TLL 2 Quick integration guide

Introduction

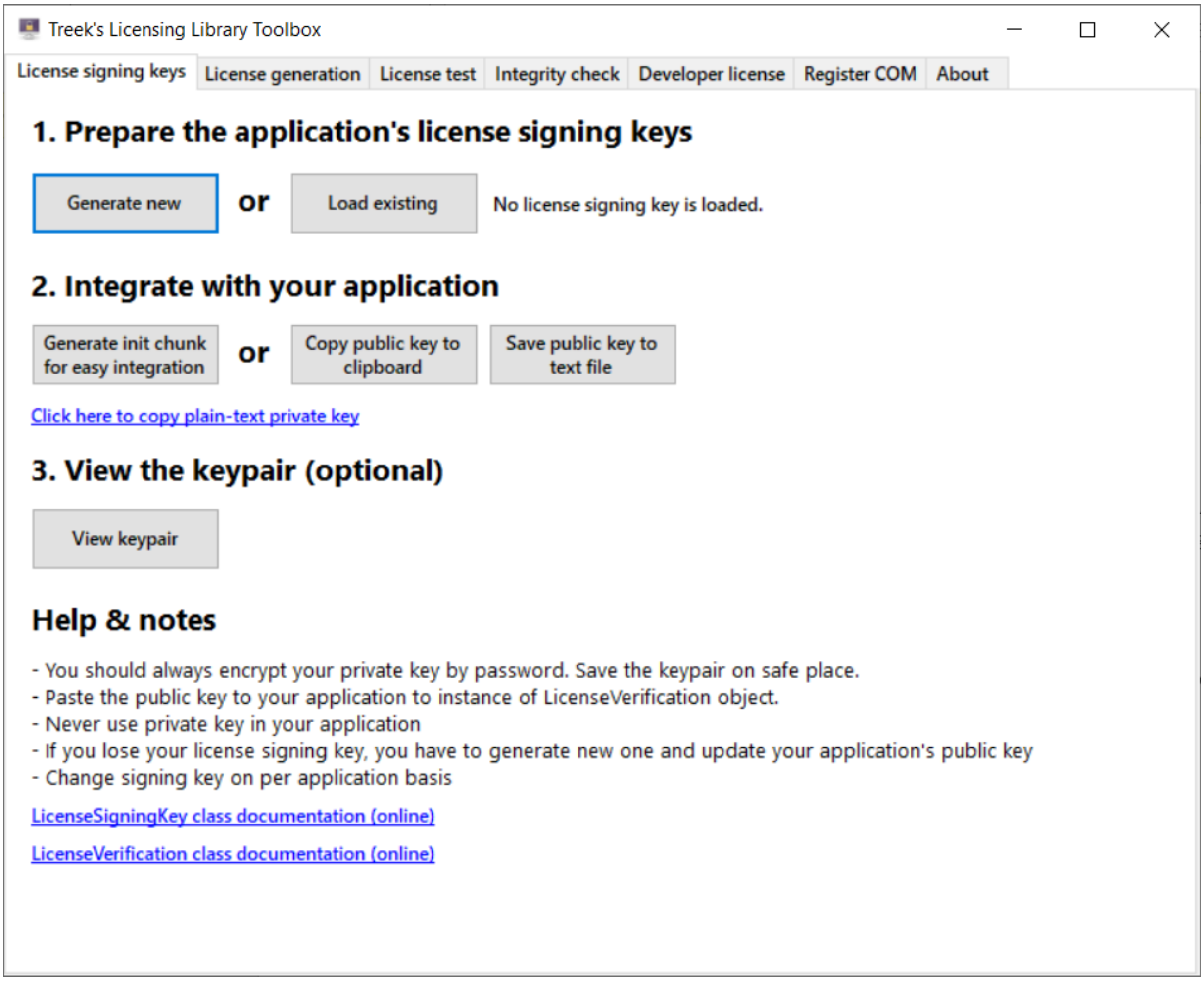
This guide will lead you through Treek's Licensing Library integration process into your app. The guide is mainly written for C#, but almost everything applies also for VB.NET and other languages.

Step 1 – Preparing keys

To secure the application licenses, you need to generate its keys. Keys will be used to sign license data, thus you must keep private part of keys (lkey file) safe. Never ever distribute the lkey file or the plain-text private key content. Also, you should create new keys for each project.

