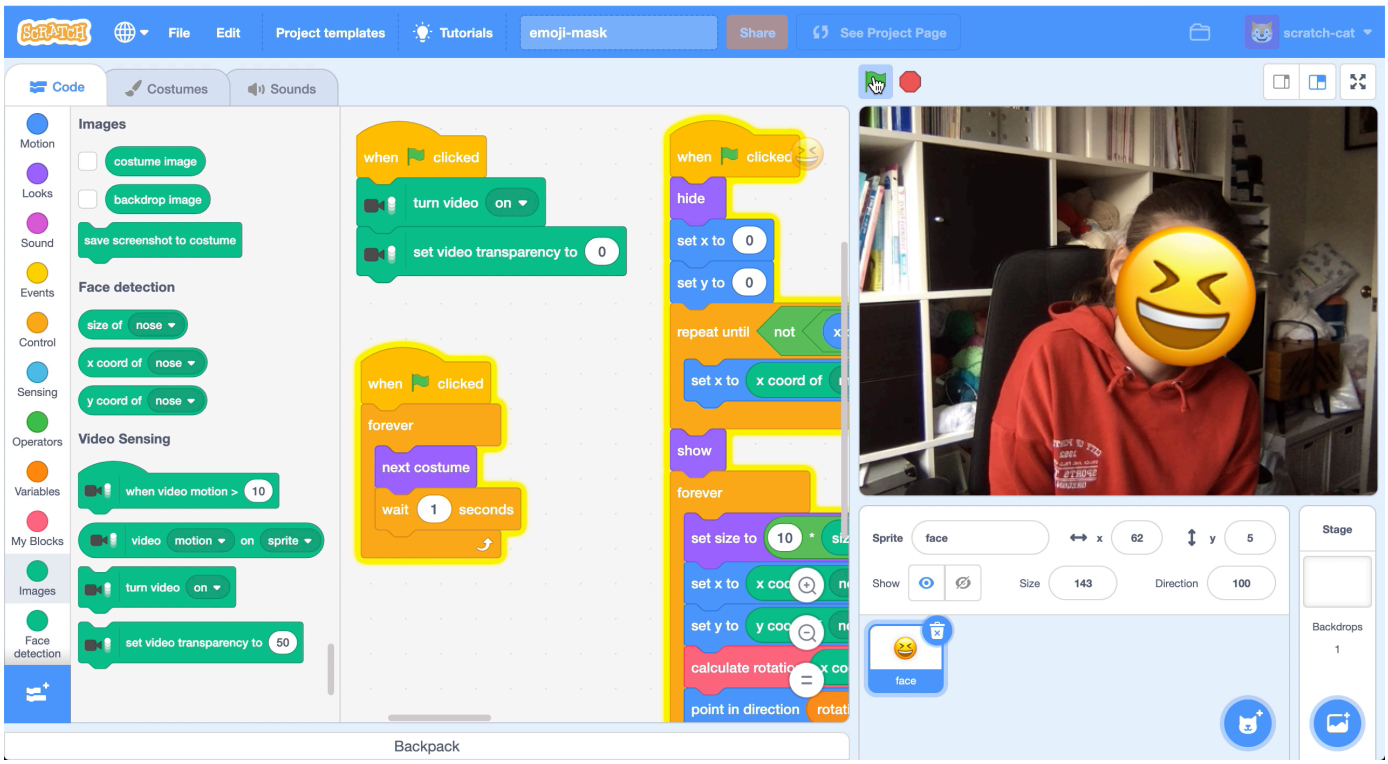
Emoji Mask

In this project you will make an AI-powered face filter that adds a cartoon mask to your face.

You will use a pre-trained machine learning model to perform face detection on a live webcam video, and code animated effects using the results.



