
# Integration Guide

## pDoc Pro Server SDK

**Version 2.1**

Last Update: March 20, 2018

# Table of Contents

**1.0 – Introduction ..... 4**

**2.0 – Installing/Uninstalling pDoc Pro Server SDK ..... 5**

2.1 – Before You Begin ..... 5

2.2 – Installing pDoc Pro Server SDK ..... 5

2.3 – Uninstalling pDoc Pro Server SDK ..... 5

**3.0 – Features of the SDK ..... 6**

**4.0 – Add pDoc Pro Server SDK Reference to Your Project ..... 7**

**5.0 – Namespaces ..... 11**

**6.0 – Enumerations ..... 12**

6.1 - The ClientStatus enumeration ..... 12

6.2 - The ClientActionType enumeration ..... 12

6.3 - The DocReceiptStatus enumeration ..... 13

6.4 - The HomeScreenType enumeration ..... 13

6.5 - The ScrollBarOptions enumeration ..... 14

# Table of Contents

**7.0 – Classes ..... 15**
7.1 - ClientInfo Class ..... 15
*7.1.1 Properties ..... 15*
7.2 - ClientHomeScreen Class ..... 16
*7.2.1 Properties ..... 16*
7.3 - SigningClient Class ..... 17
*7.3.1 Properties ..... 17*
*7.3.2 Methods ..... 18*
7.3.2.1 GetListOfClients ..... 18
7.3.2.2 SelectClient ..... 18
7.3.2.3 GetClientStatus ..... 19
7.3.2.4 DiscoverClients ..... 19
7.3.2.5 PushPDF ..... 20
7.3.2.6 SetupHomeScreen ..... 21
7.3.2.7 TerminateHomeScreen ..... 22
7.3.2.8 TerminateSigningSession ..... 23
*7.3.3 Events ..... 24*
7.3.3.1 docReceived ..... 24
7.3.3.2 clientDiscovered ..... 26

**8.0 - Parameter File ..... 27**
8.1 - Parameter File Schema (From Server to Client) ..... 27
8.2 - Parameter File Schema (From Client to Server) ..... 33

# 1.0 – Introduction

The pDoc® Pro eSign Software System provides an electronic signature solution for environments where a system operator pushes a PDF document from their PC to a client for signing or annotating. The pushed document can be a form, contract, or other document for a user to sign or annotate. The user completes a document by filling in text boxes, selecting check boxes and radio buttons, selecting items from lists, and signing or annotating the document. When the user has finished and taps the DONE button icon, the client sends the completed document back to the originator's PC where it is stored and available to the operator. Then the client goes back to sleep, playing a video or displaying a sequence of images, until it receives another document for signing or annotating.

The pDoc Signer application running on the pDoc Pro Client for signing captures the handwritten biometric signature plus the time and date of the captured signature when the document is sent for "Signing". Also, the integrity value (hash) of the document will be calculated and stored in the signature field together with the biometric signature. The captured signature is displayed on the corresponding signature field in the document and can be viewed and verified by a variety of PDF document rendering applications like Adobe Acrobat and Adobe Reader.

If the client usage is "Annotating", the pDoc Signer application running on the pDoc Pro Client for annotating captures pencil annotations in the document that can be viewed by other PDF document rendering applications like Adobe Acrobat and Adobe Reader.

The pDoc Pro Server SDK is an SDK that can be integrated into applications that want to leverage the functionality of pDoc Pro. An application running on a PC can integrate this SDK and operate or manipulate the pDoc Pro Client.

# 2.0 – Installing/Uninstalling pDoc Pro Server SDK

The pDoc Pro Server SDK is part of pDoc Pro SDK Server software.

## 2.1 – Before You Begin

- pDoc Pro SDK Server runs on Windows operating systems beginning with Windows 7.
- Your system should have a minimum of 50 MB free space on the hard drive, in addition to the free space requirements for Windows.
- Before installing a new version of pDoc Pro SDK Server, uninstall any older versions of pDoc Pro SDK Server on your PC.
- pDoc Pro Clients may be dedicated or non-dedicated Windows tablets or PCs. See the separate pDoc Pro Client User Manual for additional information.

## 2.2 – Installing pDoc Pro Server SDK

The pDoc Pro Server SDK is installed as part of the pDoc Pro SDK Server software. Installing pDoc Pro SDK Server software is accomplished by running the pDoc Pro SDK Server installation file provided. During the installation you will be provided with an option to agree or disagree to the license agreement, prompted to enter user information, and optionally set a different folder for installation. The sequence of screens displayed guide the user through the installation process.

## 2.3 – Uninstalling pDoc Pro Server SDK

The pDoc Pro Server SDK is uninstalled when the pDoc Pro SDK Server software is uninstalled from a PC. To uninstall pDoc Pro SDK Server:

- Go to Start→ Control Panel → Programs and Features.
- Select pDoc Pro SDK Server and click on the “Uninstall” button. Follow the instructions to uninstall the software.

The instructions are similar for other Windows operating systems.

*Note: Deleting the pDoc Pro SDK Server installation folder directly will not uninstall the software completely.*