To prepare keys:

1. Start TLL Toolbox
2. Click „Generate new button“ on „License signing keys“ tab (Fig. 1)
3. In opened dialog, select size, save path and password to encrypt keys (Fig. 2)
    a. If you choose the password, you will need to enter this password before generating licenses using Toolbox with this key.
4. Click OK and wait for the operation to complete. Backup your keys to make sure you won't loose them.



Treek's Licensing Library Toolbox
License signing keys
License generation
License test
Integrity check
Developer license
Register COM
About

1. Prepare the application's license signing keys

Generate new
or
Load existing

No license signing key is loaded.

2. Integrate with your application

Generate init chunk for easy integration
or
Copy public key to clipboard
Save public key to text file
Click here to copy plain-text private key

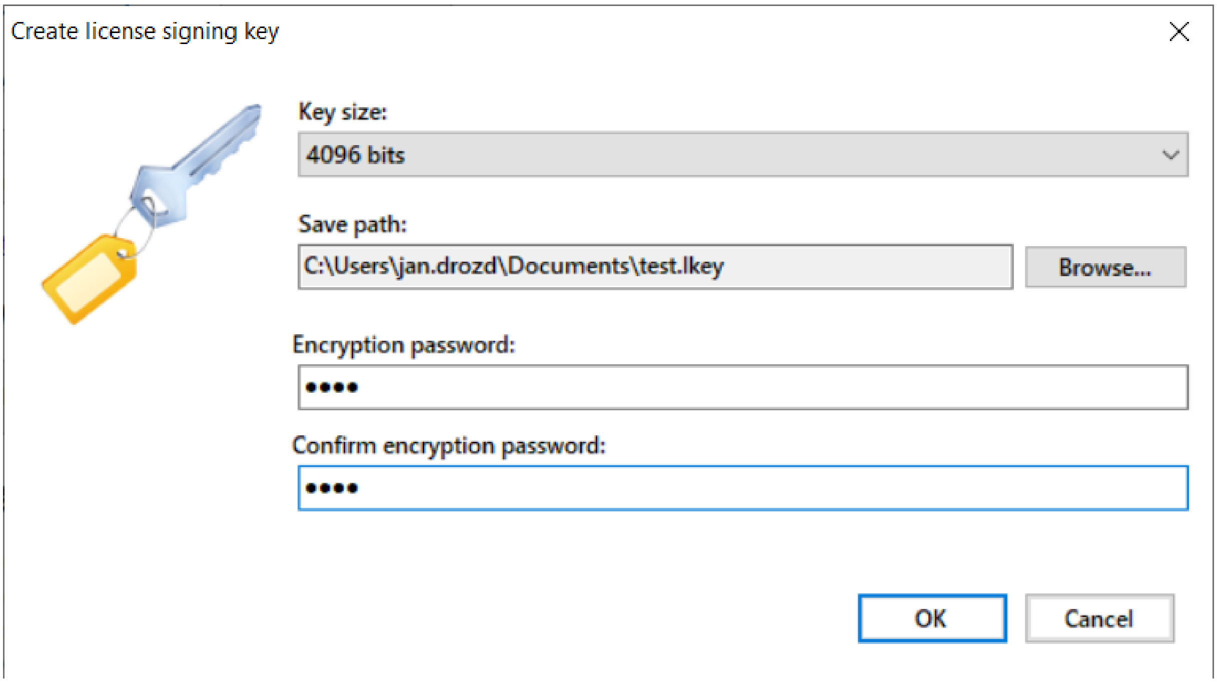
3. View the keypair (optional)

Help & notes

- You should always encrypt your private key by password. Save the keypair on safe place.
- Paste the public key to your application to instance of LicenseVerification object.
- Never use private key in your application
- If you lose your license signing key, you have to generate new one and update your application's public key
- Change signing key on per application basis

LicenseSigningKey class documentation (online)
LicenseVerification class documentation (online)

