

[AI & Machine Learning Products](https://cloud.google.com/products/machine-learning/) (<https://cloud.google.com/products/machine-learning/>)

[Cloud Vision API](https://cloud.google.com/vision/) (<https://cloud.google.com/vision/>)

[Documentation](https://cloud.google.com/vision/docs/) (<https://cloud.google.com/vision/docs/>) [Guides](#)

Cloud Vision API

Objectives

In this sample, you'll use the Google Cloud Vision API to detect faces in an image. To prove to yourself that the faces were detected correctly, you'll then use that data to draw a box around each face.

Viewing Code Samples: Most of the code samples in this tutorial are taken from larger code files located in [GitHub](https://github.com/) (<https://github.com/>). You can view and download the complete file from which a code sample is taken by clicking the "View on GitHub" button provided above a sample.

Costs

This tutorial uses billable components of Cloud Platform, including:

- Google Cloud Vision API

Use the [Pricing Calculator](https://cloud.google.com/products/calculator/) (<https://cloud.google.com/products/calculator/>) to generate a cost estimate based on your projected usage. New Cloud Platform users might be eligible for a [free trial](https://cloud.google.com/free-trial) (<https://cloud.google.com/free-trial>).

Before you begin

1. [Sign in](https://accounts.google.com/Login) (<https://accounts.google.com/Login>) to your Google Account.

If you don't already have one, [sign up for a new account](https://accounts.google.com/SignUp) (<https://accounts.google.com/SignUp>).

2. In the Cloud Console, on the project selector page, select or create a Google Cloud project.

Note: If you don't plan to keep the resources that you create in this procedure, create a project instead of selecting an existing project. After you finish these steps, you can delete the project, removing all resources associated with the project.

[GO TO THE PROJECT SELECTOR PAGE](https://console.cloud.google.com/projectselect) ([HTTPS://CONSOLE.CLOUD.GOOGLE.COM/PROJECTSELECT](https://console.cloud.google.com/projectselect))

3. Make sure that billing is enabled for your Google Cloud project. [Learn how to confirm billing is enabled for your project](https://cloud.google.com/billing/docs/how-to/modify-project) (<https://cloud.google.com/billing/docs/how-to/modify-project>).

4. Enable the Google Cloud Vision API.

[ENABLE THE API](https://console.cloud.google.com/flows/enableapi?apiid=vision.googleapis.com) ([HTTPS://CONSOLE.CLOUD.GOOGLE.COM/FLOWS/ENABLEAPI?APIID=VISION.GOOGLE](https://console.cloud.google.com/flows/enableapi?apiid=vision.googleapis.com))

5. Set up your environment for using [Application Default Credentials](https://cloud.google.com/vision/docs/common/auth#authenticating_with_application_default_credentials) (https://cloud.google.com/vision/docs/common/auth#authenticating_with_application_default_credentials)

6. Set up language-specific tasks and tools:

C#	JAVA	NODE.JS	MORE ▾
<ul style="list-style-type: none"> • Install the Google Client Library (https://cloud.google.com/vision/docs/reference/libraries#installing_the_client_library) • Install Visual Studio 2012/2015. 			

Create the service object

To access Google APIs using the official client SDKs, you create a service object based on the API's discovery document, which describes the API to the SDK. You'll need to fetch it from the Vision API's discovery service, using your credentials:

C#	JAVA	NODE.JS	MORE ▾
<p>vision/api/DetectFaces/DetectFaces.cs (https://github.com/GoogleCloudPlatform/dotnet-docs-samples/blob/master/vision/api/DetectFaces/DetectFaces.cs)</p>			

```
)UDPLATFORM/DOTNET-DOCS-SAMPLES/BLOB/MASTER/VISION/API/DETECTFACES/DETECTFACES.CS)
```

```
using Google.Cloud.Vision.V1;  
using System;  
using System.Linq;  
  
var client = ImageAnnotatorClient.Create();
```

Send a face detection request

To construct a request to the Vision API, first consult the [API documentation](https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate) (<https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate>). In this case, you'll be asking the `images` resource to `annotate` your image. A request to this API takes the form of an object with a `requests` list. Each item in this list contains two bits of information:

- The base64-encoded image data
- A list of features you'd like annotated about that image.

For this example, you'll simply request `FACE_DETECTION` annotation on one image, and return the relevant portion of the response:

C#

JAVA

NODE.JS

MORE ▾

```
vision/api/DetectFaces/DetectFaces.cs  
(https://github.com/GoogleCloudPlatform/dotnet-docs-  
samples/blob/master/vision/api/DetectFaces/DetectFaces.cs)
```

```
)UDPLATFORM/DOTNET-DOCS-SAMPLES/BLOB/MASTER/VISION/API/DETECTFACES/DETECTFACES.CS)
```

```
var response = client.DetectFaces(Image.FromFile(args[0]));
```

Process the response

Congratulations - you've detected the faces in your image! The [response](https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate#AnnotateImageResponse) (<https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate#AnnotateImageResponse>) to

our face annotation request includes a bunch of [metadata](#)

(<https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate#FaceAnnotation>) about the detected faces, which include [coordinates of a polygon](#)

(<https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate#BoundingPoly>)

encompassing the face. At this point, though, this is only a list of numbers. Let's use them to confirm that you have, in fact, found the faces in your image. We'll draw polygons onto a copy of the image, using the coordinates returned by the Vision API:

C# JAVA NODE.JS MORE ▾

[vision/api/DetectFaces/DetectFaces.cs](#)
(<https://github.com/GoogleCloudPlatform/dotnet-docs-samples/blob/master/vision/api/DetectFaces/DetectFaces.cs>)