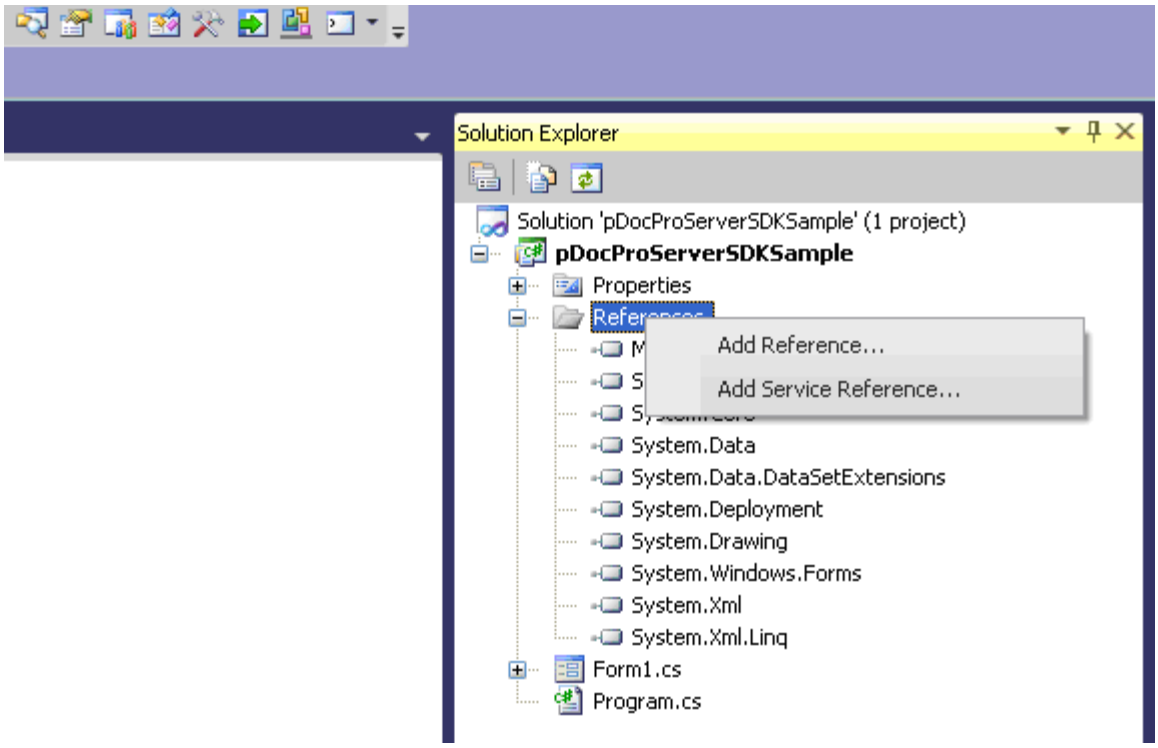
## 3.0 – Features of the SDK

- Set the Client Information. Required client information is Client Identifier and the Port Number for communication with the client
- Check the client state (available, busy, not available)
- Terminate a client session (e.g., in case of a timeout or other error).
- Specify if the document has to be signed or annotated.
- Specify the PDF file to be pushed to the client.
- Specify the parameter file to be pushed to the client.
- Push the PDF and associated parameter file to the specified client.
- Set the home screen display type (Basic, Images (default), or Video).
- Specify the list of images to display on the home screen (maximum of 10)
- Set the time interval to display each image
- Select the video to be run
- Specify whether Video Player Bar should be displayed or not
- Setup the Home Screen (i.e., push the video or images to the client)
- Terminate the home screen on the client, i.e. revert it back to a normal Windows client
- Check if signed or annotated PDF document has been received from the client
- Get the signed or annotated PDF document received from the client
- Get the updated parameter file received from the client

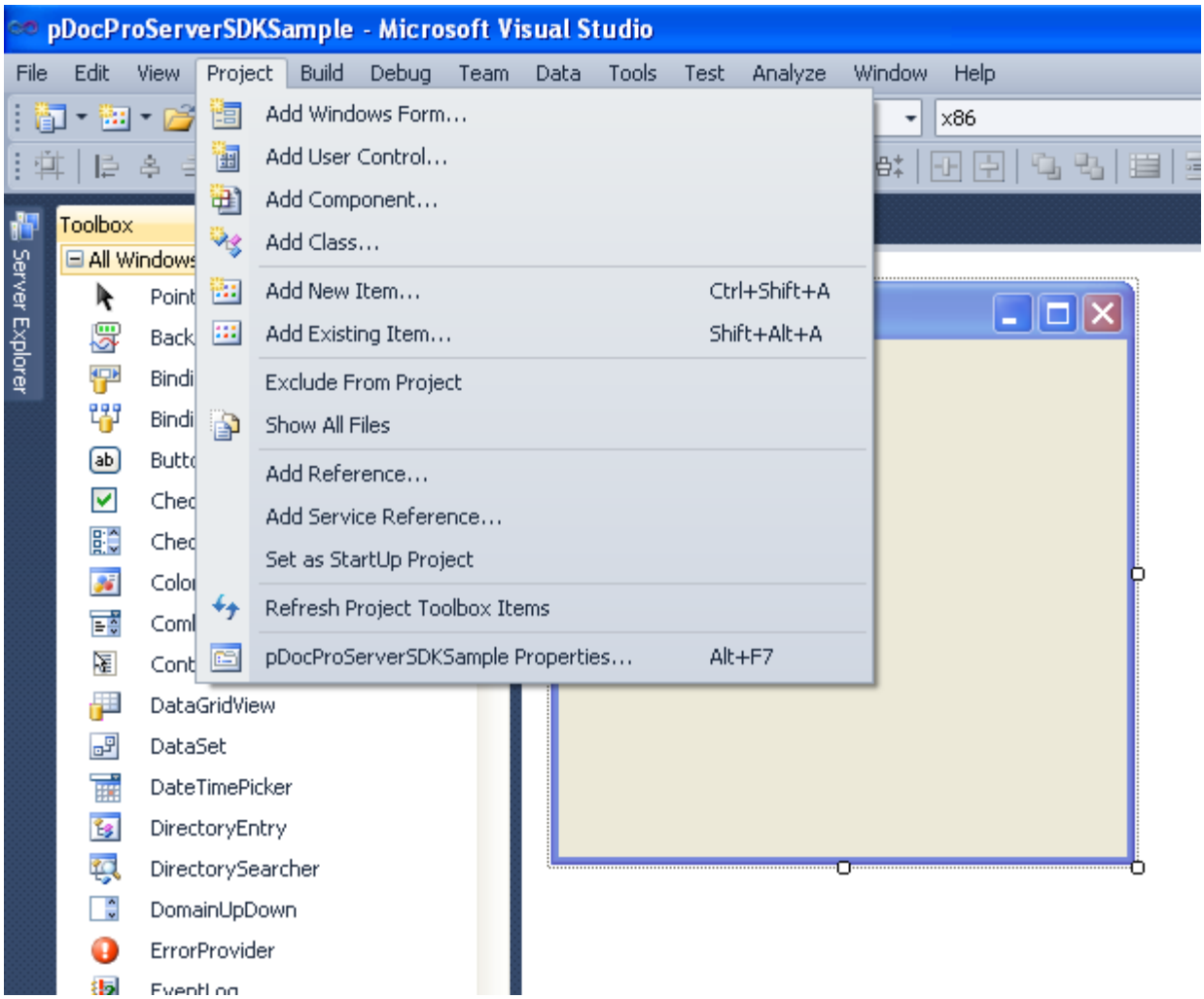
# 4.0 – Add pDoc Pro Server SDK Reference to Your Project

Before you begin to use the various API calls described in this document you should add a reference to the pDoc Pro Server SDK assembly to your project. This section describes how this can be done using Visual Studio 2008.

