Начало работы с классификатором изображений (ML) в блочном кодировании

Описание

В этом руководстве показано, как создавать модели машинного обучения с помощью классификатора изображений в PictoBlox.

- **Используемое программное обеспечение:** PictoBlox .
- **Уровень сложности:** Новички
- **Категория:** Блочное кодирование , Начало работы , Среда машинного обучения , Учебное пособие

Введение в классификатор изображений

Классификатор изображений **среды машинного обучения PictoBlox** используется для классификации изображений на различные классы на основе их характеристик.

Например, предположим, что вы хотите создать модель, позволяющую судить, носит ли человек маску правильно или нет, и носит ли он ее вообще. Вам придется разделить свое изображение на три класса:

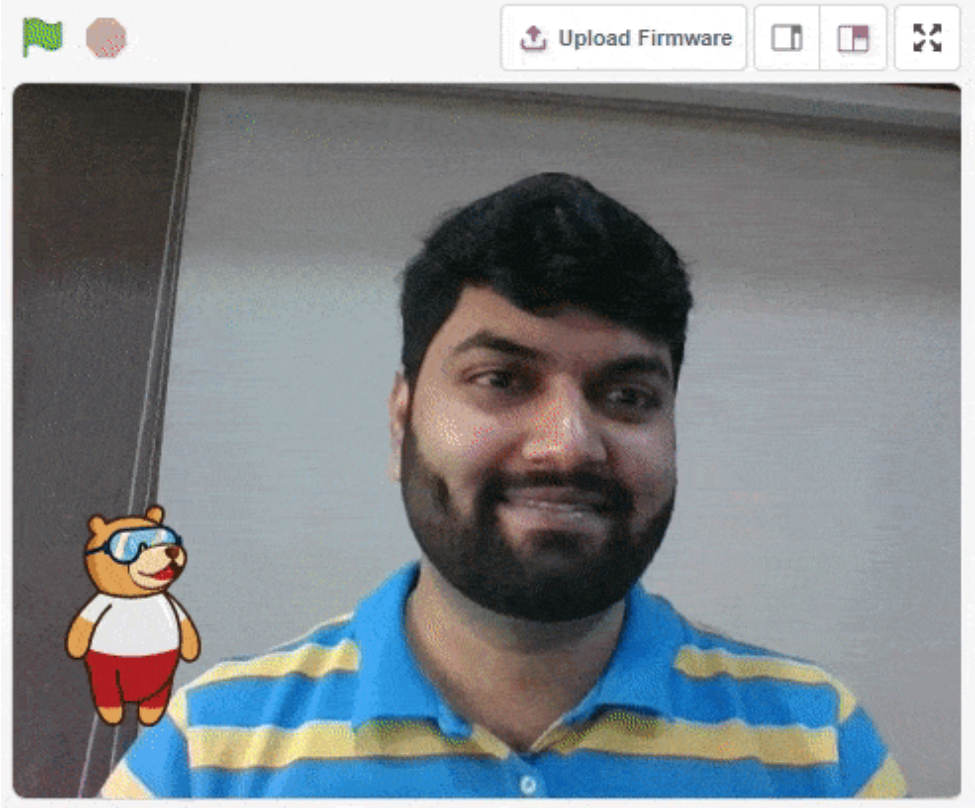
1. Ношение маски

2 Не носить маску

3 Неправильно носить маску

Это случай классификации изображений, когда вы хотите, чтобы машина пометила изображения в один из классов.

В этом уроке мы узнаем, как построить модель ML с помощью классификатора изображений PictoBlox.



Вот этапы процедуры:

1. Настройка среды
2. Сбор данных (Сбор данных)
3. Обучение модели
4. Тестирование модели
5. Экспорт модели в PictoBlox
6. Создание скрипта в PictoBlox

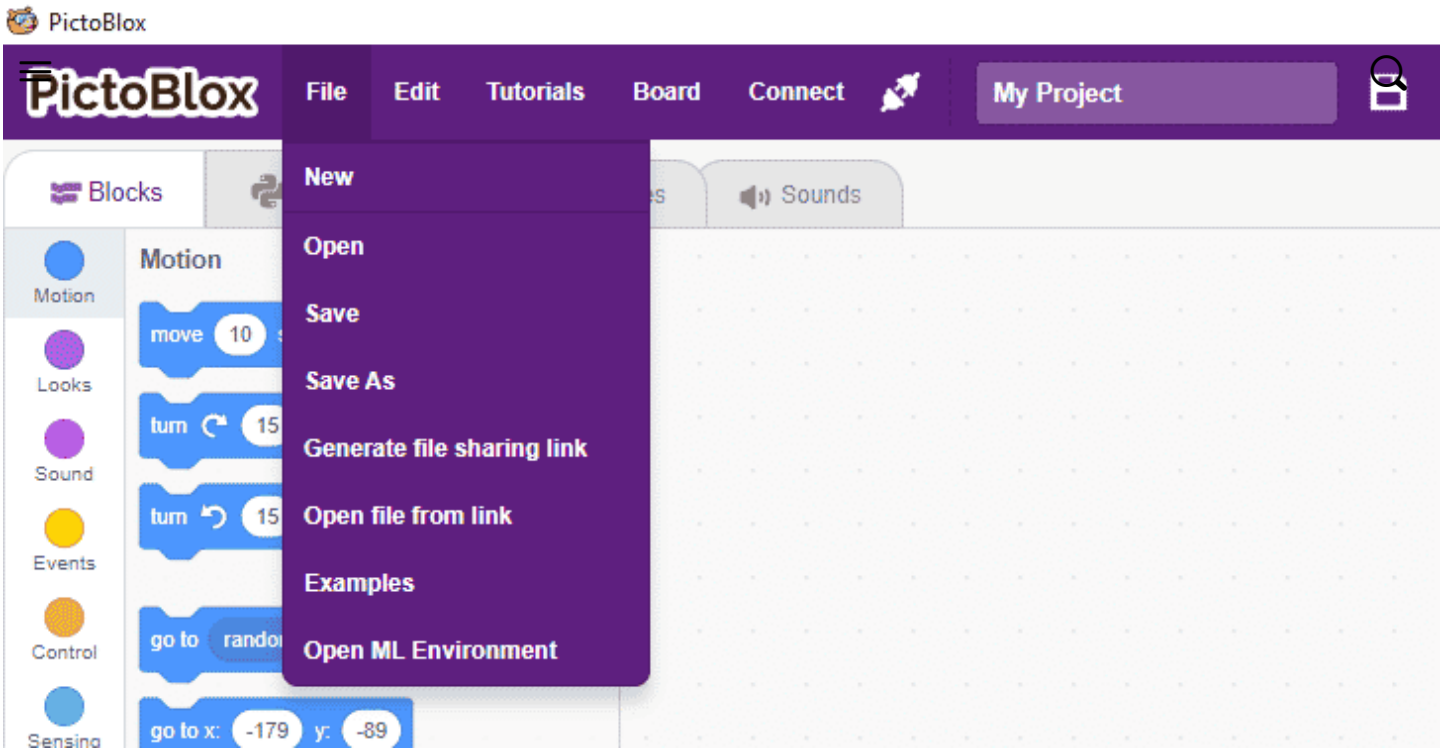
Настройка среды

Во-первых, нам нужно настроить среду машинного обучения для классификации изображений.

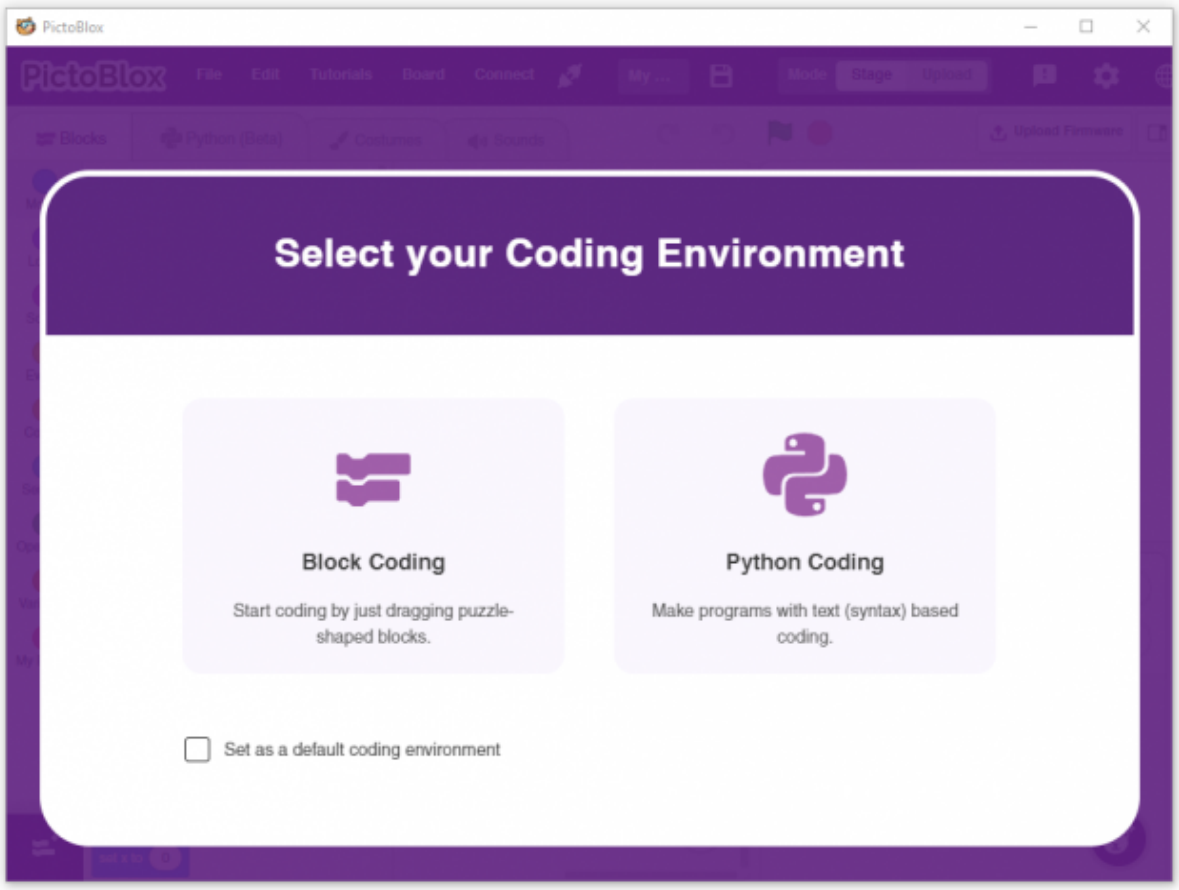
Среда машинного обучения для создания моделей доступна в единственной настольной версии PictoBlox для **Windows, macOS или Linux** . Он недоступен в веб-версиях, версиях для Android и iOS.