Fig. 1



Key size: 4096 bits

Save path: C:\Users\jan.drozd\Documents\test.lkey

Encryption password: ••••

Confirm encryption password: ••••

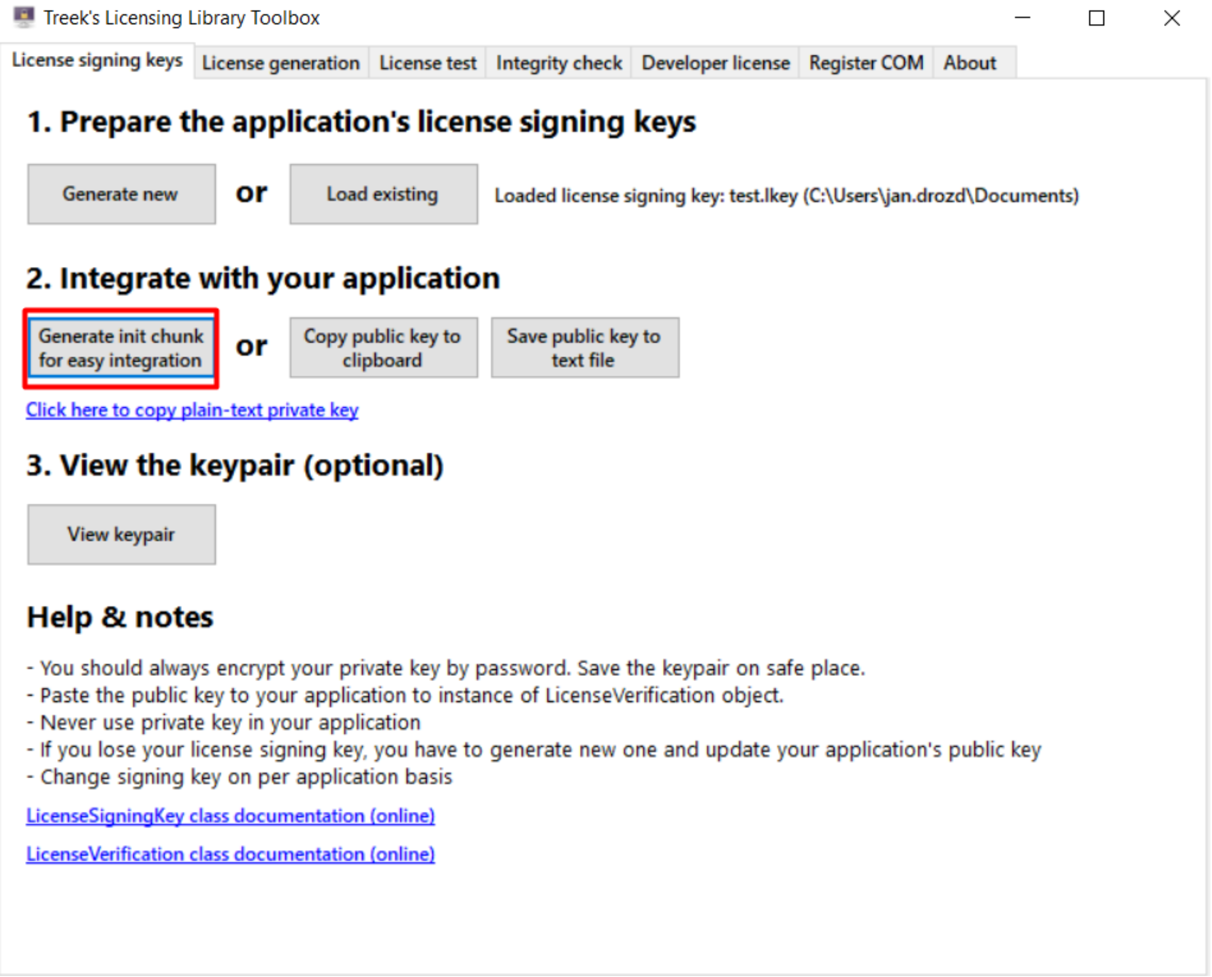
Fig. 2

Step 2 – Configuring init chunk

Init chunk is piece of encrypted data „telling“ to TLL how to work – where to save active license data, what public key should be used to verify licenses, etc. You need the init chunk to use EasyIntegration classes.

