Sandkey for FieldShield

Software Development Kit

Overview

The Sandkey for FieldShield software development kit (SDK) is a library for data masking and encryption. Data value protection can be added to any application, to achieve either dynamic data masking (DDM), or static data masking (SDM).

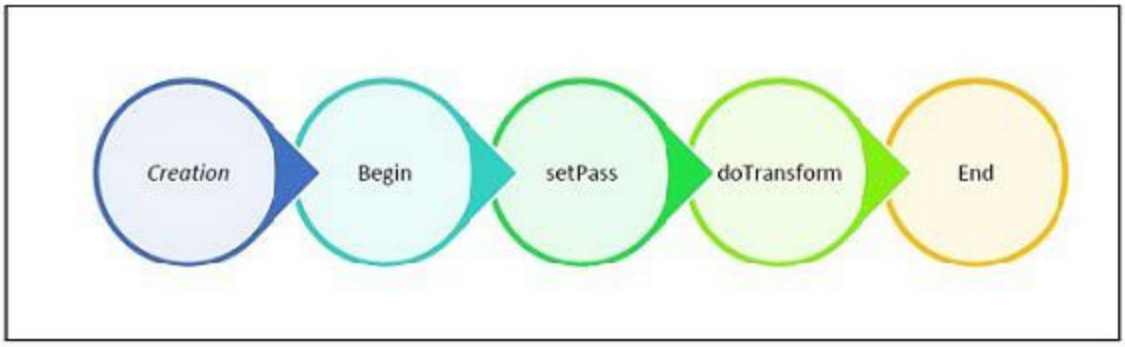
There is an application programming interface (API) for both Java and .NET. Sandkey is a group of classes that provides several ways of protecting or altering data, using encryption, decryption, hashing, redaction, and encoding.

The Sandkey encryption and decryption algorithms include:

- AES-256 (Advanced Encryption Standard); a state-of-the-art and widely accepted encryption algorithm.
- FP-ASCII (Format Preserved for American Standard Code for Information Interchange); the original data length is preserved but protected with AES-256 bit encryption.
- FP-ALPHANUM; another version of the Format Preserving algorithm above, in which only uppercase letters, lowercase letters, and numeric digits are encrypted.

Library Process

Each function in the Sandkey library is utilized in a consistent manner. The figure below shows the sequence of methods for each encryption and decryption usage. Once the password is set for a given object, it does not need to be set each time the encryption object is used.



The other classes follow a similar flow in usage, varying slightly.

The steps are defined as follows:

1. Creation
   Create an instance of the desired class. Most languages use the keyword new.

2. Begin
   After creating an instance from the class, use this instance to call the begin() method. This function will initiate the encryption or decryption process.

3. SetPass
   Next, the pass phase must be set by calling the setPass method. This method requires a String to be passed in as a parameter.

4. DoTransform
   Once the passphrase has been set, the core method, doTransform(Data), is called to encrypt or decrypt. A String is passed as a parameter, which is then encrypted or decrypted using the algorithm chosen. For the alphanumeric class, a numeric doTransform method exists to be used on numerals. The method will return the data that has been encrypted or decrypted with the appropriate type.

5. Destruct
   Finally, call the method destruct() to release the object from the occupied space in memory. Since it uses the DLL file, it is necessary to destruct the object.

Sandkey Primary Classes

The three types of available encryption/decryption algorithms include AES256, FPASCII, and FP-ALPHANUM. Each type has two classes, one for encryption and the other for decryption. The classes are:

AES-256
- enc_aes256: AES256 – Encryption
- dec_aes256: AES256 – Decryption

FP-ASCII
- enc_fp_ascii: FP- ASCII – Encryption
- dec_fp_ascii: FP- ASCII – Decryption

FP-ALPHANUM
- enc_fp_alphanum: FP- ALPHANUM – Encryption
- dec_fp_alphanum: FP- ALPHANUM – Decryption

The other classes available are for hashing, encoding in base64 or hex, and replacing characters in a String. The classes are:
- hex: Hex encoding
- hash_sha2: Hashing based on the SHA256
- base64: Base64 encoding
- eplace_char: Replacing characters (masking)

Java

In the following instructions, the Eclipse IDE was used for Java programming. Other IDEs will vary slightly.

