

## FIRMA CERTA SDK (revisione 1.5.4.1)

### Rilevazione del software di firma

Leggere il valore PathApp della seguente chiave che contiene solo il path dell' eseguibile non il nome che è fisso ed è uguale a fcsign.exe :

```
[HKEY_CURRENT_USER\Software\Nimiral\Firmacerta\fcsign.exe]
```

```
"PathApp"=""
```

se il programma è in esecuzione da USB

```
[HKEY_CURRENT_USER\Software\Nimiral\Firmacerta\fcsign.exe]
```

```
"InUSB"=1
```

in questo caso il path dell'eseguibile dovrà essere letto da

```
[HKEY_CURRENT_USER\Software\Nimiral\Firmacerta\fcsign.exe]
```

```
"PathAppUSB"=""
```

### Lancio con parametri

```
fcsign.exe <file> <action> <mode> <params>
```

N.B. : Rispettare l'ordine dei parametri così come indicato.

**file** : path completo del file su cui operare , il path può essere anche un URL e in questo caso viene prima effettuato il download del file; se si utilizza la versione ActiveX dell'sdk definendo questo parametro uguale a MEMORY si ottiene la lettura del file caricato in memoria attraverso il metodo di interfaccia

*Ifcxcore.Memory.LoadDocumentFromBuffer* in questo caso al termine del processo il file di output si otterrà tramite il metodo *Ifcxcore.Memory.SaveSignedDocumentToBuffer*

**action** : 0=Firma , 1=Firma e marca , 2=Controfirma , 3=Marca , 4=Verifica , 5=Visualizza, 8=Opzioni, 9=Mostra Certificati, 10=Verifica Dispositivo

**mode** : vale m se firma massiva altrimenti non deve essere indicato; nel caso di firma massiva il parametro <file> deve essere un file di testo contenente la lista dei file da processare uno per riga

**params**: parametri nella forma "chiave1=valore1;chiave2=valore2" (separati dal punto e virgola ;) e racchiusi da doppie virgole ("")

### Esecuzione tramite ActiveX

E' possibile eseguire le stesse operazione tramite l'utilizzo dell'ActiveX **fcx.dll** che opportunamente installato e registrato consente di effettuare le stesse operazione previste per il lancio con parametri attraverso il metodo di interfaccia:

```
function Execute(const AFile: WideString; AAction: fcx_action; AMode: fcx_mode; const AParams: WideString; out AMessage: WideString): WordBool;
```

Per ulteriori dettagli sull'utilizzo tramite ActiveX vedi fcx.chm ed esempi di integrazione.

I parametri possibili sono:

#### Parametri

- **TargetDir**=<cartella di destinazione dei file processati>
- **LogFileName**=<file di log del processo che verrà eseguito in modalità silent (senza mostrare finestre non necessarie)>
- **ShowErrorMessage**=<[0,1] : ha effetto solo se in modalità silent e forza la visualizzazione dei messaggi di errore>

- **ShowQuestionMessage**=<[0,1] : ha effetto solo se in modalità silent e forza la visualizzazione dei messaggi di richiesta>
- **OverwriteLog**=<[0,1] : se = 1 riscrive il file di log per ogni processo altrimenti accoda (default 0)>
- **Pin**=<codice pin per l'accesso alla smart card>
- **ReaderName**=< in alternativa al ReaderIndex indica il nome del lettore contenete la smartcard: necessario solo se sono presenti più lettori>
- **CertificateIndex**=<indice (da 1) del certificato da utilizzare: necessario solo se sono presenti più certificati >
- **CertificateName**=<indica il Common Name del soggetto intestatario del certificato da utilizzare; è possibile anche indicare una espressione regolare per filtrare più certificati aggiungendo il suffisso "@RE:" prima del valore>
- **UrlTSA**=<url del server per le marche temporali> se non impostato viene preso quello inserito in configurazione
- **UsernameTSA**=<utente per le marche temporali>
- **PasswordTSA**=<password per le marche temporali>
- **OverwriteFile**=<[0,1] : se = 1 sovrascrive il file processato se presente altrimenti aggiunge un underscore (\_) nel nome (default 0)>
- **PDFPassword**=<Password di protezione del pdf> obbligatorio solo se il documento pdf è protetto da passord
- **TimestampFormat**=<CADEST,PADEST,XADEST,PDF,TSR,TST : formato con cui viene generata la marca temporale in associazione al file (default CADEST)>
- **UploadUrl**=<url dove effettuare l'upload del file processato>
- **UploadMethod**=<[POST,PUT] : modalità di invio del file (default POST)>
- **UploadFileName**=<specifica il file di destinazione per l'upload in modalità POST>
- **UploadPostFieldName**=<specifica il nome del parametro in POST relativo al file (default firmacertafile)>
- **UploadSoapRequest**=<serve nel caso di invio del documento firmato ad un webservice e indica l'XML da utilizzare per generare la request SOAP che verrà inviata all'uploadUrl o in alternativa indica l'url da cui reperire l'XML. Per evitare l'interferenza di caratteri speciali è consigliabile formattare il valore con la funzione Escape(). All'interno dell'XML è possibile utilizzare la parola riservata @FCSIGN\_OUTPUT@ che verrà sostituita con il contenuto del file di output codificato in base64>
- **UploadSoapAction**=<deve essere valorizzata se si utilizza uploadSoapRequest e indica il valore di header http SOAPAction da aggiungere nella request da inviare al webservice. Per evitare l'interferenza di caratteri speciali è consigliabile formattare il valore con la funzione Escape()>
- **UploadSoapContentType**=<indica il Content-Type della request da inviare al webservice (default "text/xml; charset="utf-8"). Per evitare l'interferenza di caratteri speciali è consigliabile formattare il valore con la funzione Escape()>
- **UploadZipped**=<[0,1] : se = 1 zippa il file prima dell'upload (default 0)>
- **DownloadFileName**=<specifica il file sorgente nel caso in cui il parametro file sia un url, il nome viene utilizzato per capire di che tipo di file si tratta attraverso la lettura dell'estensione>
- **WidgetPDFSignaturePosition**=<page|x|y|width|height|ppi : (sequenza valori come indicata separati da pipe; x,y,width,height espressi in pixel e ppi sono i "pixel per inch" se non indicati viene preso il valore del sistema operativo ) se presente indica dove posizionare il campo firma nel pdf. Nel caso di logo pdf viene controllato il valore page nel caso di firma grafometrica vengono controllati width ed height se uno dei valori controllati è zero viene presentata la preview del pdf in oggetto per la selezione della area. Per indicare l'ultima pagine del pdf impostare page a 999999>
- **ContentManagementService**=<Nome servizio di archiviazione documentale da utilizzare. Indicare "NESSUNO" o "NO" per non effettuare l'archiviazione>
- **SigningTime**=<data di firma nel formato dd-mm-yyyy-hh-nn-ss : viene utilizzata come data di firma se non presente la marcatura temporale>
- **UsernameRS**=<utente/dispositivo virtuale per l'accesso al servizio di firma remota>
- **PasswordRS**=<password per l'accesso al servizio di firma remota>
- **OtpRS**=<OTP per l'accesso al servizio di firma remota>
- **DocInfoNotCompile**=<[0,1] : se = 1 non richiede la compilazione dell'informativa (defaults : exe=0, sdk=1>
- **DocInfoTargetFileName**=<path completo di dove salvare l'informativa>
- **DocInfoParams**=<parametri di compilazione informativa:  
CODICE\_FISCALE=AAAAAA99A99A999A|COGNOME\_NOME=MARIO  
ROSSI|COMUNE\_NASCITA=ROMA|DATA\_NASCITA=31121999>