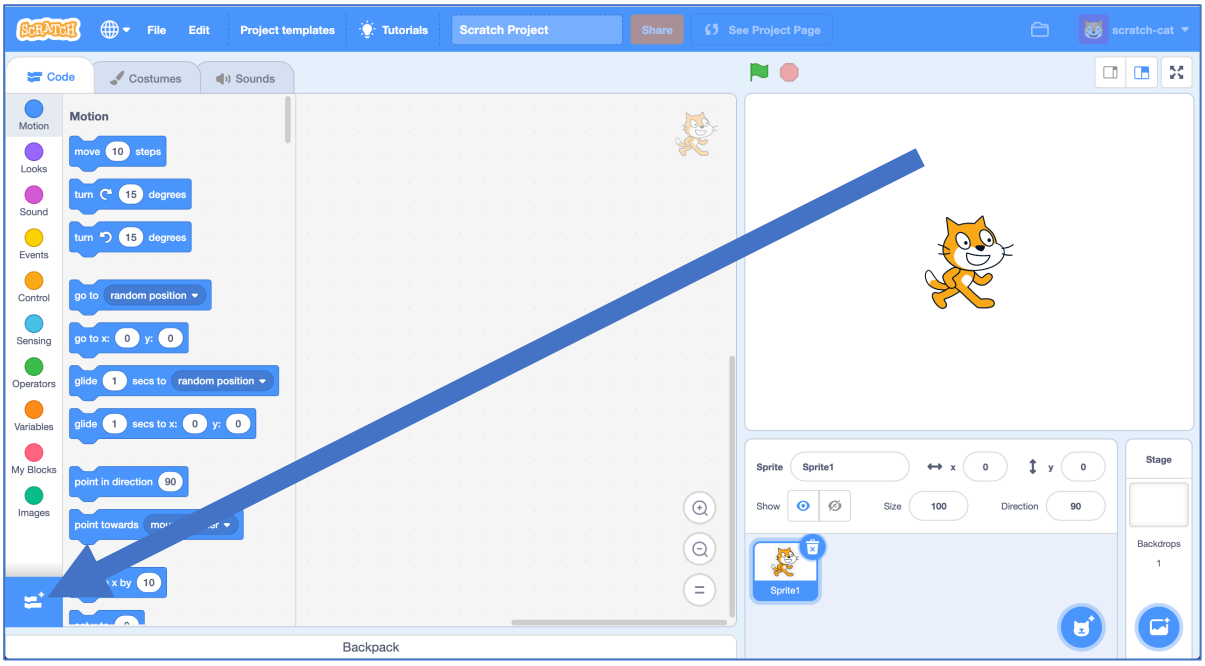
1. Go to https://machinelearningforkids.co.uk/pretrained/ in a web browser

This page displays some of the pretrained machine learning models that are available to you. For this project, we'll be using the Face Detection model.

2. Click on "**Get started**"

3. Open the **Extensions** window

Click on the blue button with the plus icon in the bottom left.



4. Click on the **Video Sensing** extension

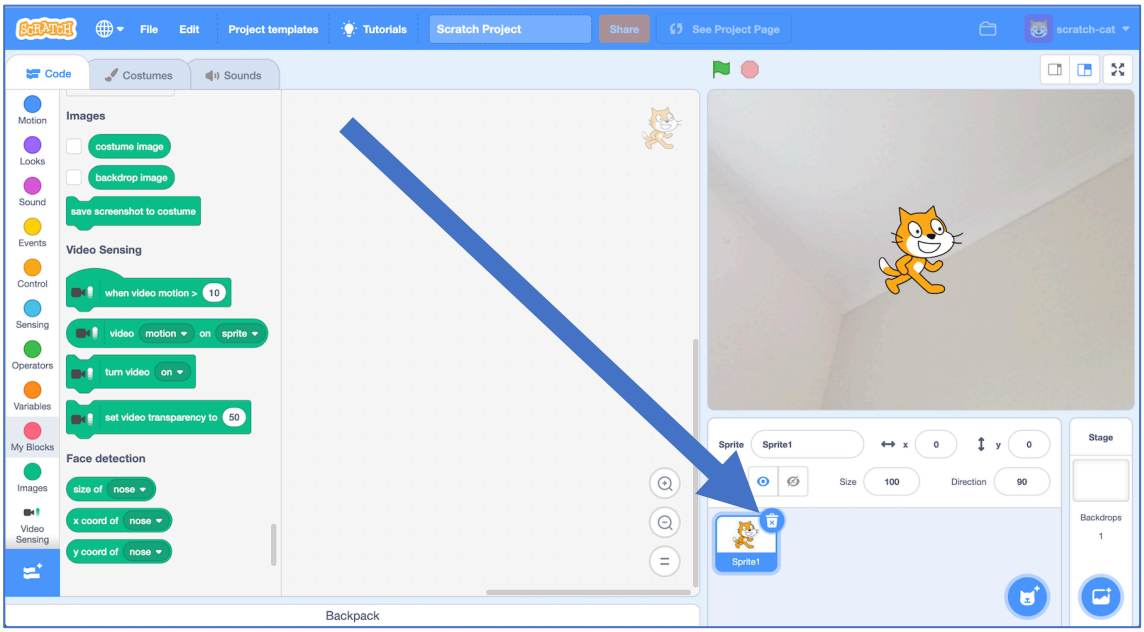
You will need this extension to use the webcam in your project.