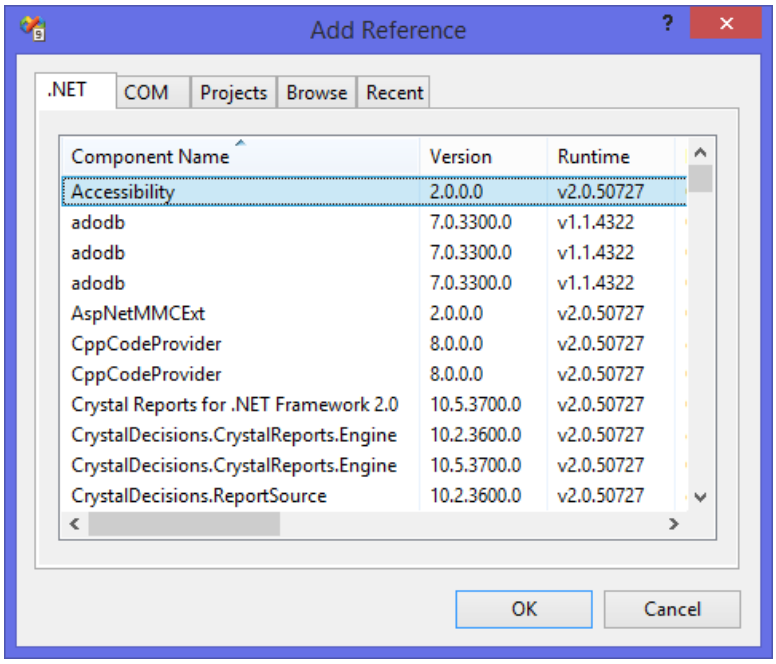
1. In the solution explorer right click on the project node and select “Add Reference”.



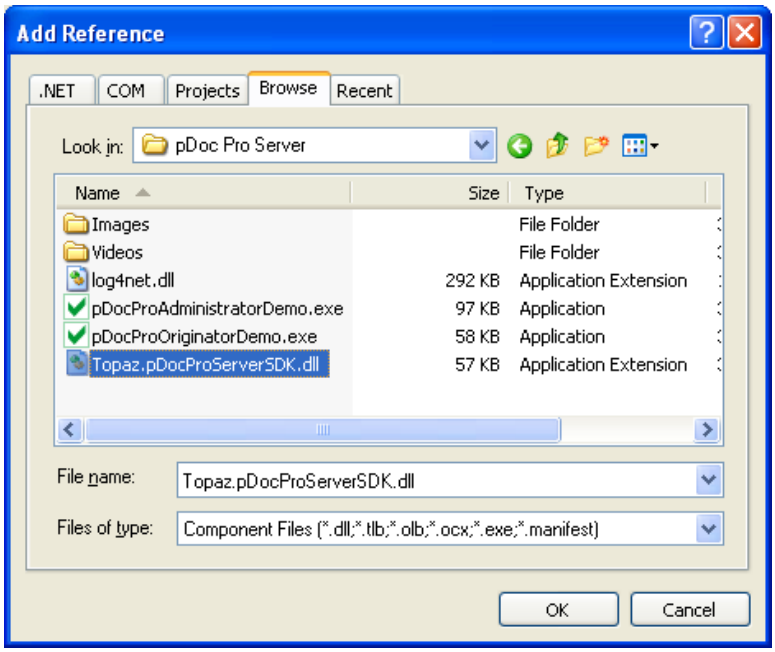

You can also add the reference using the main menu. Navigate to “Project” and then select “Add Reference”.


pDocProServerSDKSample - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools Test Analyze Window Help
x86
Add Windows Form...
Add User Control...
Add Component...
Add Class...
Add New Item... Ctrl+Shift+A
Add Existing Item... Shift+Alt+A
Exclude From Project
Show All Files
Add Reference...
Add Service Reference...
Set as StartUp Project
Refresh Project Toolbox Items
pDocProServerSDKSample Properties... Alt+F7
Toolbox
Server Explorer
DataGridView
DataSet
DateTimePicker
DirectoryEntry
DirectorySearcher
DomainUpDown
ErrorProvider