- **PdfSignInfoReason**=<Motivo di firma : gestibile solo per firme in formato PDF>
- **PdfSignInfoLocation**=<Luogo di firma : gestibile solo per firme in formato PDF>
- **PdfSignInfoContactInfo**=<Informazioni aggiuntive : gestibile solo per firme in formato PDF>
- **WidgetPDFShowTimestamp**=<[0,1] : se = 1 mostra la data e ora sul logo della firma PDF (default 1)>
- **WidgetPDFTimestampFormat**=<formato con cui esporre la data. Es. dd/mm/yyyy=mostra solo la data senza ora>
- **WidgetPDFAutoText**=<[0,1] : se = 1 genera un testo automatico sul logo della firma PDF (default 1)>
- **WidgetPDFHeader**=<se WidgetPDFAutoText = 0 imposta la testata sul logo della firma PDF>
- **WidgetPDFTitle1**=<se WidgetPDFAutoText = 0 imposta il titolo 1 sul logo della firma PDF>
- **WidgetPDFText1**=<se WidgetPDFAutoText = 0 imposta il testo 1 sul logo della firma PDF, questo valore può essere indicato al massimo su 4 righe e ogni riga deve terminare col carattere "|" (pipe); e' inoltre possibile indicare alcune informazioni aggiuntive utilizzando le variabili @SIGNERNAME (Nome dell'intestatario del certificato di firma), @ISSUENAME (Nome dell'emittitore del certificato), @SERIALNUMBER (Numero di serie del certificato), @SIGNINGTIME (data di firma) e @SIGNINGDATETIME (data e ora di firma), @SIGNERID (Codice fiscale dell'intestatario del certificato di firma), @SIGNERCOUNTRY (Paese dell'intestatario del certificato di firma), @SIGNEREMAIL (Email dell'intestatario del certificato di firma)>
- **WidgetPDFTitle2**=<se WidgetPDFAutoText = 0 imposta il titolo 2 sul logo della firma PDF>
- **WidgetPDFText2**=<se WidgetPDFAutoText = 0 imposta il testo 2 sul logo della firma PDF>
- **WidgetPDFAutoFontSize**=<[0,1] : se = 1 viene calcolata in automatico la dimensione del font da utilizzare in base alla dimensione del logo di firma (default 1)>
- **WidgetPDFHeaderFontSize**=<[numero intero positivo] indica la dimensione del font da utilizzare per la testata se WidgetPDFAutoFontSize=0>
- **WidgetPDFTimestampFontSize**=<[numero intero positivo] indica la dimensione del font da utilizzare per la data e ora se WidgetPDFAutoFontSize=0>
- **WidgetPDFTitleFontSize**=<[numero intero positivo] indica la dimensione del font da utilizzare per il titolo se WidgetPDFAutoFontSize=0>
- **WidgetPDFTextFontSize**=<[numero intero positivo] indica la dimensione del font da utilizzare per il testo se WidgetPDFAutoFontSize=0>
- **NotCheckUpgrade**=<[0,1] : se = 1 non verifica la presenza di aggiornamenti all'avvio (default 0)>
- **PDFSignatureType**=<[DOCUMENT,MDP] : se = MDP la prima firma viene generata di tipo Modification Detection & Prevention come previsto dallo standard PDF (default DOCUMENT)>
- **PDFSignatureAllowedChanges**=<[FillInForm|Comment] se PDFSignatureType=MDP è possibile definire che cosa l'utente può modificare dopo l'apposizione di tale firma; FillInForm: l'utente può compilare i campi vuoti presenti nel documento al momento della firma; Comment: l'utente può aggiungere commenti al documento >
- **OpenPdfAfterSign**=<[0,1] : se = 1 al termine del processo di firma apre il pdf con il lettore Pdf predefinito>
- **PDFEmptySignatureFieldName**=<indicare il nome di uno o più campi firma vuoti da firmare presenti nel pdf , più nomi devono essere separati dal carattere "|" (pipe); per firmare tutti gli eventuali campi presenti valorizzare il parametro con il carattere "\*" (asterisco); Esempio 1: PDFEmptySignatureFieldName=\* ; Esempio 2: PDFEmptySignatureFieldName=campofirma1|campofirma2; per filtrare i campi tramite una espressione regolare aggiungere il suffisso "@RE:" prima dell'espressione : Esempio 3: PDFEmptySignatureFieldName=@RE:campo.\* (filtra tutti i campi il cui nome inizia con "campo")>
- **NoPDFEmptySignatureFieldTooltipInDescription**=<[0,1] : se = 1 in fase di firma non vengono importati i tooltip eventualmente impostati nei campi firma vuoti presenti nel pdf e gestiti come descrizione del campo firma nel template (default 0)>
- **SignatureIndex**=<indice (da 1) della firma su cui applicare la controfirma o la marca temporale in caso di file già firmati; se non indicato verrà richiesto>
- **TargetFileName**=<nome file di destinazione; se non viene indicato comprensivo di percorso viene comunque usato il nome del file per definire l'output>
- **ShowPDFSignatureOnAllPages**=<[0,1] : se = 1 il campo firma viene mostrato su tutte le pagine del pdf>
- **SignatureTemplateId**=<come *BiometricSignatureTemplateId* ma specifico per la firma digitale>