UDPLATFORM/DOTNET-DOCS-SAMPLES/BLOB/MASTER/VISION/API/DETECTFACES/DETECTFACES.CS)

```
using (var image = System.Drawing.Image.FromFile(args[0]))
using (System.Drawing.Graphics g = System.Drawing.Graphics.FromImage(image))
{
    var cyanPen = new System.Drawing.Pen(System.Drawing.Color.Cyan, 3);
    foreach (var annotation in response)
    {
        g.DrawPolygon(cyanPen, annotation.BoundingPoly.Vertices.Select(
            (vertex) => new System.Drawing.Point(vertex.X, vertex.Y)).ToArray());
        // ...
    }
    // ...
}
```

Put it all together

C# JAVA NODE.JS MORE ▾

[vision/api/DetectFaces/DetectFaces.cs](#)
(<https://github.com/GoogleCloudPlatform/dotnet-docs-samples/blob/master/vision/api/DetectFaces/DetectFaces.cs>)

UDPLATFORM/DOTNET-DOCS-SAMPLES/BLOB/MASTER/VISION/API/DETECTFACES/DETECTFACES.CS)

```
static readonly string s_usage = @"dotnet run image-file
```

Use the Google Cloud Vision API to detect faces in the image.

Writes an output file called image-file.faces.

```
";
```

```
public static void Main(string[] args)
```

```
{
```

```
    if (args.Length < 1)
```

```
    {
```

```
        Console.WriteLine(s_usage);
```

```
        return;
```

```
    }
```

```
    var client = ImageAnnotatorClient.Create();
```

```
    var response = client.DetectFaces(Image.FromFile(args[0]));
```

```
    int numberOfFacesFound = 0;
```

```
    using (var image = System.Drawing.Image.FromFile(args[0]))
```

```
    using (System.Drawing.Graphics g = System.Drawing.Graphics.FromImage(image))
```

```
    {
```

```
        var cyanPen = new System.Drawing.Pen(System.Drawing.Color.Cyan, 3);
```

```
        foreach (var annotation in response)
```

```
        {
```

```
            g.DrawPolygon(cyanPen, annotation.BoundingPoly.Vertices.Select(
```

```
                (vertex) => new System.Drawing.Point(vertex.X, vertex.Y)).ToArray(
```

```
                // ...
```

```
            }
```

```
        // ...
```

```
    }
```

```
    // ...
```

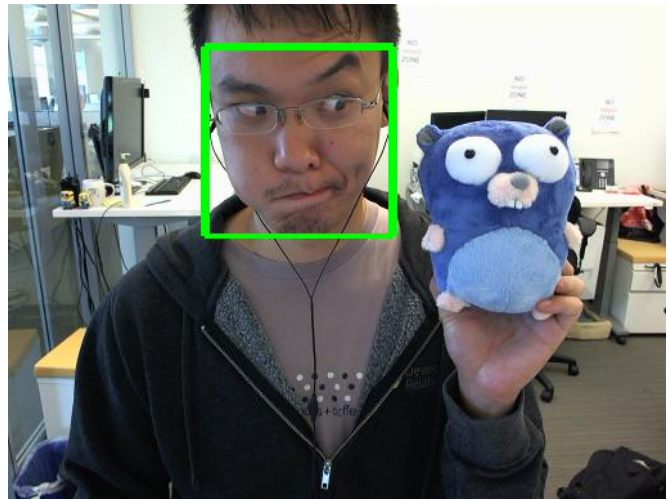
```
}
```

To build the sample, open the Solution file `dotnet-doc-samples/vision/api/Vision.sln` in Visual Studio and build the solution.

To run the sample:

```
C:\...\> cd dotnet-docs-samples\vision\api\DetectFaces\bin\Debug
```

```
C:\...\bin\Debug> DetectFaces ..\..\..\VisionTest\data\face.png
```



Cleaning up

To avoid incurring charges to your Google Cloud Platform account for the resources used in this tutorial:


Caution: Deleting a project has the following effects:

- **Everything in the project is deleted.** If you used an existing project for this tutorial, when you delete it, you also delete any other work you've done in the project.
- **Custom project IDs are lost.** When you created this project, you might have created a custom project ID that you want to use in the future. To preserve the URLs that use the project ID, such as an **appspot.com** URL, delete selected resources inside the project instead of deleting the whole project.

If you plan to explore multiple tutorials and quickstarts, reusing projects can help you avoid exceeding project quota limits.

1. In the Cloud Console, go to the **Manage resources** page.

[GO TO THE MANAGE RESOURCES PAGE \(HTTPS://CONSOLE.CLOUD.GOOGLE.COM/IAM-ADMIN/PROJ...](https://console.cloud.google.com/iam-admin/projects)

2. In the project list, select the project you want to delete and click **Delete** .

3. In the dialog, type the project ID, and then click **Shut down** to delete the project.

Except as otherwise noted, the content of this page is licensed under the [Creative Commons Attribution 4.0 License](https://creativecommons.org/licenses/by/4.0/) (https://creativecommons.org/licenses/by/4.0/), and code samples are licensed under the [Apache 2.0 License](https://www.apache.org/licenses/LICENSE-2.0) (https://www.apache.org/licenses/LICENSE-2.0). For details, see our [Site Policies](https://developers.google.com/terms/site-policies) (https://developers.google.com/terms/site-policies). Java is a registered trademark of Oracle and/or its affiliates.

Last updated January 10, 2020.