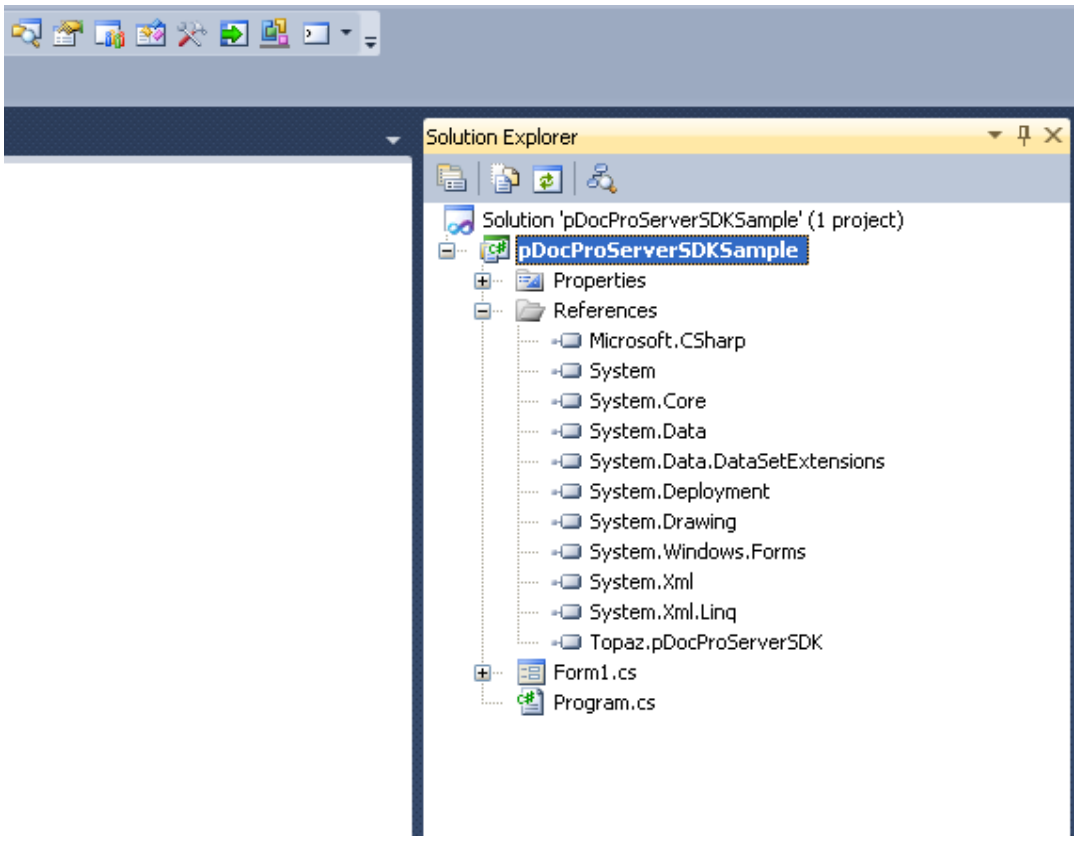
2. The following dialog will appear.




3. Select the “Browse” tab and navigate to the folder where the “Topaz.pDocProServerSDK.dll” is located (typically the pDoc Pro SDK Server installation folder) and select the assembly. Click “OK” to add the assembly to your project.



4. The assembly will be added to the project as shown. Now you can start using the API exposed by the pDoc Pro Server SDK.



## 5.0 – Namespaces

The API calls are encapsulated within the following namespaces.

- Topaz.pDocProServerSDK.ClientInfo
  - This namespace holds the class containing information about a pDoc Pro Client.
- Topaz.pDocProServerSDK.ClientHomeScreen
  - This namespace holds the class containing information about a pDoc Pro Client Home Screen.
- Topaz.pDocProServerSDK.SigningClient
  - This namespace will hold the main class required for administering a pDoc Pro Client. This class will encapsulate methods, properties, and fields required to administer the pDoc Pro Client.

# 6.0 – Enumerations

## 6.1 - The ClientStatus enumeration

Enumeration that lists the available client statuses.

**Namespace**

**Topaz.pDocProServerSDK.SigningClient**

**Syntax**

```
public enum ClientStatus
{
        Available = 1,
        Busy = 2,
        NotAvailable = 3
}
```

## 6.2 - The ClientActionType enumeration

Enumeration that lists the client action types.

**Namespace**

**Topaz.pDocProServerSDK.SigningClient**

**Syntax**

```
public enum ClientActionType
{
        Signing = 1,
        Annotating = 2
}
```

### 6.3 - The DocReceiptStatus enumeration

Enumeration that lists the DocReceiptStatus.

**Namespace**

**Topaz.pDocProServerSDK.SigningClient**

**Syntax**

```
public enum DocReceiptStatus
{
        Accepted = 1,
        Rejected = 2,
        None = 3
}
```

### 6.4 - The HomeScreenType enumeration

Enumeration that lists the HomeScreenTypes.

**Namespace**

**Topaz.pDocProServerSDK.ClientHomeScreen**

**Syntax**

```
public enum HomeScreenType
{
        Basic = 0,
        Images = 1,
        Video = 2
}
```

### 6.5 - The ScrollBarOptions enumeration

Enumeration that lists the Scroll Bar Options available for the client.

**Namespace**

**Topaz.pDocProServerSDK.ClientHomeScreen**

**Syntax**

```
public enum ScrollBarOptions
{
    NoScrollBar = 1,
    DefaultScrollBar = 2,
    CustomScrollBar = 3
}
```

# 7.0 – Classes

## 7.1 - ClientInfo Class

Class that provides information about a specific pDoc Pro Client. Contains information like Identifier (Name or IP Address) and Port Number used to communicate with the client.

### *7.1.1 Properties*

| | Name | Syntax | Description |
|---|---|---|---|
| | ClientIdentifier | public string ClientIdentifier {get;set;} | The identifier for the pDoc Pro Client. The identifier can be the name or IP Address of the client. |
| | PortNumber | public int PortNumber<br>{ get; set; } | The port number used to communicate with the pDoc Pro Client. |
| | Client_Status | public ClientStatus Client_Status<br>{ get; set; } | The status of the client (Available, Not Available or Busy) |
| | SigningSession TimeOut | public int SigningSessionTimeOut<br>{ get; set; } | Specifies the time out period for the Signing or Annotating session in minutes. Default value is 10 minutes. |
| | CheckForExternalPad | public bool CheckForExternalPad<br>{ get; set; } | Specifies whether signing with an External Signature Pad is allowed or not. Useful when the clients are Windows tablets to improve the signing experience for signing process. Default value is False. |

## 7.2 - ClientHomeScreen Class

Class that contains information about the home screen of the client.

