# Dlib FaceLandmark Detector 2.0.0


---

## 🚀 About **Dlib FaceLandmark Detector**

**Dlib FaceLandmark Detector** is an **Assets Plugin** for using Dlib19.7 C++ Library from within **Unity** cross-platform game engine that can perform **Object Detection** and **Shape Prediction**.

> ***Dlib FaceLandmark Detector*** uses ***Dlib*** under ***Boost Software License***; see Third-Party `Notices.txt` file in package for details.
> ***The Shape Predictor model files included with this asset are available for commercial use.***

---

## ✨ Our Asset Features

## 🌐 Cross Platform

- Compatible with multiple platforms, allowing for app development on major platforms
- **Mobile**: **iOS** & **Android**
- **Desktop**: **Windows** & **macOS** & **Linux** Standalone support
- **Web**: **WebGL** support
- **ChromeOS**: **ChromeOS** support
- **XR**: **visionOS** support (beta) & **Windows10 UWP** support
- **Editor**: Support for preview in the **Editor**

## 🔍 Object Detection & Shape Prediction

- **Object Detection**: Detect **frontal human faces** and other objects using **Histogram of Oriented Gradients (HOG)** feature combined with a linear classifier, an image pyramid, and sliding window detection scheme.
- **Shape Prediction**: Detect **face landmarks (68 points, 17 points, 6 points)** using dlib's implementation of the paper (One Millisecond Face Alignment with an Ensemble of Regression Trees by Vahid Kazemi and Josephine Sullivan, CVPR 2014).
- **Custom Training**: You can train your own object detector and shape predictor models using dlib's machine learning tools. (Object Detector Training Tool, Shape Predictor Training Tool)

## 🛠️ Easy to Use

- Support for `Texture2D`, `WebCamTexture`, and `Image byte array` input.
- When combined with OpenCV for Unity, you can also input from **OpenCV's** `Mat` class for enhanced image processing capabilities.
- Helper functions for easy integration with **Unity**.
- `FaceLandmarkDetector` class implement `IDisposable`, allowing you to manage resources using the `using` statement.

## 📚 Include Many Examples

- Includes a wide variety of example usage scenarios, which consist of **scene files** and **script codes**. By running these example applications, **you can learn how to develop Dlib applications effectively**.
- **Basic Examples**: Texture2D Example, WebCamTexture Example, Benchmark Example.
- **Advanced Examples**: Advanced examples using **OpenCV for Unity**. (requires OpenCV for Unity)
  **DlibFaceLandmarkDetector Examples** (GitHub)
  **EnoxSoftware repositories** (GitHub)

## 🎨 Visual Scripting Support

- By utilizing the **VisualScripting With DlibFaceLandmarkDetector Example**, you can leverage all the methods available in **DlibFaceLandmarkDetector** within the **Unity's Visual Scripting development environment**.
  **VisualScripting With DlibFaceLandmarkDetector Example** (GitHub)

---

# 🔗 Quick Links

- **Official Site** - Main product page
- **Example Code** - GitHub repository with examples
- **Dlib Official Site** - Dlib C++ Library documentation
- **Forum** - Community discussions
- **API Reference** - Complete API documentation

---

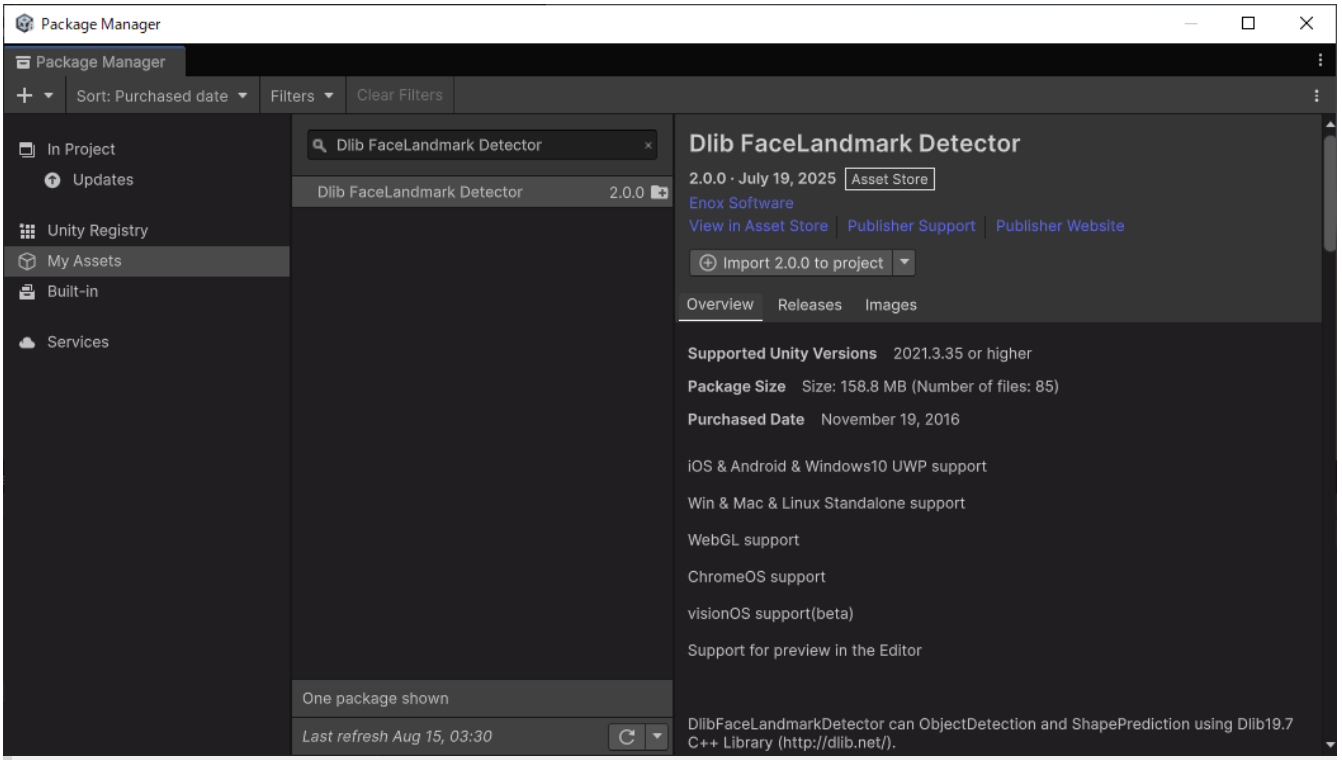
# 📋 System Requirements

| Platform | Requirement |
|---|---|
| **Windows Standalone & Editor** | **Windows 8** or later |
| **Mac Standalone & Editor** | **macOS 10.13** or later |
| **Linux Standalone & Editor** | **Ubuntu 18.04** or later |
| **Android** | API level **21** or later |
| **iOS** | **iOS** Version **12.0** or later |
| **visionOS (beta)** | **visionOS 1** or later |

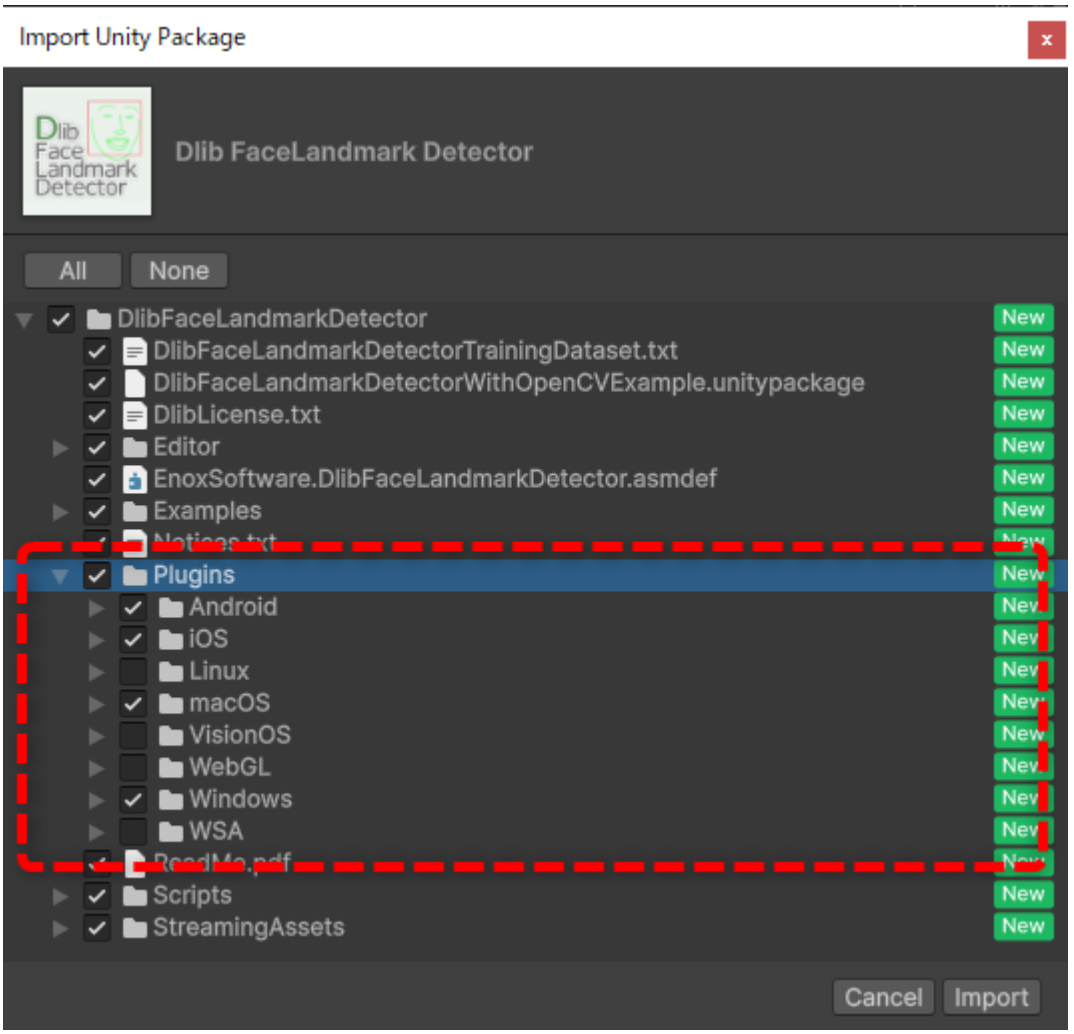
---

# Setup Guide for Getting Started (Setup Guide Video)

1. Import **Dlib FaceLandmark Detector** from the `Package Manager`.
   > ⚠ **Important**: If your project already has a previous version of the `Assets/DlibFaceLandmarkDetector` folder, issues may occur. Please delete the `Assets/DlibFaceLandmarkDetector` folder first before importing the new version.