Installing Sandkey

1. Download Sandkey. This includes two components:
    - sandkey.jar - the Java wrapper for the native shared libraries
    - sandkey-native.zip - contains platform specific native shared libraries (.dll, .so)

2. Set up the libraries.
   This library has the core functions coded in .NET. This file is native library which is tobe used in java. There are two ways to use this library:

Command line

To set the java library path through the command line, use the following command. This must be set so that the JNI can locate the .NET SDK to perform the operations. The path must be the location of the folder that contains the DLL file of .NET SDK of encryption project.

Java run configuration arguments: VM arguments:

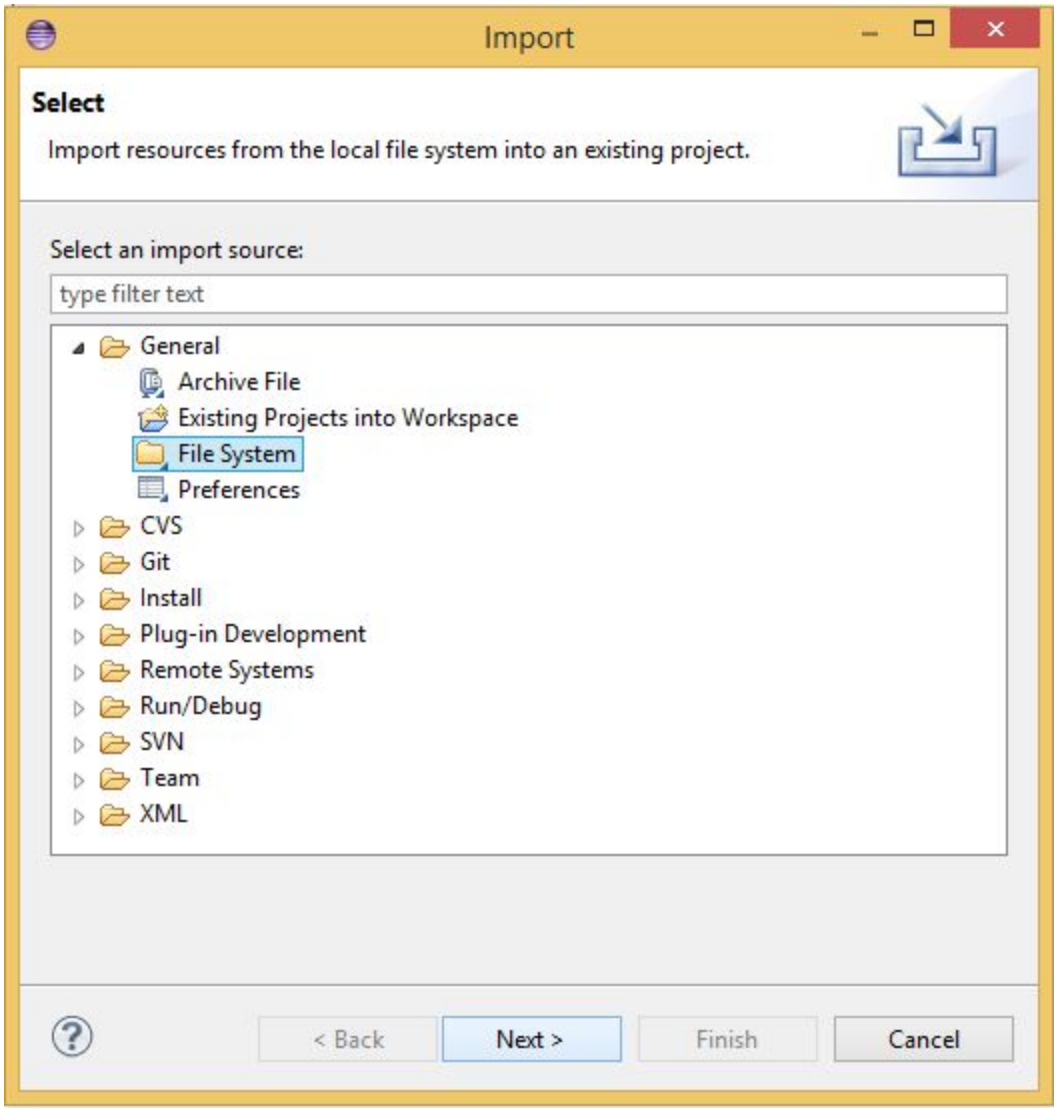
```
java -Djava.library.path=<directory_location_native_libs> TestApp
```

