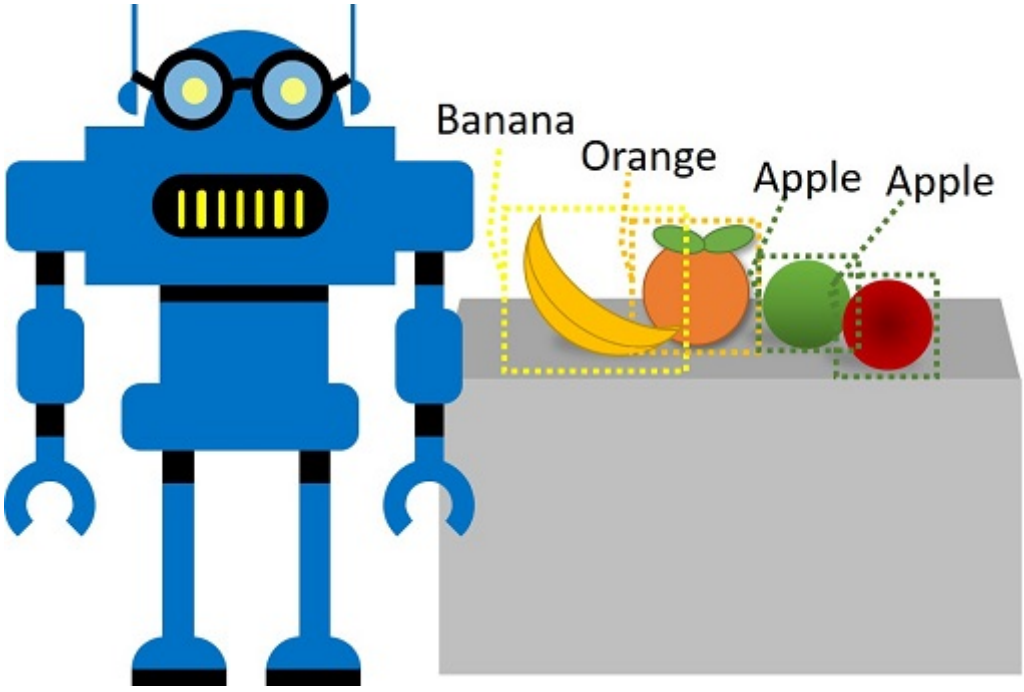
개체 감지

개체 감지는 기계 학습 모델을 학습하여 이미지에서 사물의 개별 인스턴스를 분류하고 그 위치를 표시하는 표시 상자 를 표시하는 컴퓨터 비전이다. 이것을 이미지 분류("이것이 이미지란 무엇인가?"라는 질문에 모델이 답하는)에서 모델에게 "이 이미지에 어떤 물체가 있고, 어디에 있는가?"라고 물어볼 수 있는 솔루션 구축으로 발전했다고 생각할 수 있다.



예를 들어, 식료품점에서 카메라를 사용하여 컨베이어 벨트를 스캔하는 자동 체크 시스템을 구현하기 위해 객체 감지 모델을 사용할 수 있으며, 벨트에 각 품목을 배치하고 개별적으로 스캔할 필요 없이 특정 품목을 식별할 수 있다.

Microsoft Azure에 있는 **Custom Vision** Cognitive서비스는 클라우드에서 자체의 개체 감시 모델을 생성하고 게시하는 기능을 제공한다..

Custom Vision 리소스 생성하기

Custom Vision 서비스를 사용하기 위해서 모델을 학십하는데 사용할 수 있는 Azure 리소스가 필요하고 응용 프로그램이 사용할 수 있도록 게시할수 있는 리소스가 필요하다. 이런 각각의 작업에 같은 리소스를 이용할 수도 있고 또는 리소스에 대한 비용을 따로 처리하기 위해 같은 리전에서 리소스를 분리해서 생성할 수 있다. 각 작업(또는 두 작업 모두)을 위한 리소스는 일반적인 **Cognitive 서비스**리소스가 될 수도 있고 특별한 **Custom Vision** 리소스가 될 수 있다. 다음 순서로 새로운 **Custom Vision**리소스를 만들수 있다(만일 이미 존재하는 리소스가 있다면 그것을 활용할 수도 있다).