> At that time, by unchecking platforms other than those planned to be supported in your development project, unnecessary plugin files for those platforms will not be imported, reducing the project size.

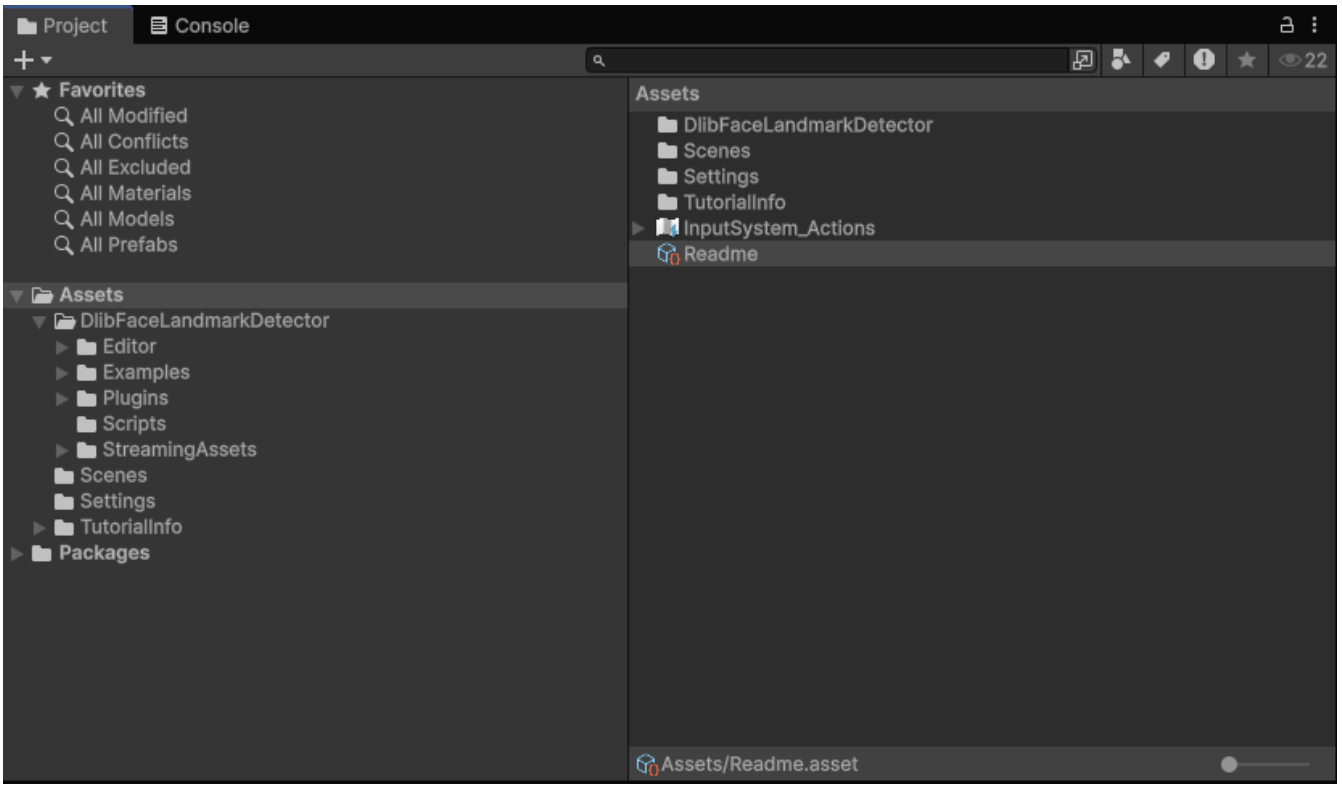



Even after import, you can easily reduce unnecessary files in the `Assets/OpenCVForUnity`

folder using the `Project Size Reducer`. Particularly, by removing plugin files for platforms not planned to be supported in your development project, you can significantly reduce the project size. For details, please see Project Size Reducer Window Description.

2. When the import is complete, the `Setup Tools` window will automatically appear. In the `Setup Tools` window, you can easily perform various settings. For details, please see Setup Tools Window Description.
3. Setup is now complete.

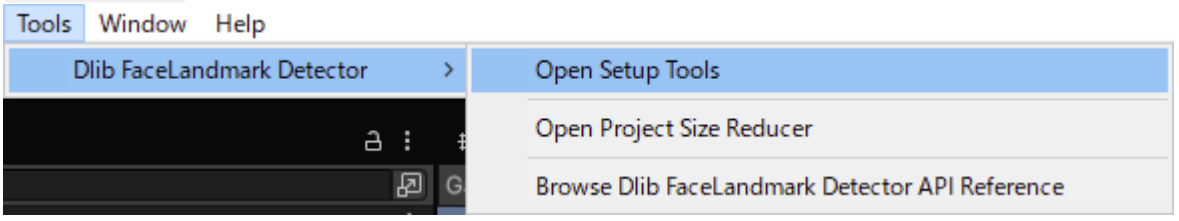
> As a starting point for development using this asset, we recommend first referring to the API usage examples in the Example Scenes.

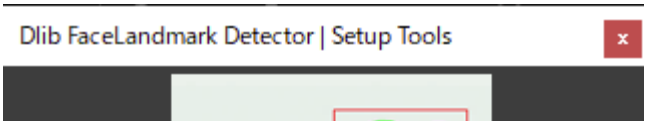


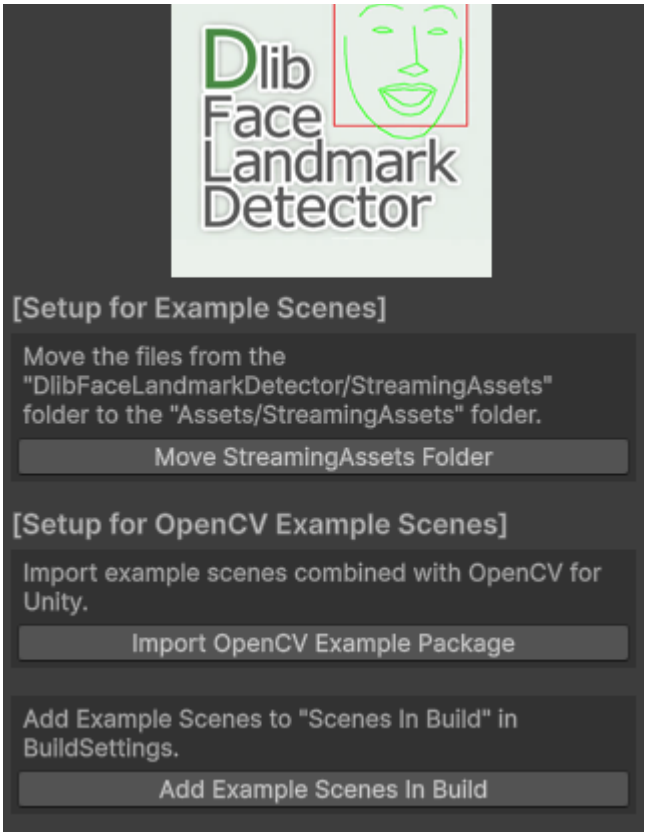

---

# Example Scenes Setup Guide (Setup Guide Video)

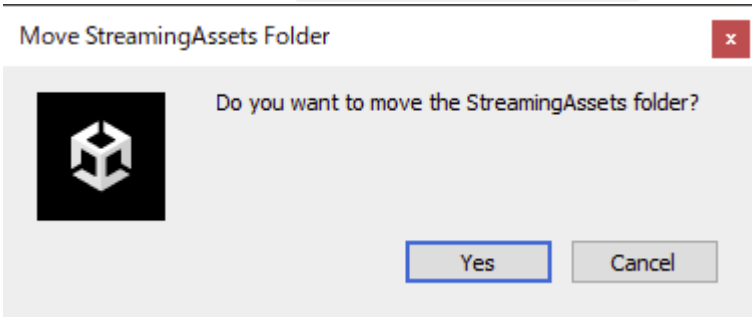
1. Complete the Setup Guide for Getting Started.
2. Select the menu item `Tools > Dlib FaceLandmark Detector > Open Setup Tools` to open the `Setup Tools` window.





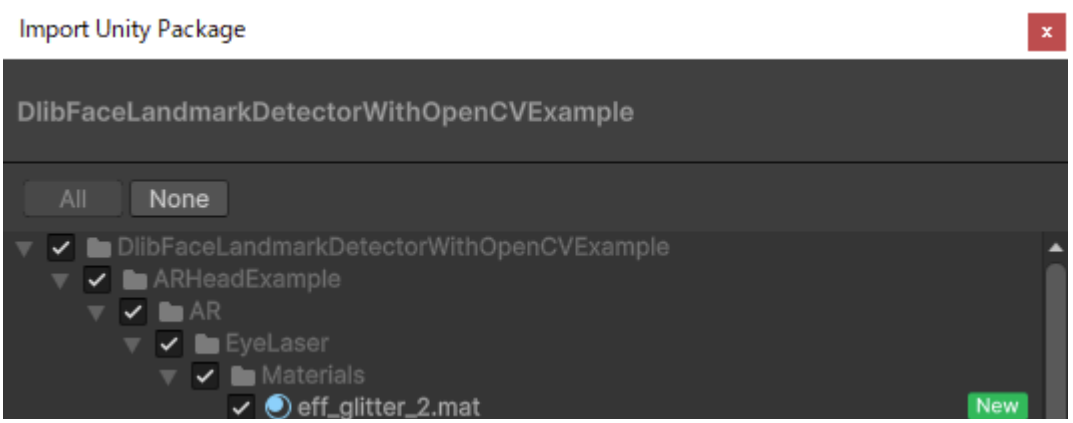

3. Click the `Move StreamingAssets Folder` button in the `Setup Tools` window to move the `Assets/DlibFaceLandmarkDetector/StreamingAssets/DlibFaceLandmarkDetectorExamples` folder to directly under the `Assets/StreamingAssets` folder.