Eclipse

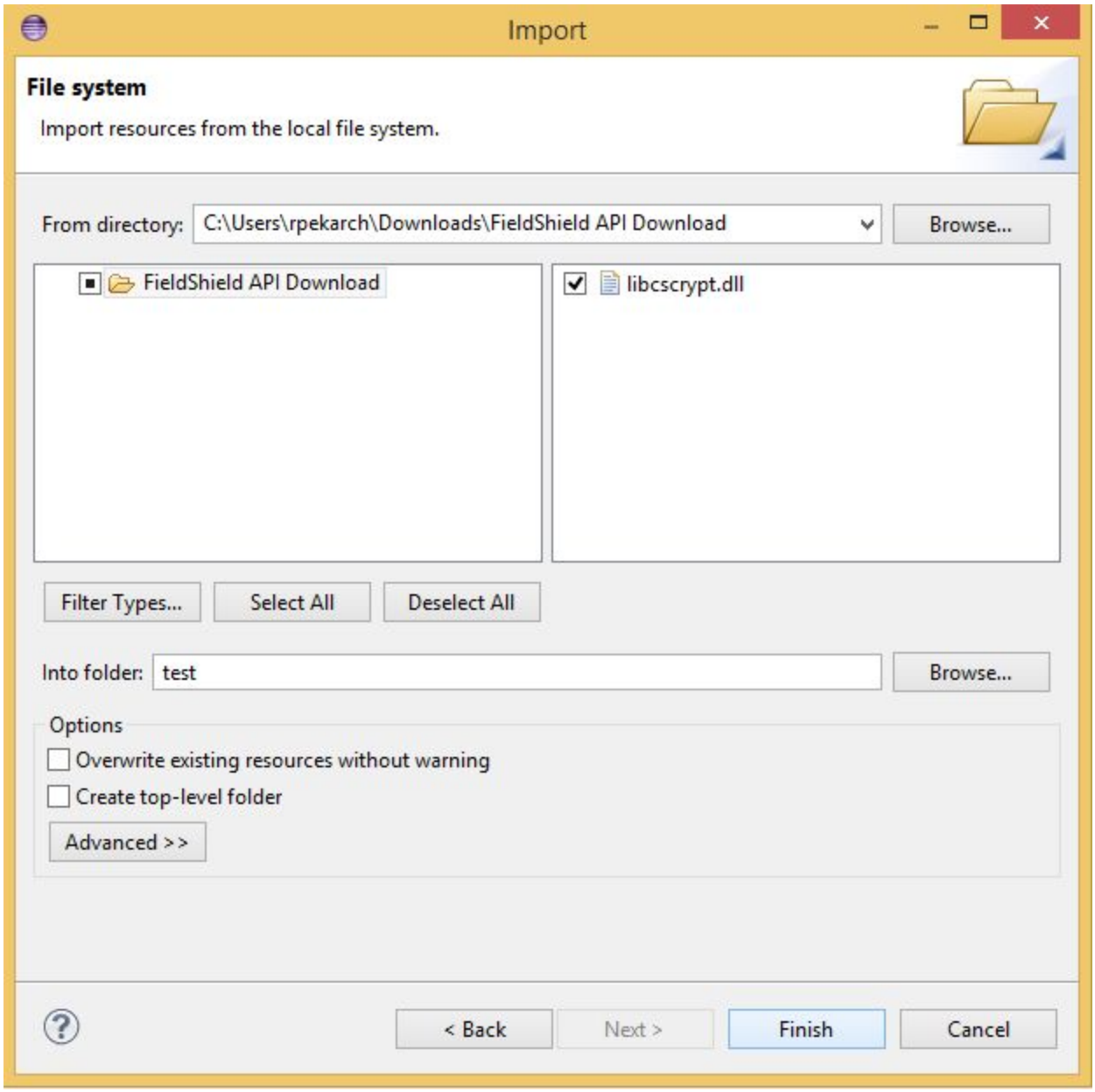
Import the files into a Project for the Eclipse IDE.

