# SAMPLE CODE GUIDELINE

## Pos Digicert Sample Code for Soft Certificate (e-Invoice)

## REVISION CONTROL AND CHANGE HISTORY

| Revision Number | Approval Date | Approved By |
|---|---|---|
| Revision 1.0 | 20th May 2024 | Nazril Bin Mohd Ghani |
| Revision 1.1 | 18th July 2024 | Nazril Bin Mohd Ghani |

## Disclaimer

The information in this document is intended only for the person or entity to which it is addressed and may contain confidential and/or privileged material. Any review, retransmission, dissemination or other use of, or taking of any action in reliance upon this information by persons or entities other than the intended recipient is prohibited. If you received this document in error, please contact the sender and delete the material from all computers or medium.

## Ownership

This is the intellectual property of Pos Digicert Sdn Bhd (Pos Digicert) and all its components belong to Pos Digicert located in Star Central, Cyberjaya, Malaysia.

## Sample Code

The sample code in this document is provided "AS IS" and any express or implied warranties, including the implied warranties of merchantability and fitness for a particular purpose are disclaimed. We voluntarily put forth our effort to enhance the industry by providing this sample code and Pos Digicert will not be held liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) sustained by the client or a third party, however caused and on any theory of liability, whether in contract, strict liability, or tort arising in any way out of the use of this sample code, even if advised of the possibility of such damage.

## Feedback

To get further clarification on the Soft Certificate or its usage, email can be sent to einvoice@posdigicert.com.my

## Confidentiality Statement

This document is the property of Pos Digicert and no part thereof shall be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without written approval from the management of Pos Digicert.

## Contents

# 1 Introduction

Soft certificate for e-invoicing typically refers to a digital certificate stored in software rather than on a physical token like a smart card or USB token. This certificate is used for electronic transactions, including e-invoicing, to ensure the authenticity, integrity, and confidentiality of the transmitted data.

This document provides guidelines and sample code for Soft Certificate e-Invoice. To execute the sample code for reading and signing using a soft certificate, you will generally require the following:

**Soft Certificate:** Ensure you have a soft certificate installed on your system. This could be a certificate file (e.g.: .pfx, .p12) along with its password or any other form of soft certificate that your system supports.

**Programming Language:** You'll need to choose a programming language that supports cryptographic operations and provides libraries or APIs for working with certificates.

**Library or API:** You'll need to use a library or API that provides functions or classes for reading and signing with certificates. For example, in Python, you might use the ‘cryptography’ library or ‘pyOpenSSL’. In Java, you might use java.security package.

**Sample Code:** You'll need a sample code that demonstrates how to read and sign using certificate.

For now, we are sharing the sample code in three languages:

i. **Java**
ii. **.NET**
iii. **PHP**

(this space is intentionally left blank)

## 2 General Guide

Here's a general guide on how you might proceed:

1. **Install Necessary Libraries:**
   Install the necessary libraries or packages for your chosen programming language If you haven't already.

2. **Import the Soft Certificate:**
   Import your soft certificate into the sample code directory.

3. **Download the Sample Code:**
   Download the sample code provided.

4. **Adjust Sample Code:**
   Adjust the sample code to use your specific soft certificate file and any other relevant details such as the password for the certificate.

5. **Run the Code:**
   Finally, run the code and verify that it works as expected. Make sure to handle any errors or exceptions that might occur during the process.

**Important Note:** The developer who implement e-Invoice Middleware / Service Provider should consult IRBM / e-Invoice SDK on the correct attributes on the XADES document regarding certificate/signed data.

(this space is intentionally left blank)

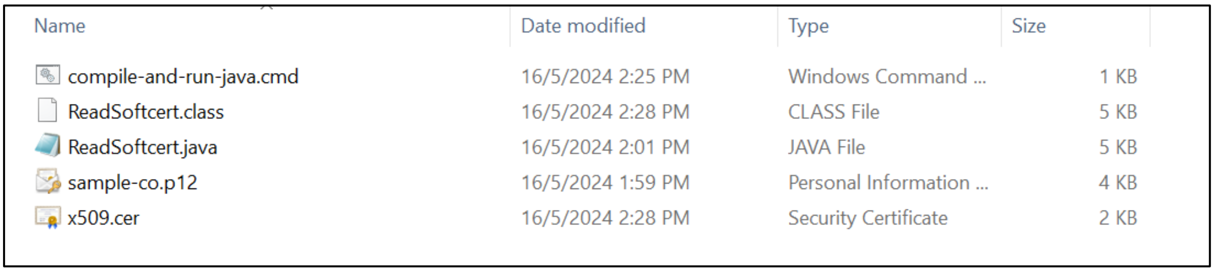
# 3 Sample Code in Java

Sample code to read and sign using soft cert in Java could be downloaded from:

| | |
|---|---|
| URL | https://posdigicertsupport.freshdesk.com/a/solutions/articles/69000855244?lang=en |
| ZIP password | MYCRS_TCNRYYC5!#$ |