## Parametri relativi alla firma grafometrica

- **BiometricData**=<[0,1] : se = 1 si predispone per la firma grafometrica>
- **BiometricDataBackgroundBMP**=<percorso del file di sfondo per il tablet: se non specificato il viene copiato lo sfondo pdf relativo all'area di firma indicata>
- **BiometricSignatureDocumentSchema**=<sequenza di caratteri che descrive come applicare il modello di firma al documento (es: TEMPLATE\_1=FIRMA\_1|||FIRMA\_2:FIRMA\_3|||FIRMA\_1| in questo caso viene specificato di caricare il file TEMPLATE\_1.fct, che deve essere presente nell'apposita cartella configurabile dalle opzioni, e di applicare FIRMA\_1 a pagina 1, FIRMA\_2 e FIRMA\_3 a pagina 4 e di nuovo FIRMA\_1 a pagina 7); il valore BiometricSignatureDocumentSchema può essere inserito in alternativa all'interno del pdf da firmare in fondo al file dopo %%EOF, in questo caso si dovrà anche cambiare l'estensione del file da .pdf a .fcd> E possibile inserire eventuali parametri aggiuntivi separati dal carattere & sempre in formato chiave=valore (es: TEMPLATE\_1=FIRMA\_1|||FIRMA\_2:FIRMA\_3|||FIRMA\_1|\$CAMPO1=VALORE1&CAMPO2=VALORE2). Al posto della sequenza di caratteri è possibile indicare il percorso completo di un file nel quale inserire la sequenza. I parametri aggiuntivi nel caso di file possono essere inseriti uno per riga a partire dalla seconda senza indicare il separatore & ; se si utilizza la versione ActiveX dell'sdk definendo questo parametro uguale a MEMORY si ottiene la lettura dello schema dalla proprietà di interfaccia *Ifcxcore.Memory.DocumentSchema*>
- **BiometricSignatureTemplateId**=<nome del file (no percorso e no estensione) relativo al modello di firma da applicare al documento che verrà ricercato nella cartella apposita definita in configurazione, in alternativa può essere indicato il path completo del file di template. In presenza di BiometricSignatureDocumentSchema, se indicato il path completo, viene considerato come template relativo allo schema indipendentemente dall'id indicato nello schema stesso; su si utilizza la versione ActiveX dell'sdk definendo questo parametro uguale a MEMORY si ottiene la lettura del template dalla proprietà di interfaccia *Ifcxcore.Memory.Template*>
- **BiometricSignatureRequired**=<[0,1] se = 1 imposta i campi firma come obbligatori; ha effetto solo se non viene indicato un template (default 0)>
- **BiometricSignatureNoBiometricData**=<[0,1] se = 1 non include i dati biometrici per i campi firma creati; ha effetto solo se non viene indicato un template (default 0)>
- **BiometricSignatureTemplateFieldId**=<indica il FieldName da utilizzare come nome campo firma nel caso di firma grafometrica effettuata senza l'utilizzo di un template (vedi SignatureFieldNameFromTemplate)>
- **SignatureFieldNameFromTemplate**=<[0,1] se = 1 imposta il nome del campo firma nel pdf utilizzando il nome del campo relativo all'interno del template (default 0) o indicato col parametro BiometricSignatureTemplateFieldId>
- **BiometricSignatureNotUseTemplateBackground**=<[0,1] se = 1 non vengono usate le immagini di sfondo delle firme salvate nel template ma vengono generate al momento dal pdf ; obbligatorio solo se presenti viene ignorato se presente BiometricSignatureDocumentSchema o BiometricSignatureTemplateId>
- **BiometricSignatureTimestampNotAllowUserCancel**=<[0,1] se = 1 non viene consentito all'utente di non apporre la marca temporale se richiesta>
- **ExternalDeviceUILanguage**=<[IT,EN,DE,FR,ES,PT] specifica la lingua da utilizzare per l'interfaccia utente proiettata sul dispositivo esterno utilizzato per la firma (default IT)>.
- **CloneFileWithoutBiometricData**=<[0,1] se = 1 viene creata una copia del file pdf con la sola parte grafica dei campi firma e senza i dati biometrici (default 0)>
- **CloneFilename**=<nome del file di copia senza i dati biometrici; se viene indicato solo il nome il file verrà generato nella stessa directory del file originale; se viene indicato anche il percorso verrà generato nel percorso indicato; se viene lasciato vuoto verrà generato un file con suffisso "\_NoBiometricData" nella stessa cartella del file originale>
- **CreateSpecimenImagesFile**=<[0,1] se = 1 viene generato un file zip nella stessa directory del documento pdf e con suffisso "\_SpecimenImages" contenente le immagini bitmap delle firme effettuate nel documento (default 0) ; su si utilizza la versione ActiveX dell'sdk definendo questo parametro uguale a MEMORY si possono leggere le immagini tramite la proprietà di interfaccia *Ifcxcore.Memory.Specimens*>
- **BiometricSignatureLight**=<[0,1] se = 1 viene eseguita la firma in modalità "leggera" senza l'inserimento dei dati biometrici e pertanto senza alcuna validità legale (default 0)>
- **BiometricSignatureLoopLength**=<[numero intero >= -1] questo parametro viene considerato solo in caso di firma con posizionamento libero da parte dell'utente e quindi in assenza di template;

indica il numero di firme da apporre nel documento; se = -1 dopo ogni firma viene chiesto all'utente se continuare a firmare (default 0)>

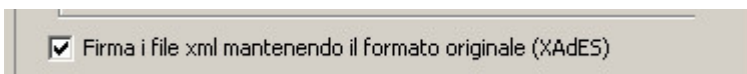
- **SpecimenImageOptions**=<[GRID|TIME|AIR] opzioni relative alla generazione dell'immagine della firma; GRID= mostra carta millimetrata, TIME= mostra durata della firma, AIR= mostra i tratti in aria>
- **SpecimenImageZoomPercentage**=<[numero intero compreso tra 30 e 300] indica la percentuale di zoom dell'immagine della firma>
- **SpecimenImageFormat**=<[BMP,JPG,PNG,GIF,TIFF] definisce il formato dell'immagine della firma (default BMP)>
- **BiometricSignaturePdfViewerLayoutOptions**=<[HIDELIST|HIDETOOLBAR|STARTFROMFIRSTPAGE|SPLITTERBIG|INITIALZOOMFITPAGE] opzioni relative alla visualizzazione del pdf per la firma; HIDELIST nasconde la lista delle firme a sinistra, HIDETOOLBAR nasconde la barra di navigazione del pdf, SPLITTERBIG aumenta la larghezza del separatore tra l'elenco delle firme e l'area di visualizzazione del pdf >
- **BiometricSignaturePdfViewerLayoutListPosition**=<[LEFT,TOP] se = LEFT la lista delle firme viene mostrata a sinistra, se = TOP la lista viene mostrata in alto (default LEFT)>
- **BiometricSignaturePdfViewerLayoutListSizePerc**=<[numero intero compreso tra 1 a 100] indica la dimensione della lista delle firme in percentuale]>
- **BiometricSignaturePdfPositionLayoutOptions**=<[SHOWMAXIMIZED] opzioni relative alla visualizzazione del pdf per la posizione del campo firma; SHOWMAXIMIZED mostra la finestra a schermo intero>
- **BiometricSignatureTemplateOrdered**=<[POSITION,FIELDID,DESCRIPTION] specifica il criterio con cui i campi del template devono essere ordinati prima di essere processati; POSITION ordina in base alla posizione del campo all'interno del pdf dall'altro verso il basso e da sinistra verso destra; FIELDID ordina in base al FieldId; DESCRIPTION ordina in base alla Descrizione se valorizzata>
- **BiometricSignatureKeySearch**=<testo di ricerca che verrà utilizzato per rilevare le posizioni dei campi firma>

#### Parametri di setting

Tramite i parametri di setting è possibile bypassare i valori relativi alle opzioni memorizzate nel registro in HKEY\_CURRENT\_USER\Software\Namirial\Firmacerta\ (accessibili tramite l'applicativo desktop da menu Strumenti>Opzioni) aggiungendoli nella forma : **Chiave.NomeValore=Valore**

I valori definiti in questo modo non verranno salvati nel registro e avranno effetto solo per il processo di firma corrente.

Per esempio per eseguire una firma di un file xml in formato XAdES senza modificare l'opzione relativa (vedi figura),



che nel registro di sistema è salvata nel valore XmlFormat della chiave Signature (vedi figura),



si dovrà aggiungere il parametro **Signature.XmlFormat=1** agli altri parametri di esecuzione.