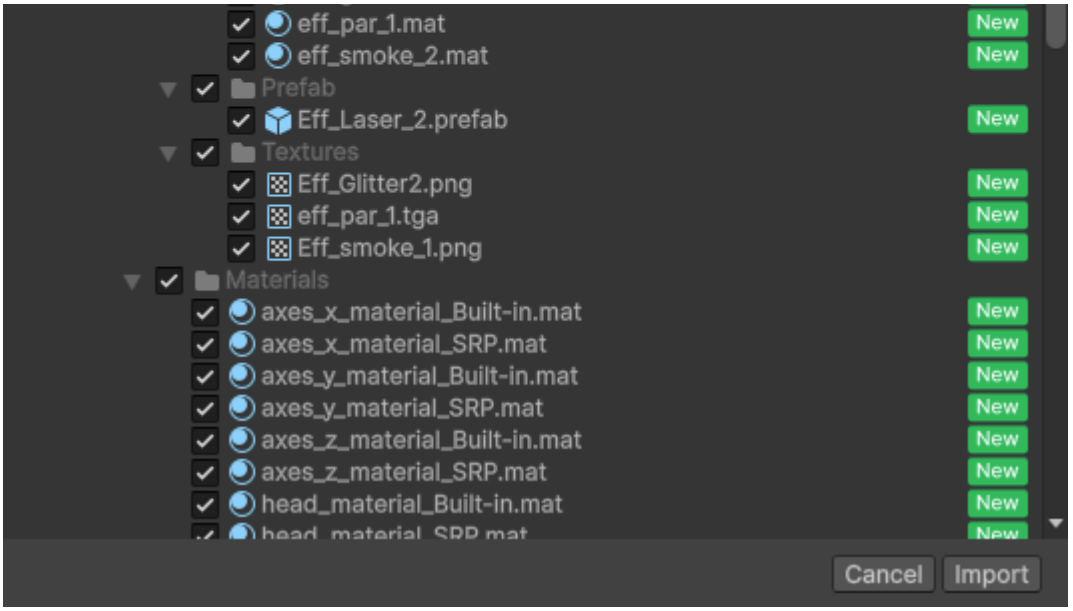



4. **Optional**

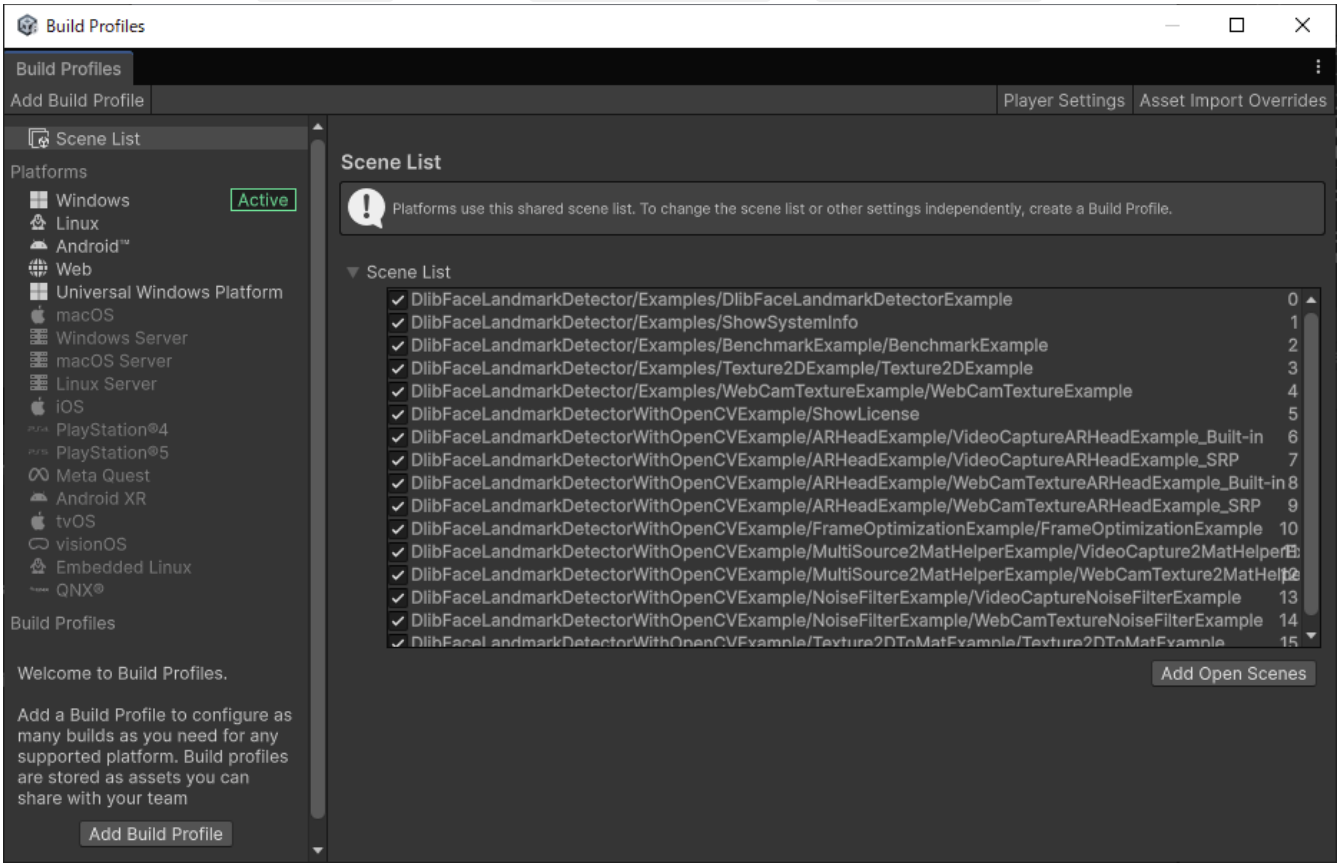
> This step is only required if you want to set up **Advanced** examples using **OpenCV for Unity**.

i. Import **OpenCV for Unity** from the `Package Manager`.
ii. Click the `Import OpenCV Example Package` button in the `Setup Tools` window to import the `DlibFaceLandmarkDetectorWithOpenCVExample.unitypackage`.





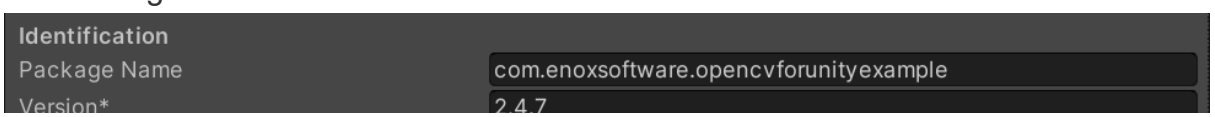

5. Click the `Add Example Scenes in Build` button in the `Setup Tools` window to add all Example Scene files in the `Examples` folder to `Scenes In Build` in `Build Settings`.

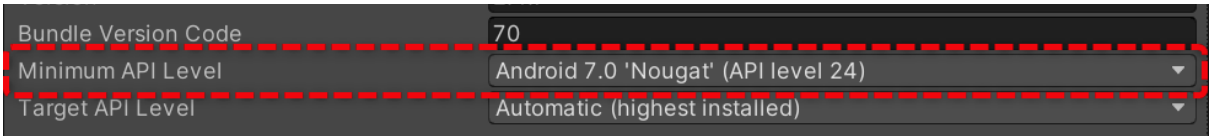



6. When building Example Scenes and running them on devices, settings may need to be changed depending on the platform.
   - **Android**

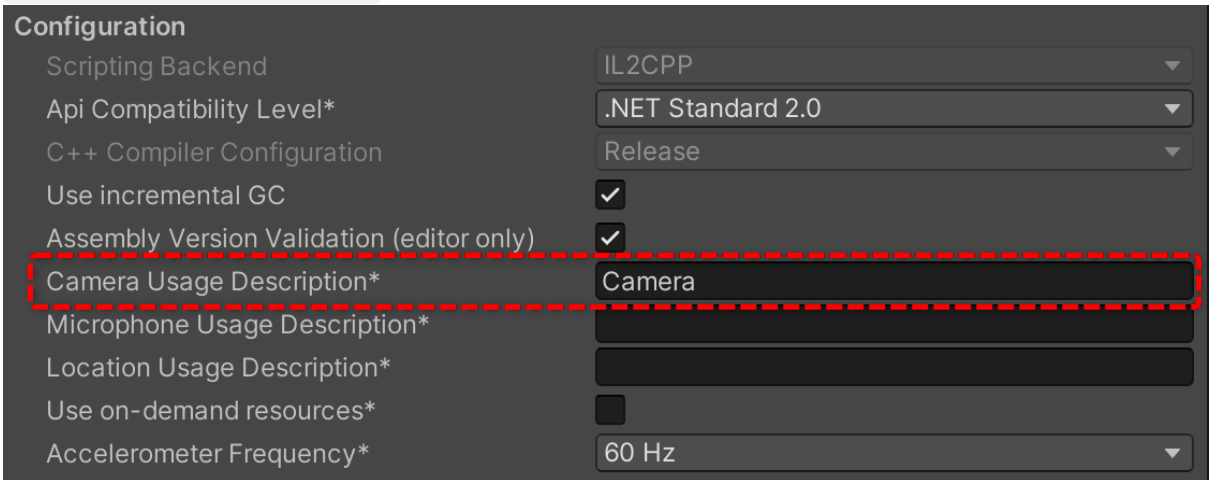
     > This step is only required if you want to set up **Advanced** examples using **OpenCV for Unity**.

     - Set `Player Settings` → `Other Settings` → `Identification` → `Minimum API Level` to **24** or higher.

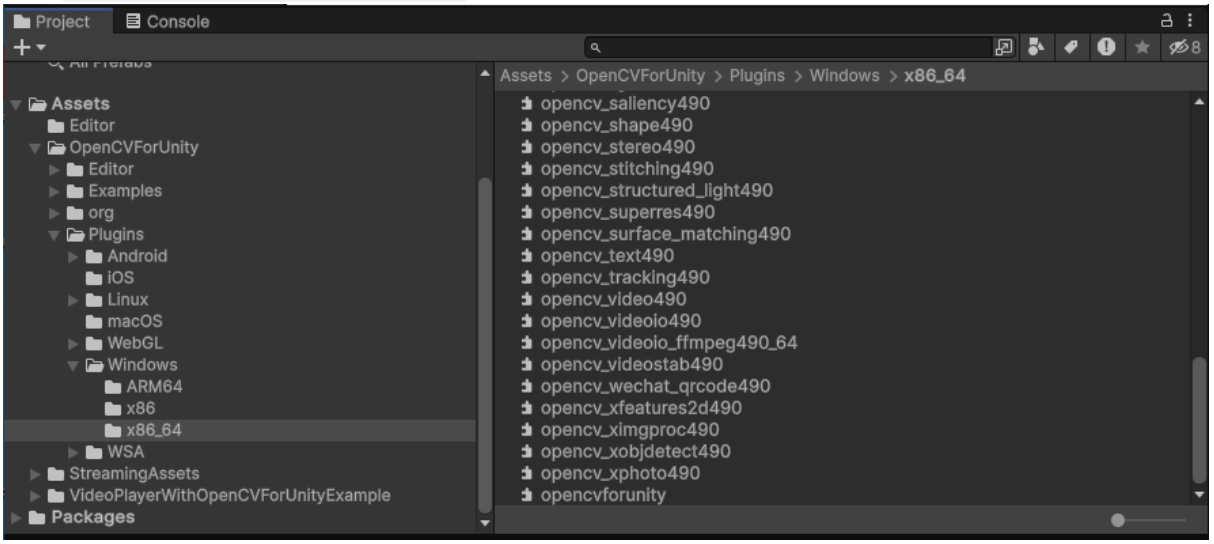





- **iOS**
  - Set `Player Settings` → `Other Settings` → `Configuration` → `Camera Usage Description`.