### *7.2.1 Properties*

| | Name | Syntax | Description |
|---|---|---|---|
| | ImageDuration | public int ImageDuration { get; set; } | Time period to display each image in seconds |
| | ImageList | public string[] ImageList { get; set; } | List of images to be run on home screen |
| | VideoFile | public string VideoFile { get; set; } | Video to be run on the home screen |
| | Password | public string Password { get; set; } | Password to terminate the program on the pDoc Pro Client |
| | ScreenType | public HomeScreenType screenType { get; set; } | Home Screen Type (i.e., Basic, Images, or Video) |
| | DisplayVideoPlayerBar | public bool DisplayVideoPlayerBar { get; set; } | Specifies if the video player bar needs to be displayed or not. Applicable only when the home screen type is set to “Video” |
| | ScrollBarType | public ScrollBarOptions scrollBarOption { get; set; } | Sets scrollbar option (i.e., NoScrollBar, DefaultScrollBar, and CustomScrollBar) |
| | ScrollBarWidth | public int ScrollBarWidth { get; set; } | Specifies the scrollbar width for the CustomScrollBar option |

### 7.3 - SigningClient Class

The primary class of the SDK that an application can use to administer a pDoc Pro Client.

#### *7.3.1 Properties*

| | Name | Syntax | Description |
|---|---|---|---|
|  | PDFFilePath | Public string PDFFilePath { get; set; } | The path of the PDF File to be pushed to the client. |
|  | ParameterFilePath | Public string ParameterFilePath { get; set; } | The path of the Parameter File to be pushed to the client. |
|  | ClientAction | Public ClientActionType ClientAction { get; set; } | The Client Action Type for the document pushed to the client. It can be either “Signing” or “Annotating”. |
|  | SendingTime | Public string SendingTime { set; } | The time at which the PDF document was sent to the client. |
|  | ErrorCode | Public string ErrorCode { set; } | Error Code returned if any of the functions is not successful. |

### 7.3.2 Methods

#### 7.3.2.1 GetListOfClients

**Method Description**

Provides the configured list of clients (by the administrator) as an array of ClientInfo classes.

**Syntax**

```
public ClientInfo[] GetListOfClients(string AuthID)
```

**Parameters**

AuthID
Type: string
Authentication Identifier.

**Return Value**

ClientInfo array

#### 7.3.2.2 SelectClient

**Method Description**

Allows selecting a pDoc Pro Client with which to start communication.

**Syntax**

```
public void SelectClient(ClientInfo clientInfo)
```

**Parameters**

*clientInfo*
Type: ClientInfo

ClientInfo class containing the Name or IP Address, and Port Number of the pDoc Pro Client with which to communicate.

**Return Value**

No return value

#### *7.3.2.3 GetClientStatus*

**Method Description**

Gets the status of the specified pDoc Pro Client.

**Syntax**

```
public ClientStatus GetClientStatus(string AuthID)
```

**Parameters**

AuthID
Type: string
Authentication Identifier.

**Return Value**

ClientStatus containing the pDoc Pro Client status (Available, Busy, and NotAvailable).

#### *7.3.2.4 DiscoverClients*

**Method Description**

Sends a broadcast over the network to discover the available pDoc Pro Clients.

**Syntax**

```
public void DiscoverClients()
```

**Return Value**

No return value. But, whenever a client responds to the broadcast message, the ClientDiscovered event is fired. Refer to the event for more information.

### *7.3.2.5 PushPDF*

**Method Description**

Pushes the selected PDF to the specified pDoc Pro Client. The PDFFilePath and ParameterFilePath properties have to be set before this method is invoked.

**Syntax**

```
public bool PushPDF(string AuthID)
```

**Parameters**

AuthID
Type: string
Authentication Identifier.

**Return Value**

Boolean indicating the status of sending the PDF to the client for signing.

If PushPDF is not successful, the return value is false and the application should query the ErrorCode property to get the error code. The following error codes are applicable.

| Error Code | Description |
|---|---|
| ERRDATACONSTRUCTION | An error occurred while preparing the data to be sent to the Client. |
| ERRSTARTCONNECTION | A connection with the client could not be established. |
| ERRMSGNULL | No response received from the client. |
| ERRAUTHFAILED | The Authentication ID provided is incorrect. |
| CLIENTBUSY | The client is busy. |
| ANOTHERTRANSRUNNING | Another transaction is running on the client. |
| PUSHPDFFAIL | Push PDF transaction failed because of an exception. Check the logs for details. |
| CLIENTNOTSUPPORTED | The client will not support the Push PDF transaction. |
| CLIENTNOTAVAILABLE | The client is not available. |

#### *7.3.2.6 SetupHomeScreen*

**Method Description**

Sets up the home screen of the specified pDoc Pro Client with the information provided.

**Syntax**

```
public bool SetupHomeScreen(ClientHomeScreen clientScreen, string AuthID)
```

**Parameters**

clientScreen
Type: ClientHomeScreen
Client Home Screen information

AuthID
Type: string
Authentication Identifier.

**Return Value**

Boolean indicating whether the home screen setup was successful or not.

If SetupHomeScreen is not successful, the return value is false and the application should query the ErrorCode property to get the error code. The following error codes are applicable.

| Error Code | Description |
|---|---|
| ERRDATACONSTRUCTION | An error occurred while preparing the data to be sent to the Client. |
| ERRSTARTCONNECTION | A connection with the client could not be established. |
| ERRMSGNULL | No response received from the client. |
| ERRAUTHFAILED | The Authentication ID provided is incorrect. |
| CLIENTBUSY | The client is busy. |
| ANOTHERTRANSRUNNING | Another transaction is running on the client. |
| NOTSUPPORTED | Setup home screen transaction is not supported by the selected client. |
| SETUPHOMESCREENFAIL | Setup home screen transaction failed because of an exception. Check the logs for details. |
| ANOTHERTRANSRUNNING | Another transaction is running on the client. |
| CLIENTNOTAVAILABLE | The client is not available. |

### *7.3.2.7 TerminateHomeScreen*

**Method Description**

Terminates the home screen of the client, regardless of its current state, and returns the client to normal usage.

**Syntax**

```
public bool TerminateHomeScreen(string AuthID)
```

**Parameters**

AuthID
Type: string
Authentication Identifier.

**Return Value**

Boolean indicating whether the home screen was terminated successfully or not. Returns true if the home screen is not currently running on the client.