5. Open the **Extensions** window again

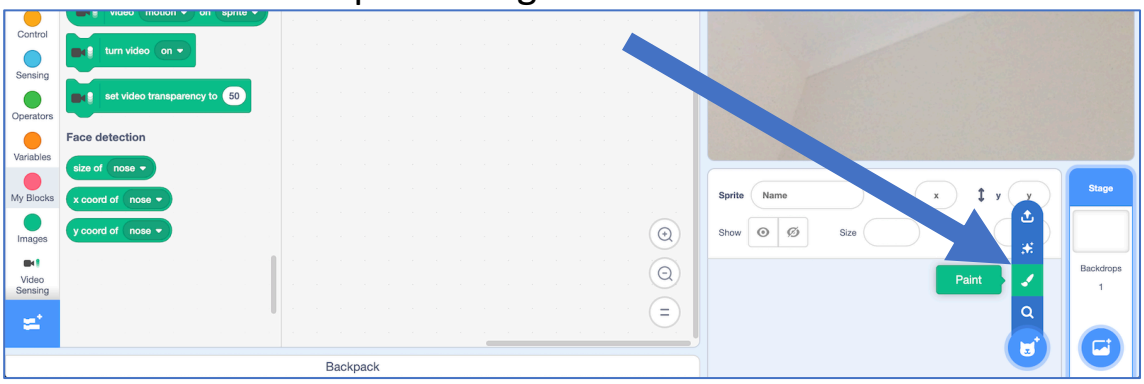
6. Click on the **Face detection** extension

You will need this extension to use the pre-trained machine learning model that identifies the location of your face in the webcam feed.

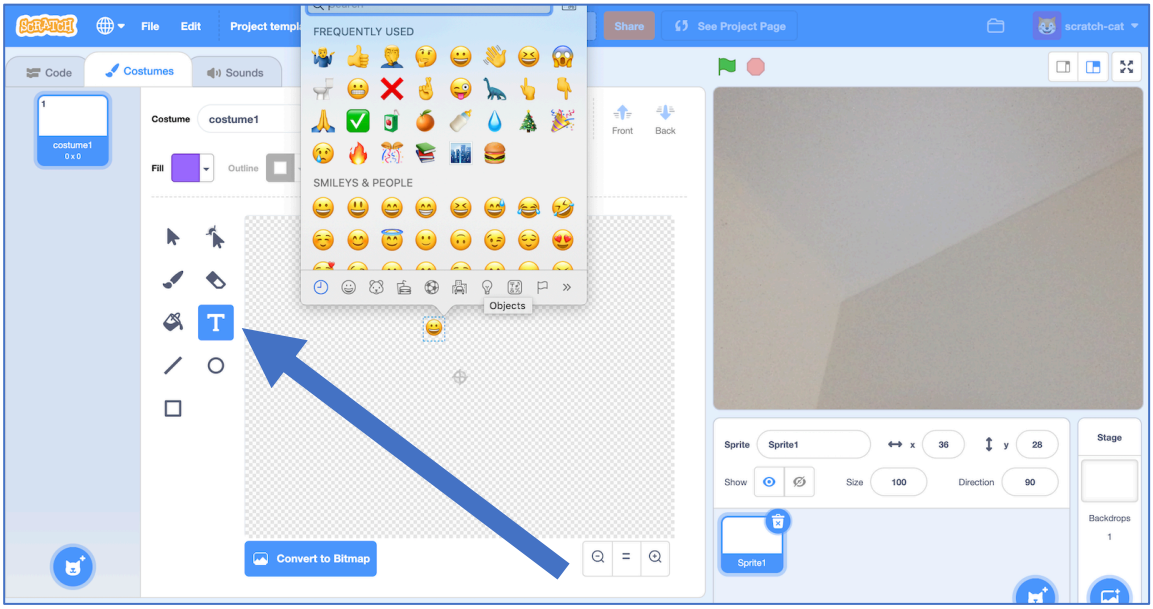
7. Delete the cat sprite by clicking on the trash can icon