N.B. : E' possibile indicare tutti i parametri in un file qualora la lunghezza totale superasse il massimo consentito dal sistema operativo ( 8191 caratteri, 2047 su Microsoft Windows 2000 o Windows NT 4.0. ). I parametri possono essere indicati o tutti nella prima riga sempre separati da ; oppure uno per riga senza carattere di separazione. In questo caso nel parametro <params> va indicato il path completo del file.

Esempi:

Firma di un documento

fcsign.exe "C:\Documents and Settings\Utente\Desktop\Documento.pdf" 0

#### Firma di un documento senza interfaccia grafica

```
fcsign.exe "C:\Documents and Settings\Utente\Desktop\Documento.pdf" 0  
"TargetDir=C:\Cartella\LogFileName=C:\firmacerta.log;Pin=999999"
```

#### Firma di piu documenti

```
fcsign.exe "C:\temp\lista.txt" 0 m  
dove "C:\temp\lista.txt" contiene:  
C:\Documents and Settings\Utente\Desktop\Documento1.pdf  
C:\Documents and Settings\Utente\Desktop\Documento2.pdf
```

#### **Altre funzioni richiamabili tramite lancio con parametro:**

- GetTimeStampInfo : Esegue una chiamata al server Namirial per ottenere informazioni relative alle marche temporali.  
Es: fcsign.exe @NOFILE 999  
"AdvancedCommand=GetTimeStampInfo;TargetFilename=C:\timestampinfo.xml;"
- WindowsCertificateAddCACerts : Aggiunge i certificati di root Namirial come identità affidabili nel Certificate Storage di Windows  
Es: fcsign.exe @NOFILE 999 "AdvancedCommand=WindowsCertificateAddCACerts;"
- InstallManagedCertificate : Installa un certificato medium (\*.fck) per la firma grafometrica  
Es: fcsign.exe @NOFILE 999  
"AdvancedCommand=InstallManagedCertificate;Filename=C:\certificate.fck;Password=12345678;"
- UninstallManagedCertificate : Rimuove un certificato medium (\*.fck) installato per la firma grafometrica  
Es: fcsign.exe @NOFILE 999 "AdvancedCommand=UninstallManagedCertificate;Name=certificate;"
- SmartCardSerial : Legge il numero seriale della smartcard. Il parametro ReaderName non è obbligatorio.  
Es: fcsign.exe @NOFILE 999 "AdvancedCommand=SmartCardSerial;ReaderName=Nome lettore;TargetFilename=C:\serial.txt;"

#### **Integrazione con web browser per Applicazioni Web**

Per poter interagire con firma certa da browser è necessario eseguire lato client l'applicazione **fchttpserver.exe** che è un servizio http locale che comunica su una porta (es: 7777) le risorse javascript necessarie per l'integrazione. La pagina web dovrà contenere nell' <head> il seguente script:

```
<head>  
  <script type="text/javascript" src="http://localhost:7777/files/fcsign.js"></script>  
</head>
```

Se tutto va a buon fine dopo il caricamento della pagina si avrà in memoria l'oggetto globale **fcsign** con i seguenti metodi e proprietà:

- **templateUrl**: String indica l'eventuale template da utilizzare per firmare il documento
- **uploadSoapRequest**: String questa proprietà serve nel caso di invio del documento firmato ad un webservice e indica l'XML da utilizzare per generare la request SOAP da inviare all'uploadUrl o in alternativa indica l'url da cui reperire l'XML. All'interno dell'XML è possibile utilizzare la parola riservata @FCSIGN\_OUTPUT@ che verrà sostituita con il contenuto del file firmato codificato in base64.
- **uploadSoapAction**: String deve essere impostata in coppia con uploadSoapRequest e indica il valore di SOAPAction da aggiungere nell'Header della request al webservice.

- **uploadSoapContentType**: String indicare il Content-Type della request al web service (default 'text/xml; charset="utf-8"')
- **documentSchema**: String indica l'eventuale schema da utilizzare (per il formato vedi *BiometricSignatureDocumentSchema*)
- **uploadPostFieldName**: String indica il nome del campo contenente il file inviato in multipart/form-data (se non indicato verrà impostato con "firmacertafire")
- **uploadMethod**: String indica il metodo di upload; valori possibili [POST,PUT] (default POST)
- **extraParams**: String sono i soliti parametri definiti sopra validi anche per la modalità web
- **callback**: function(data:Object)  
Indica la funzione da chiamare al termine dell'operazione.
- **version**: function()  
Ritorna la versione del file fcx.dll in callback data successMessage  
callback data: success:Boolean, errorMessage:String, errorCode:Integer, successMessage:String
- **installManagedCertificate**: function(certFile:String, password:String)  
Installa in locale il certificato di firma "medium"  
certFile: url del certificato  
password: eventuale password che si vuole assegnare per l'accesso alla chiave privata contenuta nel certificato (non obbligatorio); se non indicata al momento della firma non verrà richiesta alcuna password  
callback data: success:Boolean, errorMessage:String, errorCode:Integer
- **uninstallManagedCertificate**: function(name:String)  
Disinstalla il certificato di firma "medium"  
name: nome del certificato ottenibile dalla chiamata readers  
callback data: success:Boolean
- **tabletStatusById**: function(vendorId:Integer, productId:Integer)  
Rileva la connessione di un dispositivo al PC  
vendorId: Identificativo del produttore del dispositivo  
productId: Identificativo del prodotto  
callback data: success:Boolean, successMessage:String=[Unknown,Connected,Disconnected],  
errorCode:Integer
- **loadDocumentInMemory**: function(filename)  
Carica il documento in memoria  
filename: indica l'url del file da caricare in memoria; il file può anche essere compresso (zip)  
callback data: success:Boolean, errorMessage:String, document:String
- **isDocumentInMemory**: function()  
Verifica se esiste un documento in memoria  
callback data: success:Boolean, document:String
- **clearMemory**: function()  
Cancella la memoria
- **uploadSignedDocumentFromMemory**: function(uploadUrl:String, uploadFilename:String, uploadZipped:Boolean)  
Invia il file firmato in memoria al server  
uploadUrl: Url di destinazione  
uploadFilename: nome file di destinazione  
uploadZipped: Indica se comprimere il file prima dell'invio  
callback data: success:Boolean, successMessage:String, errorMessage:String, errorCode:Integer