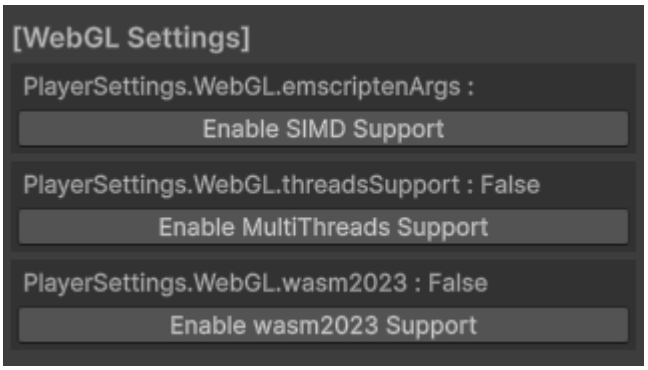



  - Set `Target minimum iOS Version` to **12.0** or higher.




- **WebGL**
  - If you want to enable multi-threading or SIMD optimization, click the `Enable MultiThreads Support` button or `Enable SIMD Support` button in the `Setup Tools` window. Setup Tools Window Description

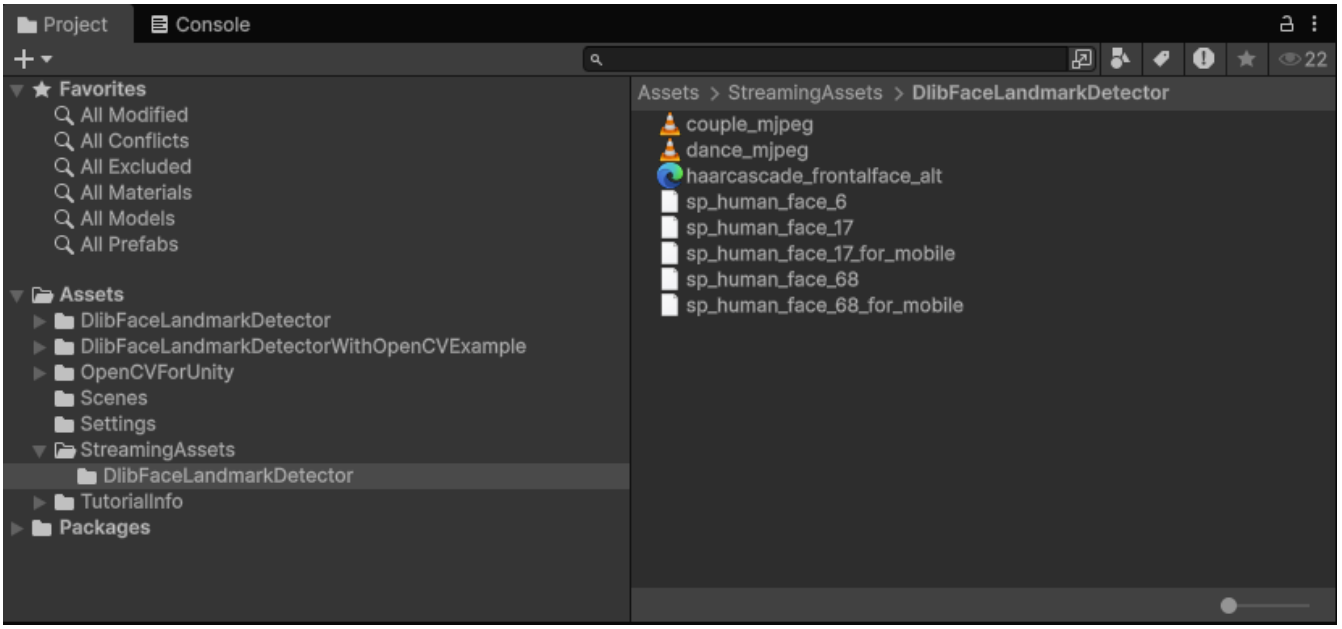
> ⚠ The browser you run must support multi-threading and SIMD.




- **Windows10 UWP**

- If using the `WebCamTexture` class, check `WebCam` in `Player Settings` → `PublishingSettings` → `Capabilities`.


7. Setup is now complete.




For each Example, please see Example Scenes Description.

---

# Example Scenes Description

The Example Scenes in **Dlib FaceLandmark Detector** are structured to allow you to learn various **Dlib** features progressively. They are divided into the following **2** categories, and by starting from **Basic** and progressing to **Advanced**, you can learn **Dlib** features practically.

## `Basic` Folder

Contains Examples related to **basic functionality and setup**:

- **Texture2DExample**: Example of detecting faces and face landmarks in Unity's `Texture2D` class
- **WebCamTextureExample**: Basic example of detecting faces and face landmarks in Unity's

`WebCamTexture` class video

- **BenchmarkExample**: Example of measuring performance for face detection and face landmark detection

## `Advanced` Folder

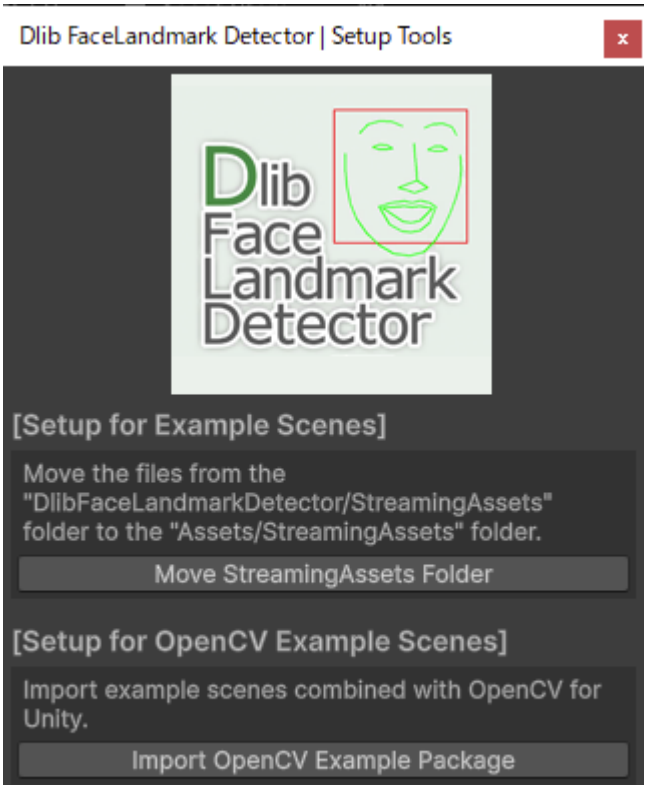
Contains **advanced implementation examples** using **OpenCV for Unity**:

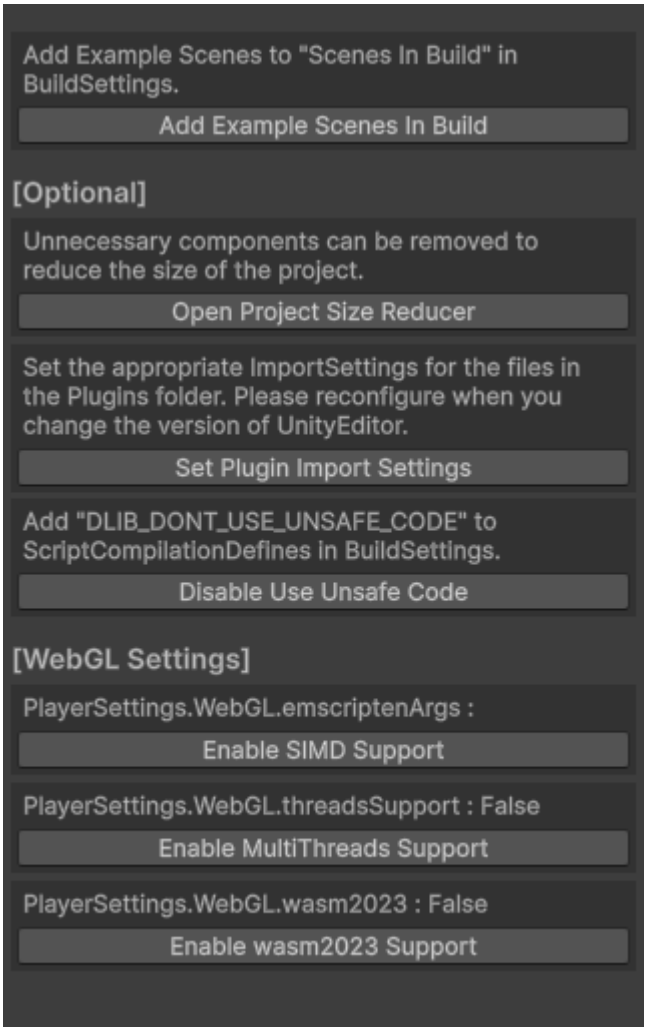
- **Texture2DToMatExample**: Example of detecting faces and face landmarks using OpenCV's `Mat` class
- **MultiSource2MatHelperExample**: Example of using the `MultiSource2MatHelper` class to efficiently detect faces and face landmarks from multiple video sources
- **ARHeadExample**: Example of Augmented Reality head tracking using face landmarks with head pose estimation and 3D face mesh rendering
- **NoiseFilterExample**: Example of applying various noise filters including `LowPassFilter`, `KalmanFilter`, and `OpticalFlowFilter` to improve face landmark detection stability
- **FrameOptimizationExample**: Example of frame optimization techniques including frame downscaling and frame skipping to improve performance

---

## `Setup Tools` Window Description

Select the menu item `Tools > Dlib FaceLandmark Detector > Open Setup Tools` to open the `Setup Tools` window.