1. 새 브라우저 탭에서 https://portal.azure.com에서 (https://portal.azure.com에서) Azure 포털을 열고 Azure 구독과 연결된 Microsoft 계정을 사용하여 로그인한다.
2. **+ 를 선택한다. 리소스 만들기 버튼을 선택하고,, *Custom Vision*을 검색한 뒤에 다음과 같이** Custom Vision** 리소스를 생성한다:
3. **+ 리소스 만들기** 버튼을 선택하고, *Custom Vision*을 검색한 뒤에 다음과 같이 **Custom Vision** 리소스를 생성한다:

    - **만들기 옵션**: 둘 다
    - **구독**: *Azure* 구독 옵션
    - **리소스 그룹**: 유일한 이름으로 리소스 그룹 생성하기
    - **이름**: 유일한 이름*(*영문과 숫자 적용*)*

- **학습 위치**: 가능한 위치 선택
- **교육 가격 책정 계층**: F0
- **예측 위치**: 학습위치와 동일한 지역 선택
- **예측 가격 책정위치**: F0

> **알림**: 여러분들이 이미 F0 Custom Vision서비스를 사용하고 있다면 여기서는 **S0**을 선택함.

1. 리소스가 생성되길 기다린다.

Custom Vision 프로젝트 생성하기

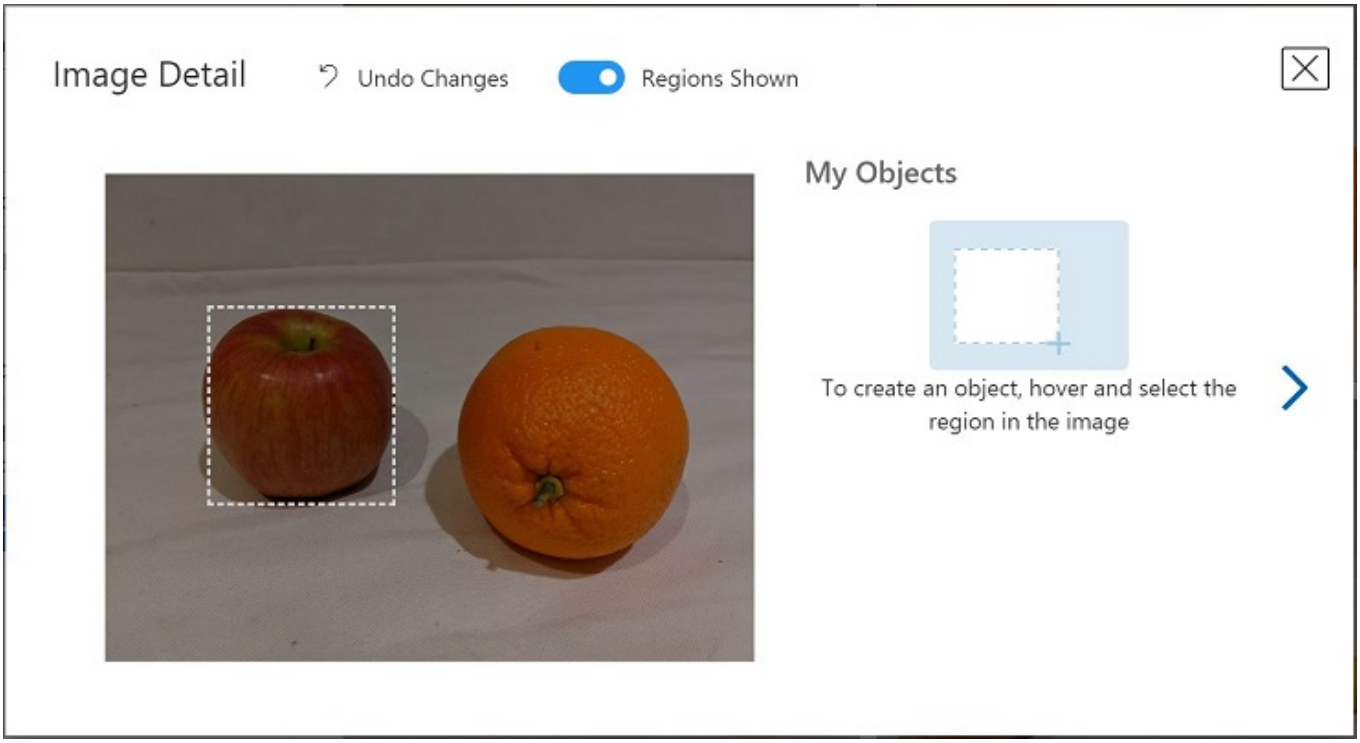
객체 감지 모델을 학습시키기 위해 학습 리소스를 바탕으로 하여 Custom Vision 프로젝트를 생성하여야 한다. 이렇게 하기 위해서는 Custom Vision 포털을 사용하게 될 것이다.