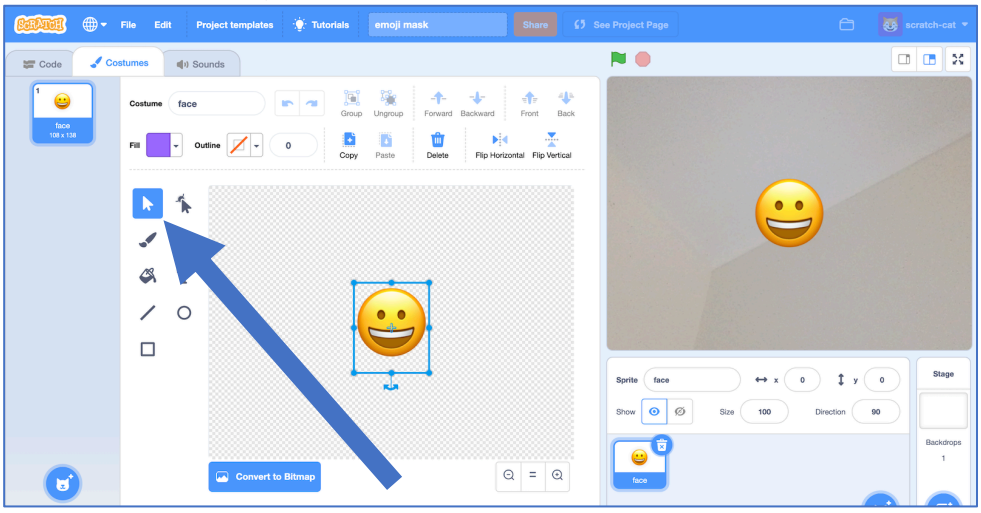
8. Create a new sprite using the **Paint** brush button



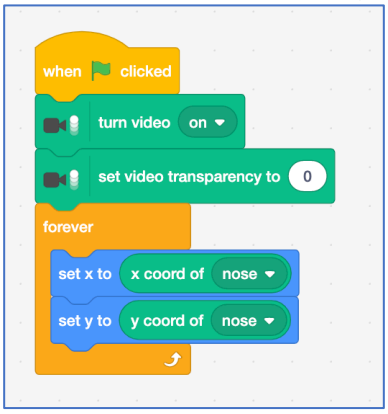
9. Click on the **T** text tool and enter a face emoji



10. Use the arrow tool to drag the face to be larger and move it to the middle of the sprite canvas



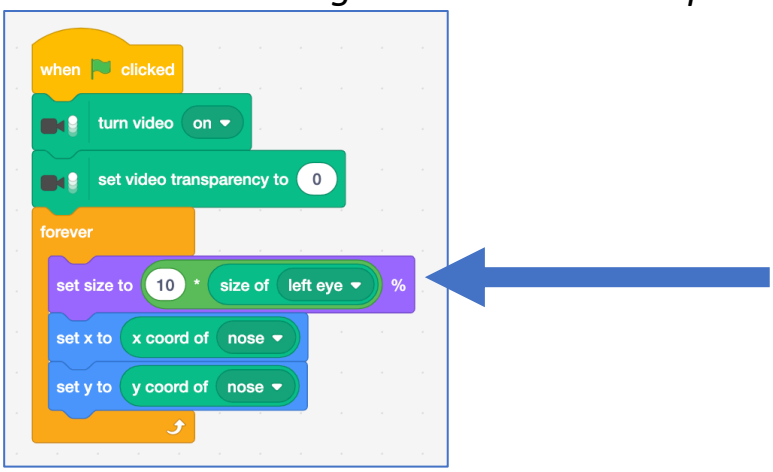
11. Create this script



12. It's time to test! Click on the **Green Flag**.

13. Update the script to adjust the size to match the size of your face

You'll need to change the number 10 depending on the size of your sprite



14. Click on the **Green Flag** to test again

What have you done so far?

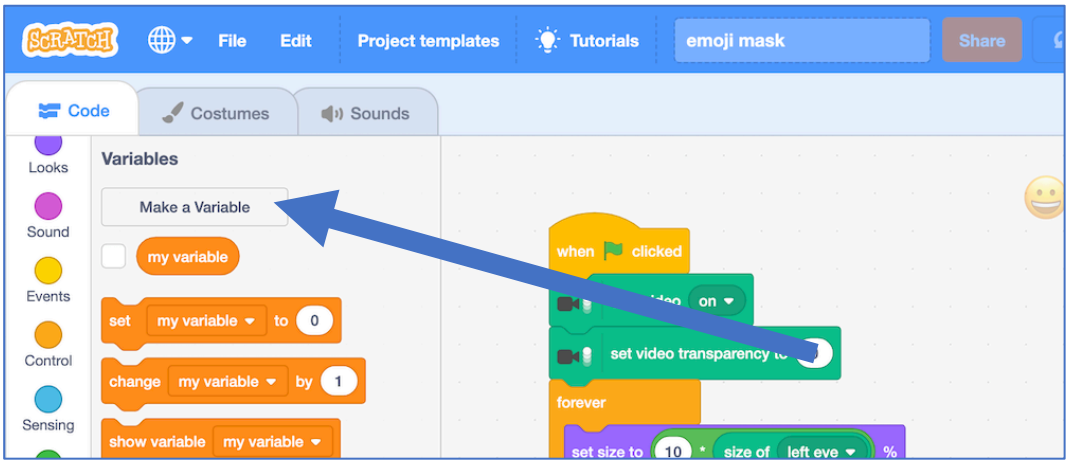
You've made a Scratch project using a pre-trained machine learning model.