Выполните следующие действия:

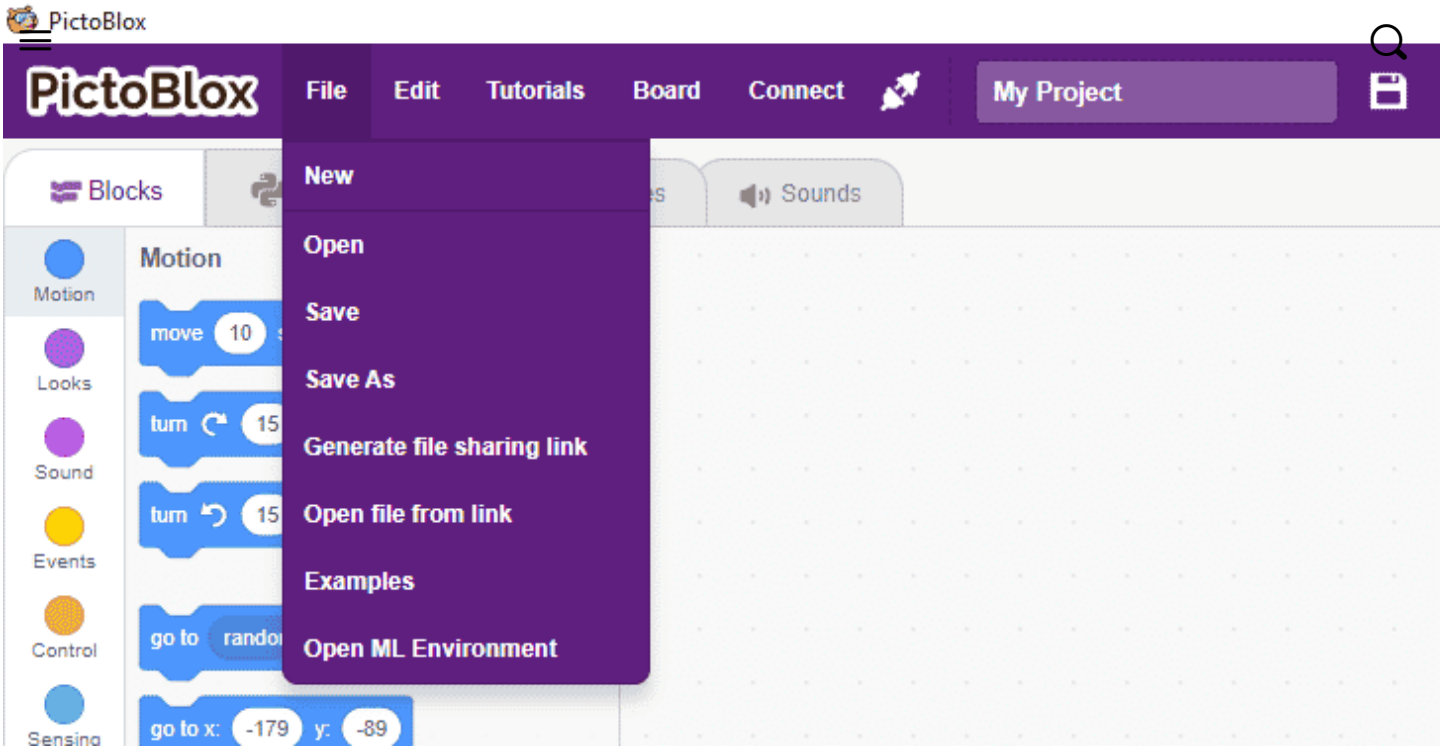
1. Откройте **PictoBlox** и создайте новый файл.



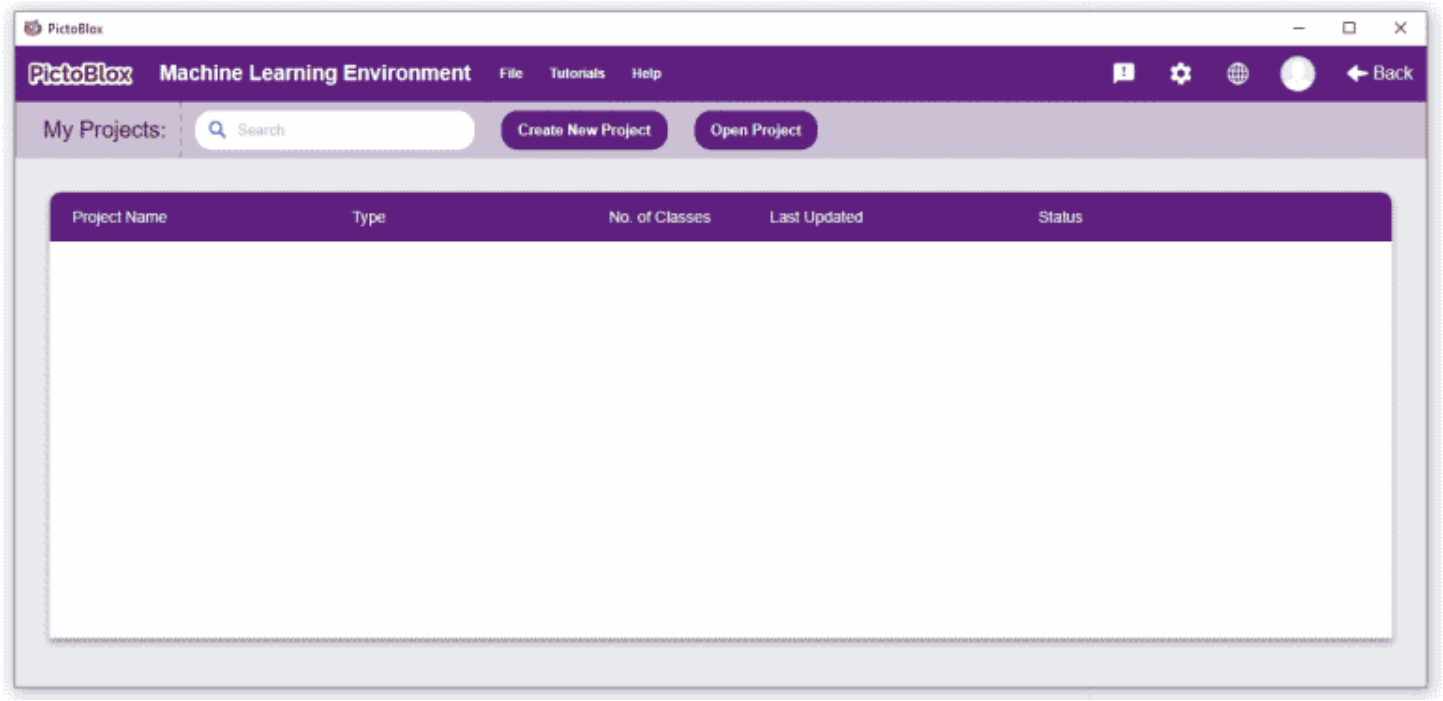
2 Выберите среду кодирования как **Block Coding** .



3 Чтобы получить доступ к среде ML, выберите параметр « **Открыть среду ML** » на вкладке « **Файлы** ».