This window is a collection of tools to make **Dlib FaceLandmark Detector** project setup easier. Each section has a specific role and is structured as follows.

## Setup for Example Scenes

This section contains tools for preparing and setting up files necessary to run Example Scenes.

- `Move StreamingAssets Folder`
  Moves downloaded files from the `Assets/DlibFaceLandmarkDetector/StreamingAssets` folder to the `Assets/StreamingAssets` folder.

## Setup for OpenCV Example Scenes

This section contains tools for preparing and setting up files necessary to run Advanced Example Scenes.

- `Import OpenCV Example Package`
  Import Advanced Example Scenes combined with OpenCV for Unity.
- `Add Example Scenes In Build`
  Automatically adds Example Scenes to `Scenes In Build` in `Build Settings`.

## Optional

This section contains convenient tools for optional use, such as project size reduction.

- `Open Project Size Reducer`
  Opens the `Project Size Reducer` window that removes unnecessary components and reduces project size. For details, please see Project Size Reducer Window Description.
- `Set Plugin Import Settings`
  Sets import settings for files in the `Plugins` folder appropriately in bulk.
  > Please execute when changing Unity editor version or settings.

- `Disable Use Unsafe Code`
  Adds `DLIB_DONT_USE_UNSAFE_CODE` to `ScriptCompilationDefines` in `Build Settings`. With this setting, Dlib FaceLandmark Detector's internal code will not use unsafe code.
  > Enabling unsafe code often results in faster code, so we recommend keeping it enabled.

## WebGL Settings

This section is for settings to improve performance during **WebGL** builds.

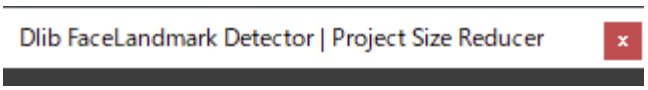
- `Enable SIMD Support`
  Adds settings to enable **SIMD** (Single Instruction Multiple Data) instruction set to `PlayerSettings.WebGL.emscriptenArgs` to improve performance.
  > Please enable when aiming for significant speedup in WebGL builds.

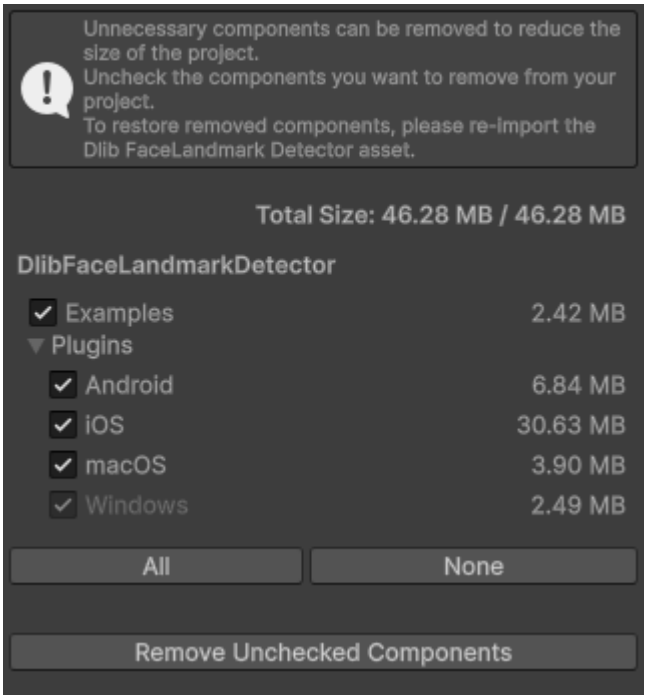
- `Enable MultiThreads Support`
  Enables `PlayerSettings.WebGL.threadsSupport` to enable multi-threading in WebGL.
  > Please enable when aiming for speedup in WebGL builds.

- `Enable wasm2023 Support`
  Enables `PlayerSettings.WebGL.wasm2023` to enable the latest **WebAssembly 2023** support.
  > This setting automatically enables SIMD, WebAssembly native exceptions, BigInt, WebAssembly.Table, optimized data operations, and faster data type conversions.
  > This setting is supported in Unity 2023.3 and later.
  > For details, please see Unity Official Documentation.

## `Project Size Reducer` Window Description

Select the menu item `Tools > Dlib FaceLandmark Detector > Open Project Size Reducer` to open the `Project Size Reducer` window. It can also be opened from the `Setup Tools` window.






In this window, you can remove unnecessary components to reduce project size. Uncheck the components you want to remove from the project and click the `Remove Unchecked Components` button. To restore deleted components, reimport the **Dlib FaceLandmark Detector** asset.

---

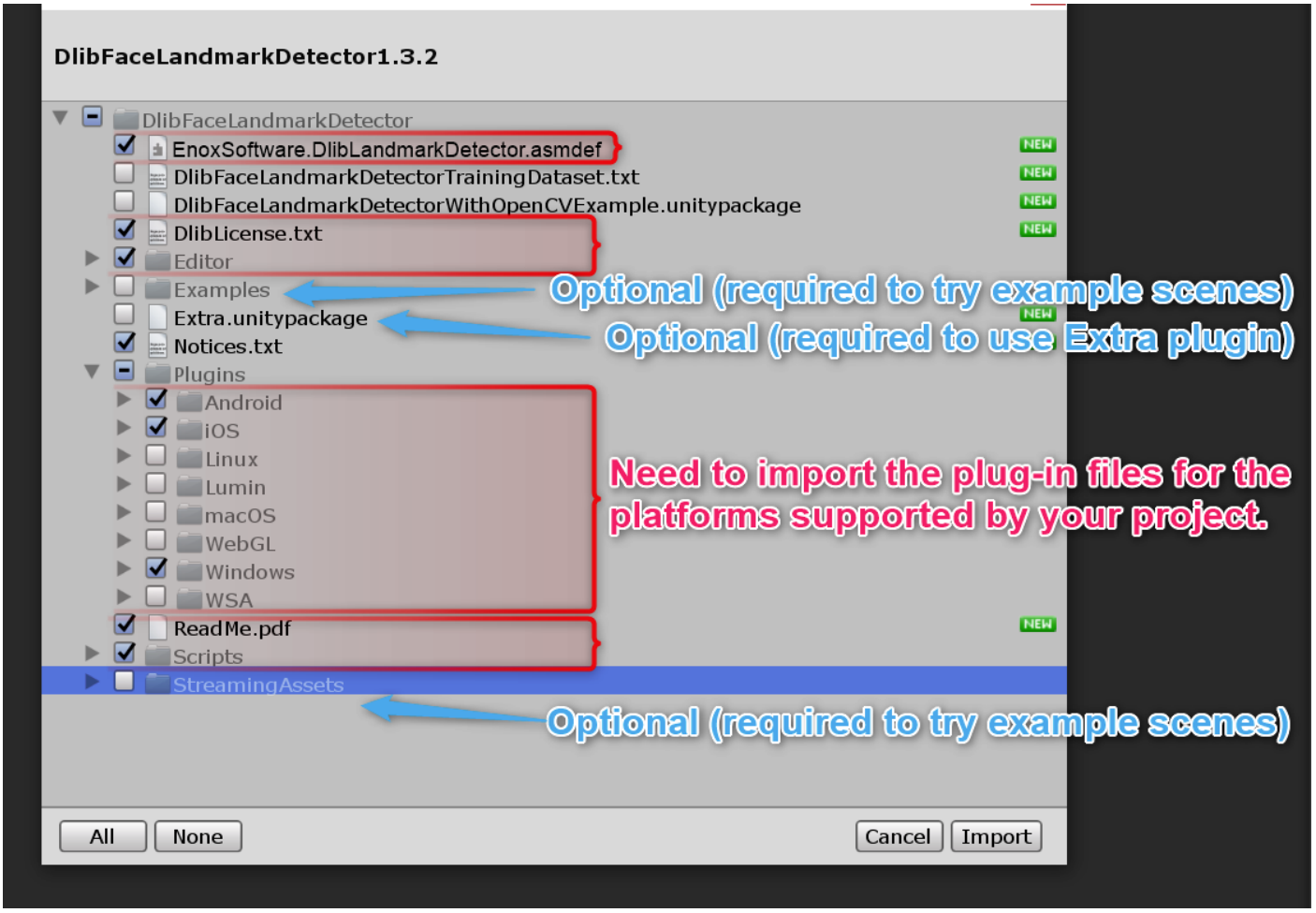
# ❓ Q & A

## Q1. The asset package size is large. Is there a way to reduce it?

**A1.** Using the `Project Size Reducer`, you can easily reduce unnecessary files in the `Assets/DlibFaceLandmarkDetector` folder. Particularly, by removing plugin files for platforms not planned to be supported in your development project, you can significantly reduce the project size. For details, please see Project Size Reducer Window Description.

## Q2. What is the minimum file configuration required for the asset to work?

**A2.** You don't need to import all files for the asset to work. If you don't need to try Example Scenes, the minimum file configuration required is as follows.




## Q3. When running Example Scenes, `DllNotFoundException: dlibfacelandmarkdetector` appears in the console.

**A3.** Plugin does not seem to be loaded correctly. Please check the setup procedure.

## Q4. When running Example Scenes, `Level 'Texture2DExample' (-1) could not be loaded because it h` appears in the console.