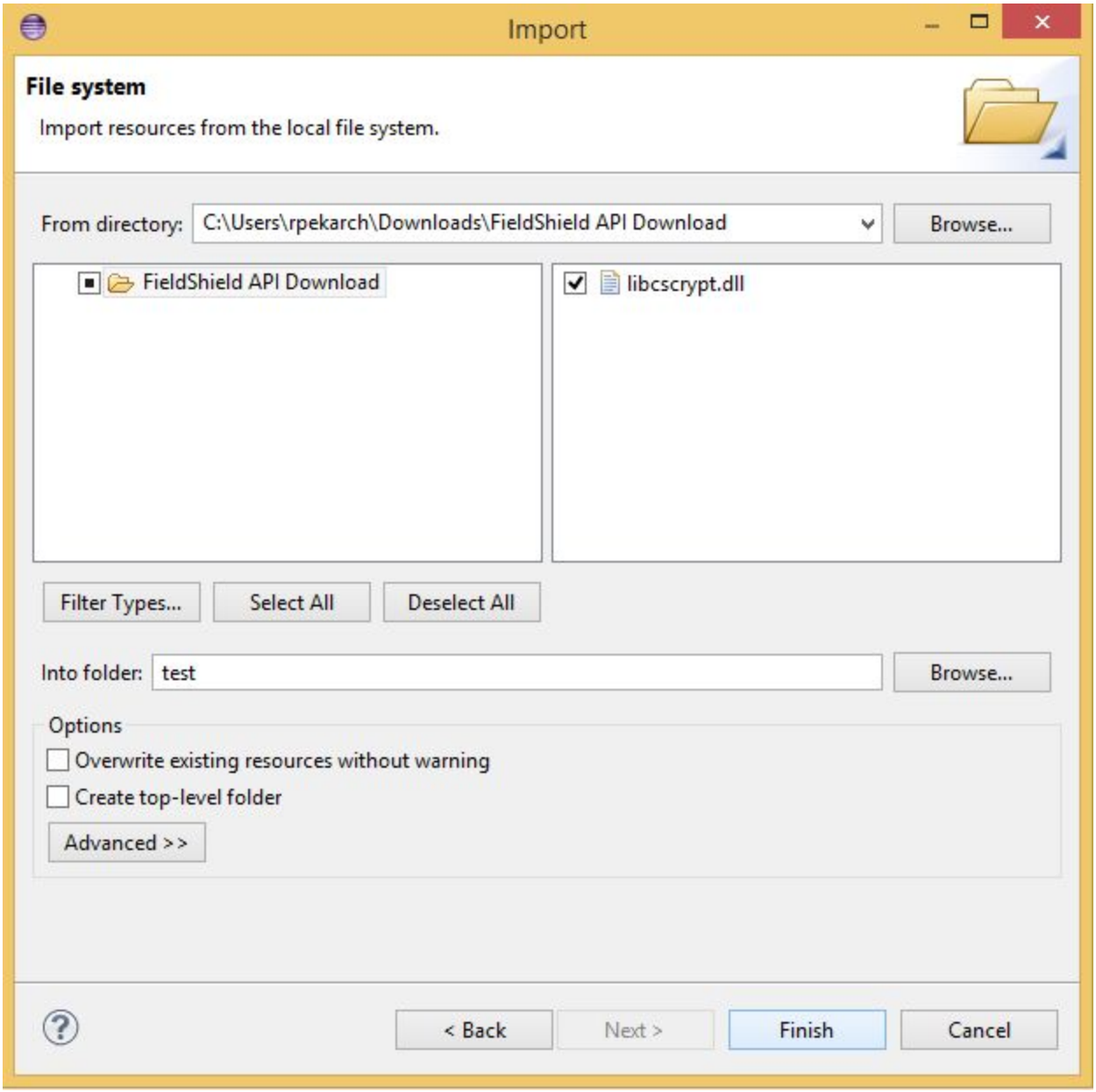
Both the jar file and DLL file must be imported into the project in which the classes are to be used.

1. Copy the DLL file into the Project folder of your eclipse java project, or right-click the project and select **Import**. On the Import page, select **General**, and the select **File System**. Click **Next**.



2. On the File System page, browse for the directory with the libsandkey.dll, then select that DLL. Click **Finish**. The libsandkey.dll is now included in the Project folder.