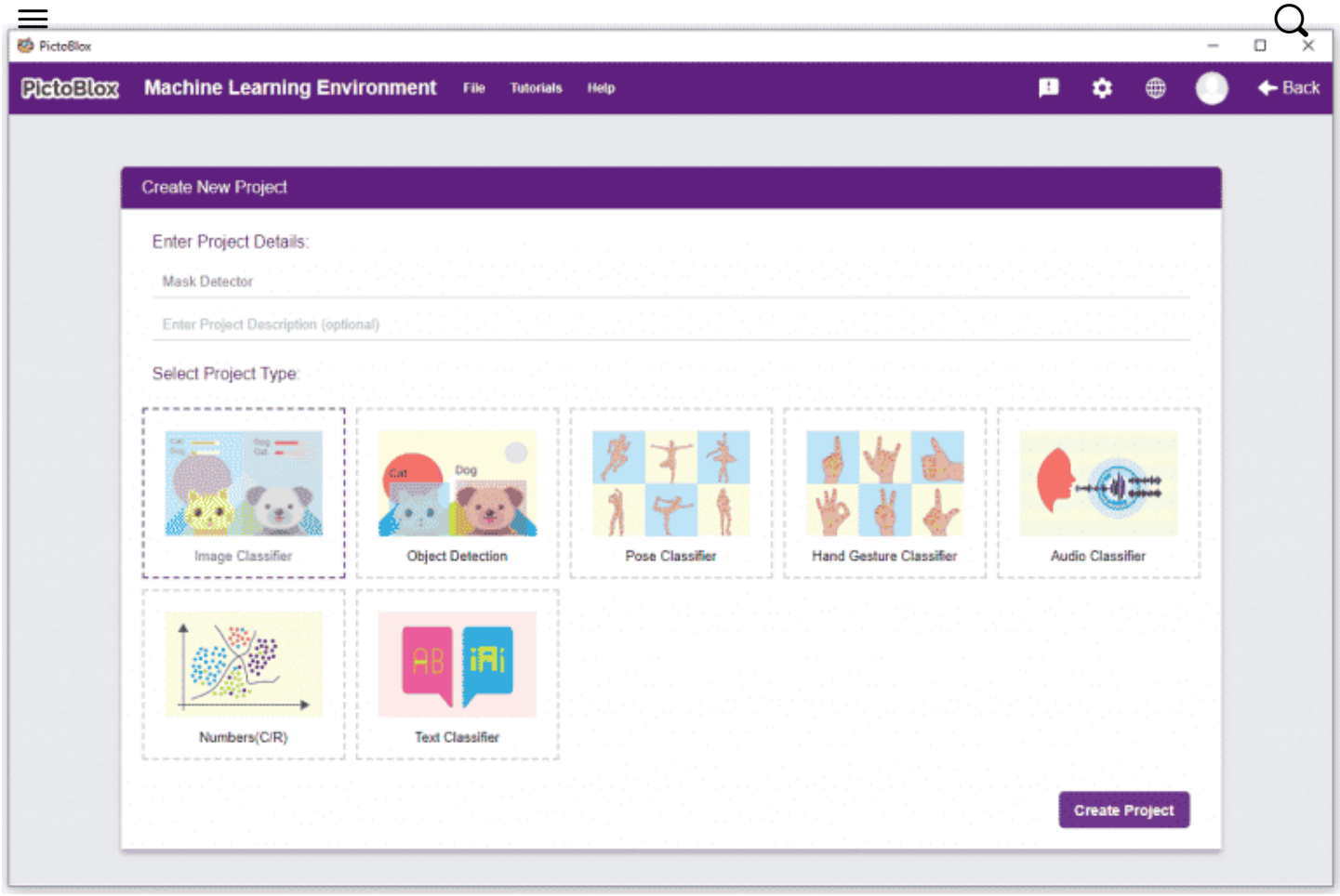
4 Вас встретит следующий экран.



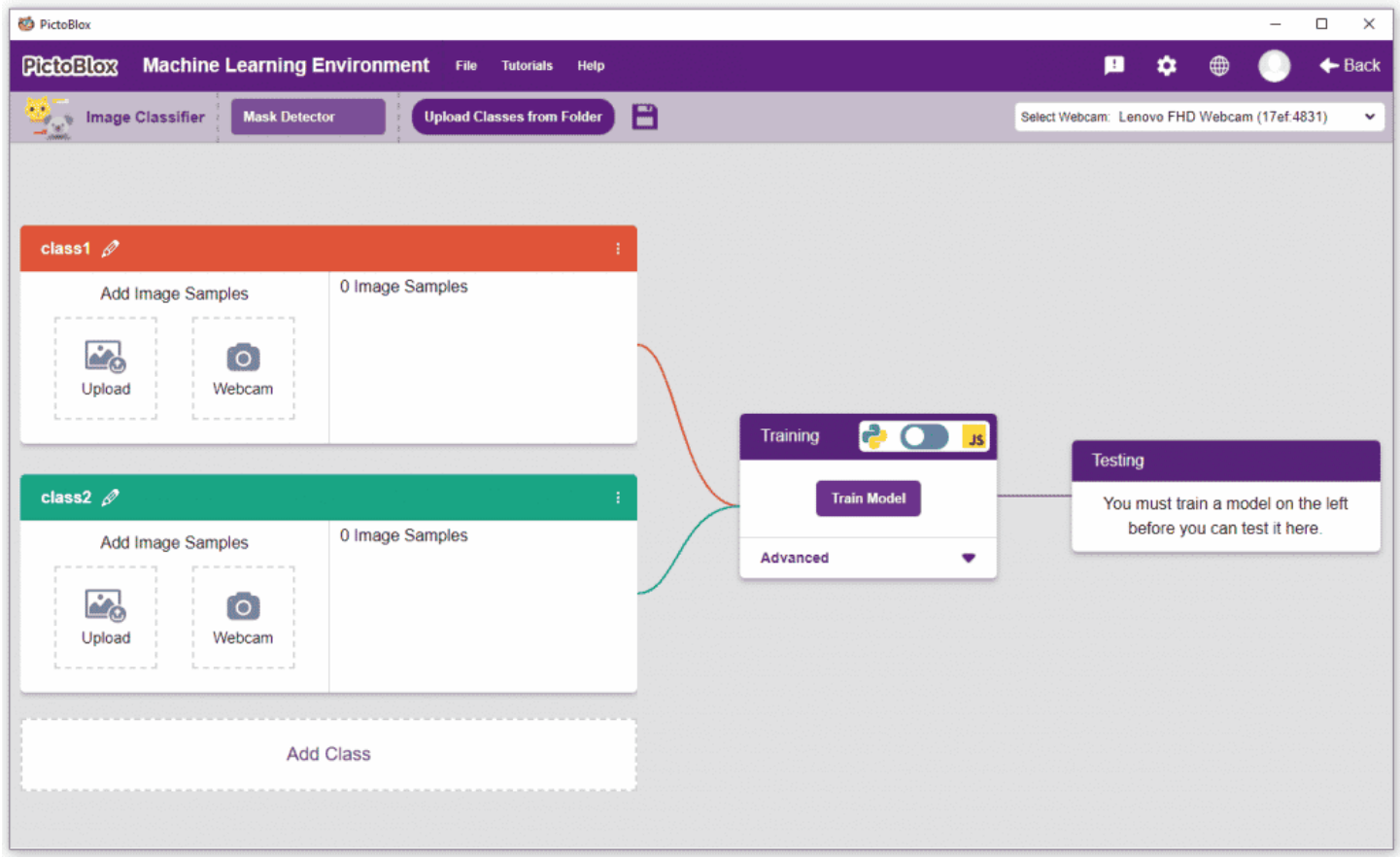
Нажмите « **Создать новый проект** ».

5 Откроется окно. Введите **имя проекта** по вашему выбору и выберите расширение « **Классификатор изображений** ». Нажмите кнопку « **Создать проект** », чтобы открыть окно

Классификатора изображений.

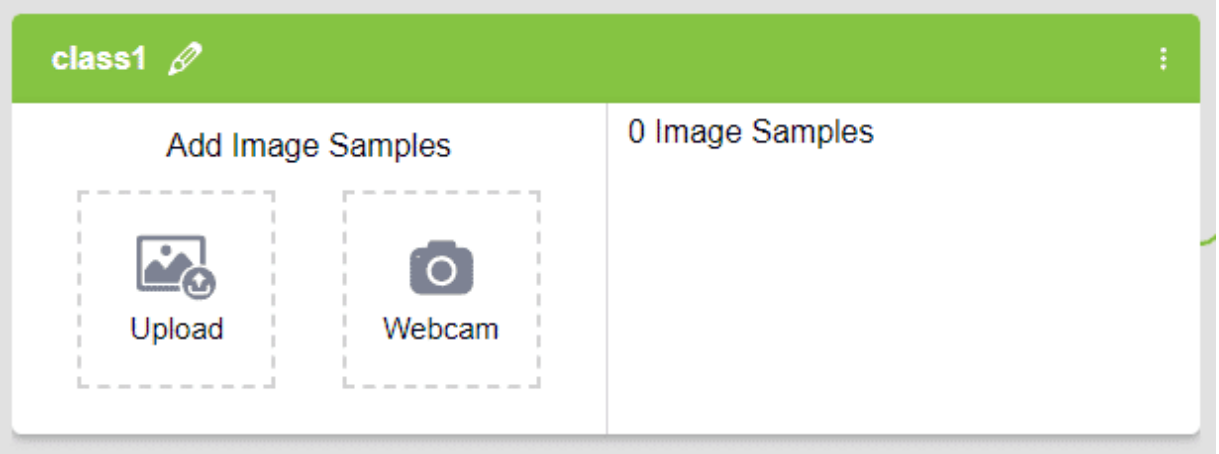


6 Вы увидите **рабочий процесс классификатора изображений** с двумя уже созданными для вас классами. Ваша среда настроена. Теперь пришло время загрузить данные.



Сбор и загрузка данных

Класс — это категория, по которой модель машинного обучения классифицирует изображения. Подобные изображения помещаются в один класс.



Есть 2 вещи, которые вы должны предоставить в классе:

1 Имя класса

2 Данные изображения: эти данные можно получить с **веб-камеры** , **загрузить** из локального хранилища или с диска Google.