If TerminateHomeScreen is not successful, the return value is false and the application should query the ErrorCode property to get the error code. The following error codes are applicable.

| Error Code | Description |
|---|---|
| ERRDATACONSTRUCTION | An error occurred while preparing the data to be sent to the Client. |
| ERRSTARTCONNECTION | A connection with the client could not be established. |
| ERRMSGNULL | No response received from the client. |
| ERRAUTHFAILED | The Authentication ID provided is incorrect. |
| CLIENTBUSY | The client is busy. |
| ANOTHERTRANSRUNNING | Another transaction is running on the client. |
| NOTSUPPORTED | Terminate home screen transaction is not supported by the selected client. |
| TERMINATEHOMESCREENFAIL | Terminate Home Screen transaction failed because of an exception. Check the logs for details. |
| CLIENTNOTAVAILABLE | The client is not available. |
| HOMESCREENNOTRUNNING | Terminate Home Screen transaction failed because the home screen is not running on the client. |

### *7.3.2.8 TerminateSigningSession*

**Method Description**

Terminates the signing session in progress, resets to the home screen, and waits for the next signing session to be initiated.

**Syntax**

```
public bool TerminateSigningSession(string AuthID)
```

**Parameters**

AuthID
Type: string
Authentication Identifier.

**Return Value**

Boolean indicating whether the signing session was terminated successfully or not. Returns true if the signing session is not currently active on the client.

If TerminateSigningSession is not successful, the return value is false and the application should query the ErrorCode property to get the error code. The following error codes are applicable.

| Error Code | Description |
|---|---|
| ERRDATACONSTRUCTION | An error occurred while preparing the data to be sent to the Client. |
| ERRSTARTCONNECTION | A connection with the client could not be established. |
| ERRMSGNULL | No response received from the client. |
| ERRAUTHFAILED | The Authentication ID provided is incorrect. |
| CLIENTBUSY | The client is busy. |
| ANOTHERTRANSRUNNING | Another transaction is running on the client. |
| TERMINATESIGNINGSESSIONFAIL | Terminate Signing Session transaction failed because of an exception. Check the logs for details. |
| CLIENTNOTAVAILABLE | The client is not available. |
| SIGNSESSIONNOTRUNNING | Terminate Signing Session transaction failed because the signing session is not running on the client. |

### 7.3.3 Events

#### 7.3.3.1 docReceived

**Event Description**

This event is raised when the document is received back from the pDoc Pro Client after completion of the signing or annotating process in the client. The DocEventArgs will contain the PDF File Name, Parameter File Name, etc.

When the document is received back from the pDoc Pro Client after completion of an annotating process in the client, an invisible signature is added to the document for each annotated session.

**Syntax**

```
void SigningClient_docReceived(object sender,
Topaz.pDocProServerSDK.SigningClient.DocReceivedArgs DocArgs)
```

**Parameters**

DocReceivedArgs contain the following information.

clientIdentifier
portNumber
sentPDFFileName,
sentTime,
receivedPDFFilePath
receivedParameterFilePath
receivedTime
docSignedStatus
receivedMessage

clientIdentifier – The client identifier from which the document was received

portNumber – Port Number from which the document was received

sentPDFFileName – The name of the PDF that was sent to the client

sentTime – The time at which the PDF was sent to the client

receivedPDFFilePath – The path of the processed PDF that was received and saved by the SDK. The PDF is received back from the client only when the signer clicks the “DONE” button in the client either after signing or without signing. In all other cases like Cancelled,

Terminated, Error Occurred, License Expired or Timed Out, the document is not returned back and this parameter would be empty.

receivedParameterFilePath– The path of the updated parameter file that was received and saved by the SDK. The updated parameter file is received back from the client only when the signer clicks the “DONE” button in the client either after signing or without signing. In all other cases like Cancelled, Terminated, Error Occurred, License Expired or Timed Out, the updated parameter file is not returned back and this parameter would be empty

receivedTime – The time at which the response was received from the client.

docSignedStatus – This is a Boolean value that indicates if the document was signed or annotated during the session or the document was returned back without any signatures or annotation in that session. A value of “1” indicates that the document was signed or annotated in the session, and a value of “0” indicates that the document was not signed or annotated in the session.

receivedMessage – Contains the return message from the client. The value can be one of the following

- SIGNSUCCESS
- SIGNCANCELLED
- SIGNERROR
- PDOCEXPIRED
- SIGNINGTERMINATED
- SIGNINGTIMEDOUT
- COMMUNICATIONERROR
- RECEIPTRECEIVEDSUCCESS
- RECEIPTRECEIVEDFAILED

### *7.3.3.2 clientDiscovered*

**Event Description**

This event is raised after the DiscoverClients() method is called to discover the available pDoc Pro Clients on the network. As and when the clients respond to the broadcast, the information of the client which has responded is passed to the host application via this event.

**Syntax**

```
void SigningClient_clientDiscovered(object sender, Topaz.pDocProServerSDK.ClientDiscoveredArgs args)
```

**Parameters**

ClientDiscoveredArgs contain the following information.

clientIdentifier
portNumber
signingSessionTimeOut,
checkForSignaturePad

clientIdentifier – The client identifier for the discovered client

portNumber – Port Number for the discovered client

signingSessionTimeOut – The signing session timeout for the discovered client

checkForSignaturePad – Whether the client checks for the availability of the external signature pad for a signing session

# 8.0 - Parameter File

When the system operator pushes a PDF document to a client machine, the server software also sends a parameter file to the client along with the PDF document. The parameter file controls the pDoc Pro Client behavior on the client.

The parameter file contains the information configured by the Administrator using the Administrator Demo module. The default parameter file, named DefaultParam.xml, is installed by the pDoc Pro SDK Server installation into the Common Application Data folder of the server machine. Typically, this folder is “C:\ProgramData\pDoc Pro SDK Server”. Custom applications can manipulate this file directly for writing any other information required to control the pDoc Pro Client behavior. When the parameter file is pushed to the client, it should have the same name as the PDF that is currently being pushed. Section 8.1 below explains the schema of the parameter file along with a sample file.

When the signed PDF is received by the system operator, along with the PDF document, another variant of the parameter file is also received that contains the information about the signing or annotating session and also the sequence of events that happened during the signing or annotating session. Section 8.2 explains the schema of this file along with a sample file.