1. 브라우저의 다릅 탭을 열고, Custom Vision portal 사이트 https://customvision.ai (https://customvision.ai) 를 입력한다. 입력 창에 Azure 구독에 사용된 Microsoft 계정 정보를 입력한다..
2. 새로운 프로젝트를 생성할 때 다음과 같이 입력하다.:
   - **Name**: Grocery Detection
   - **Description**: 채소에 대한 이미지 감지.
   - **Resource**: 앞에서 만든 *Custom Vision* 리소스
   - **Project Types**: Object Detection
   - **Domains**: General
3. 프로젝트게 만들어지고 브라우저에서 열릴때까지 기다린다.

이미지 추가하고 태그하기

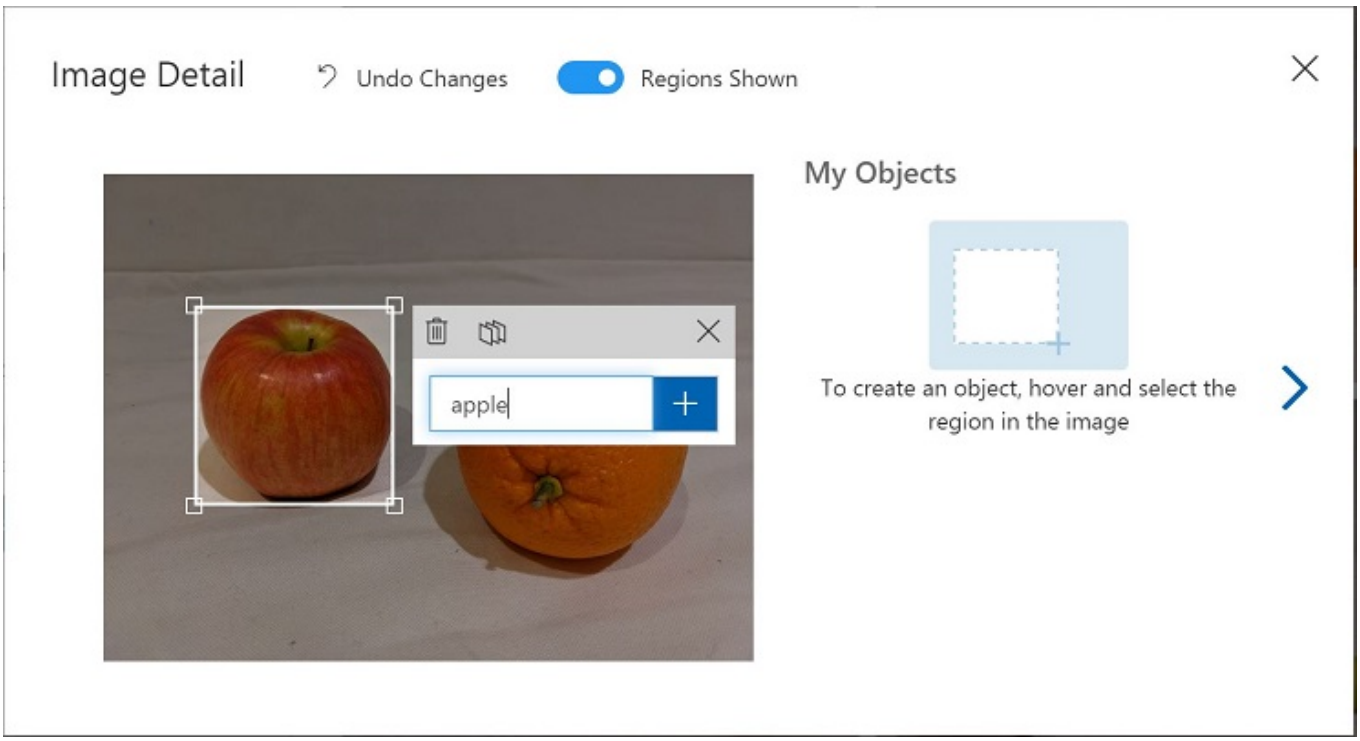
개체 검지 모델을 학습시키기 위해서는 모델이 인지하도록 하기 원하는 클래스들이 포함되어 있는 이미지들을 업로드해야 하고 이미지에 있는 각 개체에 대해서 표시 상자를 이용하여 태그를 달아야 한다.