Для этого проекта нам понадобятся три класса:

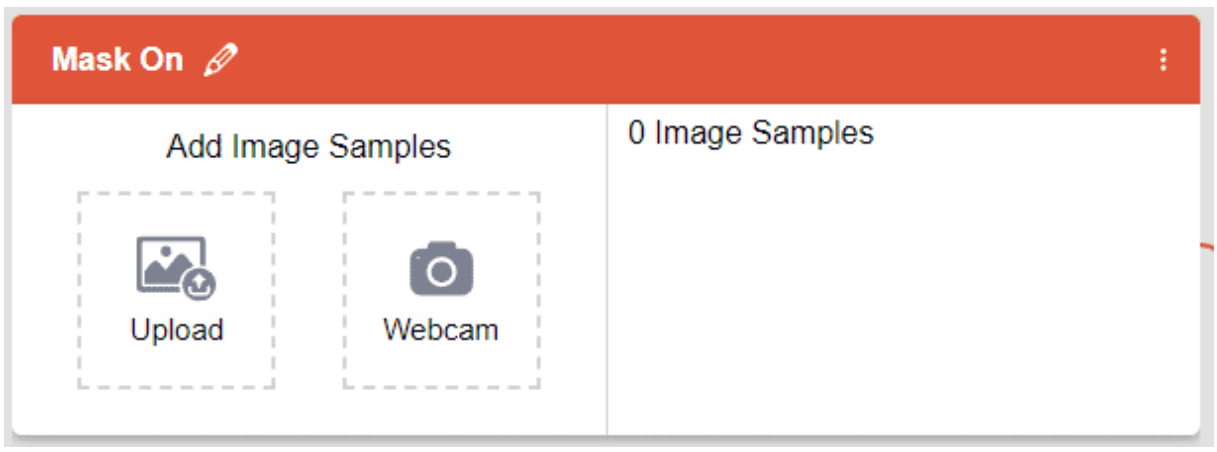
1 Ношение маски – **Маска включена**

2 Без маски – **маска выключена.**

3 Неправильное ношение маски – **маска неправильная**

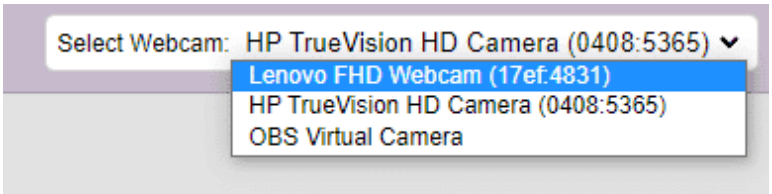
Выполните следующие действия, чтобы загрузить данные для классов:

1 Переименуйте имя первого класса ***в «Маска».***

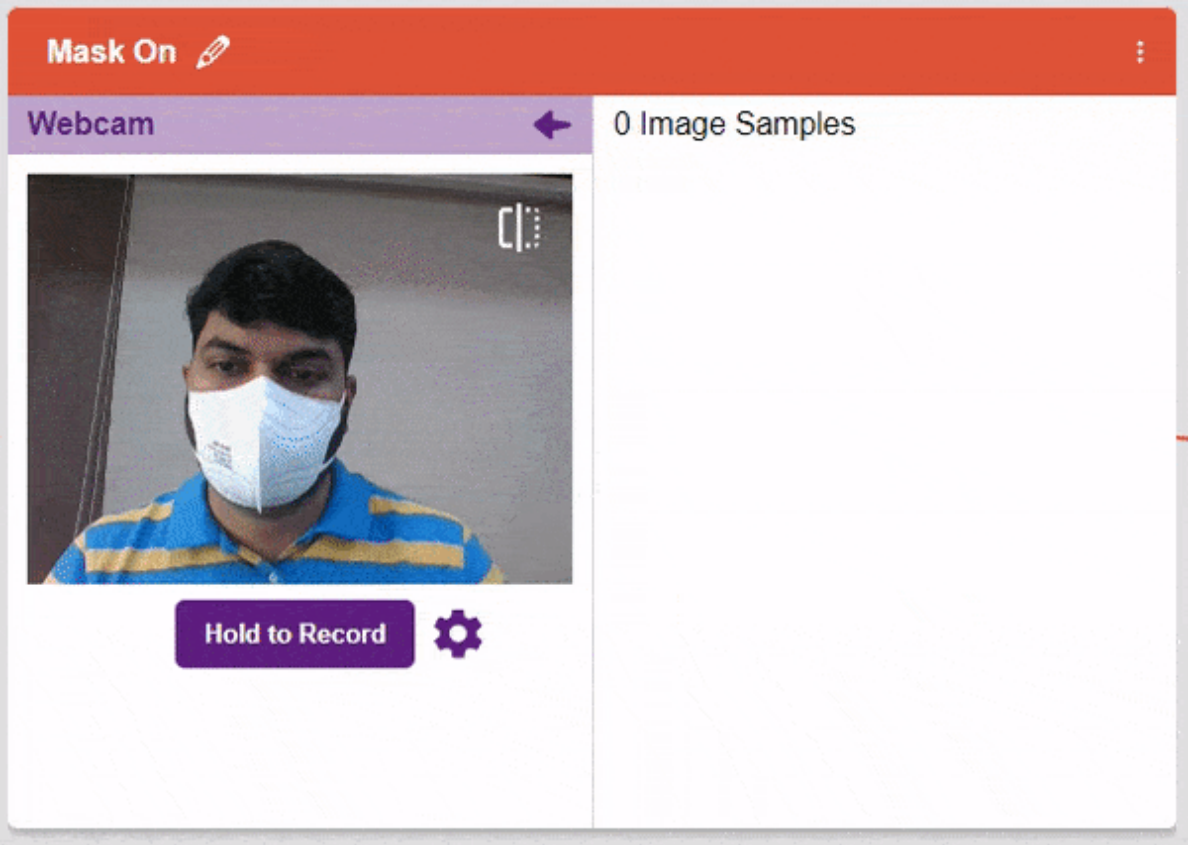


2 Нажмите кнопку ***«Веб-камера»*** .

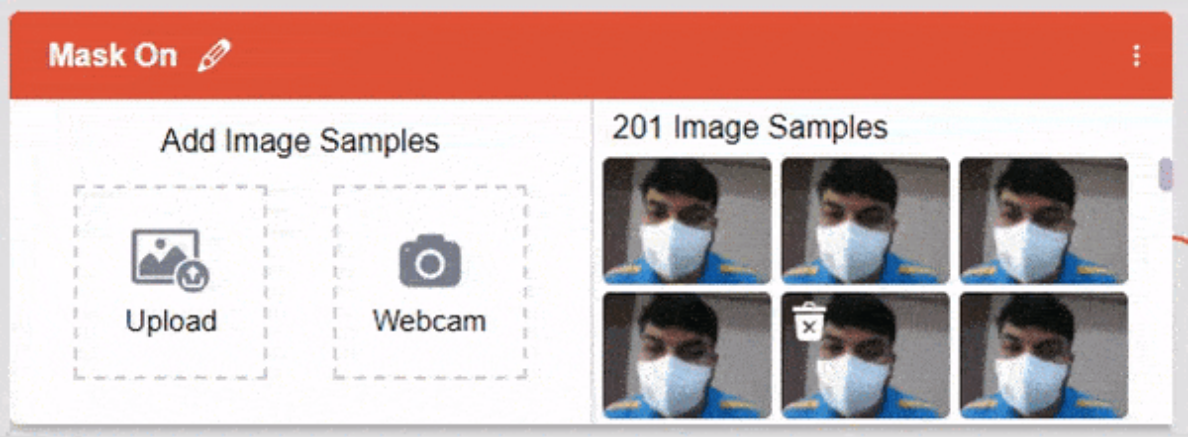
Если вы хотите изменить изображение с камеры, вы можете сделать это с помощью переключателя веб-камеры в правом верхнем углу.



3 Затем нажмите кнопку «Удерживать для записи», чтобы сделать снимок без маски. **Сделайте 200 фотографий с разной ориентацией головы.**

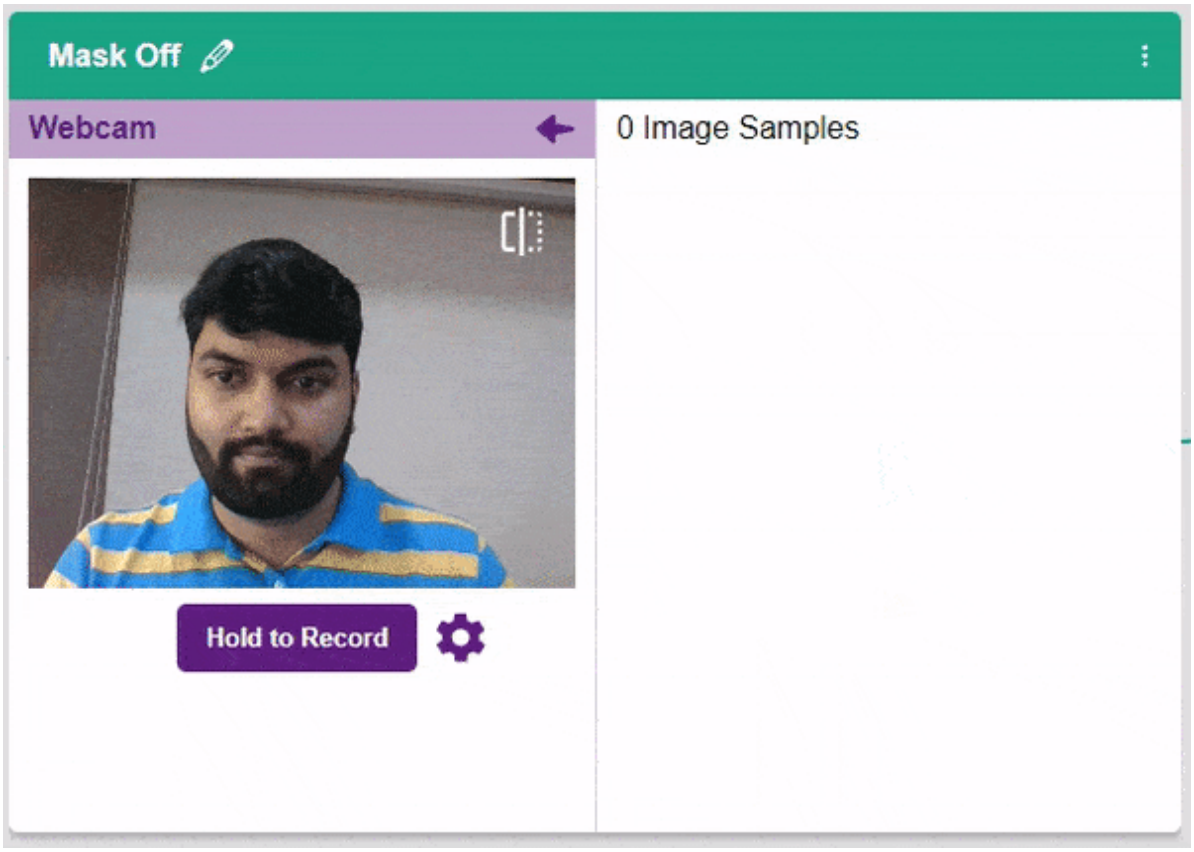


Если вы хотите удалить какое-либо изображение, наведите курсор на изображение и нажмите кнопку удаления.