- **readDocumentArray**: function()  
legge il file da firmare in formato array di byte javascript  
callback data: success:Boolean, content: Array[byte:Integer]
- **readSignedDocumentArray**: function()  
legge il file firmato in formato array di byte javascript  
callback data: success:Boolean, content: Array[byte:Integer]
- **uploadSignedDocumentFlatFromMemory**: function(uploadUrl:String, uploadFilename:String, uploadZipped:Boolean)  
Invia il file firmato appiattito in memoria al server  
uploadUrl: Url di destinazione  
uploadFilename: nome file di destinazione  
uploadZipped: Indica se comprimere il file prima dell'invio  
callback data: success:Boolean, successMessage:String, errorMessage:String, errorCode:Integer
- **readSignedDocumentFlatArray**: function()  
legge il file firmato appiattito in formato array di byte javascript  
callback data: success:Boolean, content: Array[byte:Integer]
- **uploadSpecimensFromMemory**: function(uploadUrl:String,uploadFilename:String)  
Invia il file zip contenente le immagine degli specimen di firma al server  
uploadUrl: Url di destinazione  
uploadFilename: nome file di destinazione  
callback data: success:Boolean, successMessage:String, errorMessage:String, errorCode:Integer
- **sign** :  
function(downloadUrl:String,downloadFilename:String,uploadUrl:String,uploadFilename:String, signAction:Integer)  
Esegue la firma  
downloadUrl: indica il pdf da firmare  
downloadFilename : se impostato indica il nome del file pdf altrimenti viene prelevato dal parametro downloadUrl  
uploadUrl : indica l'url dove effettuare l'upload del file al termine delle operazioni se non indicata il file viene salvato in locale  
uploadFilename : indica il nome del file che verrà inviato se non indicato verrà effettuata una chiamata in post su upload Url senza l'invio del file  
signAction : se indicato vengono eseguite le seguenti operazioni di firma digitale: 0=Firma , 1=Firma e marca , 2=Controfirma , 3=Marca  
callback data: success:Boolean, successMessage:String, errorMessage:String, errorCode:Integer , specimens:Array[{signature:String, photo:String}] , document:String, signedDocument:String, signedDocumentFlat:String, specimensFile:String
- **readers** : function(onlyLocal:Boolean, localStorage:Boolean)  
Rileva la lista di lettori presenti nel PC  
callback data: success:Boolean, readers:Array[{name:String, remote:Boolean, smartCard:Boolean, servId:String}]
- **getCertificates** : function(token:Fccertificate)  
Legge i certificati contenuti in una smartcard  
token: indica i parametri per filtrare un particolare certificato. E un oggetto di tipo Fccertificate :  
certificateName: Nome (Common Name) del certificato da leggere, è possibile indicare anche una espressione regolare,  
certificateIndex: Indice corrispondente al certificato da leggere (in alternativa al certificateName);  
readerName: nome del lettore su cui si trova la smartcard con il certificato da leggere;  
userName:utente per l'accesso al servizio (solo per lettori remoti (HSM));  
password: [non utilizzato in questa funzione];  
otp: [non utilizzato in questa funzione]

callback data: success:Boolean, errorMessage:String, errorCode:Integer,  
certificates:Array[{Issuer:Object{country:String, stateOrProvince:String, locality:String,  
organization:String, organizationUnit:String, commonName:String, emailAddress:String,  
serialNumber:String, name:String, surname:String}, Subject:Object{country:String,  
stateOrProvince:String, locality:String, organization:String, organizationUnit:String,  
commonName:String, emailAddress:String, serialNumber:String, name:String, surname:String},  
notRepudiation:Boolean, validFrom:String(GGMMAAAA), validTo:String(GGMMAAAA),  
isDemo:Boolean}]

- **displayManager** : function(action:string,imageUrl:string:autoDetect:Boolean)  
Gestisce lo slide show nel tablet di firma. Il parametro imageUrl è obbligatorio solo nel caso in cui action=show, se autoDetect=true l'immagine viene inviata a tutti i tablet connessi altrimenti viene inviata solo al tablet configurato nelle opzioni.  
action: [status|pause|resume|show|reset] ;  
callback data (action=status): success:Boolean, isPlaying:Boolean  
callback data (action=pause|resume|show|reset): success:Boolean
- **getSupportedTablets** : function(connected:boolean)  
Restituisce l'elenco dei device di firma gestiti  
connected: se true vengono mostrati solo i device connessi al PC  
callback data: success:Boolean, tablets:Array[{name:String, signatureAreaWidth:Integer,  
signatureAreaHeight:Integer, vendorId:Integer, productId:Integer, modelAsString:String,  
maxPressure:Integer, isExternalDevice:Boolean, isTabletPC:Boolean, model:Integer,  
modelId:String}]
- **showDocumentToTheSigner** : function(pdfFile:string, pdfPassword:string,  
forceOnPrimaryMonitor:boolean, buttonOkCaption:string, buttonCancelCaption:string)  
Mostra il pdf a tutto schermo utilizzando il viewer interno  
pdfFile: path locale o url del file pdf da visualizzare  
pdfPassword: eventuale password di accesso al file pdf  
forceOnPrimaryMonitor: se true il pdf viene mostrato sempre nel monitor principale altrimenti viene mostrato nel monitor selezionato nelle opzioni di firma grafo metrica  
buttonOkCaption: imposta la caption del bottone di conferma  
buttonCancelCaption: imposta la caption del bottone di annullamento  
callback data: success:Boolean
- **selectSignaturePosition** : function(pdfFile:string, pdfPassword:string)  
Mostra il pdf utilizzando il viewer interno al fine di selezionare la posizione di firma o il campo firma vuoto eventualmente già presente.  
pdfFile: path locale o url del file pdf da visualizzare  
pdfPassword: eventuale password di accesso al file pdf  
callback data: success:Boolean , position:Object{top:Integer, left:Integer, width:Integer,  
height:Integer, page:integer, fieldname:String}
- **verifyDevice** : function(readerName:string, pin:string)  
Verifica il corretto funzionamento di una smartcard testando l'accesso tramite il pin.  
readerName: nome del lettore contenente la smartcard  
pin: password di accesso alla smartcard  
callback data: success:Boolean
- **turnOffAllSecondaryMonitors** : function()  
Spegne tutti gli eventuali monitor secondari compresi i tablet di firma con monitor  
callback data: success:Boolean
- **turnOnAllSecondaryMonitors**: function()  
Accende tutti gli eventuali monitor secondari compresi i tablet di firma con monitor  
callback data: success:Boolean

- **settingShowUI:** function()  
Mostra la form interna per la modifica delle opzioni  
callback data: success:Boolean

Per ricevere la notifica su un eventuale blocco del local server è possibile settare la variabile globale **fcHttpServerOfflineCallback** con una propria funzione.

Es: fcHttpServerOfflineCallback = function () { alert('Server offline'); }

L'applicazione fchttpserver.exe di default non presenta alcuna possibilità di interazione da parte dell'utente ma se si esegue con il parametro **-admin** si avrà la possibilità di accedere all'interfaccia utente tramite un doppio click sull'icona presente nella traybar.

