VisionLabs

MACHINES CAN SEE

VisionLabs LUNA SDK Licensing

written for LUNA SDK Mobile Android version 5.8.0

Contents

1 Introduction

This document includes information about LUNA SDK licensing. This information is required for any engineer or developer who wants to utilize SDK features.

Here you can find:

- General information about LUNA SDK licensing;
- The list of licensed features;
- Online and offline license activation description.

2 General information

FaceEngine supports per-features node-locked licensing for every supported platform. This means that a final customer version of FaceEngine might be customized by activating and deactivating the licensed features. For that reason, not all algorithms and modules described in this book might be available to you.

Each SDK instance should be activated on every device. License, which was activated for one device, could not be used on some other device.

Interface for License objects ILicense (see file ILicense.h) gives you possibility to:

- check, if license is already activated;
- save license for next usage to some file;
- load license from file;
- check if some feature of FaceEngine is available for this license.

Typical usage scenario:

- Create a IFaceEngine object (see file FaceEngine.h);
- Get license pointer through `fsdk::IFaceEngine::getLicense`;
- Make activation for that license object through `fsdk::activateLicense`. This method requires full or relative path to the "license.conf" file.

3 Licensed features

To work with license features in code the ***LicenseFeature*** enum should be used (see file ILicense.h). Some features are not available for some platforms.

> Descriptor license feature allows creating no more than 1000 descriptors on mobile platforms.

Full list of features and their associated estimators available on mobile platforms:

- **Detection** - enables you to to find person face in a frame Available instances:

    – IDetector

- **BestShot** - enables you to select bestshot from the provided images. Available instances:

    – IBestShotQualityEstimator
    – IAGSEstimator (deprecated, use IBestShotQualityEstimator instead)
    – IHeadPoseEstimator (deprecated, use IBestShotQualityEstimator instead)
    – IQualityEstimator

- **FaceFeatures** - enables you to estimate face features. Available instances:

    – IEyeEstimator
    – IGlassesEstimator

- **Descriptor** - enables you to extract face features of a concrete person and work with them. Available instances:

    – IDescriptor
    – IDescriptorBatch
    – IDescriptorExtractor
    – IDescriptorMatcher

- **MobileLiveness** - enables you to predict whether the person's face is real or fake (photo, printed image). Available instances:

    – ILivenessOneShotRGBEstimator

- **Liveness** - enables you to get a bestshot from video. Available instances:

    – IBestShotMobile

- **MedicalMaskDetection** - enables you to detect a medical face mask on the face in the source image.

    Available instances:

    – IMedicalMaskEstimator

- **TrackEngine** - enables you to track person on video. Available instances:

    – ITrackEngine

> Other features such as Attributes, Emotions, Liveness, DescriptorIndex, LivenessEngine, HumanDetection are not available for mobile platforms.

> See FeatureMapMobile.htm for additional information about these features.

If the license for the selected feature is invalid, the factory instantiation method will return result with error code and empty object. For example, method: ***IFaceEngine::createAGSEstimator*** will return ***ResultValue*** with ***FSDKError::LicenseError*** empty ***IAGSEstimatorPtr*** object in case if LicenseFeature::BestShot in not available.

4 License activation

This section describes license activation.

License activation should be performed after the distribution package is unarchived.

> The generated license can only be used on the device where it was activated as the fingerprint of the device is used for its creation.

NOTE! Always remember that incorrect config may huck the things up very badly. Pay attention to what you configure and how. Always double-check what you deploy.

4.1 Online activation

Online activation is performed when you have an Internet connection on the device where the LUNA SDK license is activated.

Licensing configuration options are specified via the "license.conf" file which is an XML document with special tag formatting.

> You can find the "license.conf" file in the "data" directory in the distribution package.

This file is mandatory for license activation. You must fill it with the correct values before launching LUNA SDK.

> Fill in this file carefully. Incorrectly entered data may cause problems with activation on the device.

4.1.1 Online activation process

The activation process is as follows:

- Request the **Server**, **EID**, and **ProductID** from VisionLabs
- Go to the "data" directory
- Open the "license.conf" file
- Enter the received parameters
- Save changes in the "license.conf" file

The license key will be generated and saved to the "data" directory.

Now you can use LUNA SDK.

4.2 License file parameters description

License activation and next processing requires parameters listed below.

Parameter	Description	Type	Default value
Server	Activation server URL	"Value::String"	(empty)
EID	Entitlement ID	"Value::String"	(empty)
ProductID	Product ID	"Value::String"	(empty)
Filename	Default license filename	"Value::String"	license.dat
ContainerMode	If run in container	"Value::Int1"	0
ConnectionTimeout	Request timeout (in seconds)	"Value::Int1"	15

Server, **EID**, and **ProductID** - this information must be requested from VisionLabs and written to the file. It is mandatory for activation procedure.

EID field should be filled in for the online and offline activation.

Server and **ProductID** fields should be filled in for the online activation only.

Filename - the name of the file to save license after activation. The maximum length of the setting string is 64 symbols. Do not change this name!

ContainerMode - 0 - ignore any virtual network interfaces in the formation of the fingerprint.
1 - take into account all virtual interfaces in the formation of a fingerprint. The last option used while running in docker container due to physical network interface is not available by default. Although, you can make physical interface available with –net=host and set ContainerMode to 0 In this case you can be sure fingerprint will not be changed among containers. More reading in "Licensing in docker container"

ConnectionTimeout - set the maximum time in seconds that you allow to transfer operation to take. Normally, name lookups can take a considerable time and limiting operations risk aborting perfectly normal operations. Timeout 0 (zero) means it never times out during transfer. The ConnectionTimeout can't be set to a negative value, or a value that is large than maximal, which is 300 seconds.

By default, the file is created in the "data" directory. The file has a binary format. At the next launch of the product on the same device, a license will be read from this file.

An example of the "license.conf" file:

```
<section name="Licensing::Settings">
    <param name="Server" type="Value::String" text=""/>
    <param name="EID" type="Value::String" text=""/>
    <param name="ProductID" type="Value::String" text=""/>
    <param name="Filename" type="Value::String" text="license.dat"/>
    <param name="ContainerMode" type="Value::Int1" x="0"/>
    <param name="ConnectionTimeout" type="Value::Int1" x="15"/>
```

```
    </section>
```