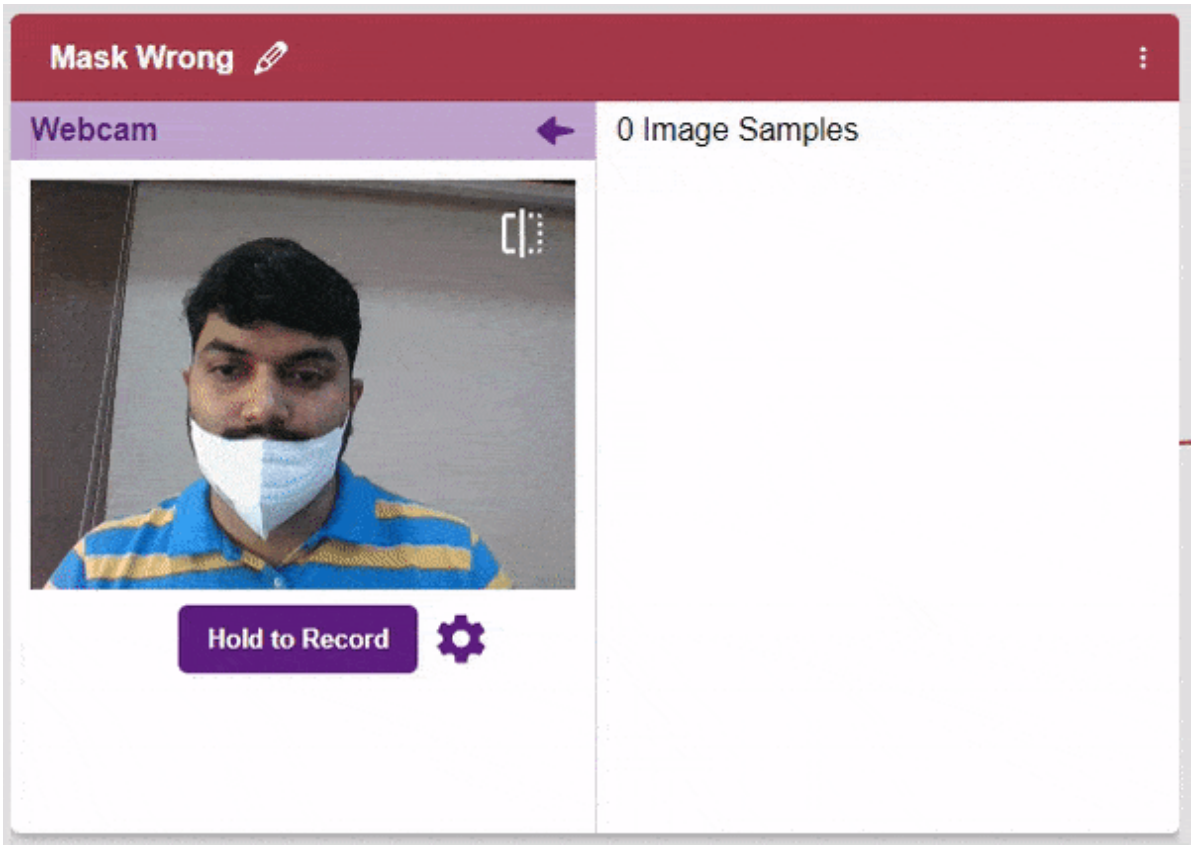


После загрузки вы сможете увидеть изображения в классе.

4 Переименуйте Класс 2 в «Выкл. маску» и возьмите образцы с веб-камеры.



5 Нажмите кнопку « **Добавить класс** », и вы увидите новый класс в своей среде. Переименуйте имя класса в «Неверная маска».



Примечание. *Для обучения модели необходимо добавить не менее 20 образцов в каждый класс. Больше образцов приведет к лучшим результатам.*

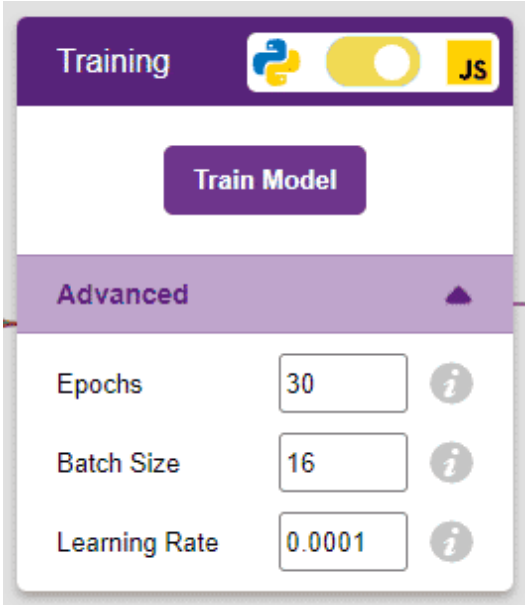
Как видите, теперь у каждого класса есть данные, из которых можно извлечь шаблоны. Чтобы извлечь и использовать эти шаблоны, мы должны обучить нашу модель.

Обучение модели

Теперь, когда мы собрали данные, пришло время научить нашу модель классифицировать новые, невидимые данные по этим трем классам. Для этого нам нужно **обучить** модель. Обучая модель, мы извлекаем из изображений значимую информацию, что, в свою очередь, обновляет

веса . Как только эти веса будут сохранены, мы сможем использовать нашу модель для прогнозирования ранее невидимых данных.

Однако перед обучением модели вам следует знать несколько гиперпараметров. Нажмите на вкладку «Дополнительно», чтобы просмотреть их.



Примечание. *Эти гиперпараметры могут в значительной степени повлиять на точность вашей модели. Поэкспериментируйте с ними, чтобы найти то, что лучше всего подходит для ваших данных.*

Здесь есть три гиперпараметра, с которыми вы можете играть:

1 **Эпохи** — общее количество раз, когда ваши данные будут передаваться через модель обучения. Следовательно, за 10 эпох набор данных будет пропущен через обучающую модель **10 раз** . **Увеличение количества эпох часто может привести к повышению производительности.**

2 **Размер партии** — размер набора образцов, которые будут использоваться за один шаг. Например, если в вашем наборе данных 160 выборок данных, а размер пакета равен 16, каждая эпоха будет завершена за **160/16=10 шагов** . **Вам редко придется изменять этот гиперпараметр.**

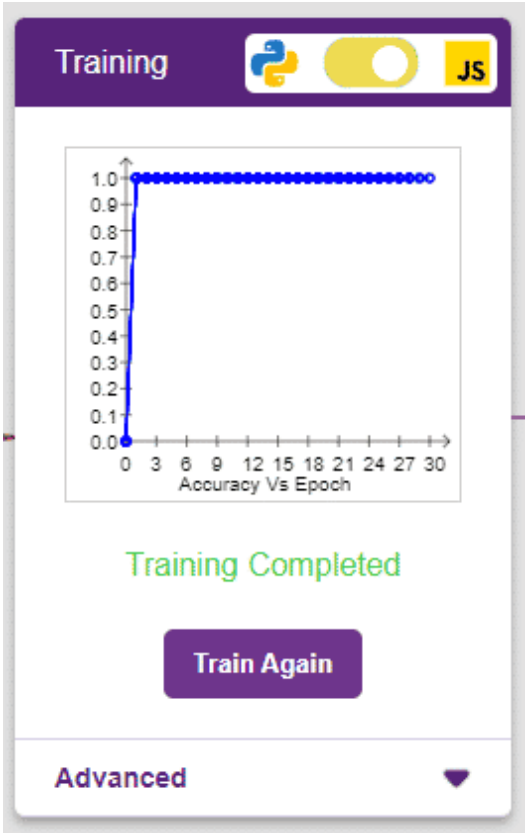
3 **Скорость обучения** . Определяет скорость, с которой ваша модель обновляет веса после итерации шага. **Даже небольшие изменения этого параметра могут оказать огромное влияние на производительность модели** . Обычный диапазон лежит между **0,001 и 0,0001.**

Примечание. *Наведите указатель мыши на вопросительный знак рядом с гиперпараметрами, чтобы увидеть их описание.*

Давайте обучим модель стандартным гиперпараметрам и посмотрим, как она работает. Вы можете обучать модель как на JavaScript, так и на Python. Чтобы выбрать между ними, нажмите на переключатель в верхней части поля «Обучение».

