**Detect objects in images and generate image tags**

**Object detection**

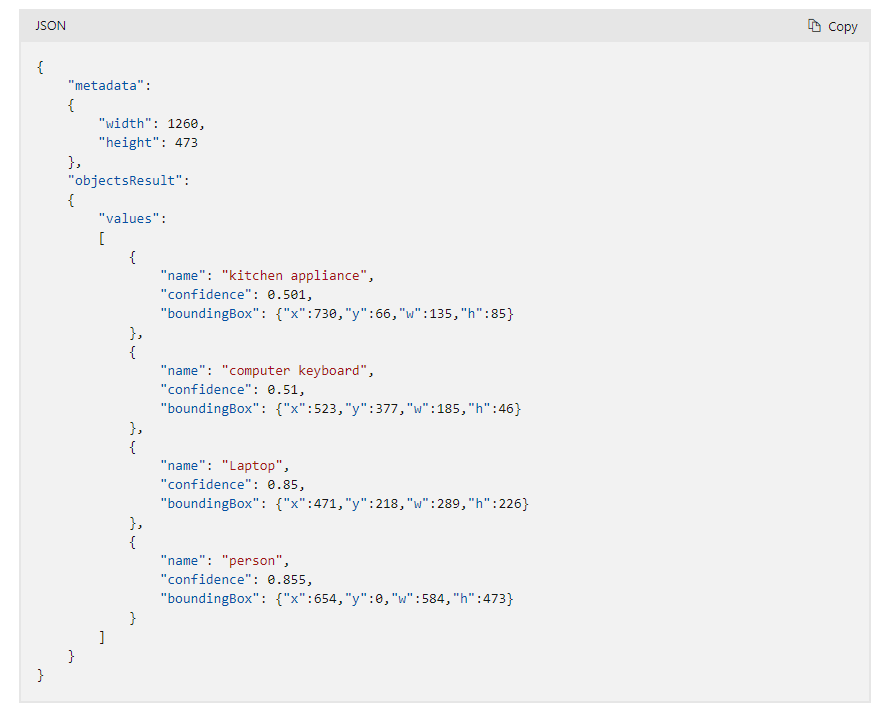
Object detection is similar to [tagging](https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/concept-tag-images-40), but the API returns the bounding box coordinates (in pixels) for each object found in the image. For example, if an image contains a dog, cat, and person, the object detection operation lists those objects with their coordinates in the image. You can use this functionality to process the relationships between the objects in an image. It also lets you determine whether there are multiple instances of the same object in an image.

The object detection function applies tags based on the objects or living things identified in the image. There's no formal relationship between the tagging taxonomy and the object detection taxonomy. At a conceptual level, the object detection function only finds objects and living things, while the tag function can also include contextual terms like *indoor*, which can't be localized with bounding boxes.

**Object detection example**

The following JSON response illustrates what the Image Analysis 4.0 API returns when detecting objects in the example image.





**Limitations**

It's important to note the limitations of object detection so you can avoid or mitigate the effects of false negatives (missed objects) and limited detail.

* Objects are generally not detected if they're small (less than 5% of the image).
* Objects are generally not detected if they're arranged closely together (a stack of plates, for example).
* Objects aren't differentiated by brand or product names (different types of sodas on a store shelf, for example). However, you can get brand information from an image by using the [Brand detection](https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/concept-brand-detection) feature.

**Image tagging**

Image Analysis can return content tags for thousands of recognizable objects, living beings, scenery, and actions that appear in images. Tags aren't organized as a taxonomy and don't have inheritance hierarchies. A collection of content tags forms the foundation for an image [description](https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/concept-describing-images) displayed as human readable language formatted in complete sentences. When tags are ambiguous or not common knowledge, the API response provides hints to clarify the meaning of the tag in context of a known setting.

After you upload an image or specify an image URL, the Analyze Image API can output tags based on the objects, living beings, and actions identified in the image. Tagging is not limited to the main subject, such as a person in the foreground, but also includes the setting (indoor or outdoor), furniture, tools, plants, animals, accessories, gadgets, and so on.

**Image tagging example**

The following JSON response illustrates what Azure AI Vision returns when tagging visual features detected in the example image.

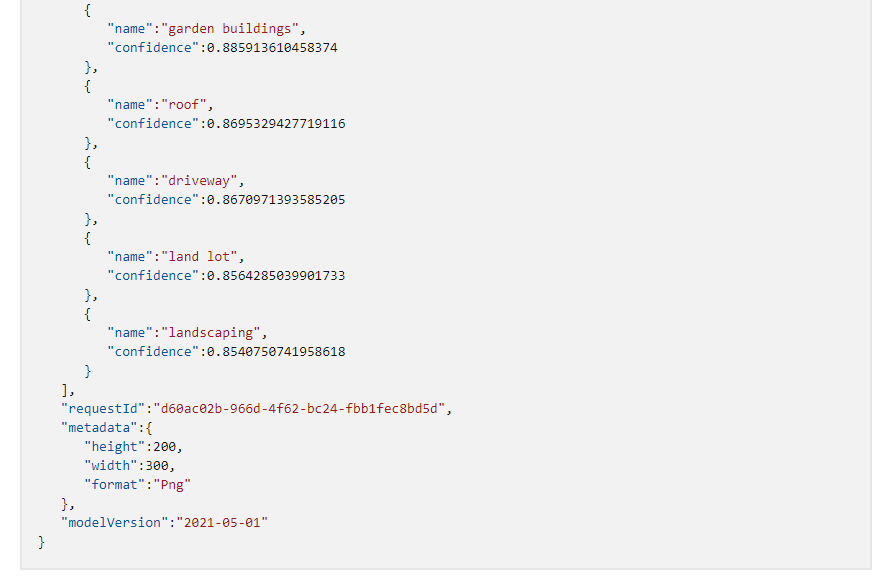
.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