Over 32,000 photos were collected by academics at a university, who went through them all and noted the location of the 390,000 faces they found in them. All of those examples of what bits of photos look like faces were used to train a machine learning model how to recognize faces in photos.

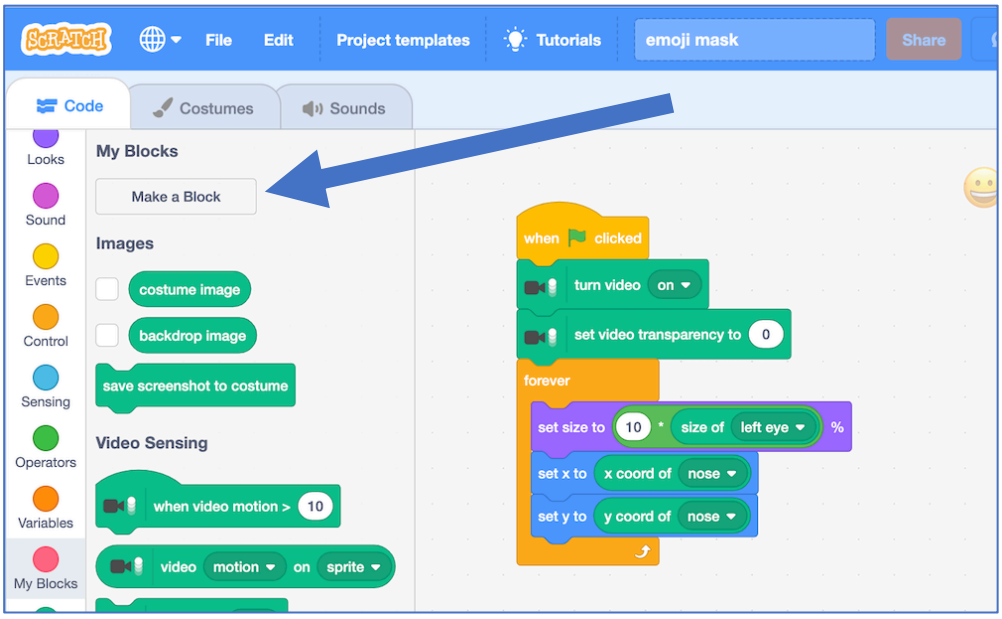
Real-world machine learning projects often use models already trained by other people. It is a good way to quickly make a project when you don't have the time to collect your own training data.

The next step is to update your project so the sprite matches the angle of your face, so the mask matches your face when you tilt your head!