Внимание: *для обучения модели на Python необходимо загрузить зависимости. По умолчанию будет выбран JavaScript.*

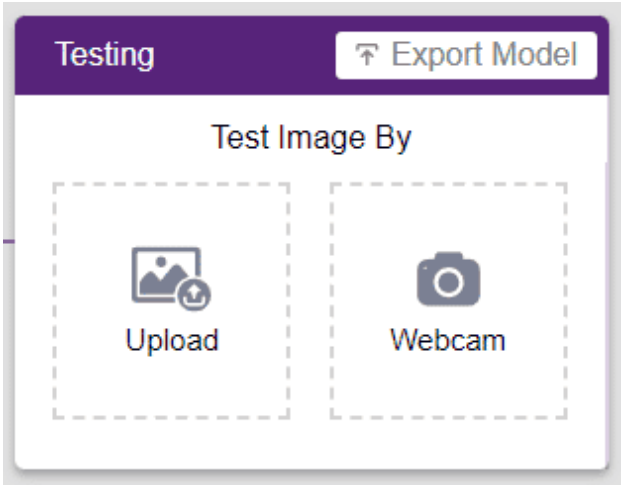
Мы будем обучать эту модель на JavaScript. Нажмите кнопку «Обучить модель», чтобы начать обучение. Нам не нужно менять какие-либо гиперпараметры для этой модели.



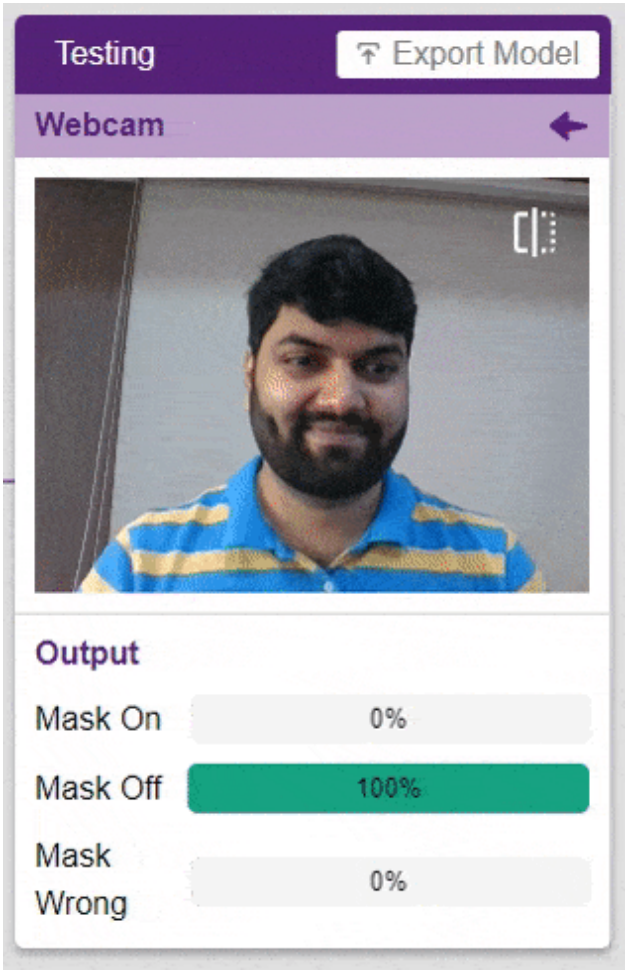
Модель показывает отличные результаты! Помните: чем выше значение на графике точности, тем лучше модель. По оси X графика показаны эпохи, а по оси Y представлена соответствующая точность. Диапазон точности от 0 до 1.

Тестирование модели

Теперь, когда модель обучена, давайте посмотрим, дает ли она ожидаемые результаты. Мы можем протестировать модель, используя камеру устройства или загрузив изображение из хранилища устройства. Давайте для начала воспользуемся нашей веб-камерой.



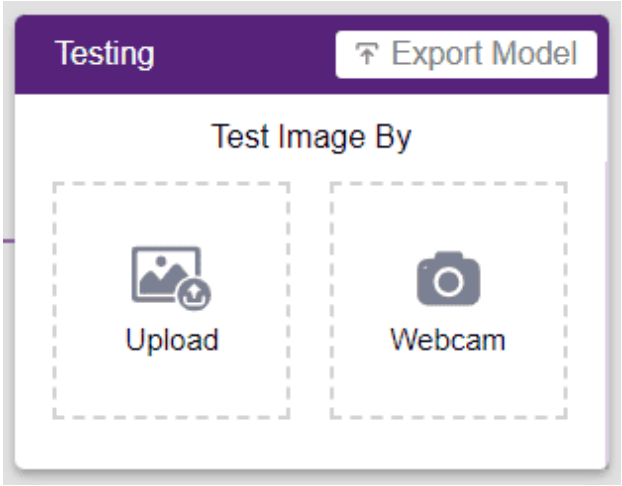
Нажмите на опцию «Веб-камера» в поле тестирования, и модель начнет прогнозировать на основе изображения в окне.



Большой! Модель способна делать прогнозы в режиме реального времени. Теперь закройте окно, щелкнув крестик в правом верхнем углу окна тестирования.

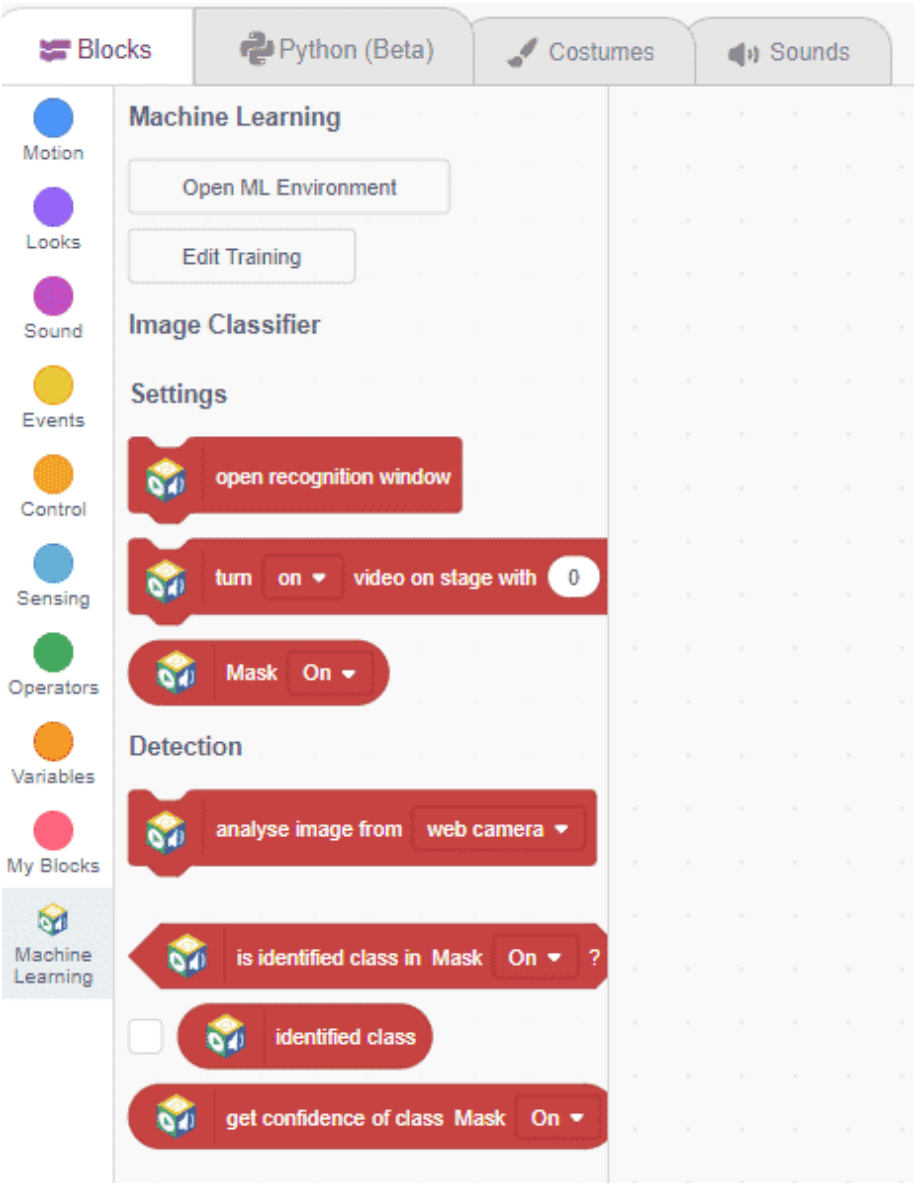
Теперь мы можем экспортировать нашу модель и создать проект в **среде блочного кодирования** .

Экспорт модели в среду блочного кодирования



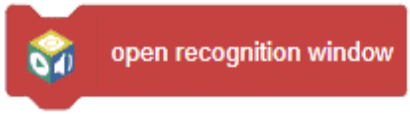
Нажмите кнопку « **Экспортировать модель** » в правом верхнем углу окна «Тестирование», и PictoBlox загрузит вашу модель в среду блочного кодирования.