**Use the API**

The tagging feature is part of the [Analyze Image](https://learn.microsoft.com/en-us/rest/api/computervision/analyze-image/analyze-image?view=rest-computervision-v3.2&tabs=HTTP) API. You can call this API through a native SDK or through REST calls. Include Tags in the **visualFeatures** query parameter. Then, when you get the full JSON response, parse the string for the contents of the "tags" section.

**Call the Image Analysis 4.0 Analyze API**

**Prerequisites**

This guide assumes you've followed the steps mentioned in the [quickstart](https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/quickstarts-sdk/image-analysis-client-library-40) page. This means:

* You have [created a Computer Vision resource](https://portal.azure.com/#create/Microsoft.CognitiveServicesComputerVision)and obtained a key and endpoint URL.
* You have the appropriate SDK package installed

**Create and authenticate the client**

To authenticate against the Image Analysis service, you need a Computer Vision key and endpoint URL. This guide assumes that you've defined the environment variables VISION\_KEY and VISION\_ENDPOINT with your key and endpoint.

Start by creating a [ImageAnalysisClient](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.imageanalysisclient) object.

A screenshot of a computer program

Description automatically generated

**Select the image to analyze**

You can select an image by providing a publicly accessible image URL, or by passing binary data to the SDK.

**Image URL**

Create a [Uri](https://learn.microsoft.com/en-us/dotnet/api/system.uri) object for the image you want to analyze.

A white rectangular object with red text

Description automatically generated

**Image buffer**

Alternatively, you can pass the image data to the SDK through a [BinaryData](https://learn.microsoft.com/en-us/dotnet/api/system.binarydata) object. For example, read from a local image file you want to analyze.

A close-up of a computer code

Description automatically generated

**Select visual features**

The Analysis 4.0 API gives you access to all of the service's image analysis features. Choose which operations to do based on your own use case. See the [overview](https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/overview-image-analysis) for a description of each feature. The example in this section adds all of the [available visual features](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.visualfeatures), but for practical usage you likely need fewer.

A white rectangular object with a black border

Description automatically generated

**Select analysis options**

Use an [ImageAnalysisOptions](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.imageanalysisoptions) object to specify various options for the Analyze Image API call.

* **Language**: You can specify the language of the returned data. The language is optional, with the default being English. See [Language support](https://aka.ms/cv-languages) for a list of supported language codes and which visual features are supported for each language.
* **Gender neutral captions**: If you're extracting captions or dense captions (using [VisualFeatures.Caption](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.visualfeatures) or [VisualFeatures.DenseCaptions](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.visualfeatures)), you can ask for gender neutral captions. Gender neutral captions are optional, with the default being gendered captions. For example, in English, when you select gender neutral captions, terms like **woman** or **man** are replaced with **person**, and **boy** or **girl** are replaced with **child**.
* **Crop aspect ratio**: An aspect ratio is calculated by dividing the target crop width by the height. Supported values are from 0.75 to 1.8 (inclusive). Setting this property is only relevant when [VisualFeatures.SmartCrops](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.visualfeatures) was selected as part the visual feature list. If you select [VisualFeatures.SmartCrops](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.visualfeatures) but don't specify aspect ratios, the service returns one crop suggestion with an aspect ratio it sees fit. In this case, the aspect ratio is between 0.5 and 2.0 (inclusive).

A screenshot of a computer

Description automatically generated

**Call the Analyze API**

Call the [Analyze](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.imageanalysisclient#methods) method on the [ImageAnalysisClient](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.imageanalysisclient) object, as shown here. The call is synchronous, and blocks execution until the service returns the results or an error occurred. Alternatively, you can call the non-blocking [AnalyzeAsync](https://learn.microsoft.com/en-us/dotnet/api/azure.ai.vision.imageanalysis.imageanalysisclient#methods) method.

Use the input objects created in the above sections. To analyze from an image buffer instead of URL, replace imageURL in the method call with the imageData variable.

A screenshot of a computer

Description automatically generated

**Get results from the service**

The following code shows you how to parse the results of the various Analyze operations.

 A screenshot of a computer program

Description automatically generated

 A screenshot of a computer

Description automatically generated

**Exception handling**

When you interact with Image Analysis using the .NET SDK, any response from the service that doesn't have a 200 (success) status code results in an exception being thrown. For example, if you try to analyze an image that is not accessible due to a broken URL, a 400 status is returned, indicating a bad request, and a corresponding exception is thrown.

In the following snippet, errors are handled gracefully by catching the exception and displaying additional information about the error.

A screenshot of a computer program

Description automatically generated