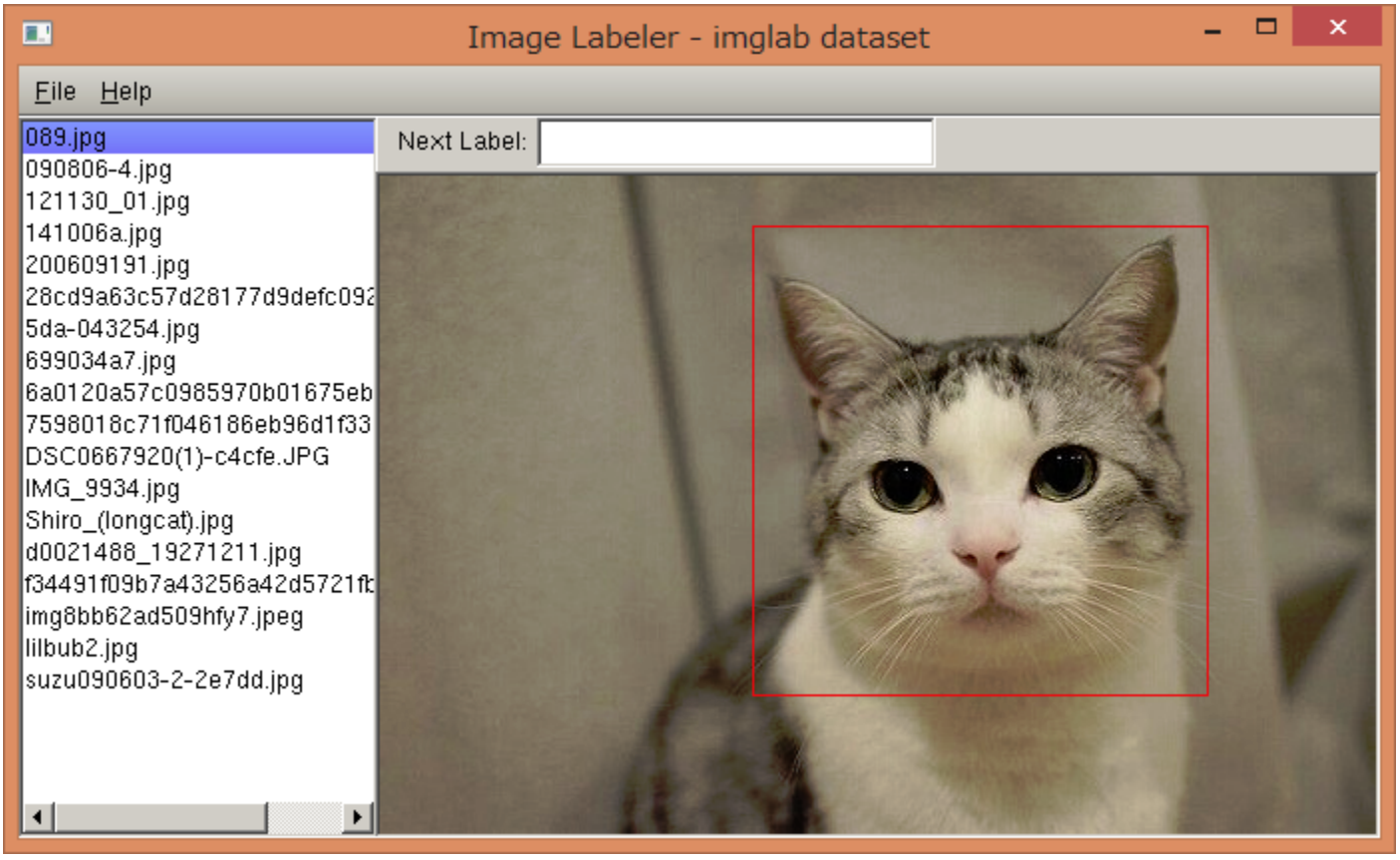
**A4.** Please Add all of the `***.unity` in the "DlibFaceLandmarkDetector" folder to `Build Settings` → `Scenes In Build`.

## Q5. In `Texture2DExample`, red rectangle is not displayed around a face.

**A5.** Please move the `Assets/DlibFaceLandmarkDetector/StreamingAssets/` folder to the `Assets/` folder.

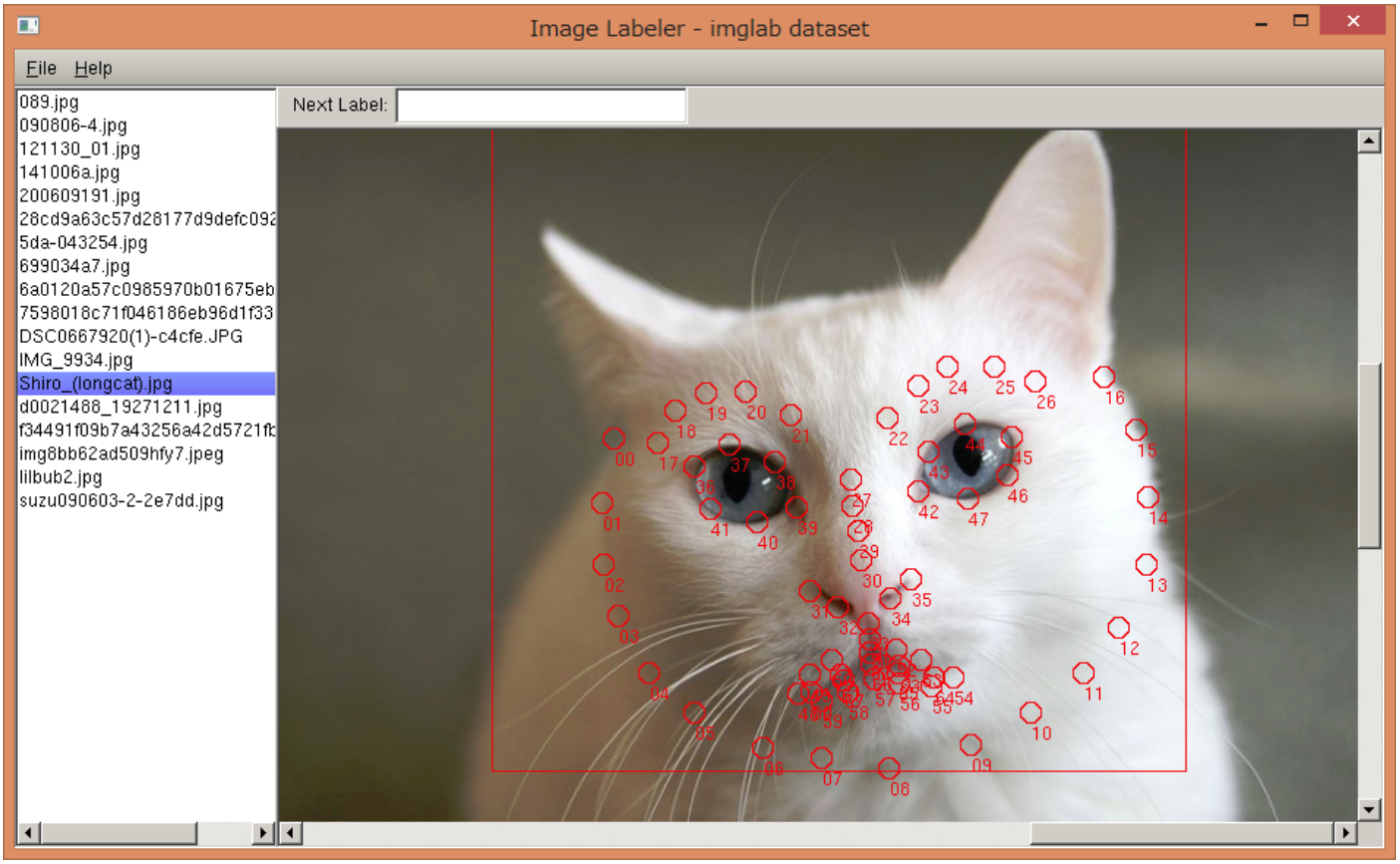
## Q6. How can I train object detector?