## 8.1 - Parameter File Schema (From Server to Client)

```
<?xml version="1.0" encoding="utf-16"?>
<!--Root element for the Parameter File-->
<pDocProServer SchemaVersion="1.0">
  <!—Name of the PDF file-->
  <PDFName />
  <!—Size of the PDF file-->
  <PDFSize />
  <!—Specified timeout period for Signing or Annotating-->
  <SigningTimeOut />
  <!—Specified whether extended monitor is allowed or not-->
  < EnableTabletDisplay/>
  <!— Specifies if the signature field list window is enabled or not (TRUE or FALSE)-->
  < EnableSignFieldList/>
```

```
<!— Specifies if the attachment list window is enabled or not (TRUE or FALSE)-->
< EnableAttachmentList/>
<!— Text to be displayed on the pDoc Signing Screen user interface before clicking on a signature field -->
< DisplayTextBeforeSigning/>
<!— Text to be displayed on the pDoc Annotating Screen user interface before pencil annotation -->
< DisplayTextBeforeAnnotating/>
<!--Text to be displayed on the pDoc Signing Screen user interface after clicking on a signature field-->
<DisplayTextSign></DisplayTextSign>
<!-- Specifies if on screen keyboard is enabled or not (1 or 0)-->
<OnScreenKeyboard/>
<!--Signature information like First Name, Last Name and Email can be added here if known-->
<SignatureInformation>
  <FirstName />
  <LastName />
  <Email />
</SignatureInformation>
<!-- Signature Capture Related information-->
<SignatureCaptureData>
  <!—Specified whether signing with external signature capture devices is allowed or not -->
  <ExternalDevice />
  <!--Signature Position with respect to details (i.e., behind, beside, and Image Only) -->
  <SignPosition />
  <!--Specified if Maximum Enlargement Feature needs to be enabled (TRUE or FALSE)-->
  <EnableMEF>
    <!--Maximum Enlargement Factor percentage-->
    <MEFPercent />
  </EnableMEF>
  <!--Inking color for the signature-->
  <InkColor />
```

```
    <!--Inking thickness for the signature-->
    <InkThickness/>
<!-- Specifies the option for capturing the signer photo during the signing process (0 or 1 or 2)-->
  <SignerPhoto/>
<!-- Specifies the option for checking the signature spots available (0 or 1)-->
  <CheckSignatureSpots/>
<!-- Specifies the option for automatically confirm signer (0 or 1)-->
  <AutomaticallyConfirmSigner/>
<!-- Specifies the automatically confirm wait period (default value is 1)-->
<AutomaticallyConfirmWaitPeriod/>  </SignatureCaptureData>
  <!—Attributes for the Text Comments feature. Specifies whether text comments feature is enabled or
not and if enabled, different attributes for font-->
  <TextComments Enabled="">
    <FSize />
    <FStyle />
    <FEffect />
    <FColor />
    <FName />
  </TextComments>
  <!—Attributes for the Pencil markup feature. Specifies whether pencil markup feature is enabled or
not. If enabled, the line color and line width for the pencil markups.-->
  <PencilMarkups Enabled="">
    <LineColor />
    <LineWidth />
  </PencilMarkups>
  <!--"DONE" button options-->
  <DoneButton>
    <!--Display text for the DONE button-->
    <DisplayText></DisplayText>
    <!-- Specifies if confirmation message should be displayed or not (1 or 0)-->
    <ShowConfirmationMsg/>
```

```
    <!—Message that needs to be displayed when the DONE button is clicked-->
    <MessageText>MessageText>
    <!--Text for the "Cancel" button in the message-->
    <CancelButTextMsg> </CancelButTextMsg>
    <!--Text for the "Continue" button in the message-->
    <ContinueButTextMsg> </ContinueButTextMsg>
  </DoneButton>
  <!--"Cancel" button options-->
  <CancelButton>
    <!--Display text for the Cancel button-->
    <DisplayText></DisplayText>
    <!—Message that needs to be displayed when the Cancel button is clicked-->
    <MessageText></MessageText>
    <!--Text for the "Yes" button in the message-->
    <YesButTextMsg></YesButTextMsg>
    <!--Text for the "No" button in the message-->
    <NoButTextMsg></NoButTextMsg>
  </CancelButton>
</pDocProServer>
```

The following is a sample parameter file. The name of the PDF that is pushed in this example is pDoc Pro Sample Document.pdf and hence the parameter file name should be pDoc Pro Sample Document.xml.

```
<?xml version="1.0" encoding="utf-16"?>
<pDocProServer SchemaVersion="1.0">
  <PDFName>pDoc Pro Sample Document.pdf</PDFName>
  <PDFSize>651047</PDFSize>
  <SigningTimeOut> 10 </SigningTimeOut>
  <EnableTabletDisplay> True </EnableTabletDisplay>
  <EnableSignFieldList>False</EnableSignFieldList>
  <EnableAttachmentList>False</EnableAttachmentList>
```

```
  <DisplayTextBeforeSigning>To sign, first touch a blue signature spot.</DisplayTextBeforeSigning>
  <DisplayTextBeforeAnnotating>Use the attached pen or your finger to annotate the document
.</DisplayTextBeforeAnnotating>
  <DisplayTextSign>Sign your name in the signing window.</DisplayTextSign>
  <OnScreenKeyboard>1</OnScreenKeyboard>
  <SignatureInformation>
    <FirstName>
    </FirstName>
    <LastName>
    </LastName>
    <Email>
    </Email>
  </SignatureInformation>
  <SignatureCaptureData>
    <ExternalDevice>True</ExternalDevice>
    <SignPosition>Automatic</SignPosition>
    <EnableMEF>
      <MEFPercent>25</MEFPercent>
    </EnableMEF>
    <InkColor>0000ff</InkColor>
<InkThickness>2</InkThickness>
<SignerPhoto>0</SignerPhoto>
<CheckSignatureSpots>0</CheckSignatureSpots>
<AutomaticallyConfirmSigner>0</AutomaticallyConfirmSigner>
<AutomaticallyConfirmWaitPeriod>1</AutomaticallyConfirmWaitPeriod>
  </SignatureCaptureData>
  <TextComments Enabled="false">
    <FSize>26</FSize>
    <FStyle>Regular</FStyle>
    <FEffect>False</FEffect>
```

```
    <FColor>#000000</FColor>
    <FName>Times New Roman</FName>
  </TextComments>
  <PencilMarkups Enabled="false">
    <LineColor>#000000</LineColor>
    <LineWidth>2</LineWidth>
  </PencilMarkups>
  <DoneButton>
    <DisplayText>DONE</DisplayText>
   <ShowConfirmationMsg>1</ShowConfirmationMsg>
    <MessageText>Do you agree that the information you have provided is complete and correct, and that you want to conduct business electronically?</MessageText>
    <CancelButTextMsg>I do not agree</CancelButTextMsg>
    <ContinueButTextMsg>I agree</ContinueButTextMsg>
  </DoneButton>
  <CancelButton>
    <DisplayText>CANCEL</DisplayText>
    <MessageText>This action will delete any changes you made to the document. Do you want to continue this cancel process?</MessageText>
    <YesButTextMsg>Yes</YesButTextMsg>
    <NoButTextMsg>No</NoButTextMsg>
  </CancelButton>
</pDocProServer>
```