## Slide Show

Tramite **fchttpserver.exe** è anche possibile mostrare delle immagini nel tablet di firma durante il periodo di inattività. Per configurare questa funzionalità occorre editare il file Setting.ini impostando i seguenti valori:

```
[HTTPSERVER]
SlideShow=1                #indica se attivare o meno l'invio delle immagini al tablet
SlideShowAutoDetectTablet=1 #indica se inviare le immagini su tutti i tablet rilevati
SlideShowIntervalSec=2     #indica l'intervallo in secondi tra una immagine e la successiva
SlideShowPath=C:\Images\   #indica il percorso dove reperire le immagini, sono ammesse immagini in
                             formato bmp, jpg, gif, png, xml (vedi "Specifiche XML per generazione
                             immagine")
```

ATTENZIONE: Se si utilizza fcsign.exe, questa funzionalità dovrà essere messa in pausa prima di ogni firma grafometrica in quanto il tablet deve essere libero al momento dell'acquisizione dei dati biometrici, per fare questo è possibile eseguire degli appositi comandi:

fcsign.exe @NOFILE 999 "AdvancedCommand=HttpServerDisplayManger;Action=status;"  
(se l'exitcode del comando è 0 indica che lo slide show è attivo)

fcsign.exe @NOFILE 999 "AdvancedCommand=HttpServerDisplayManger;Action=resume;"  
(se l'exitcode del comando è 0 indica che lo slide show è stato messo in pausa)

fcsign.exe @NOFILE 999 "AdvancedCommand=HttpServerDisplayManger;Action=pause;"  
(se l'exitcode del comando è 0 indica che lo slide show è stato ripristinato)

## Abilitare protocollo http in modalità https

Quando si esegue il server in modalità https è possibile abilitare anche il protocollo http. Per configurare questa funzionalità occorre editare il file Setting.ini impostando i seguenti valori:

```
[HTTPSERVER]
DefaultPortHTTP=7778        #indica la porta che verrà utilizzata per il protocollo http
DefaultPortHTTPS=7777       # indica la porta che verrà utilizzata per il protocollo https
SSLEnabled=1
```

I valori delle porte per i due protocolli devono essere diversi.

## Utilizzo DNS esterno per servizio in HTTPS

Se si desidera utilizzare un certificato SSL emesso da GeoTrust senza bisogno di aggiungere identità affidabili nel Windows Certificate Storage occorre impostare il seguente valore:

```
[HTTPSERVER]
ExternalDNS=1 #impostando questo valore l'url dovrà essere modificata da localhost a
fchttp.namirial.com
```

```
<script type="text/javascript" src="http://fchttp.namirial.com:7777/files/fcsign.js"></script>
```

## Esecuzione in modalità multi utente

In caso di distribuzione di una applicazione web su Citrix o Windows Terminal Server è possibile eseguire più istanze del local server selezionando, in fase di setup, l'opzione "Modalità multi utente". In questa modalità ogni local server all'avvio occuperà la prima porta TCP libera diversa per ogni utente e si dovrà quindi modificare il metodo di caricamento dello script fcsign.js utilizzando la risorsa javascript distribuita tramite il setup nella cartella \resources\js\loadfcsign.js , e chiamando il metodo:

**loadfcsign** : function (sessionId, useHttpProtocol, localServerAddress, portMin, portMax)

Esegue un controllo per capire quale local server usare relativamente all'utente passato nel apposito parametro.

sessionId: nome utente di windows (obbligatorio)

useHttpProtocol: se true verrà utilizzato il protocollo http invece del https (default false)

localServerAddress: indirizzo del local server (default 'localhost')

portMin: porta TCP da cui iniziare il controllo (default 7777)

portMax: ultima porta TCP che verrà eventualmente controllata (default 7827). Normalmente la porta del servizio relativa all'utente desiderato viene trovata o al primo o al secondo tentativo.

**fcHttpServerOnlineCallback** : funzione di callback che verrà chiamata al termine del caricamento se avvenuto con successo.

Esempio:

```
<head>
  <script type="text/javascript" src="loadfcsign.js"></script>
</head>
<body>
<script type="text/javascript">
```

```
fcHttpServerOnlineCallback = serverOnLine
```

```
loadfcsign('utente1'); // nome utente di windows
```

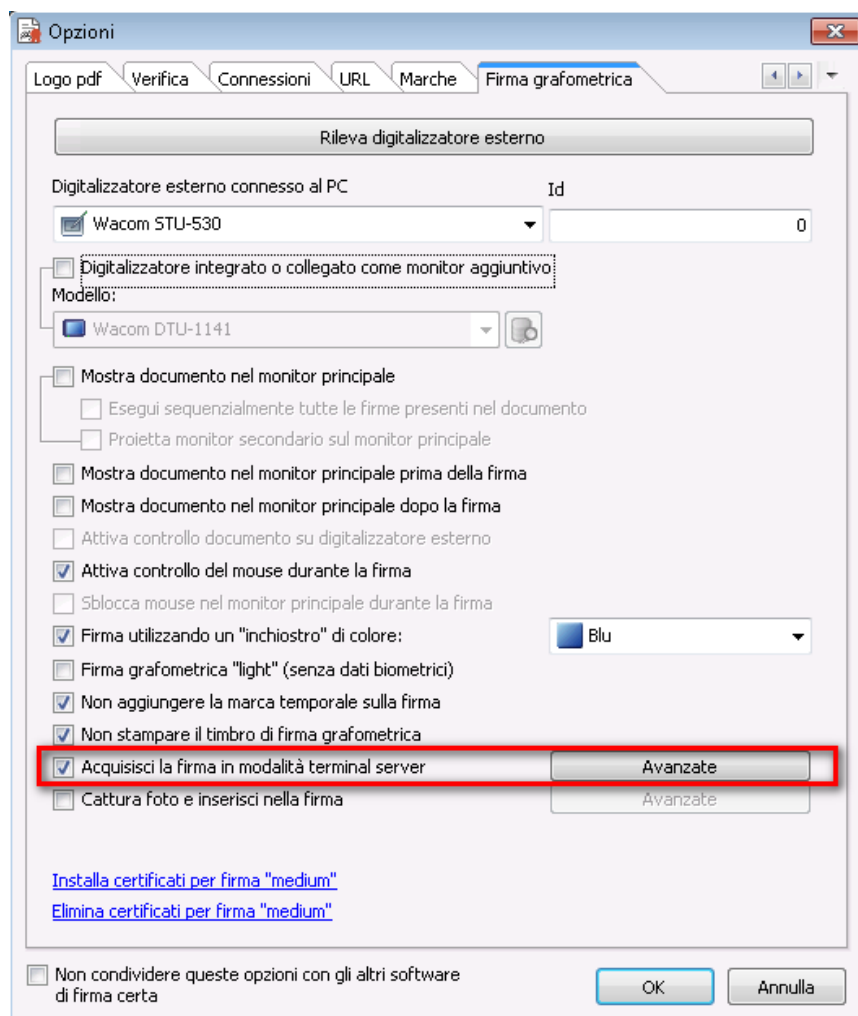
```
function serverOnLine()
{
  alert('Server online');
}
```

```
</script>
```

## Firma grafometrica in Terminal Server

Tramite sdk è possibile operare anche modalità terminal server per quanto riguarda la firma grafometrica che tipicamente implica la presenza di un dispositivo esterno lato client che non può essere condiviso come risorsa per questioni di sicurezza legate al transito non sicuro dei dati biometrici.

Per attivare questa modalità è sufficiente installare degli appositi add-on lato server e lato client ed attivare l'opzione "Acquisisci la firma in modalità terminal server" (modificabile anche da sdk settando `fcxSetting.AdvancedSignatureTerminalServerMode = true`) così facendo a fronte di una firma di un pdf lato server partirà una richiesta al client che rileverà i dati biometrici dal dispositivo e li invierà criptati al server sfruttando un apposito virtual channel della connessione terminal server. I vantaggi di questo tipo di soluzione sono la sicurezza dei dati biometrici e la bassa occupazione di banda (circa 40 KB a firma). La soluzione è compatibile sia con Windows Terminal Server che con Citrix ([certificazione Citrix Ready](#)).