The downloaded zip file will include:

a. Source code
b. Sample p12 certificate
c. Windows script to compile and run the code



| Name | Date modified | Type | Size |
|---|---|---|---|
| compile-and-run-java.cmd | 16/5/2024 2:25 PM | Windows Command ... | 1 KB |
| ReadSoftcert.class | 16/5/2024 2:28 PM | CLASS File | 5 KB |
| ReadSoftcert.java | 16/5/2024 2:01 PM | JAVA File | 5 KB |
| sample-co.p12 | 16/5/2024 1:59 PM | Personal Information ... | 4 KB |
| x509.cer | 16/5/2024 2:28 PM | Security Certificate | 2 KB |

(this space is intentionally left blank)

## Sample Code

*****************************************************************************************

* THIS SAMPLE CODE COMES WITHOUT ANY WARRANTY.

* YOU ARE FREE TO ALTER AT ANY TIME BASED ON YOUR SYSTEM REQUIREMENT, WHERE APPROPRIATE.

* THE SAMPLE CODE IS BASED ON JAVA 1.8 USING STANDARD JAVA LIBRARY.

* NO THIRD-PARTY LIBRARY IS REQUIRED TO RUN THIS SAMPLE CODE.

* YOU MIGHT WANT TO USE THIRD PARTY LIBRARIES TO EASE DEVELOPMENT, IN ADDITION TO THIS SAMPLE CODE.

****************************************************************************************

```java
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.security.InvalidKeyException;
import java.security.KeyStore;
import java.security.KeyStoreException;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.Signature;
import java.security.SignatureException;
import java.security.UnrecoverableKeyException;

public class ReadSoftcert {

        public static void main(String[] args) {

                String PIN = "12345678";
                String softcertFile = "sample-co.p12";
                String dataToSign = "<xml>e-invoice data</xml>";

                byte[] signData = null;
                byte[] softcertBytes = null;
                PrivateKey privateKey = null;
                String alias = "";
                X509Certificate x509 = null;

                try {
```

```
    /****************************************************************************
    * 1. Read soft-cert into bytes
    ****************************************************************************/
    softcertBytes = Files.readAllBytes(Paths.get(softcertFile));
   KeyStore store = ReadSoftcert.loadKeyStore(softcertBytes, PIN);

   /****************************************************************************
   * 2. Find private key and user x509 certificate
   ****************************************************************************/
Enumeration<String> e = store.aliases();
                     for (; e.hasMoreElements();) {

                     alias = (String) e.nextElement();

                     if (store.isKeyEntry(alias)) {
                     privateKey = (PrivateKey) store.getKey(alias, PIN.toCharArray());

                     x509 =  (X509Certificate) store.getCertificate(alias);
                     CertificateFactory cf = CertificateFactory.getInstance("X.509");
                     x509 = (X509Certificate)cf.generateCertificate(new
                     ByteArrayInputStream(x509.getEncoded()));

                     //print certificate details
                     System.out.println(x509.toString());

                     //write x509 certificate into file
                     Files.write(Paths.get("x509.cer"), x509.getEncoded());
                            }

                     }
  /****************************************************************************
  * 3. Perform signing with SHA256RSA algorithm
   ****************************************************************************/
                      Signature sig = Signature.getInstance("SHA256withRSA");
                      sig.initSign(privateKey);
                       sig.update(dataToSign.getBytes());

                      signData = sig.sign();

  /****************************************************************************
  * 4. Convert signed data and x509 certificate to Base64 format
  ****************************************************************************/

                String signedData = new String(Base64.getEncoder().encode(signData));
                System.out.println("\n SignatureValue : " + signedData);


                String certBase64 = new String(Base64.getEncoder().encode(x509.getEncoded()));
                System.out.println("\n X509Certificate : " + certBase64);


                 String X509IssuerName = x509.getIssuerDN().getName();
                 System.out.println("\n X509IssuerName : " + X509IssuerName);
                 } catch (IOException e) {
                     e.printStackTrace();
                 } catch (KeyStoreException e) {
                      e.printStackTrace();
                 } catch (NoSuchAlgorithmException e) {
                      e.printStackTrace();
                 } catch (CertificateException e) {
                      e.printStackTrace();
                 } catch (UnrecoverableKeyException e1) {
                      e1.printStackTrace();
                 } catch (InvalidKeyException e1) {
                       e1.printStackTrace();
                 } catch (SignatureException e1) {
                     e1.printStackTrace();
                 }


         }

         public static KeyStore loadKeyStore(byte[] fileInBytes, String PIN)
          throws KeyStoreException, NoSuchAlgorithmException, CertificateException, IOException
         {
                      KeyStore keyStore = KeyStore.getInstance("PKCS12");
                      keyStore.load(new ByteArrayInputStream(fileInBytes), PIN.toCharArray());


                      return keyStore;

          }

     }
```