Import
File system
Import resources from the local file system.
From directory:
C:\Users\rpekarch\Downloads\FieldShield API Download
Browse...
FieldShield API Download
libcscrypt.dll
Filter Types...
Select All
Deselect All
Into folder:
test
Browse...
Options
Overwrite existing resources without warning
Create top-level folder
Advanced >>
< Back
Next >
Finish
Cancel

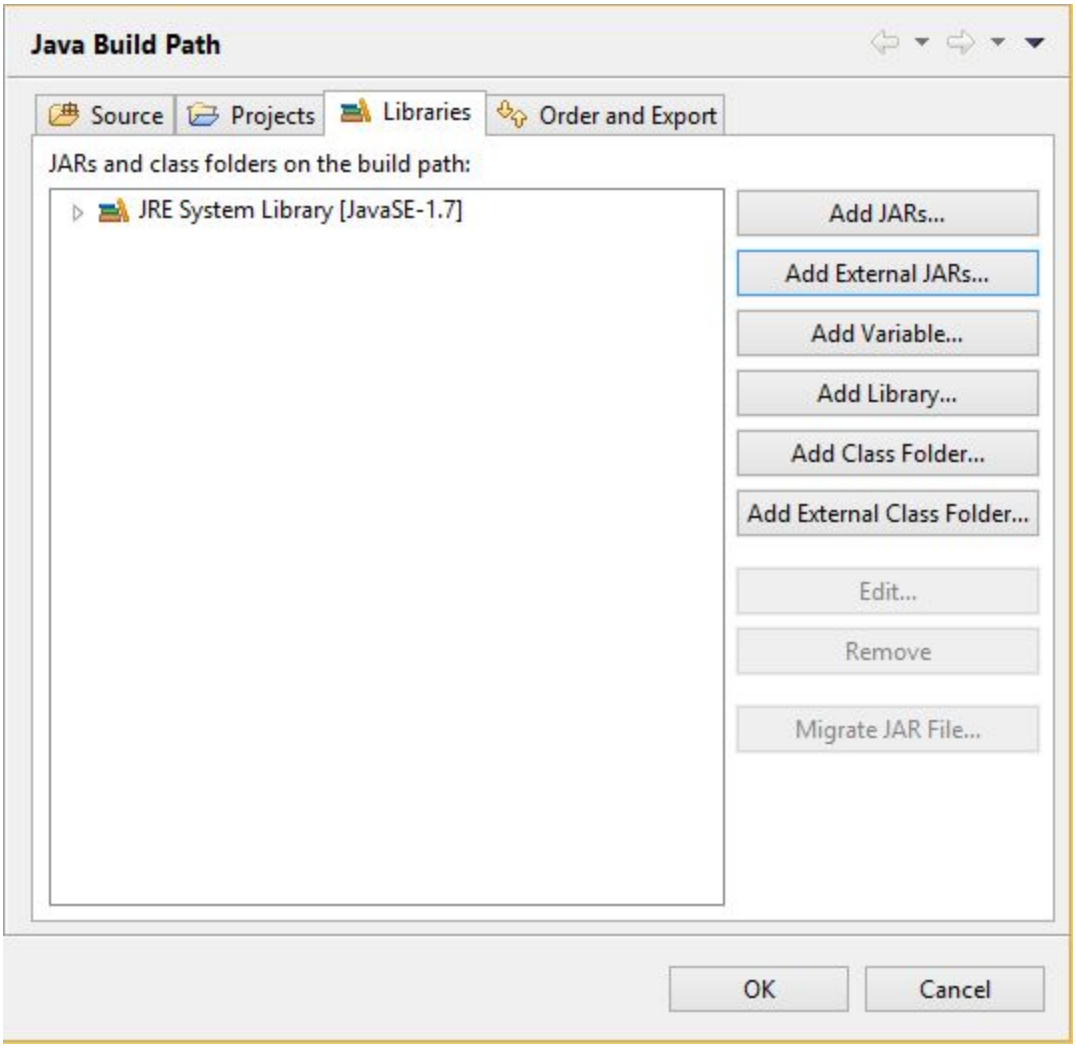


The native library has been renamed from libcscrypt to libsandkey.

Add the jar to the Project

1. Right-click on the project and select **Build Path**. Select either **Add External Archives** or **Configure Build Path**. The Properties page open. Go to the Libraries tab.

2. Click **Add External Jars** and find and select the downloaded jar file. Click **Open**.



3. Select **sandkey.jar** and click **OK**. The jar will now be included under the libraries folder in the project. In order to use the classes in the project, you must import UtilityClass.* Refer to the API documentation to create and use objects of these classes, or see the included examples.

NOTE: The JAR file works on the functions defined in the .NET DLL file that has the core code of the encryption/decryption operations. The latest .NET DLL version must be replaced to be able to use the updated/extended functions in java.