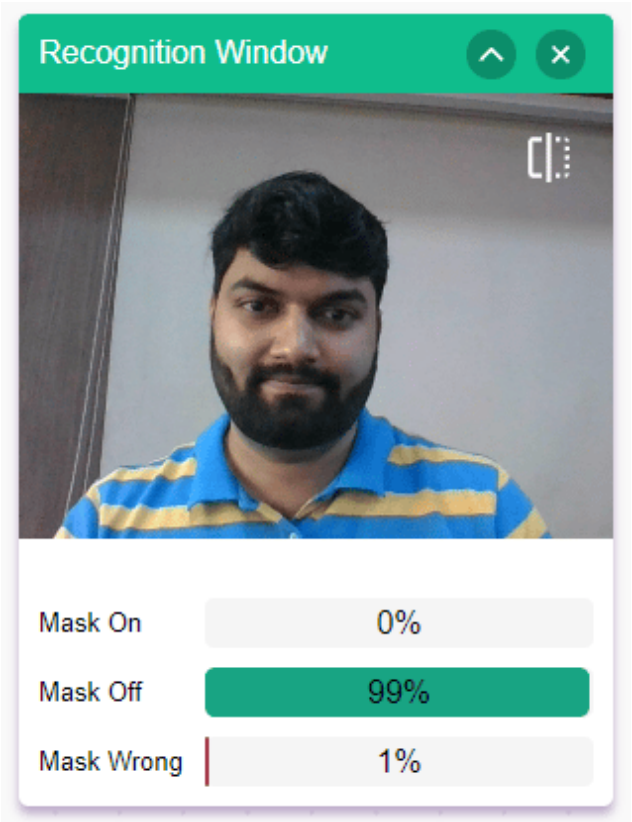
Посмотрите, как на левой боковой панели расположены блоки, относящиеся к модели, которую мы только что обучили.



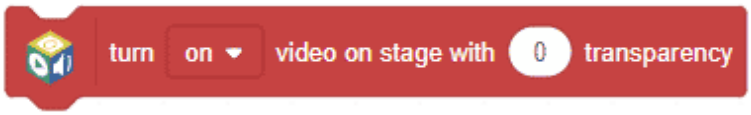
Каждый блок здесь имеет определенную функцию:

1 **открытое окно распознавания:** открывает окно, которое использует камеру устройства для прогнозирования.

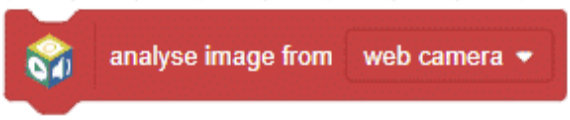




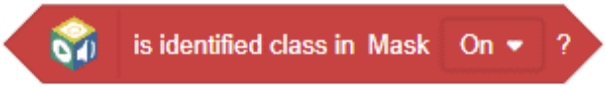
2 **Turn () видео на одну сцену с () прозрачностью:** показывает видео, снятое камерой на сцене.



3 **анализировать изображение из ():** анализировать входное изображение.



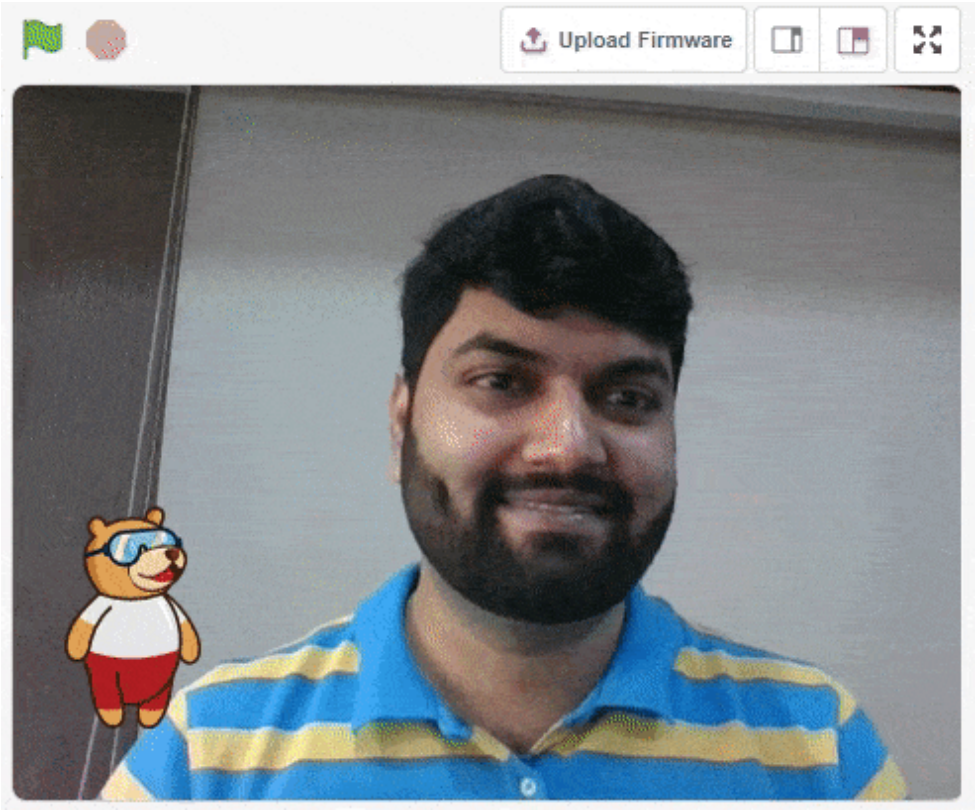
4 **is идентифицировать класс в ():** определяет, принадлежит ли анализируемое изображение определенному классу.



5 **получить достоверность в классе ():** возвращает уровень достоверности классификации изображений. Может использоваться как **пороговое значение.**

Сценарий в среде блочного кодирования

Теперь давайте применим нашу модель в реальном проекте. Для этого мы будем использовать нашу среду блочного кодирования.



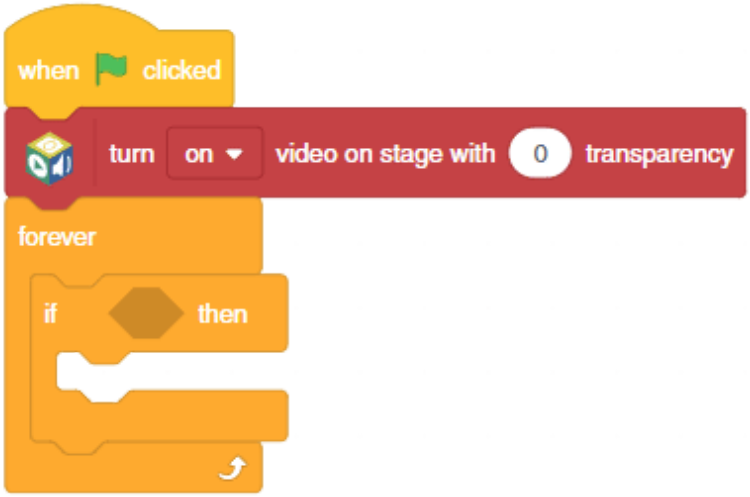
Мы создадим скрипт, который будет использовать нашу модель для анализа изображения с камеры устройства. Как только это будет сделано, наш спрайт Тоби сообщит нам, носит ли человек на изображении маску, носит маску неправильно или не носит маску вообще.

Давай начнем!

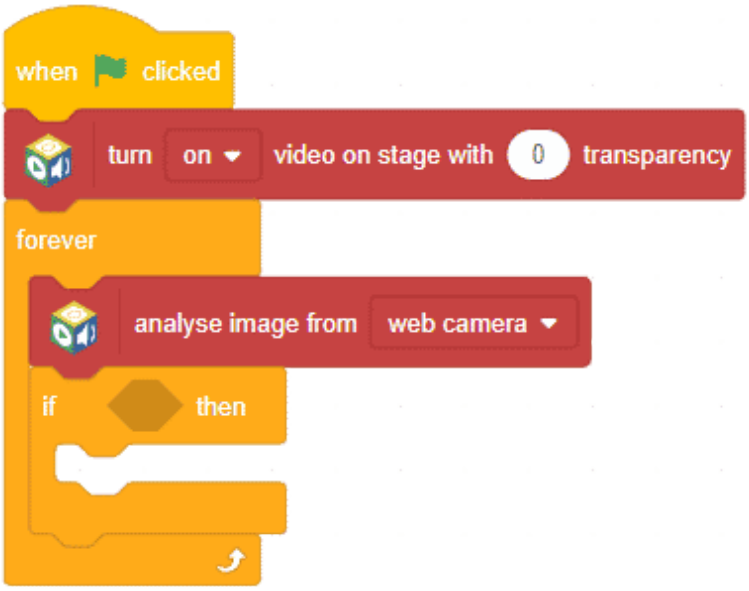
1 Добавьте блок **«когда флажок нажат»** и блок **«навсегда»** в область сценариев и соедините их вместе.

2 Добавьте **видео поворота () на сцену с** блоком прозрачности () над блоком **навсегда** . Выберите ON и 0 в качестве прозрачности.

3 Перетащите блок **if** внутрь блока **навсегда** .

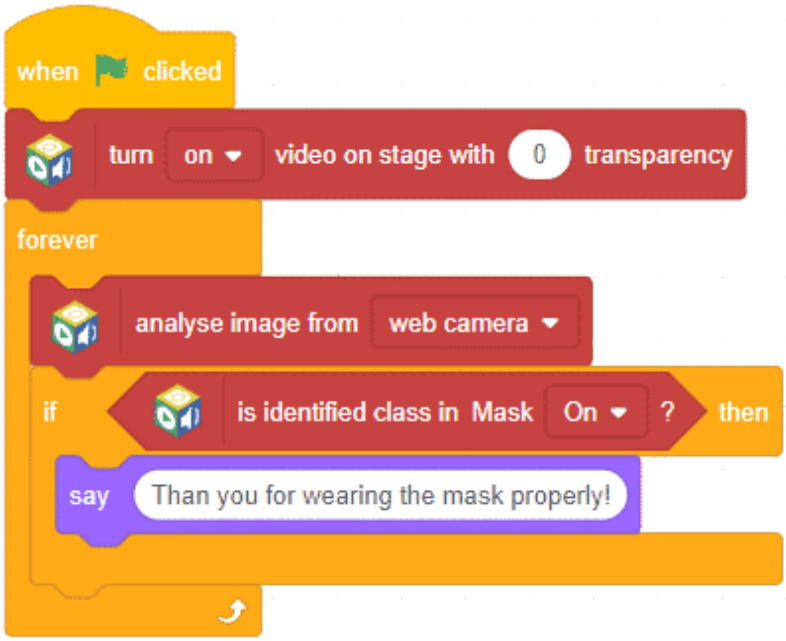


4 Затем добавьте **изображение анализа из** блока () над блоком **if ()** и выберите ***веб-камеру*** в качестве канала.