1. 학습 이미지를 https://aka.ms/fruit-objects (https://aka.ms/fruit-objects) 에서다운로드하고 압축을 해제한다. 압축해제된 폴더에는 과일 이미지들이 포함되어 있다.
2. Custom Vision 포털에서,개체 감지 프로젝트를 선택하고 **Add images** 를 선택한다음 압축해제된 이미지들을 업로드한다.
3. 이미지들이 업로드가 된 후에 첫번째 이미지를 선택하여 연다.
4. 이미지에 있는 각 개체위에 마우스를 올려두고 아래와 같이 자동으로 검지된 영역을 표시할 때까지 기다린다. 그 다음 해당 개체를 서낵하고 개체를 둘러 쌀 수 있도록 크기를 조절한다.

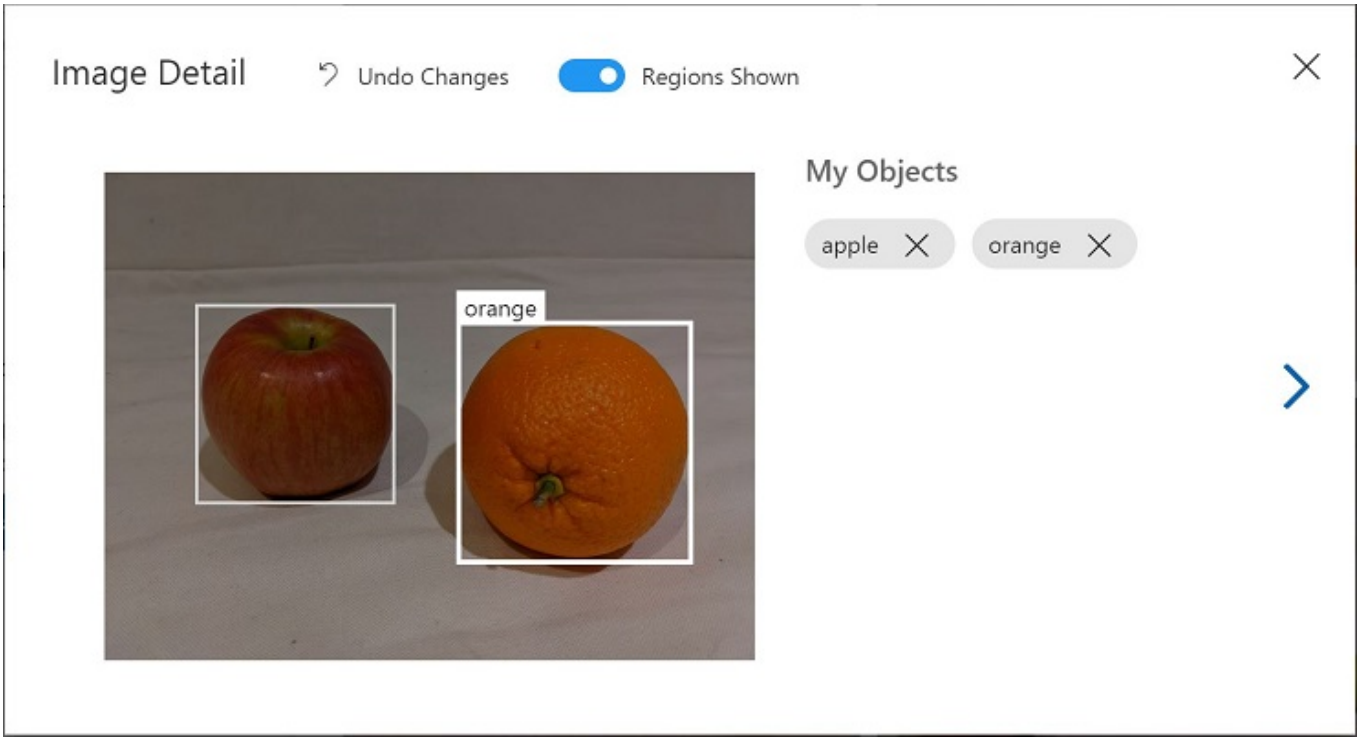


선택적으로 영역을 정하기 위해 개체 주위로 마우스를 드래그할 수 있다.

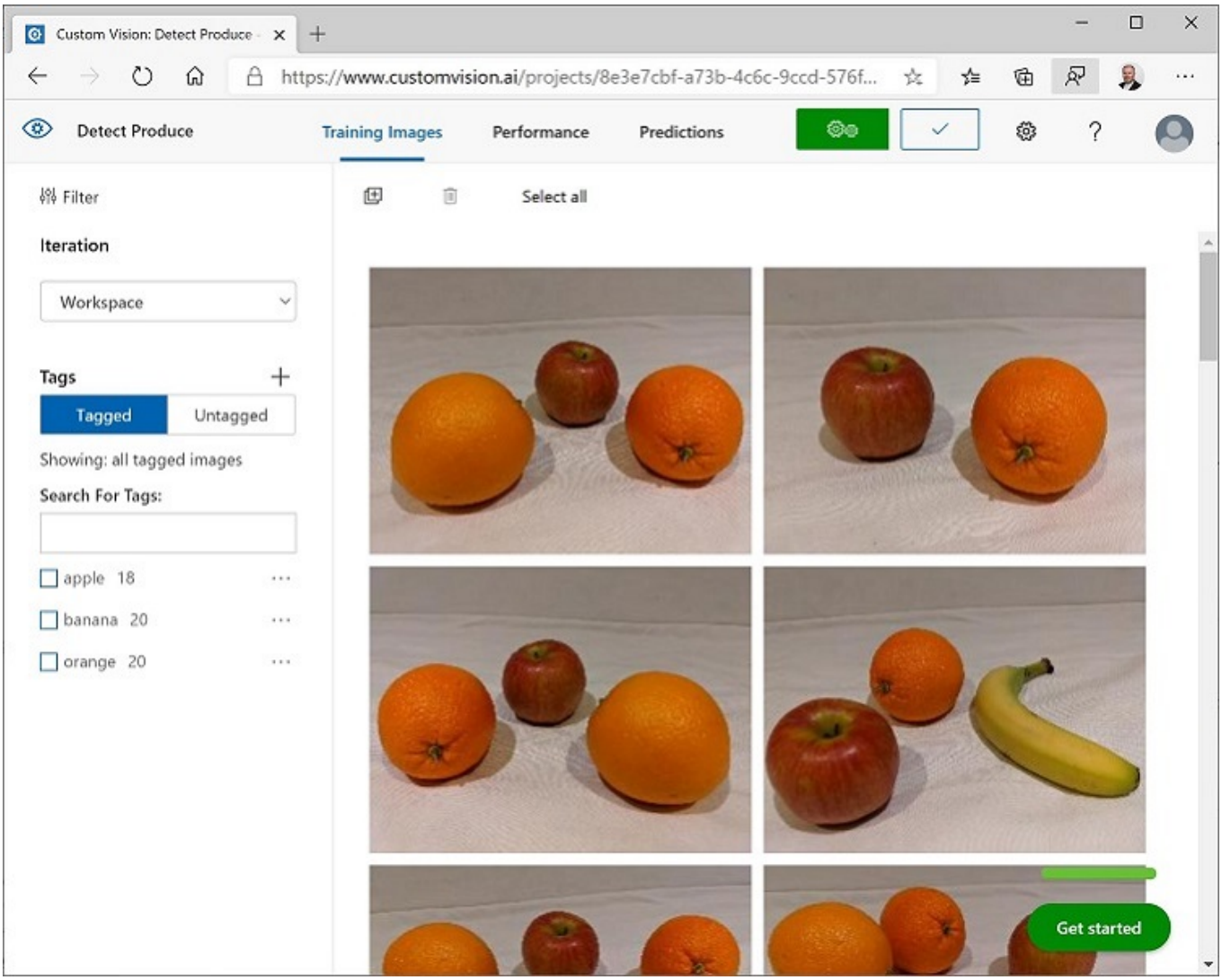
1. 개체 주위로 감싼 후에 개체의 종류에 따라 새로운 태그를 추가한다(아래의 예시처럼 *apple*, *banana*, *orange* 붙인다) .