FieldShield Library Usage

In order to use the methods in Java, you must do several steps to create the object to be passed to the DLL. Following is the code sample to use the AES256 Encryption.

Once the object is instantiated, pass the pointer to the begin() method. Then set the password with the setPass method, providing the arguments of the pointer and String that is to be the password. Use the doTransform method by passing the pointer and the String (or double if it's the alphanumeric version for numbers). This method will return the encrypted String. Finally, you must deallocate the object by passing the pointer to the destruct or end method. The other two encryptions work in the same way.

Java Example

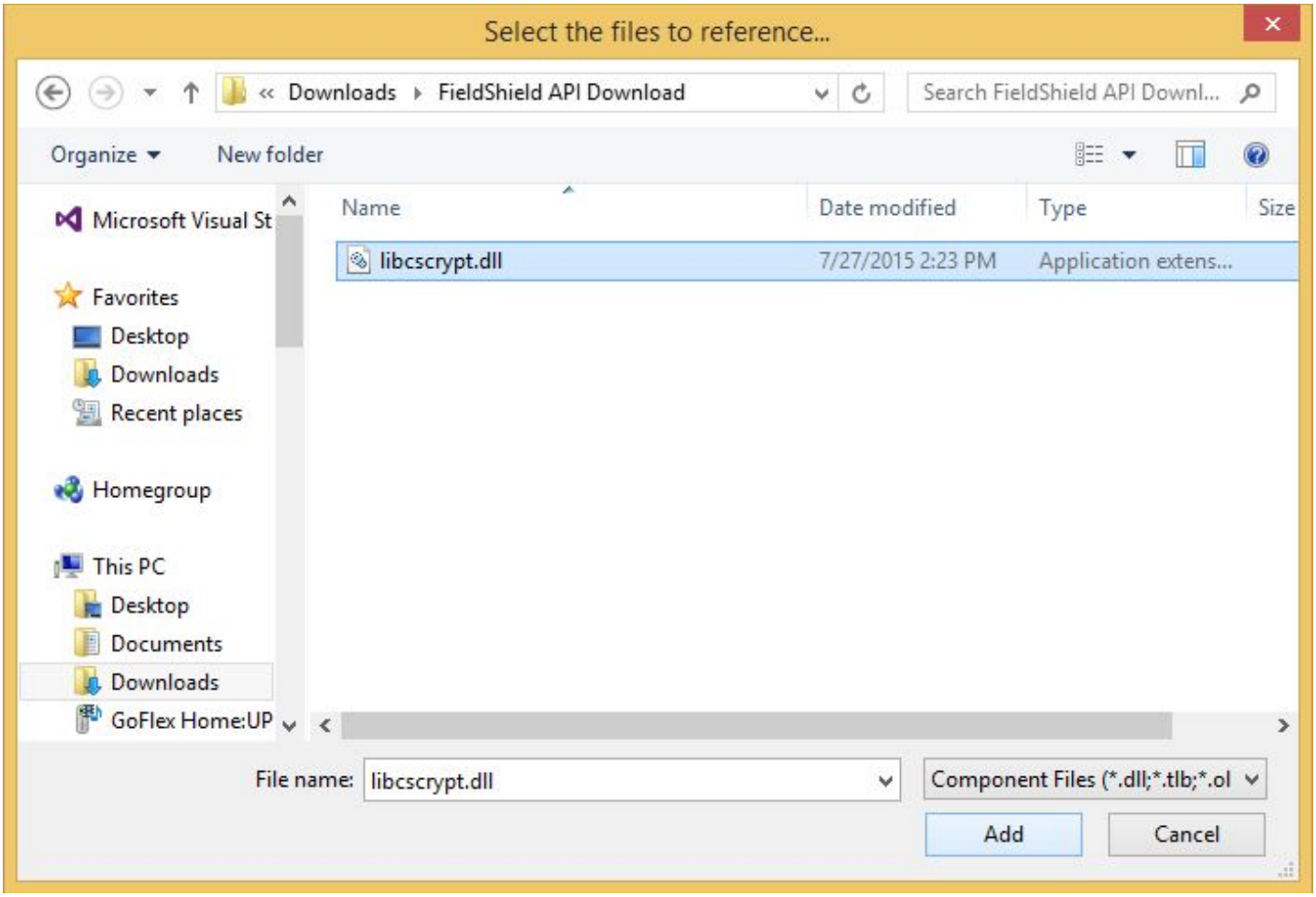
```
enc_aes256 obj = new enc_aes256();
iResult = obj.begin(obj.getptr());
obj.setPass(obj.ptr, "password");
String enc = obj.doTransform(obj.getptr(), "String to perform encryption on");
obj.destruct(obj.getptr());
```

Microsoft Visual Studio

Visual Studio 2010 was used when implementing the .NET objects for the API. Any of the languages supported in Visual Studio are capable of using the FieldShield SDK. Support documentation has been done for C++,C#, F# and Visual Basic. If using a language other than the ones documented, refer to the language's syntax in order to use the library.