(this space is intentionally left blank)

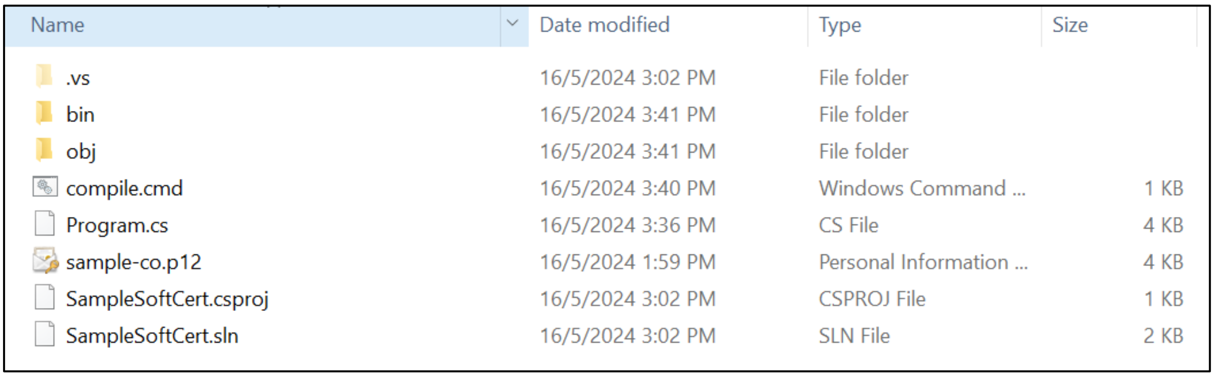
# 4 Sample Code in .NET

Sample code to read and sign using soft cert in .NET could be downloaded from:

| | |
|---|---|
| URL | https://posdigicertsupport.freshdesk.com/a/solutions/articles/69000855245?lang=en |
| ZIP password | MYCRS_TCNRYYC5!#$ |

The downloaded zip file will include:

a. Source code
b. Sample p12 certificate
c. Windows script to compile and run the code



| Name | Date modified | Type | Size |
|---|---|---|---|
| .vs | 16/5/2024 3:02 PM | File folder | |
| bin | 16/5/2024 3:41 PM | File folder | |
| obj | 16/5/2024 3:41 PM | File folder | |
| compile.cmd | 16/5/2024 3:40 PM | Windows Command ... | 1 KB |
| Program.cs | 16/5/2024 3:36 PM | CS File | 4 KB |
| sample-co.p12 | 16/5/2024 1:59 PM | Personal Information ... | 4 KB |
| SampleSoftCert.csproj | 16/5/2024 3:02 PM | CSPROJ File | 1 KB |
| SampleSoftCert.sln | 16/5/2024 3:02 PM | SLN File | 2 KB |

(this space is intentionally left blank)