1. 이미지에 다른 개체에 대해서도 선택하고 태그를 하고 영역을 다시 조절하거나 필요에 따라 새로운 태그를 부여한다.



1. **>** 를 사용하여 다음 오른쪽 이미지로 넘어가서 개체에 태그를 부여한다. 그리고 나서 전체 이미지들에 대해서 apple, banana, 그리고 orange로 태그를 부여한다.
2. 마지막 이미지에 대해서 태그 작업을 마쳤다면 **Image Detail** 편집기를 닫고 **Training Images** 페이지에서, **Tags** 아래, **Tagged** 를 선택하여 태그한 이미지들을 모두 본다.



모델 학습하기 및 테스트하기

이제 프로제트에 있는 이미지들에 태그 작업을 마쳤으므로 모델을 학습시킬 차례다.

1. Custom Vision 프로젝트에서 **Train**을 클릭하여 태그된 이미지를 이용하여 개체 검지 모델 학습시킨다. **Quick Training** 옵셥을 클릭한다.
2. 학습이 끝나길 기다린다(보통 10분 정도 걸릴 것이다). 그리고 나서 *Precision*, *Recall*, 및 *mAP* 성능 매트릭을 확인한다-이 값들은 분류의 예측 정확도를 측정하는데 사용되며 높을수록 좋다.
3. 페이지의 오른쪽 상단에 있는 **Quick Test** 를 클릭하고, **Image URL** 상자에 `https://aka.ms/apple-orange` 을 입력하고 생성하는 예측값을 확인한다. **Quick Test**창을 닫는다.

개체 검지 모델 게시하고 사용하기

이제 학습시킨 모델을 게시하고 클라이언트 응용프로그램에서 사용할 수 있게 되었다.

1. 페이지의 상단 왼편에 있는 ✓ **Publish** 를 클릭하여 학습한 모델을 다음과 같은 내용으로 게시를 한다.

   - **Model name**: detect-produce
   - **Prediction Resource**: *Custom Vision*의 ***prediction*** 리소스.
2. 게시가 끝나면 **Performance**페이지의 상단 오른편의 *settings* (⚙) 아이콘을 클릭하여 프로젝트의 설정을 확인한다. 그리고 나서 **General** (왼쪽)아래에 있는 내용 중 **Project Id** 를 복사하여 아래 코드 셀의 **YOUR_PROJECT_ID** 란에 붙여넣기를 한다.

> _**Note**: 이 학습이 초기에 **Custom Vision** 리소스를 사용하는 대신 **Cognitive Services** 리소스를 사용했다면, 프로젝트 설정의 오른편에 있는 키와 엔트포인트를 복사해서 아래의 코드셀에 붙여 넣고 결과를 확인하기 위해서 실행한다.그렇지 않으면, Custom Vision 예측 리소스에 대한 키와 엔드포인트를 얻기 위해 아래와 같은 작업을 완료한다.

1. **Project Settings** 페이지의 상단 왼편에 있는 *Projects Gallery* (👁) 아이콘을 클릭하여 Custom Vision 포털 홈페이지로 돌아가서 프로젝트 목록들이 나타나도록 한다.
2. Custom Vision 포털 홈페이지에서 윗쪽 오른편에서 *settings* (⚙) 아이콘을 클릭하여 Custom Vision 서비스에 대한 설정을 본다. 그리고 나서 **Resources** 밑에 *prediction* 리소스를 클릭하고 resource (학습리소스가 아님) **키**와 **엔드포인트**값을 복사해서 아래의 **YOUR_KEY** 와 **YOUR_ENDPOINT**를 대체한다..
3. 프로젝트 ID, 키, 엔드포인트값 변수 값을 설정한 후에 **Run cell** (▷) 버튼(셀의 왼편에 있음)을 클릭하여 실행한다