15. Click on **Make a Variable** and create a variable called "angle"

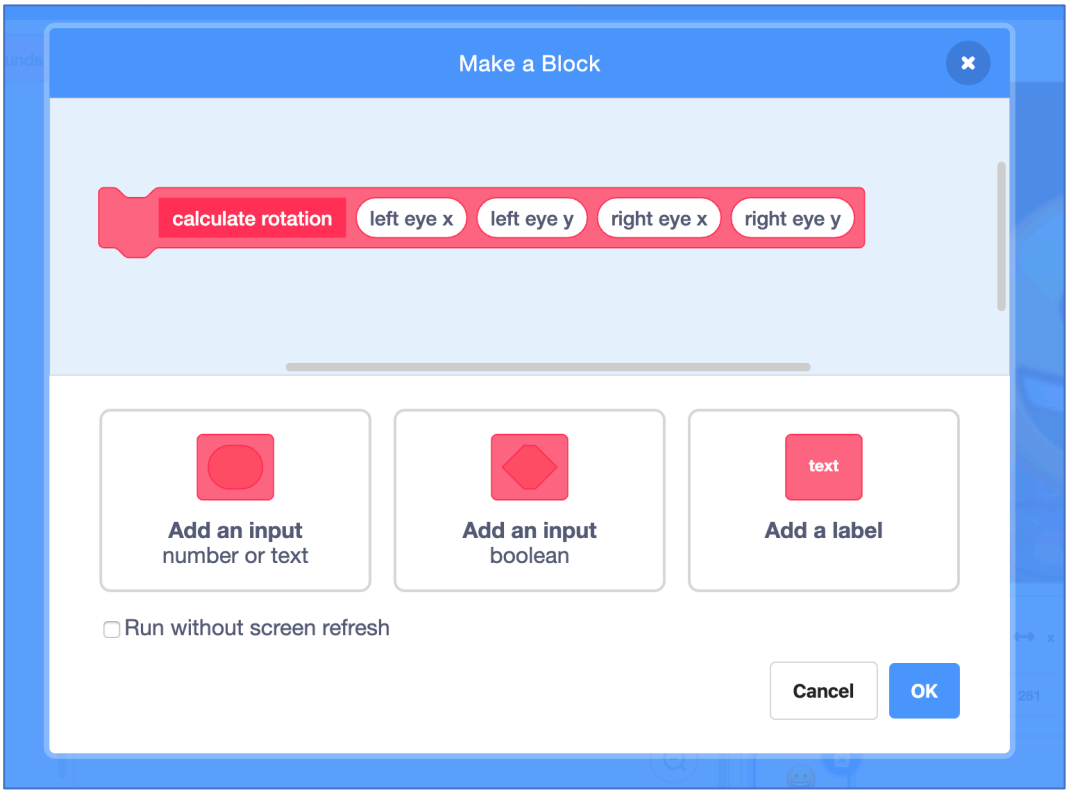


16. Click on the **Make a Block** button



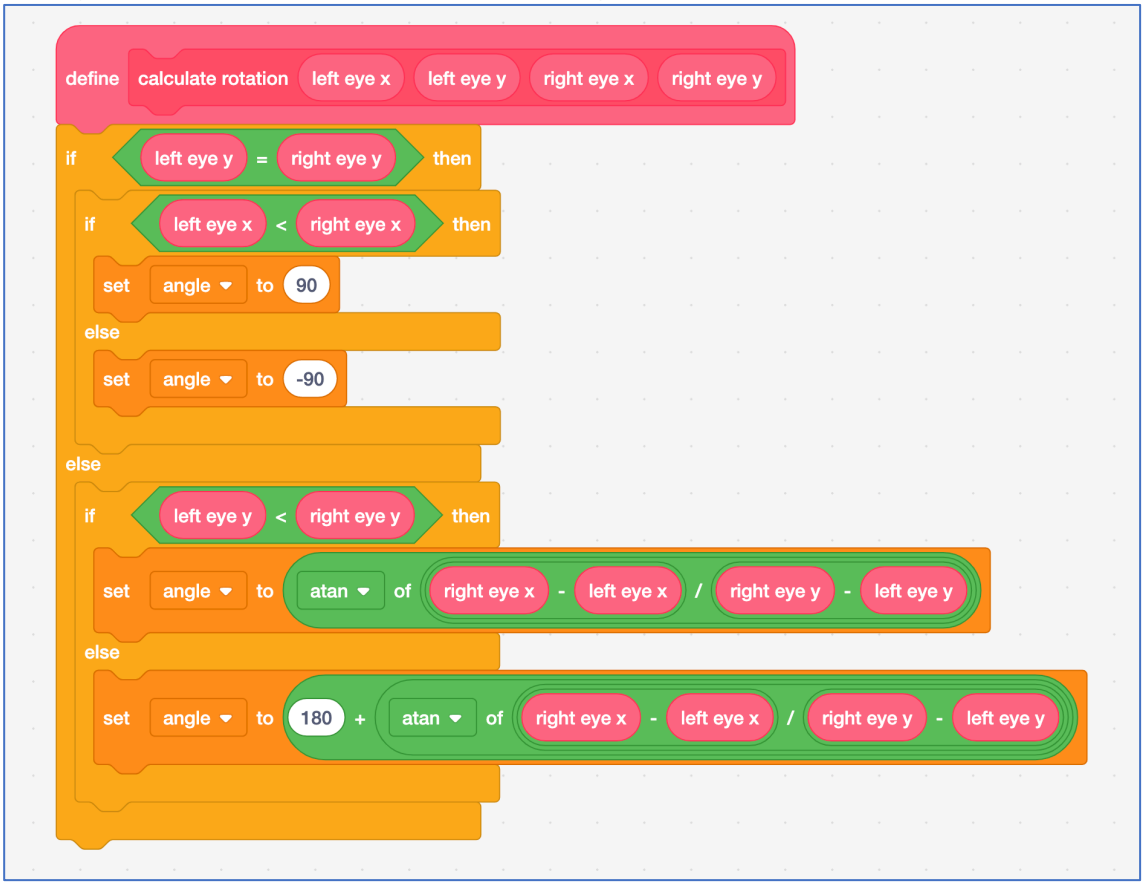
17. Create a script called **calculate rotation**

This will use the machine learning model's prediction of where your eyes are, and calculate the angle between those two points.
The script needs four input numbers: two for the location of the left eye, two for the location of the right eye.

