light". A black arrow points from the "vehicle = world" property in the left panel to the "the entire world" option in the dropdown menu.

Congratulations! You have just made your first Alice world. There are many more things that you can do with Alice, so keep exploring it!



Extensions (in order of awesomeness)

- Add more scenery.
- Make the coconuts fall from the tree when you press spacebar.
- Make your boat your **favorite color** (hint: you'll have to change the color AND texture).
- Figure out how to change the weather; make a foggy sunset.
- Check out the “functions” tab (next to “methods” and “properties”) and do something interesting with them.
- Download Alice **Version 2.3** for free from alice.org
- Dream up many more worlds, where magical, impossible things happen. If you ever need help, try out some more try out some more tutorials on [Duke's Alice webpage](#).

“Its important to have specific dreams. Dream Big. Dream without fear.”

- Randy Pausch, creator of Alice