### Installazione add-on Terminal Server (Virtual channel)

Lista software da installare in aggiunta ai software di Firma Certa standard per il funzionamento in Terminal Server da richiedere a parte:

FirmaCerta\_SDK\_TerminalServer[x-x-x] : da installare lato server

FirmaCerta\_TS : da installare lato client

FirmaCerta\_SDK\_Citrix\_Client[x-x-x] : da installare lato client solo se protocollo Citrix

FirmaCerta\_SDK\_WTS\_Client\_x32[x-x-x] : da installare lato client solo se protocollo Windows Terminal Server RDP su PC a 32 bit

FirmaCerta\_SDK\_WTS\_Client\_x64[x-x-x] : da installare lato client solo se protocollo Windows Terminal Server RDP su PC a 64 bit

### Licenze firma grafometrica in ambiente Terminal Server

Se come certificato di firma si utilizza quello contenuto nella smartcard (firma strong) il sistema licenze non cambia rispetto alla versione stand-alone se invece si intende utilizzare un certificato software distribuito tramite un file .fck (firma medium) ci sono 2 possibili strade:

- 1) **Licenza Standard** : deve essere indicato un numero massimo di utenti contemporanei che verrà inserito nel certificato.

Vantaggi : la gestione della licenza è identica alla modalità stand-alone per cui il certificato medium deve essere installato e attivato lato server una sola volta per tutti gli utenti

Svantaggi :

- vengono conteggiati tutti gli utenti connessi al server indipendentemente dal fatto che stiano o meno utilizzano il software di firma
- il numero massimo di utenti contemporanei non può più essere modificato se non acquistando un nuovo certificato di firma.

- 2) **Licenza Advanced** : oltre al numero di utenti contemporanei occorre anche comunicare il "License Repository": path di una cartella sul disco locale del server o anche in rete dove gli utenti devono avere accesso in lettura e scrittura. Non si dovrà più gestire il certificato tramite il file .fck ma si dovrà utilizzare un nuovo file di licenza .fctsk che verrà appositamente creato e che dovrà essere passato aggiungendo i parametri seguenti:

"BiometricDataPdfSignerCertFile=<path assoluto del file .fctsk>;BiometricDataSignerCertFile=<path assoluto del file MasterKeyPublic.pem>"

N.B. : Il file MasterKeyPublic.pem è la parte pubblica della Master Key utilizzata per cifrare i dati biometrici (nel caso Namirial è il file NMR\_4096.pem e verrà rilasciato assieme al file .fctsk).

Vantaggi :

- non si devono effettuare installazioni e attivazioni del certificato di firma
- vengono conteggiati solamente gli utenti che utilizzano il software di firma
- a parità di "License Repository" il numero di utenti contemporanei può essere modificato in qualsiasi momento
- possibilità di monitoraggio del numero di licenze in uso tramite Firma Certa SDK (fcxTerminalServerLicenseManager)

Svantaggi : la gestione del certificato cambia rispetto alla versione stand-alone.

ATTENZIONE: Una volta generato il file .fctsk il "License Repository" non può più essere modificato se non acquistando un nuovo file di licenza .fctsk.

### Sistemi operativi supportati in ambiente Terminal Server:

Server: Windows Server 2003, Windows Server 2003 R2 , Windows Server 2008, Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2

Client: Windows XP sp3, Windows Vista, Windows 7 , Windows 8 , Windows 8.1, Windows 10

## Specifiche XML per generazione immagine

```
<@ObjectType>
  <Background>/9j/4AAQSkZJRg...==</Background>
  <Logo Position="TopLeft">/9j/4AAQSk...</Logo>
  <Draw>
    <Line Color="#000000" Join="Round" Cap="Round" Width="2.1" X1="10.1" Y1="12.3" X2="50.5" Y2="12.3"/>
    <Rectangle LineColor="#000000" LineWidth="1.5" X1="0" Y1="0" X2="WIDTH" Y2="HEIGHT-50"/>
    <Ellipse LineColor="#000000" FillColor="#000000" LineWidth="1" CX="100" CY="100" RX="50" RY="60"/>
    <Image X="10.5" Y="20.5">/9j/4AAQSk...</Image>
  </Draw>
  <Fonts>
    <Font Id="f1" FontName="CourierNew" FontSize="20" Bold="1" Underline="0" Color="#00008B"/>
    <Font Id="f2" FontName="Arial" FontSize="20" Bold="0" Underline="1" Color="#00008B"/>
  </Fonts>
  <Text FontId="f1">
    <Row Align="Center" FontId="f2">testo riga 1</Row>
    <Row> testo riga 2</Row>
    <Row/>
    <Row> testo riga 4</Row>
    <Row/><Row/><Row/>
    <Row FontId="f2" DeltaX="20.3" DeltaY="50.1">Testo ultima riga</Row>
  </Text>
</@ObjectType>
```

### @ObjectType

Se uguale **Operation** indica lo sfondo del campo di firma grafometrica.

Se uguale **Bitmap** indica una immagine bitmap da utilizzare come logo di firma digitale.

### Bitmap:

Se ObjectType=Bitmap occorre indicare obbligatoriamente anche la larghezza e l'altezza.

Attributi:

- Width : <numero intero> larghezza dell'immagine (obbligatorio)
- Height : <numero intero> altezza dell'immagine (obbligatorio)
- Rotation : 0 (default), 90, 180, 270
- BackgroundColor : colore = <codice html del colore da utilizzare> (default #FFFFFF)
- MarginWidth : <numero intero> distanza dai margini (default 5)

### Background

Se presente indica l'immagine da utilizzare come sfondo. Il formato deve essere jpeg o bitmap e deve essere codificato in base 64. Le dimensioni dell'immagine verranno eventualmente adattate alle dimensioni del display di firma.

### Logo

Se presente indica l'immagine da utilizzare come logo. Il formato deve essere jpeg o bitmap e deve essere codificato in base 64. Le dimensioni dell'immagine non verranno modificate.

Attributi:

- Position : definisce la posizione del logo = TopRight (default), TopLeft, BottomLeft, BottomRight, CenterLeft, CenterRight, CenterTop, CenterBottom

### Draw

Se presente contiene la lista degli elementi grafici (Line,Rectangle,Ellipse) da disegnare.

### Line

Disegna una linea.

Attributi:

- Color: colore = <codice html del colore da utilizzare> (default #000000)
- Join : giunzione linee = Miter (default), Round, Bevel
- Cap : estermità linea = Butt (default), Square, Round
- Width : spessore linea = <numero reale> (default 1)
- X1 : coordinata x1 in alto a sinistra = <numero reale> (default 0) (\*)
- Y1 : coordinata y1 in alto a sinistra = <numero reale> (default 0) (\*)
- X2 : coordinata x2 in basso a destra = <numero reale> (default 0) (\*)
- Y2 : coordinata y2 in basso a destra = <numero reale> (default 0) (\*)

### Rectangle

Disegna un rettangolo.