### 8.2 - Parameter File Schema (From Client to Server)

```
<?xml version="1.0" encoding="utf-16"?>
<!--Root element for the Parameter File-->
<pDocProServer SchemaVersion="1.0">
<!—Name of the PDF file-->
  <PDFName />
  <!—Size of the PDF file-->
  <PDFSize />
<!—Pages visited during the signing or annotating session -->
  <PagesVisited />
<!—Whether the document was signed or annotated during this session or not. 0 – Not Signed or Not Annotated and 1 – Signed or Annotated-->
  <DocumentSigned/>
<!—Sequence of events during the signing session -->
  <Audit>
    <Record1/>
    <Record2/>
  </Audit>
</pDocProServer>
```

The following is a sample parameter file. The name of the PDF that is pushed in this example is Introducing pDoc Signer.pdf and hence the parameter file name should be Introducing pDoc Signer-D20170227T102415.xml.

```
<?xml version="1.0" encoding="utf-16"?>
<pDocProServer SchemaVersion="1.0">
  <PDFName>Introducing pDoc Signer.pdf</PDFName>
  <PDFSize>601394</PDFSize>
  <PagesVisited>1,2,3,4,5,6,7,8,9</PagesVisited>
```

```
<DocumentSigned>1</DocumentSigned>
<Audit>
<Record1>2~2~10/26/15 02:05:09 PM</Record1>
<Record2>1~2~1~1~2</Record2>
<Record3>1~3~1~2~3</Record3>
<Record4>1~4~1~3~4</Record4>
<Record5>1~4~10~100~75</Record5>
<Record6>1~4~9~75~100</Record6>
<Record7>1~5~1~4~5</Record7>
<Record8>1~6~1~5~6</Record8>
<Record9>1~6~2~Favorite Color~ ~Green</Record9>
<Record10>1~6~3~Eastern~Eastern~Pacific</Record10>
<Record11>1~7~1~6~7</Record11>
<Record12>1~7~6~Sales~Off~ales</Record12>
<Record13>1~6~1~7~6</Record13>
<Record14>1~6~2~Favorite Color~Green~Red</Record14>
<Record15>1~7~1~6~7</Record15>
<Record16>1~7~5~States~Alabama~California</Record16>
<Record17>1~7~5~Multiple Selection List Box~High School~High
School,Bachelor's,Doctorate</Record17>
<Record18>1~8~1~7~8</Record18>
<Record19>1~8~4~Loan~Auto~Home Improvement</Record19>
<Record20>1~9~1~8~9</Record20>
<Record21>1~9~7~sample Annotation~sample Annotation Text</Record21>
<Record22>1~9~6~Maternal Grandmother~Off~MGrandma</Record22>
<Record23>1~9~6~Maternal Grandpa~Off~MGrandpa</Record23>
<Record24>1~9~5~List Box~Internet Search~Internet Search,Newspaper Ad</Record24>
<Record25>1~9~8~Patient~10/26/15 02:06:50 PM~ </Record25>
<Record26>2~1~10/26/15 02:06:52 PM</Record26>
</Audit>
```

</pDocProServer>

The following events are recorded during the signing session.

Page Level Events:

1. Page Navigation
2. Text Box Change
3. Radio Button Selection Change
4. Combo Selection Change
5. List Selection Change
6. Check Box Selection Change
7. Text Annotation Change
8. Signature Field Signed
9. Zoom In
10. Zoom Out

Document Level Events

1. Done
2. Open

Below is the format of the event under the <Audit> tag for each of the above events.

1. Page Navigation

1 ~ Page Number ~ 1 ~ Current Page ~ New Page

2. Text Box Change

1 ~ Page Number ~ 2 ~Field Name~ Previous Value ~ New Value

3. Radio Button Selection Change

1 ~ Page Number ~ 3 ~Field Name~ Previous Export Value ~ New Export Value

4. Combo Selection Change

1 ~ Page Number ~ 4 ~Field Name~ Previous Value ~ New Value

5. List Selection Change

1 ~ Page Number ~ 5 ~Field Name~ Previous Value(s) with comma as seperator ~ New Value(s) with comma as seperator

6. Check Box Selection Change

1 ~ Page Number ~ 6 ~Field Name~ Previous Export Value(s) ~ New Export Value(s)

7. Text Annotation Change

1 ~ Page Number ~ 7 ~Field Name~ Previous Value ~ New Value

8. Signature Field Signed

1 ~ Page Number ~ 8 ~ Signature Field Name ~ Date Time ~ Signer Name

9. Zoom In

1 ~ Page Number ~ 9 ~ Current Zoom Percentage ~ New Zoom percentage

10. Zoom Out

1 ~ Page Number ~ 10 ~ Current Zoom Percentage ~ New Zoom percentage

Document Level Events

1. Done

2 ~ 1 ~ Date Time at which the “DONE” button was clicked

2.Open

2~2~Date Time at which the Document was opened in the pDoc Pro Client