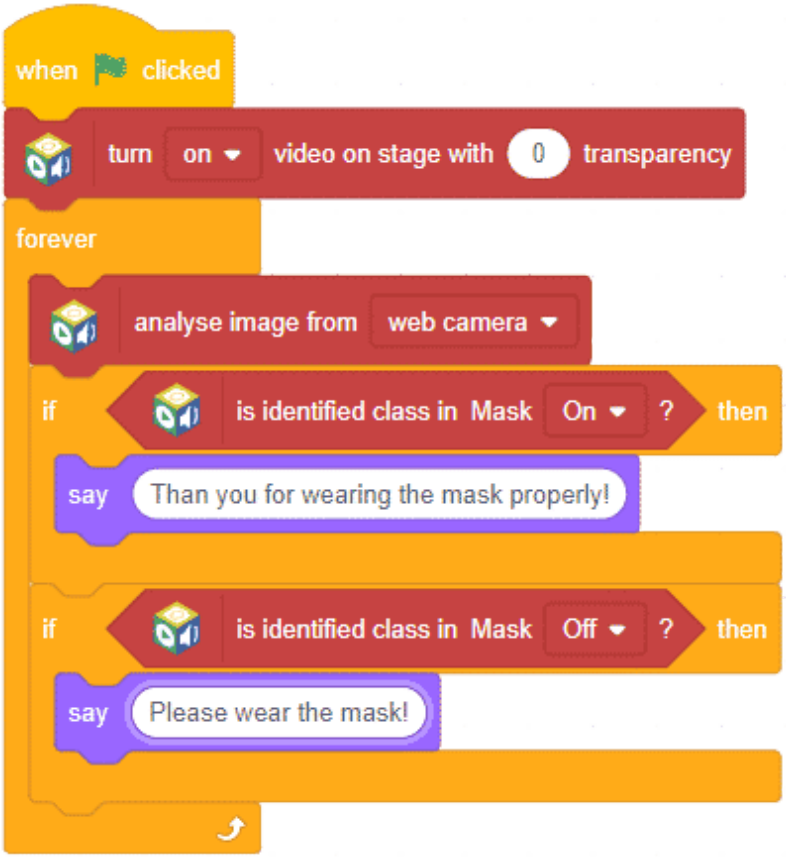


5 Затем добавьте **идентифицированный класс ()?** блок в пространстве условий. Выберите класс как **Mask ON** .

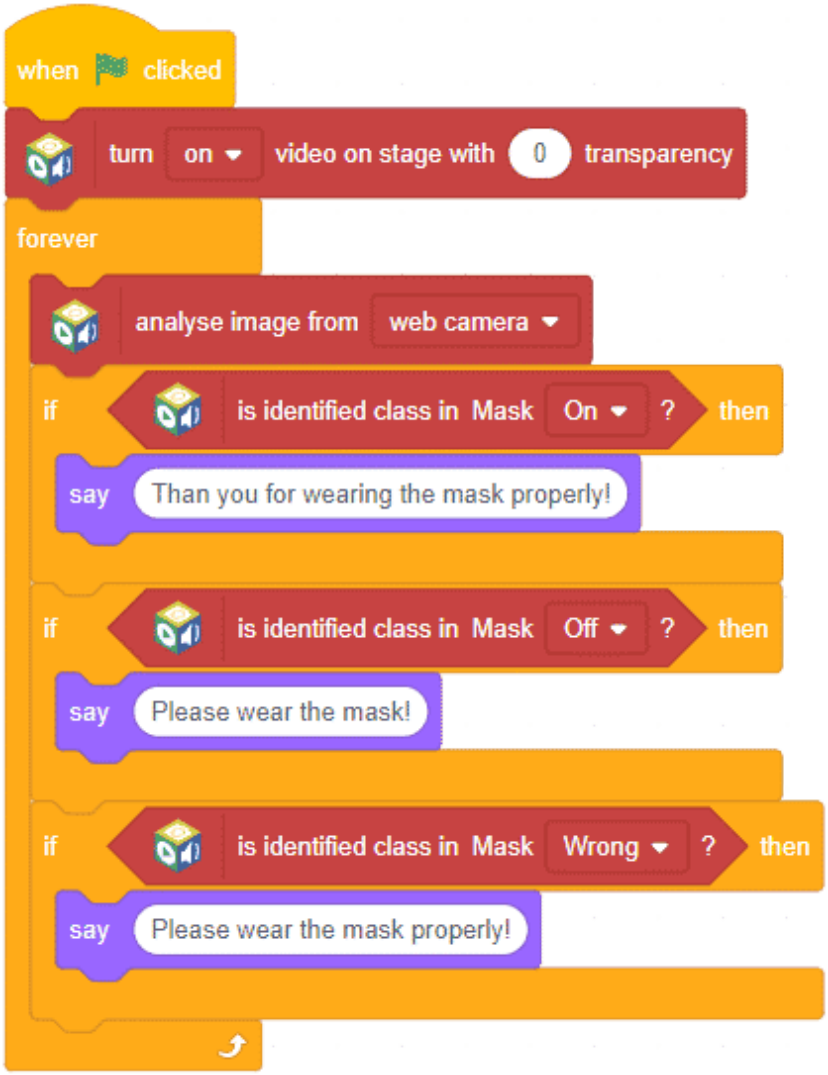
6 В палитре ***«Внешний вид»*** добавьте блок **Say()** внутри блока **if** . Напишите сообщение – Спасибо за правильное ношение маски!



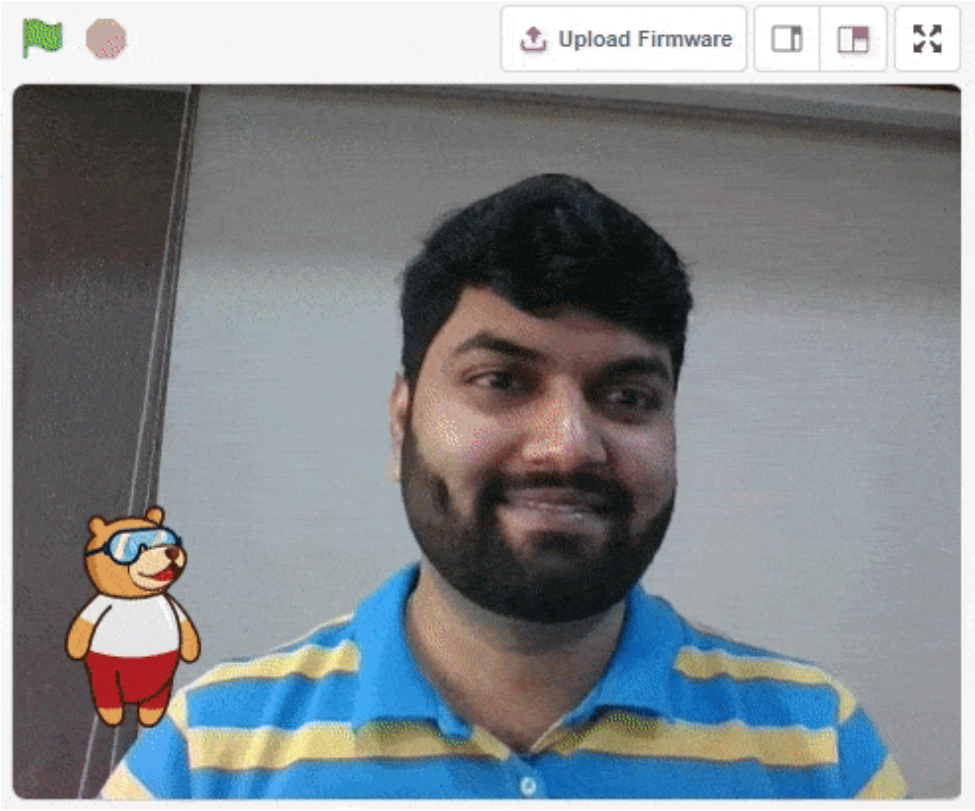
7 Дублируйте блок **if** и прикрепите его под первым блоком **if** . Выберите класс в качестве **маски. Выкл.** Во втором **идентифицирован класс ()?** блокировать. Измените текст в блоке «Сказать» на «Пожалуйста, наденьте маску!»



8 Дублируйте блок **if** и прикрепите его под первым блоком **if** . Выберите класс в качестве **маски. Неправильно** во втором **идентифицирован класс из () есть ()?** блокировать. Измените текст в блоке сказать на «Пожалуйста, носите маску правильно!»



Сценарий завершен. Нажмите **зеленый флаг** , чтобы запустить скрипт.



Сохраните проект как **Mask Detector** .

Вот оно! Вы только что использовали классификацию изображений, чтобы создать свой собственный проект обнаружения масок.

Документация продукта

Кварки Наборы
наборы для выживания
Программное обеспечение ПиктоБлокс
Расширения и библиотеки PictoBlox
Приложение Даббл
Ардуино с PictoBlox

Связаться

Связаться с нами
Забронировать демо
Запрос цитаты

Подписывайтесь на нас

PictoBlox — программирование на блоках и Python для детей

GET IT ON Google Play
Download for Windows
Download for MacOS

Приложение Dabble — одно приложение. Бесконечный контроль.

GET IT ON Google Play
Download on the App Store