To create init chunk:

1) Open or create license signing keys according to step 1
2) Click „Generate init chunk for easy integration“ button „License signing keys“ (Fig. 3)
3) Set the configuration of init chunk (according to help below)
4) Click „Generate C# or VB snippet“
5) Copy the snippet code to your application


Treek's Licensing Library Toolbox
License signing keys
License generation
License test
Integrity check
Developer license
Register COM
About
Generate new
or
Load existing
Generate init chunk for easy integration
or
Copy public key to clipboard
Save public key to text file
Click here to copy plain-text private key
View keypair
LicenseSigningKey class documentation (online)
LicenseVerification class documentation (online)

1. Prepare the application's license signing keys

Loaded license signing key: test.lkey (C:\Users\jan.drozd\Documents)

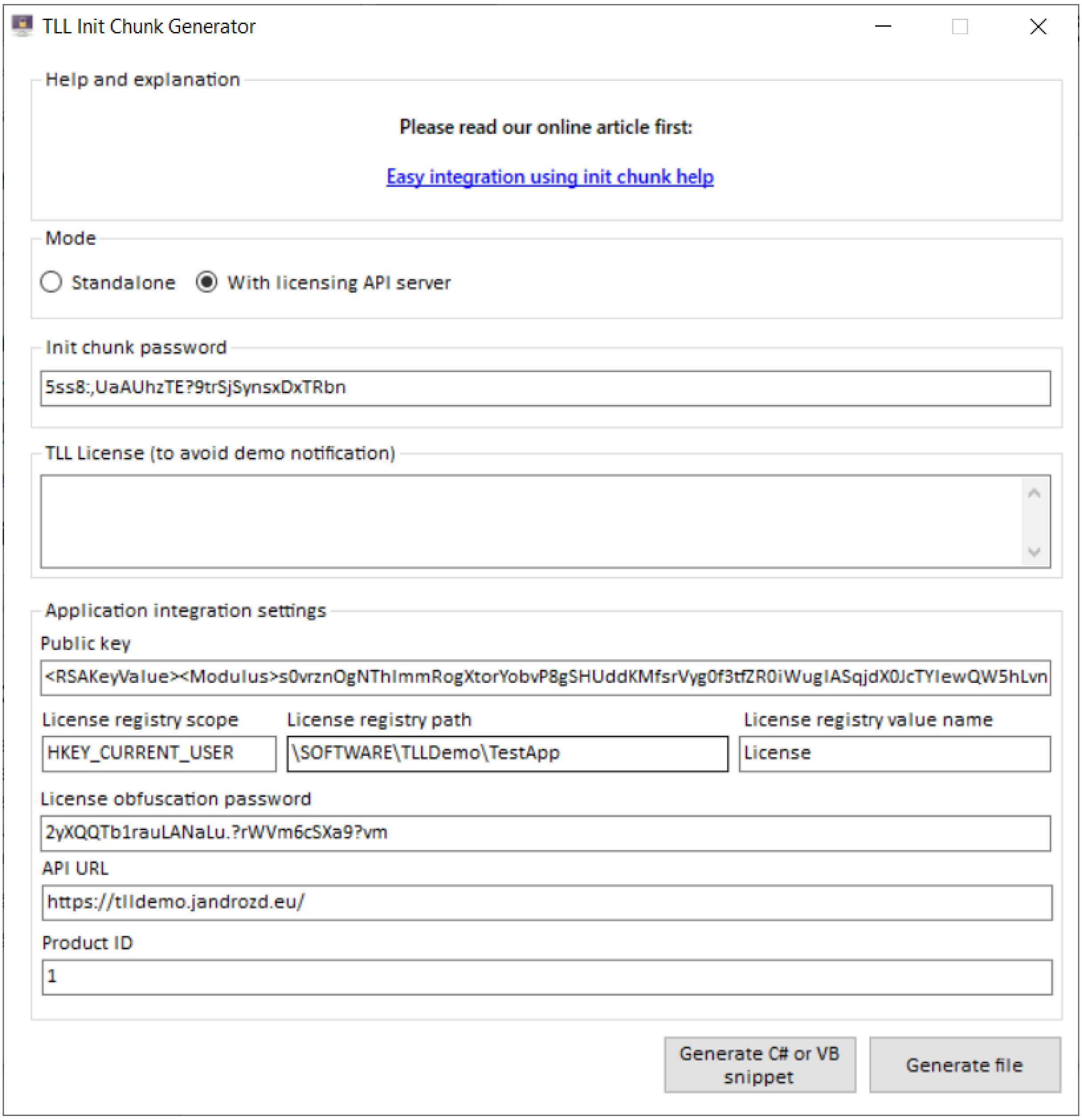
2. Integrate with your application

3. View the keypair (optional)

Help & notes

- You should always encrypt your private key by password. Save the keypair on safe place.
- Paste the public key to your application to instance of LicenseVerification object.
- Never use private key in your application
- If you lose your license signing key, you have to generate new one and update your application's public key
- Change signing key on per application basis