Attributi:

- LineColor: colore linea = <codice html del colore da utilizzare> (default #000000)
- FillColor: colore di riempimento = <codice html del colore da utilizzare> (se non definito non verrà riempito)
- LineWidth : spessore linea = <numero reale> (default 1)
- X1 : coordinata x1 in alto a sinistra = <numero reale> (default 0) (\*)
- Y1 : coordinata y1 in alto a sinistra = <numero reale> (default 0) (\*)
- X2 : coordinata x2 in basso a destra = <numero reale> (default 0) (\*)
- Y2 : coordinata y2 in basso a destra = <numero reale> (default 0) (\*)

### Ellipse

Disegna un'ellisse.

Attributi:

- LineColor: colore linea = <codice html del colore da utilizzare> (default #000000)
- FillColor: colore di riempimento = <codice html del colore da utilizzare> (se non definito non verrà riempito)
- LineWidth : spessore linea = <numero reale> (default 1)
- CX : coordinata x del centro = <numero reale> (default 0) (\*)
- CY : coordinata y del centro = <numero reale> (default 0) (\*)
- RX : lunghezza del raggio orizzontale = <numero reale> (default 0) (\*)
- RY : lunghezza del raggio verticale = <numero reale> (default 0) (\*)

### Image

Disegna un'immagine. Il formato deve essere jpeg o bitmap e deve essere codificato in base 64. Le dimensioni dell'immagine non verranno modificate.

Attributi:

- X : coordinata x del vertice in alto a sinistra = <numero reale> (default 0) (\*)
- Y : coordinata y del vertice in alto a sinistra = <numero reale> (default 0) (\*)

### Fonts

Contiene la lista dei font utilizzati.

### Font

Definisce un font.

Attributi:

- Id (obbligatorio) : definisce univocamente il font = <codice alfanumerico a piacere>
- FontName : nome = <font windows di tipo truetype > (default Arial)
- FontSize : dimensione = <numero intero positivo> (default 22)
- Bold : grassetto = <vale 1 o 0> (default 0)
- Underline : sottolineato = <vale 1 o 0> (default 0)
- Color : colore = <codice html del colore da utilizzare> (default #00008B)

### Text

Contiene al lista delle righe di testo da disegnare.

Attributi:

- FontId : definisce l'Id del font di default da utilizzare se non specificato nelle singole righe = <codice alfanumerico presente tra quelli definiti nei fonts>
- Orientation : definisce l'orientamento del testo = TopBottom (default), BottomTop

In alternativa al FontId posso essere definiti direttamente gli attributi:

- FontName : nome = <font windows di tipo truetype > (default Arial)
- FontSize : dimensione = <numero intero positivo> (default 22)
- Bold : grassetto = <vale 1 o 0> (default 0)
- Underline : sottolineato = <vale 1 o 0> (default 0)
- Color : colore = <codice html del colore da utilizzare> (default #00008B)

### Row

Definisce il testo della riga da disegnare (indicare <Row/> per definire una riga vuota). Se il testo indicato supera la larghezza del display di firma viene mandato a capo automaticamente. Nel caso di @ObjectType=Bitmap e' inoltre possibile indicare alcune informazioni aggiuntive utilizzando le variabili @SIGNERNAME (Nome dell'intestatario del certificato di firma), @ISSUERNAME (Nome dell'emittitore del certificato), @SERIALNUMBER (Numero di serie del certificato), @SIGNINGTIME (data di firma) e @SIGNINGDATETIME (data e ora di firma), @SIGNERID (Codice fiscale dell'intestatario del certificato di firma), @SIGNERCOUNTRY (Paese dell'intestatario del certificato di firma), @SIGNEREMAIL (Email dell'intestatario del certificato di firma)

Attributi:

- FontId : definisce l'Id del font di default da utilizzare per quella riga, se non specificato verrà utilizzato il font definito in Text = <codice alfanumerico presente tra quelli definiti nei fonts>
- Align : allineamento del testo = Left (default), Center, Right
- DeltaX : indica lo sfasamento della X rispetto al valore calcolato = <numero intero> (\*)
- DeltaY : indica lo sfasamento della Y rispetto al valore calcolato = <numero intero> (\*)

(\*) L'attributo può essere definito tramite una formula in cui è possibile utilizzare le variabili seguenti:

WIDTH = larghezza dell'immagine di sfondo

HEIGHT= altezza dell'immagine di sfondo

FIELD\_LEFT = coordinata X del punto in alto a sinistra del campo di firma

FIELD\_TOP = coordinata Y del punto in alto a sinistra del campo di firma

FIELD\_RIGHT = coordinata X del punto in basso a destra del campo di firma

FIELD\_BOTTOM = coordinata Y del punto in basso a destra del campo di firma

FIELD\_WIDTH = Larghezza del campo di firma

FIELD\_HEIGHT = Altezza del campo di firma

IMAGE\_WIDTH = Larghezza dell'immagine (utilizzabile solo negli attributi dell'elemento <Image>)

IMAGE\_HEIGHT = Altezza dell'immagine (utilizzabile solo negli attributi dell'elemento <Image>)

### Opacità colore

Per ogni attributo di tipo colore si può indicare l'eventuale opacità aggiungendo un valore da 0 a 255 (FF) in esadecimale. Se non viene indicato viene preso come valore di default 255 (FF) che indica una opacità pari a 0. Per esempio per ottenere un colore rosso con opacità del 50% occorre indicare #FF000032.

### Testo variabile

Se @ObjectType=Operation nell'XML può essere inserita la variabile @DESCRIPTION che verrà sostituita al momento della firma con la descrizione dei campi eventualmente definite nel template.

Per differenziare le impostazioni dei campi firma ricercati tramite testo, altrimenti ereditate dalla definizione del campo "padre", è possibile aggiungere dopo il testo i valori delle proprietà nel formato seguente:

**@ep[chiave1=valore1 | chiave2=valore2]**

Esempi:

Posizionamento di tre campi identificati dall'etichetta #firma1# :

il primo è obbligatorio :

#firma1#@ep[req=1]

il secondo non è obbligatorio e acquisisce solo il tratto senza dati biometrici :

#firma1#@ep[req=0|ndb=1]

Il terzo ha dimensione diversa:

#firma1#@ep[ppi=96|wid=300|hei=200]

Elenco chiavi:

fid = <stringa: fieldid> (utilizzare un doppio due punti [::] per indicare lo spazio)

des = <stringa : descrizione del campo firma> (utilizzare un doppio due punti [::] per indicare lo spazio)

req = <[0,1] : se = 1 campo obbligatorio>

ndb = <[0,1] : se = 1 campo senza dati biometrici>

nts = <[0,1] : se = 1 campo senza marca temporale>

nph = <[0,1] : se = 1 campo senza foto>

nbg = <[0,1] : se = 1 campo senza sfondo di firma>

hei = <[numero intero > 0] : altezza campo>

wid = <[numero intero > 0] : larghezza campo>

ksp = <[TL(Top-Left),ML(Middle-Left),BL(Bottom-Left),BC(Bottom-Center),BR(Bottom-Right),MR(Middle-Right),TR(Top-Right),TC(Top-Center),MC(Middle-Center)] : posizione del campo rispetto al testo trovato (default TL)>

x = <[numero intero <> 0] : traslazione del campo lungo l'asse x>

y = <[numero intero <> 0] : traslazione del campo lungo l'asse y>

ppi = <[numero intero >= 96] : pixel per inch di riferimento nel considerare la larghezza e/o l'altezza eventualmente indicate>