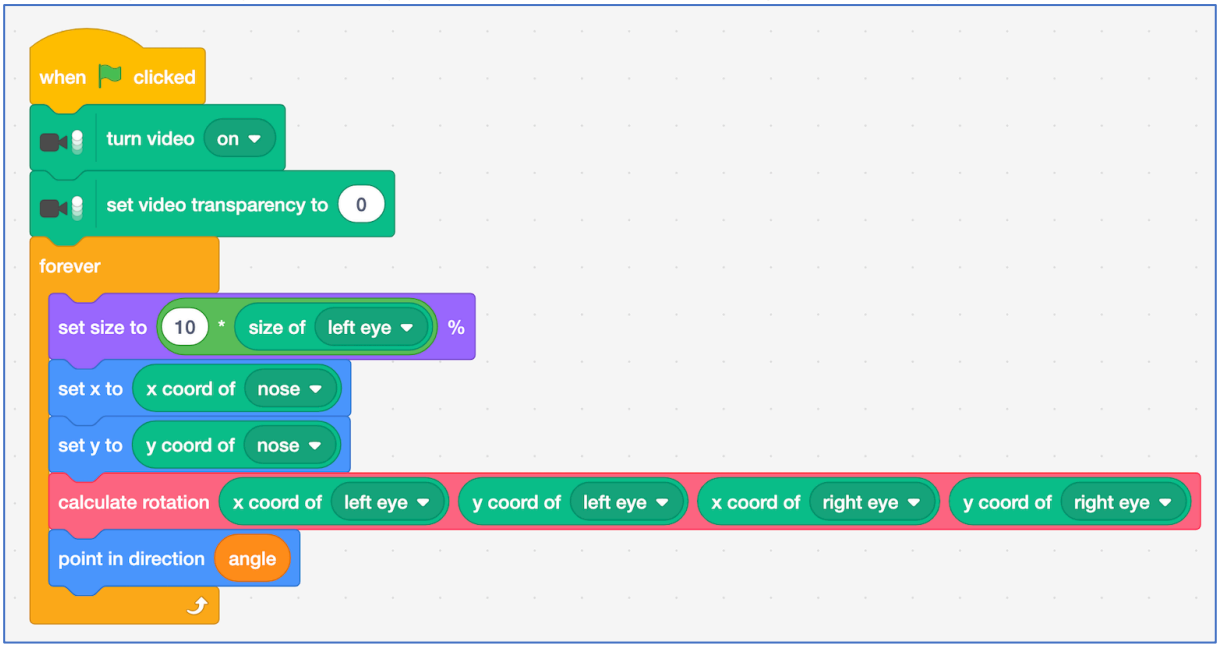


18. Time for some trigonometry!

Create this script which uses atan to compute the angle between two points. It's a little complicated – you need to copy it carefully. Can you understand what it's doing?



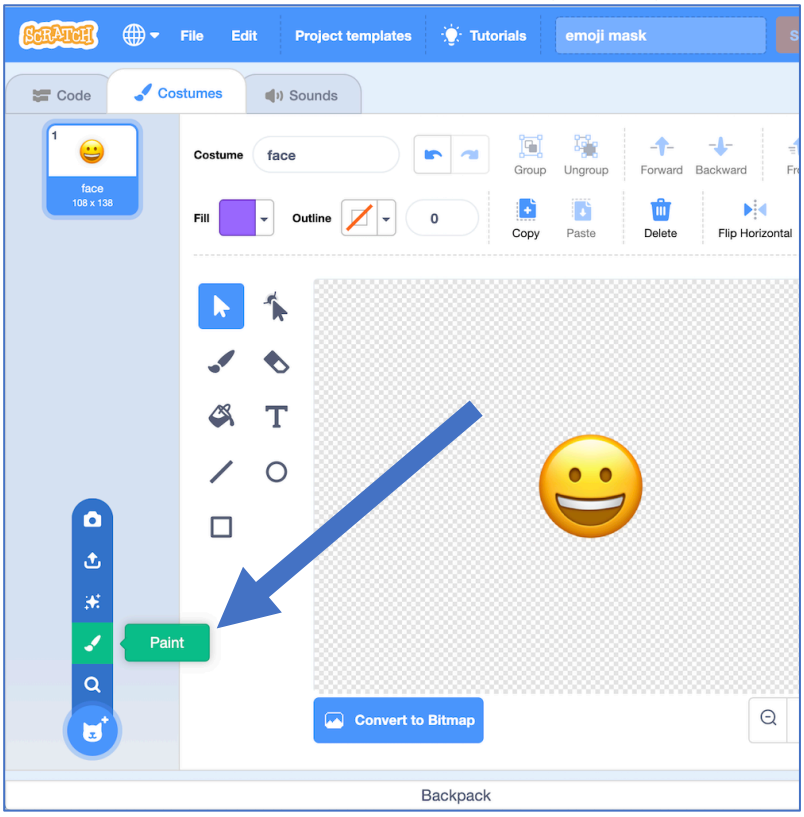
19. Update your script to use your new trigonometry function