**A6.** Please refer to this page.




# Q7. How can I train shape predictor?

**A7.** Please refer to this page.

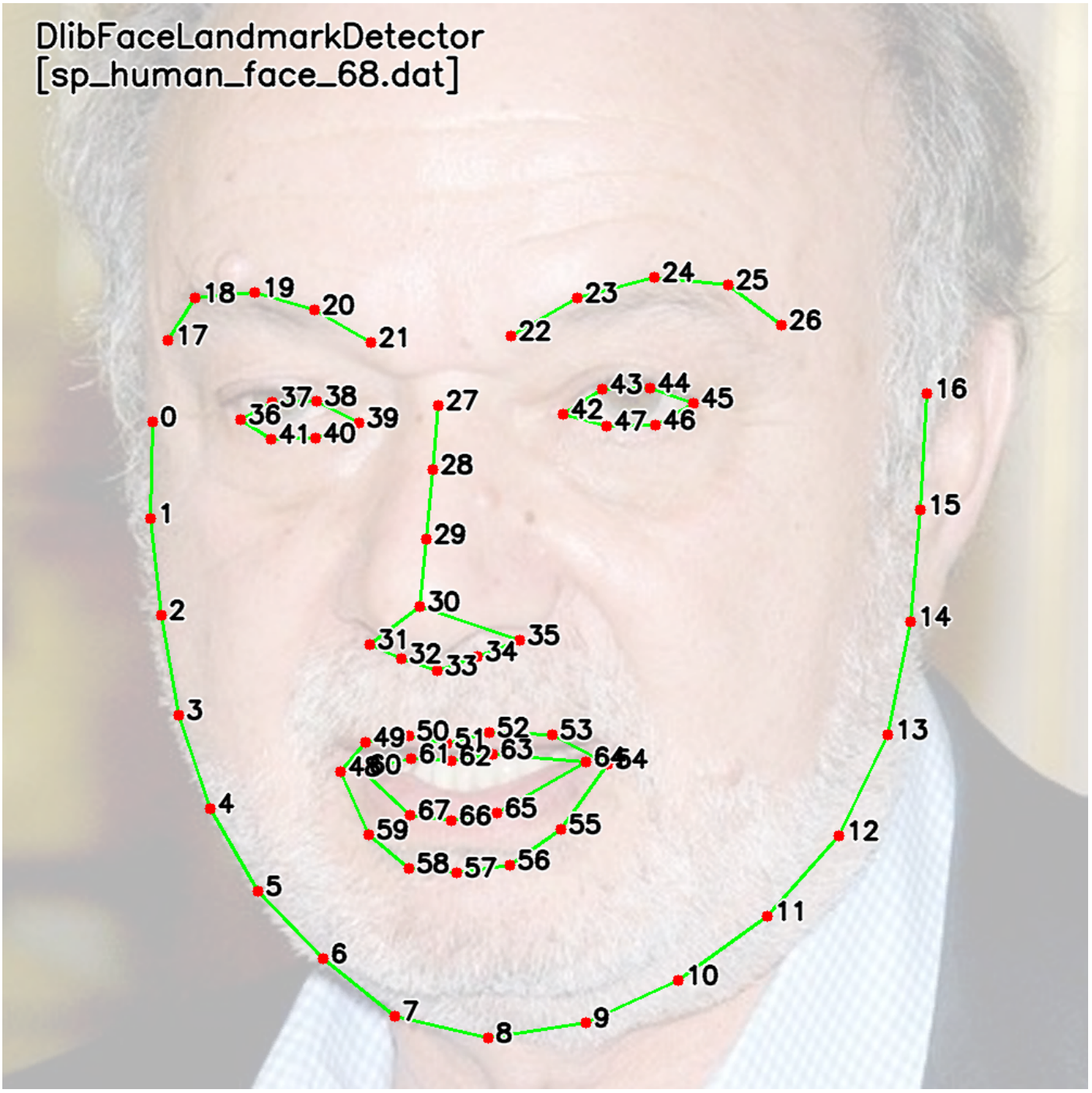


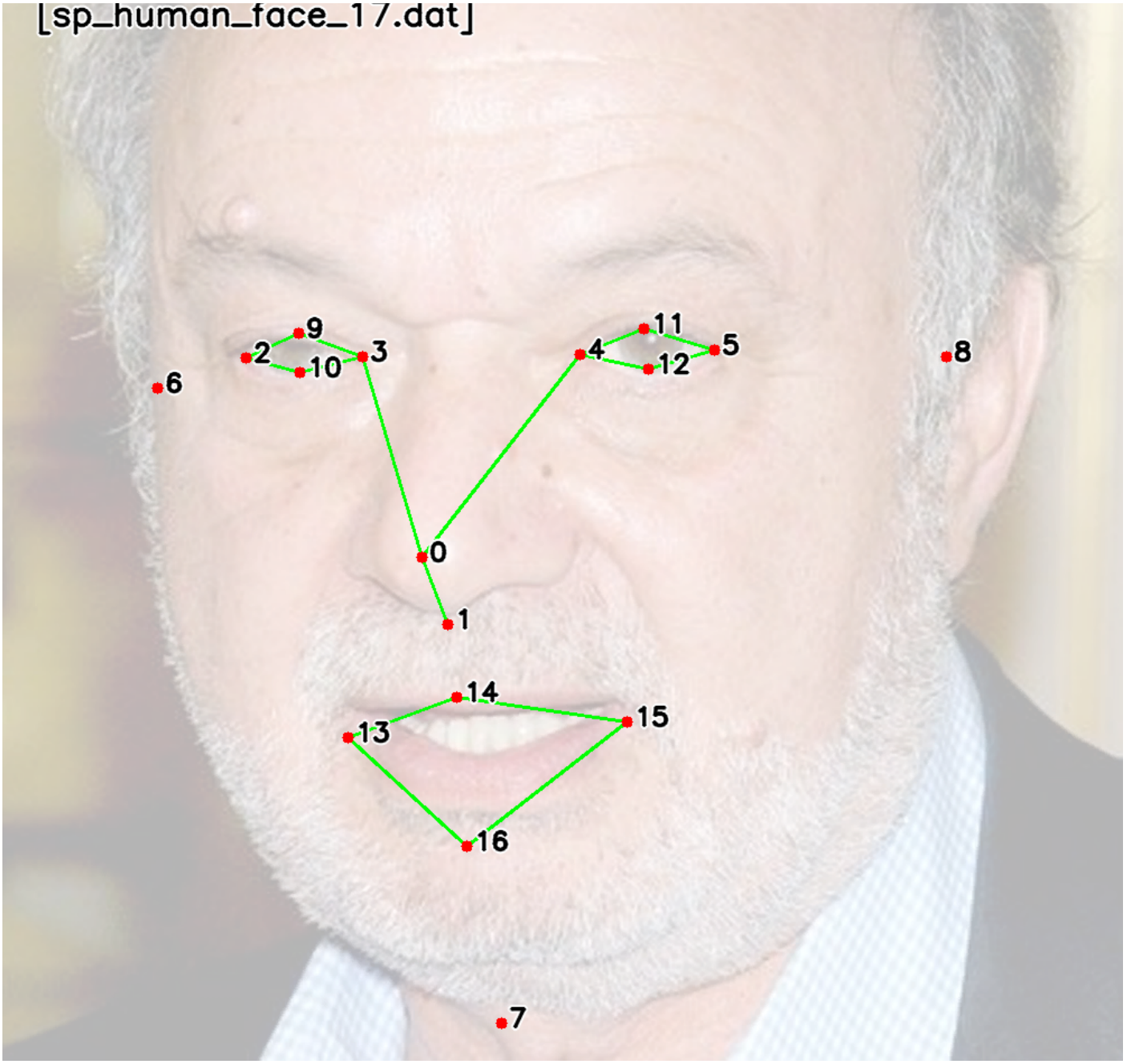
## Q8. The size of the `human_face_68_sp.dat` is too large.

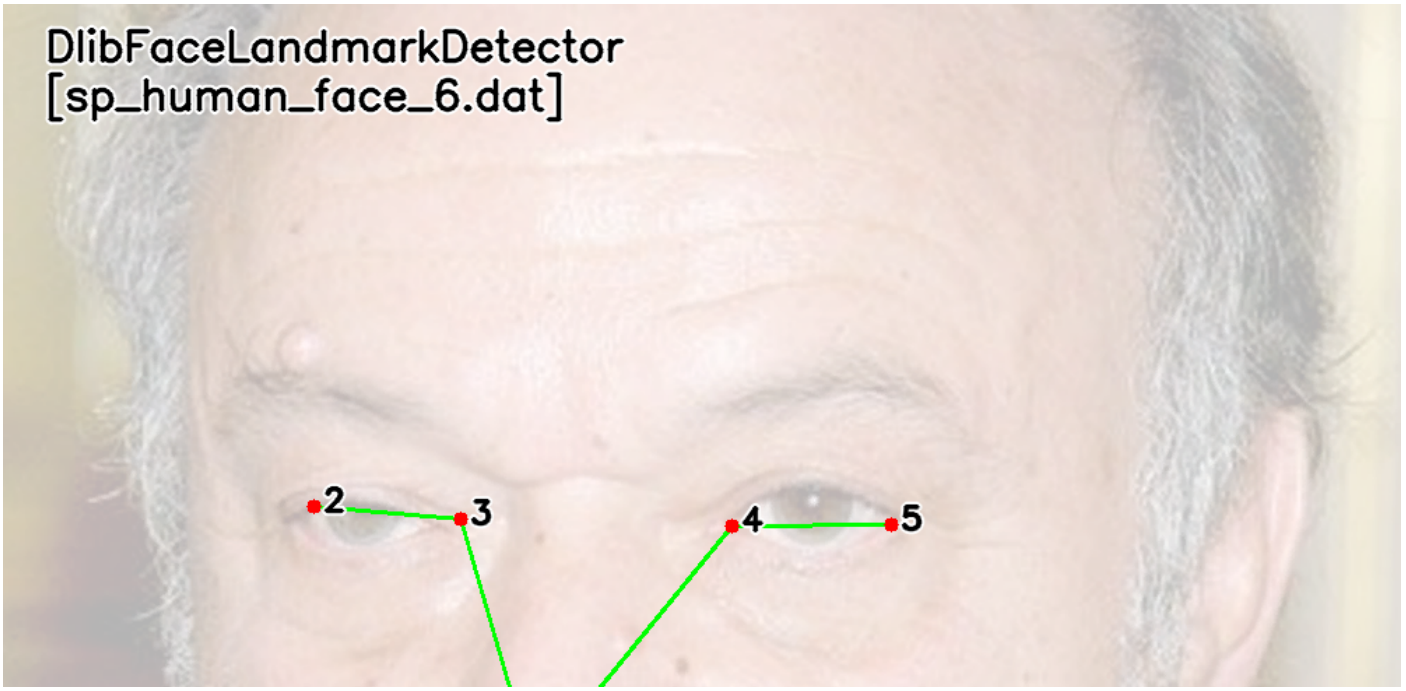
**A8.** Please use the `human_face_68_sp_for_mobile.dat`. The `human_face_68_sp_for_mobile.dat` is less accurate than the `human_face_68_sp.dat`, but it is smaller size.

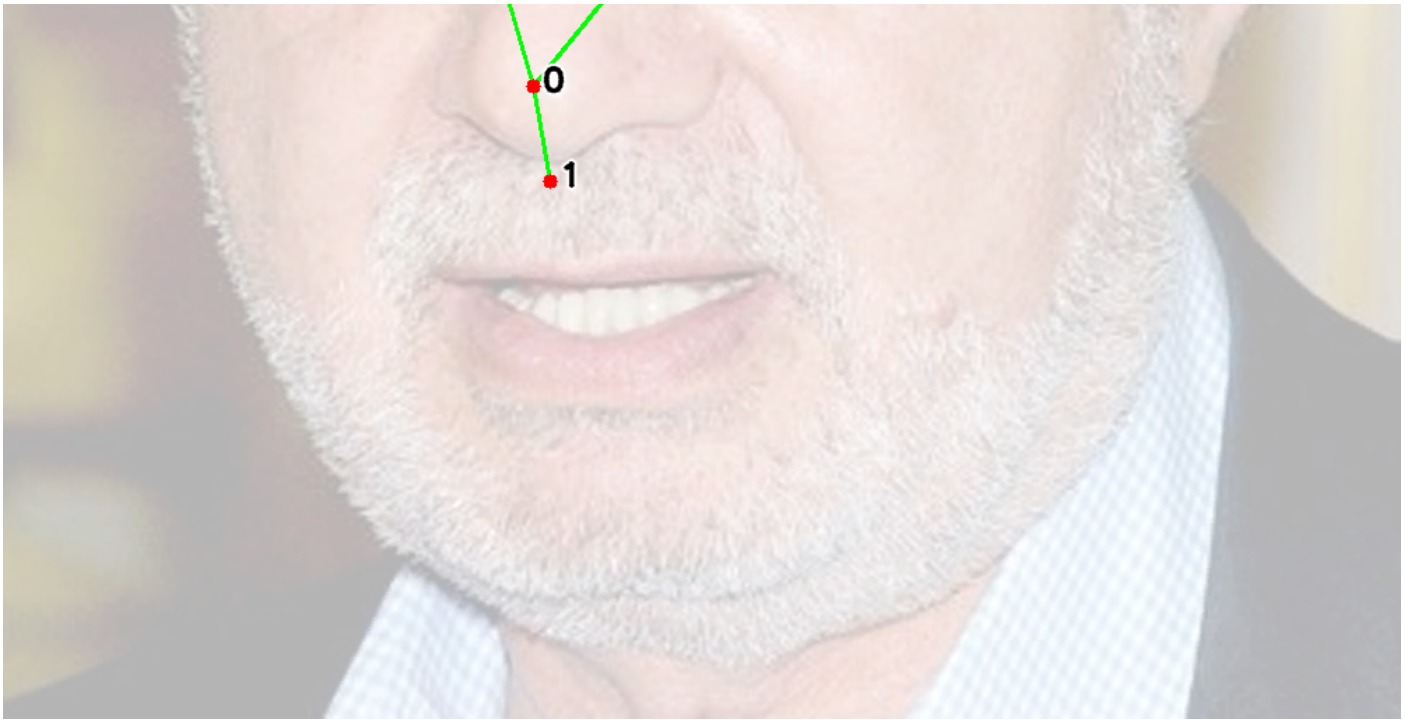
## Q9. What is the index of face landmark points that can be obtained using the `human_face_68_sp.dat`?