Fig. 3



TLL Init Chunk Generator
Help and explanation
Please read our online article first:
Easy integration using init chunk help
Mode
Generate C# or VB snippet
Generate file

○ Standalone ● With licensing API server

Init chunk password: 5ss8:,UaAUhzTE?9trSjSynsxDxTRbn

TLL License (to avoid demo notification):

Application integration settings

Public key: <RSAKeyValue><Modulus>s0vrznOgNThImmRogXtorYobvP8gSHUddKMfsrVyg0f3tfZR0iWugIASqjdX0JcTYIewQW5hLvn

License registry scope: HKEY_CURRENT_USER

License registry path: \SOFTWARE\TLLDemo\TestApp

License registry value name: License

License obfuscation password: 2yXQQTb1rauLANaLu.?rWVm6cSXa9?vm

API URL: https://tlldemo.jandrozd.eu/

Product ID: 1

Fig. 4

Configuration explanation

- Mode
  - Standalone: TLL Core WITHOUT activation server
  - With licensing API server: For those who purchased TLL activation server
- Init chunk password
  - This is password used to encrypt init chunk file content. You'll have to pass it to TLLInterface object.
- Your public key
  - Automatically pre-filled, public key used for license information verification
- TLL License
  - Put here your developer license to avoid TLL demo notification.
  - If you set developer license in the "Developer license" tab of TLL Toolbox, this information will be filled automatically
  - You can keep this empty for testing purposes
- License registry scope

  - Scope of registry used to save end user’s license
  - Can be HKEY_CURRENT_USER or HKEY_LOCAL_MACHINE
  - Remember, you need administrator permission to store license in HKEY_LOCAL_MACHINE
- License registry key
  - The path of registry key to store license
  - Always should start with backslash “\”
- License registry value name
  - Registry value name to store end user’s license
- License obfuscation password
  - Password used to encrypt end user’s license in registry
- API URL
  - URL of activation server
- Product ID
  - This value is generated by activation server per product

Step 3 – add configured TLL to your project

Integration into your C# or VB project can be summarized into few steps:

1) Add reference to TLL (TreeksLicensingLibrary2.dll)
   a. Either manually or via nuget package manager
2) Create a new class which will hold licensing code (i.e. TLLIntegration.cs)
3) Add to new class snippet from step 2 and create new instance of EasyIntegration.TLLInterface class using the snippet byte data.
4) Check license data and call functions to activate license during runtime at places you choose

Please view list of available properties and methods at http://msdn2.treekslicensinglibrary.com/html/d3950b8e-ea7d-6a9a-b1ac-7f22206445cb.htm

How to check license

If no license is installed TLLInterface property MyLicense will be null (nothing in VB). If trial version license is present, MyLicense will hold the trial license details and its IsDemo property will be true. MyLicense won’t be null in such occasion.

Displaying license details

TLLInterface MyLicense property is License type object. Check the list of available information at http://msdn2.treekslicensinglibrary.com/html/bb934f2d-6379-8165-5ab2-ab3bc5dfad72.htm

In case you are activation server user and your product has automatic silent reactivation enabled, the real license expiration date (if any) won’t be in MyLicense.ExpirationDate property, but in Info1 property. ExpirationDate property in such occasion holds the forced automatic online reactivation date untill which user must become online or the product license stops working.

Registering user’s license

There are two options:

1) You can create your own custom registration GUI
2) You can call ShowRegistrationForm method to display registration window designed by us

a. See http://msdn2.treekslicensinglibrary.com/html/7f0237e0-1f0e-b00e-1e1f-3110b305a4b5.htm for optional parameters

b. In case you are in WPF application, you can use null (nothing) as owner parameter.