## 4.1 Sample Code

```
using System;

using System.Security.Cryptography;
using System.Security.Cryptography.X509Certificates;
using System.Numerics;

namespace SampleSoftCert
{
        class Program
        {

                  // Helper method to convert a hexadecimal string to a byte array
                  private static byte[] HexStringToByteArray(string hex)
                  {
                       int length = hex.Length;
                       byte[] bytes = new byte[length / 2];
                       for (int i = 0; i < length; i += 2)
                       {
                           bytes[i / 2] = Convert.ToByte(hex.Substring(i, 2), 16);
                       }
                        return bytes;
                    }

                    static void Main(string[] args)
                   {


                       String PIN = "12345678";
                       String softcertFile = "D: \\WORKSPACE\\ SampleSoftCert\\sample-co.p12";
                       String dataToSign = "<xml>e-invoice data</xml>";

                 // Load the .p12 file into an X509Certificate2 object
                 X509Certificate2 certificate = new X509Certificate2(softcertFile, PIN);

                 // Check if the certificate has a private key
                 if (!certificate.HasPrivateKey)
                 {
                    Console.WriteLine("Certificate does not have a private key.");
                    return;

                 }
        // Get the private key from the certificate
        RSA privateKey = certificate.GetRSAPrivateKey();
        Console.WriteLine(certificate.Issuer);
        Console.WriteLine(certificate.SerialNumber);
        // Export the certificate to a byte array
        byte[] certBytes = certificate.Export(X509ContentType.Cert);

        // Convert the byte array to a Base64 string
       string certBase64 = Convert.ToBase64String(certBytes);

       // Display the Base64 encoded certificate
       Console.WriteLine("Base64 Encoded Certificate:");
       Console.WriteLine(certBase64);

       // Get the serial number as a hexadecimal string
       string serialNumberHex = certificate.SerialNumber;

       // Convert the hexadecimal string to a byte array
       byte[] serialNumberBytes = HexStringToByteArray(serialNumberHex);

        // Reverse the byte array to match the little-endian format
        Array.Reverse(serialNumberBytes);

        // Convert the byte array to a BigInteger
        BigInteger serialNumberBigInt = new BigInteger(serialNumberBytes);

        // Display the serial number as an integer
        Console.WriteLine("Serial Number (BigInteger): " + serialNumberBigInt);

       // Initialize the signature object
       using (RSA rsa = privateKey)
             {
               // Create an instance of the SHA256 hash algorithm
               using (SHA256 sha256 = SHA256.Create())
               {
               // Compute the hash of the data
 byte[] hash = sha256.ComputeHash(System.Text.Encoding.UTF8.GetBytes(dataToSign));

              // Create an RSA signature formatter
 RSAPKCS1SignatureFormatter rsaFormatter = new RSAPKCS1SignatureFormatter(rsa);
 rsaFormatter.SetHashAlgorithm("SHA256");
```

```
            // Create the signature
            byte[] signature = rsaFormatter.CreateSignature(hash);

            // Convert the signature to a base64 string for display
            string signatureBase64 = Convert.ToBase64String(signature);

            Console.WriteLine("Signature: " + signatureBase64);



                  }
                 }


              }
    }

}
```

(this space is intentionally left blank)

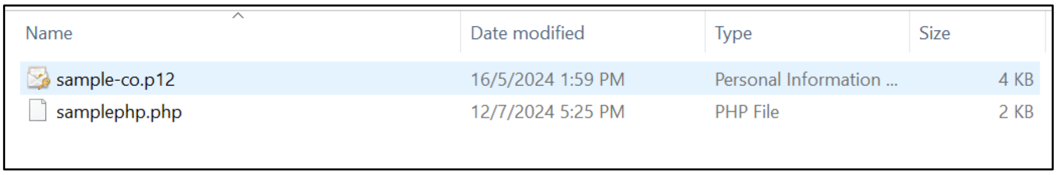
# 5 Sample Code in PHP

Sample code to read and sign using soft cert in PHP could be downloaded from:

| | |
|---|---|
| URL | https://posdigicertsupport.freshdesk.com/a/solutions/articles/69000857292?lang=en |
| ZIP password | MYCRS_TCNRYYC5!#$ |

The downloaded zip file will include:

a. Source code
b. Sample p12 certificate



| Name | Date modified | Type | Size |
|---|---|---|---|
| sample-co.p12 | 16/5/2024 1:59 PM | Personal Information ... | 4 KB |
| samplephp.php | 12/7/2024 5:25 PM | PHP File | 2 KB |

(this space is intentionally left blank)