**A9.** The index of face landmark points that can be obtained using the `human_face_68_sp.dat` is as follows:





[sp_human_face_17.dat]
9
2
3
10
6
11
4
5
12
8
0
1
14
15
13
16
7


DlibFaceLandmarkDetector
[sp_human_face_6.dat]
2
3
4
5




## Q10. Can Shape Predictor model files be used commercially?

**A10.** Since the training dataset consists of flickr CC0 licensed images, the Shape Predictor model files are available for commercial use. For details of the training data set, please refer to `Assets/DlibFaceLandmarkDetector` folder.

## Q12. How to catch native Dlib's errors code (CVException handling)

**A12.** In order to display the native Dlib's error code, please enclose the code in `Utils.setDebugMode(true)` and `Utils.setDebugMode(false)`.

> ***Utils.setDebugMode(true);***
> // your Dlib calls here
> ***Utils.setDebugMode(false);***

For details, please see this page. (How to catch native Dlib's errors code (CVException handling))

---

# Version Changes

**2.0.0** [Common] Updated FpsMonitor to version 1.0.4. [visionOS] Updated beta support for the VisionOS platform on Unity 2022.3.18f1 and later. [Android] Updated native library libdlibfacelandmarkdetector.so for Android to be compatible with 16KB page size. [Common]

Removed CatDetectionExample. [Common] Changed namespace of the Utils class from UnityUtils to UnityIntegration, split into function-specific classes, and marked old classes as deprecated. [Common] Moved the OpenCVForUnityUtils class bundled with DlibFaceLandmarkDetectorWithOpenCVExample to the main Dlib module as DlibOpenCVUtils. [Common] Changed return types of DetectValueTuple and DetectLandmark to List. [Common] Removed the Extra folder. [Common] Increased the resolution of video files used for examples. [Common] Added overloads of GetDetectResult and GetDetectLandmarkResult methods that accept Span. [Common] Added overloads of the DrawFaceLandmark method that support ReadOnlySpan, ReadOnlySpan, and ReadOnlySpan. [Common] Added IsDebugMode and IsThrowException methods to DlibDebug. [Common] Added GetFilePathCoroutine, GetFilePathAsync, and GetFilePathTaskAsync methods to DlibEnv.

**1.4.2** [Common] Changed the FpsMonitor display system from IMGUI to uGUI. [Common] Added support for the new Input System (UnityEngine.InputSystem).

**1.4.1** [Common] Added method overloads in DlibFaceLandmarkDetector class that use ValueTuples. [Common] Added method overloads in OpenCVForUnityUtils class that use structs and ValueTuples corresponding to the data formats of OpenCV base classes. Accordingly, the minimum required version of OpenCVForUnity is now 2.6.4. [Common] Added Utils.getFilePathAsyncTask() and Utils.getMultipleFilePathsAsyncTask() methods. It is an asynchronous Task-return version of the existing Utils.getFilePathsAsync() method. It can be seamlessly integrated with other asynchronous code. [iOS] Fixed an issue where AddToEmbeddedBinaries was processed redundantly with each incremental build.

**1.4.0** [Common] Changed the minimum supported version to Unity2021.3.35f1. [Common] Separated the examples using the Built-in Render Pipeline and Scriptable Render Pipeline.

**1.3.9** [iOS] Added separate plugin files for iOS for devices and simulators. [WebGL] Added plugin files with only simd enabled.

**1.3.8** [Common] Changed to use unsafe code by default. [Common] Optimized the amount of memory allocation in the FaceLandmarkDetector class.

**1.3.7** [Common] Changed the minimum supported version to Unity2020.3.48f1. [WebGL] Added support for "WebAssembly 2023". [iOS] Changed "Target minimum iOS Version" to 11.0.

**1.3.6** [WebGL] Added a plugin file with threads and simd enabled for the WebGL platform. This update removes support for the WebGL platform in Unity 2021.1 and below. (Select MenuItem[Tools/Dlib FaceLandmark Detector/Open Setup Tools/WebGL Settings])

**1.3.5** [Windows] Added support for ARM64. [WebGL] Added Unity2023.2 or later support. [Lumin] Removed Lumin platform support (for MagicLeapOne). [Common] Added a button to SetupTools to automatically add scenes under the "Examples" folder to "Scenes In Build".

**1.3.4** [Common] Changed the setup procedure to use the SetupToolsWindow. [Common] Changed the namespace under "DlibFaceLandmarkDetector/Editor" folder from "DlibFaceLandmarkDetector" to "DlibFaceLandmarkDetector.Editor". [Common] Added "DlibFaceLandmarkDetector" folder under "StreamingAssets" folder. [Common] Added a function to automatically move the StreamingAssets folder.

**1.3.3** [Android] Added support for ChromeOS (x86 and x86_64 architectures).
**1.3.2** [Common] Added Assembly Definitions.
**1.3.1** [Common] Fixed a small issue.
**1.3.0** [UWP] Added ARM64 architecture.
**1.2.9** [Common] Added optimization code using NativeArray class (requires PlayerSettings.allowUnsafeCode flag, "DLIB_USE_UNSAFE_CODE" ScriptingDefineSymbol, and Unity2018.2 or later). [Common] Added support for Unicode file path (objectDetectorFilePath and shapePredictorFilePath). [Common] Added ImageOptimizationHelper to ARHeadWebCamTextureExample. [Common] Added some converter methods to OpenCVForUnityUtils.cs.
**1.2.8** [Lumin] Added the code for MagicLeap.
**1.2.7** [WebGL] Added Unity2019.1 or later support.
**1.2.6** [Common] Added "sp_human_face_17.dat", "sp_human_face_17_mobile.dat" and "sp_human_face_6.dat". [Common] Changed the training dataset of Shape Predictor model (now CC0 licensed and available for commercial use). [Common] Added BenchmarkExample.
**1.2.5** [Common] Re-assigned namespace. [Common] Support for OpenCVforUnity 2.3.3 or later.
**1.2.4** [macOS] Removed 32bit architecture (i386) from dlibfacelandmarkdetector.bundle.
**1.2.3** [Android,UWP] Fixed Utils.setDebugMode() method on the IL2CPP backend.
**1.2.2** [iOS] Added a function to automatically remove the simulator architecture (i386, x86_64) at build time. [Common] Improved DlibFaceLandmarkDetectorMenuItem.setPluginImportSettings() method. [Common] Updated to WebCamTextureToMatHelper.cs v1.0.9. [Common] Added support for Utils.setDebugMode() method on all platforms.
**1.2.1** [Common] Updated to WebCamTextureToMatHelper.cs v1.0.8. [Common] Updated to LowPassPointsFilter v1.0.1. Updated to KFPointsFilter v1.0.2. Updated to OFPointsFilter v1.0.2. [Common] Added updateMipmaps and makeNoLongerReadable flag to DrawDetectResult() and DrawDetectLandmarkResult() methods. [Common] Fixed Utils.getFilePathAsync() method (Changed #if UNITY_2017 && UNITY_2017_1_OR_NEWER to #if UNITY_2017_1_OR_NEWER).
**1.2.0** [Common] Updated to WebCamTextureToMatHelper.cs v1.0.7. [Common] Fixed WebCamTextureExample and OpenCVForUnityUtils. [Common] Added NoiseFilterVideoCaptureExample and NoiseFilterWebCamTextureExample. [Common] Added useLowPassFilter option to ARHeadVideoCaptureExample and ARHeadWebCamTextureExample. [Common] Added throwException flag to Utils.setDebugMode() method. [Common] Added drawIndexNumbers flag to DrawFaceLandmark() method.
**1.1.9** [Android] Added arm64-v8a architecture.
**1.1.8** [Common] Updated WebCamTextureExample (support Portrait ScreenOrientation). [Common] Updated to WebCamTextureToMatHelper.cs v1.0.4.
**1.1.7** [Common] Updated "human_face_68_sp.dat" and "human_face_68_sp_for_mobile.dat".
**1.1.6** [Common] Updated to dlib19.7. [Common] Updated to WebCamTextureToMatHelper.cs v1.0.3. [Common] Updated "human_face_68_sp.dat" and "human_face_68_sp_for_mobile.dat".
**1.1.5** [Common] Switched to the shape predictor file trained using new datasets.

**1.1.4** [Common] Updated WebCamTextureToMatHelper.cs v1.0.2. [Common] Improved Utils.getFilePathAsync().
**1.1.3** [Common] Fixed to improve the pose estimation performance. [Common] Changed DetectLandmarkArray(int left, int top, int width, int height) to DetectLandmarkArray(double left, double top, double width, double height). [WebGL] Fixed Utils.getFilePathAsync() method.
**1.1.2** [Common] Updated WebCamTextureToMatHelper.cs and OptimizationWebCamTextureToMatHelper.cs (Changed several method names). [Common] Changed the Example name.
**1.1.1** [Common] Improved Utils.getFilePath() and Utils.getFilePathAsync().
**1.1.0** [Win][Mac][Linux][UWP] Added the native plugin file enabled SSE4 or AVX compiler option.
**1.0.9** [WebGL] Added WebGL Plugin for Unity5.6.
**1.0.8** [Common] Changed the name of asset project ("Sample" to "Example"). [Common] Fixed VideoCaptureARExample and WebCamTextureARExample.
**1.0.7** [Common] Fixed WebCamTextureToMatHelper.cs (flipVertical and flipHorizontal flag).
**1.0.6** [Common] Fixed OpenCVForUnityMenuItem.cs (No valid name for platform: 11 Error). [Common] Added OptimizationWebCamTextureToMatHelper.cs.
**1.0.5** [Common] Fixed WebCamTextureToMatHelper class. [Common] Added Utils.getVersion(). [Common] Fixed Utils.getFilePathAsync().
**1.0.4** [Common] Updated shape_predictor_68_face_landmarks_for_mobile.dat.
**1.0.3** [WebGL] Added WebGL (beta) support (Unity5.3 or later). [Common] Fixed missing script error (WebCamTextureToMatHelper.cs). [Common] Added shape_predictor_68_face_landmarks_for_mobile.dat.
**1.0.2** [Common] Improved WebCamTextureHelper class.
**1.0.1** [Common] Added OptimizationSample. [Common] Added DetectRectDetection() method.
**1.0.0** Initial version.