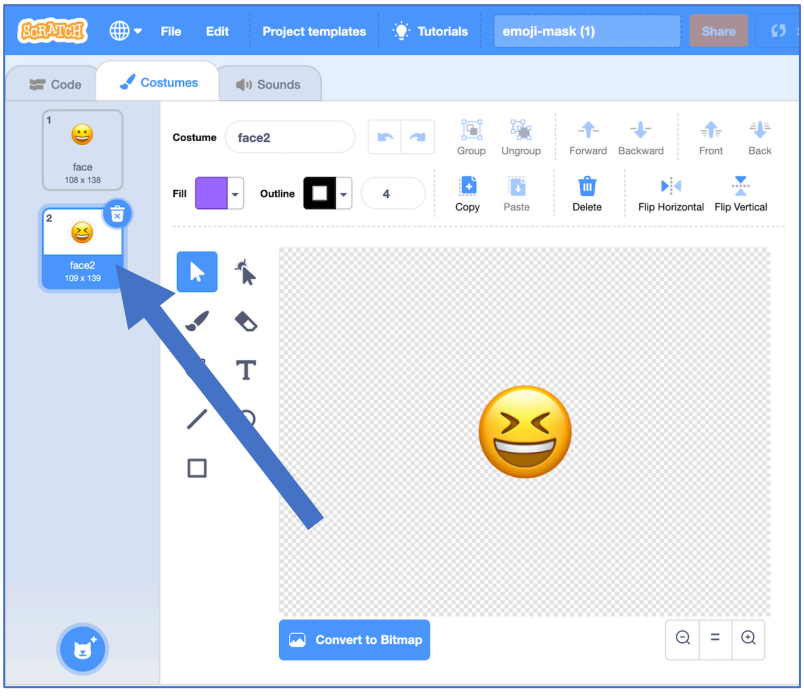
20. Test your project again by clicking on the **Green Flag**

Try tilting your head and check that the emoji mask rotates to match.

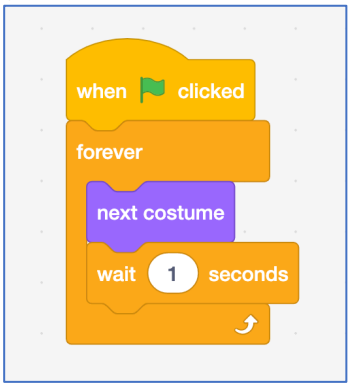
21. Add a second costume to your sprite



22. Choose a different face emoji. Make it match the size and location of the face in the first costume.



23. Add a new script to switch between your two costumes, so that your mask animates.



24. Click on the **Green Flag** to test

Your emoji mask should move to match the location of your face in the webcam.
If you tilt your head it should rotate to match.
If you move closer to the webcam, it should increase size to match.
If you move further away from the webcam, it should decrease size to match.

What have you done?

You've made a Scratch project using a machine learning technique known as face detection: detecting the location of faces in photos.

There are two stages to how it does this.

First stage: "object detection". It finds the part of the photo that looks like it contains a face. Think of it as the computer drawing a box around where it predicts a face is.

The second stage: shape prediction. It predicts where the eyes, nose and mouth are most likely to be in the box drawn in the first stage. This is sometimes described as detecting "facial landmarks".

How is this technology used?

What you're doing is **not** "facial recognition". Your project isn't recognizing whose face is in the photo. That is because the pre-trained model that you are using hasn't been trained with examples of photos of a particular person.

It is just looking for something that looks like a human face, because it has been trained with examples of photos of lots of different faces.

"Face detection" is a useful capability. You might've seen mobile apps use video face filters to add fun effects to video, like you did in this project.

Other real-world uses include being able to automatically blur people's faces in photos when you don't have permission to publish their faces, or automatically counting the number of people that a video camera can see.