In [ ]:

```
project_id = 'YOUR_PROJECT_ID' # 프로제트 ID로 바꿈
cv_key = 'YOUR_KEY' # 예측 리소스의 첫번째 키로 바꿈
cv_endpoint = 'YOUR_ENDPOINT' # 예측리소스의 엔드포인트로 바꿈

model_name = 'detect-produce' # 이것은 모델을 게시할 때 사용했던 이름과 정확히 동일하게 맞추어야 함(대소문자 포함)!
print('Ready to predict using model {} in project {}'.format(model_name, project_id))
```

Python에서 Custom Vision 서비스를 사용하기 위해 Azure Cognitive Services Custom Vision 패키지를 설치해야 한다.

In [ ]:

```
!pip install azure-cognitiveservices-vision-customvision
```

이제 여러분의 키와 엔드포인트를 사용하여 Custom Vision 클라이드로 Custom Vision 개체 감지 모델에 연결할 수 있다. 아래의 코드 셀을 실행하여 이미지에 있는 개별 개체 항목을 검지하는데 모델을 사용할 수 있다.

> **Note**: 코드의 너무 자세한 내용에 대해서 걱정하지 말라. 이것은 Computer Vision SDK for Python를 사용하여 이미지를 모델로 보내고 검지된 개체에 대한 예측값을 가져오는데 사용된다. 각 예측값에는 클래스 이름(*apple*, *banana*, 또는 *orange*)과 이미지에서 예측된 개체가 검지된 부분을 표시 상자로 나타낸다. 이 정보를 이용하여 이미지에 있는 각 개체를 둘러썬 레이블 상자를 그리게 된다.

In [ ]:

```
from azure.cognitiveservices.vision.customvision.prediction import CustomVisionPredicti
onClient
from msrest.authentication import ApiKeyCredentials
from matplotlib import pyplot as plt
from PIL import Image, ImageDraw, ImageFont
import numpy as np
import os
%matplotlib inline

# 테스트 이미지를 불러오고 Diemension을 저장한다.
test_img_file = os.path.join('data', 'object-detection', 'produce.jpg')
test_img = Image.open(test_img_file)
test_img_h, test_img_w, test_img_ch = np.array(test_img).shape

# 개체 검지 모델을 사용하기 위한 예측 클라이언트를 가져온다.
credentials = ApiKeyCredentials(in_headers={"Prediction-key": cv_key})
predictor = CustomVisionPredictionClient(endpoint=cv_endpoint, credentials=credentials)

print('Detecting objects in {} using model {} in project {}...'.format(test_img_file, model_nam
e, project_id))

# 테스트 이미지에 있는 개체를 검지한다.
with open(test_img_file, mode="rb") as test_data:
    results = predictor.detect_image(project_id, model_name, test_data)

# 결과를 나타내기 위한 그림을 그린다.
fig = plt.figure(figsize=(8, 8))
plt.axis('off')

# 검지된 개체 주위로 상자를 이미지에 나타낸다.
draw = ImageDraw.Draw(test_img)
lineWidth = int(np.array(test_img).shape[1]/100)
object_colors = {
    "apple": "lightgreen",
    "banana": "yellow",
    "orange": "orange"
}
for prediction in results.predictions:
    color = 'white' # 기본값으로 'other' 개체 태그를 부여한다.
    if (prediction.probability*100) > 50:
        if prediction.tag_name in object_colors:
            color = object_colors[prediction.tag_name]
        left = prediction.bounding_box.left * test_img_w
        top = prediction.bounding_box.top * test_img_h
        height = prediction.bounding_box.height * test_img_h
        width =  prediction.bounding_box.width * test_img_w
        points = ((left,top), (left+width,top), (left+width,top+height), (left,top+height),(left
,top))
        draw.line(points, fill=color, width=lineWidth)
        plt.annotate(prediction.tag_name + ": {0:.2f}%".format(prediction.probability * 100),(l
eft,top), backgroundcolor=color)
plt.imshow(test_img)
```

각 예측에 대한 검지된 개체와 확률등을 나타내는 예측 결과의 내용을 확인한다.