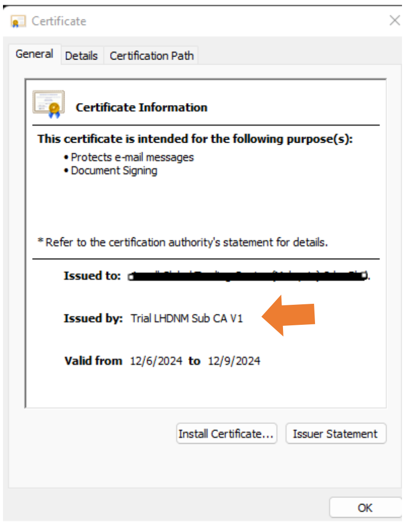
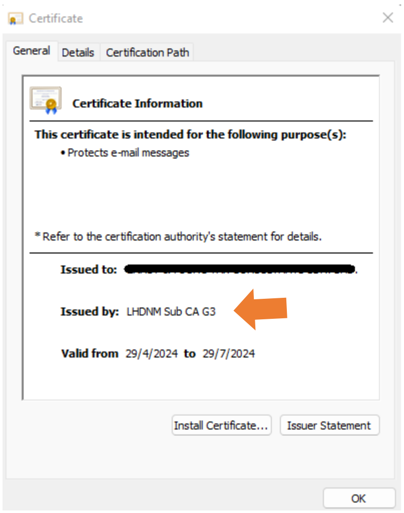
## Sample Code

```php
<?php


// Get the current script path
$currentPath = dirname(__FILE__);

// Output the current script path
echo 'Current script path: ' . $currentPath . PHP_EOL;

// Path to the .p12 file
$p12FilePath = $currentPath . DIRECTORY_SEPARATOR ."sample-co.p12"; //include your softcert here
$p12Password = "zaq12wsx"; //include your softcert pin here
echo $p12FilePath ;
echo "\n";

// Check if the file exists and is readable
if (!file_exists($p12FilePath)) {
  die('The .p12 file does not exist: ' . $p12FilePath);
}
if (!is_readable($p12FilePath)) {
  die('The .p12 file is not readable: ' . $p12FilePath);
}


// Load the .p12 file
try {
    $p12Content = file_get_contents($p12FilePath);
    if ($p12Content === false) {
        die('Failed to read the .p12 file');
    }
}

//catch exception
catch(Exception $e) {
 echo 'Message: ' .$e->getMessage();
}
```

```php
// Extract the certificate and private key from the .p12 file
$certs = [];
if (!openssl_pkcs12_read($p12Content, $certs, $p12Password)) {
    // Detailed error message
    while ($error = openssl_error_string()) {
        echo "OpenSSL Error: $error\n";
    }
    die('Failed to parse the .p12 file');
}

// The private key
$privateKey = openssl_pkey_get_private($certs['pkey']);

// Data to sign
$data = 'This is the data to sign';

// Sign the data
$signature = '';
if (!openssl_sign($data, $signature, $privateKey, OPENSSL_ALGO_SHA256)) {
    die('Failed to sign the data');
}

// Free the private key from memory
openssl_free_key($privateKey);

// Output the signature in base64 format
echo 'Signature: ' . base64_encode($signature) . PHP_EOL;

// If needed, output the certificate
echo 'Certificate: ' . $certs['cert'] . PHP_EOL;
?>
```

# 6 Soft Certificate Details

Below are the details to differentiate Soft Certificate Chain for Sandbox / Preprod and the Production environment.

| ENVIRONMENT | SANDBOX / PREPROD | PRODUCTION |
|---|---|---|
| Pos Digicert Certificate Chain | **Intermediate:**<br>**CN** = Trial LHDNM Sub CA V1<br>**OU** = Terms of use at http://www.posdigicert.com.my<br>**O** = LHDNM<br>**C** = MY<br>**Root:**<br>**CN** = Trial Pos Digicert Class 2 Root CA V1<br>**OU** = 457608-K<br>**O** = Pos Digicert Sdn. Bhd.<br>**C** = MY<br>How to identify:<br><br> Certificate<br>General<br>Details<br>Certification Path<br>Install Certificate...<br>Issuer Statement<br>OK<br>Certificate Information<br>This certificate is intended for the following purpose(s):<br>• Protects e-mail messages<br>• Document Signing<br>* Refer to the certification authority's statement for details.<br>Issued to: [redacted].<br>Issued by: Trial LHDNM Sub CA V1<br>Valid from 12/6/2024 to 12/9/2024 | **Intermediate:**<br>**CN** = LHDNM Sub CA G3<br>**OU** = Terms of use at http://www.posdigicert.com.my<br>**O** = LHDNM<br>**C** = MY<br>**Root:**<br>**CN** = Pos Digicert Class 2 Root CA G3<br>**OU** = 457608-K<br>**O** = Pos Digicert Sdn. Bhd.<br>**C** = MY<br>How to identify:<br><br> Certificate<br>General<br>Details<br>Certification Path<br>Install Certificate...<br>Issuer Statement<br>OK<br>Certificate Information<br>This certificate is intended for the following purpose(s):<br>• Protects e-mail messages<br>* Refer to the certification authority's statement for details.<br>Issued to: [redacted].<br>Issued by: LHDNM Sub CA G3<br>Valid from 29/4/2024 to 29/7/2024 |

(End of Document)