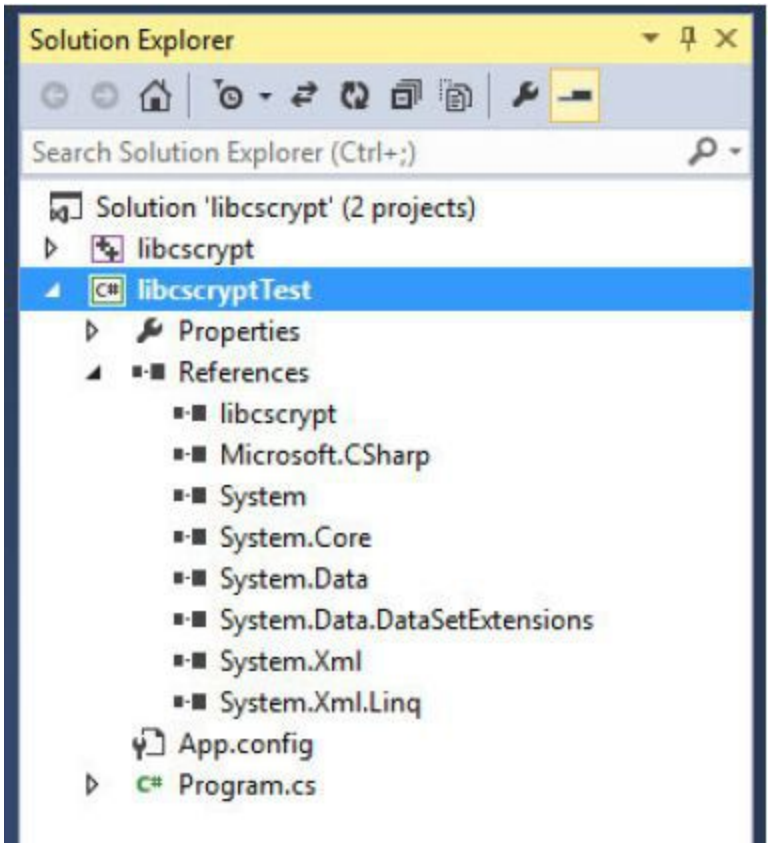
Installing Sandkey

1. Download Sandkey: libsandkey.dll
2. From Microsoft Visual Studio, open the project into which the DLL will be imported.
3. From the Solution Explorer window, right-click the project to add the reference, and select **Add** -> **Reference**. The Reference Manager opens.
4. Click **Browse** in the directory with the DLL. Select the DLL, and then click **Add**.



The native library has been renamed from libcscrypt to libsandkey.

5. Click **OK** to add the reference. It will display in Solution Explorer under the References section. Check to make sure the DLL is added. Once this is done, use the proper import keyword for the particular .NET language in which the methods will be used.



The following is the code sample to use the AES256 Encryption in C#. The other classes work in a similar manner. The HTML help file can be referenced for specific implementation in another language.

.NET Example in C#

```
enc_aes256 encryptAES256 = new enc_aes256();
iResult = encryptAES256.begin();
encryptAES256.setPass("password");
string test1 = encryptAES256.doTransform("String to perform encryption on");
encryptAES256.end();
```

Deliverables

Javadoc style documentation for the Sandkey libraries, as well as the Flow, Scl, and Ddf models, is available in the archive file: *com-iri-javadocs.zip*

The runtime libraries are in the file: *sandkey.jar*
Required platform specific shared native libraries are in: *sandkey-native.zip*

These files can all be downloaded together in a comprehensive archive:

> Sandkey support libraries and javadocs for data protection.
> Download token: 1de9048a5d31dee7a573a6baac70396b7b67e4ea
> File: *sandkey.zip*

Additional Resources

The latest revision of this document is available in Google Docs.