

Teknisk Handbok

För eBREV Webb via Webservice
Gäller från den 25 maj 2009



Innehåll

1	Introduktion	3
1.1	Allmänt.....	3
1.2	Så skickar ni eBrevWebb via Webservice	3
1.3	Testrutiner.....	3
4	Beskrivning av eBREV Webbservices.....	4
4.1	Kommunikationsöversikt.....	4
5	Filstruktur på zip-arkiv	5
5.1	Brevfil.....	5
5.2	Adresslista	6
5.2.1	Utlandsförsändelser	6
5.3	.Konfigurationsfil eBREV-WebbD.cfg.....	7
6	API	9
	Autentisering	9
	Exceptions och felkoder	9
	Returvärden	9
6.1	Metoder.....	9
6.1.1	sendWithAddressing	9
6.1.2	Ping.....	9
6.2	Exceptions och felkoder.....	11
7	Kundstöd	12
8	FAQ	13
9	Exempel.....	14
9.1	Konfigurationsfil med separat adressfil	14
9.2	Konfigurationsfil med adresslista och giro information.....	15
9.2	Kodexempel.....	16
9.2.1	Sändning med Java (>=1.5).....	16
9.2.2	Sändning med C# (MS Visual Studio .NET 3.5 WCF).....	17

1 Introduktion

Med eBREV Webb kan ni göra stora enhetliga utskick av brev. Dokument och adressfil skickas elektroniskt till Posten, som skriver ut och distribuerar fysiska försändelser.

eBrev Webb är öppen dygnet runt, de beställningar ni skickar innan 20.00 levereras nästa dag (A-post) eller inom tre dagar (B-post). Ni kan skicka dokument i färg (endast B-post) eller svart/vitt.

På www.posten.se under finns all information om eBrev Webb. Ni kan lämna dokument på två sätt till eBREV Webb. Antingen med ID på www.posten.se, eller genom att använda dator till dator-kommunikation via Webbservice.

1.1 Allmänt

Senaste versionen av den här handboken finns på www.posten.se.

För att använda Webbservice krävs att

- ni är upplagd som kund, vilket görs på www.posten.se
- ni tillsammans med Posten gör tester innan första uppdrag sänds in.

1.2 Så skickar ni eBrevWebb via WebService

1. Ni levererar ett ziparkiv till Posten.

2. Vi kontrollerar att filens struktur och varnar för eventuella felaktigheter som inte hänger ihop med utskriftsdatat.

3. Filen fördelas mellan de olika utskriftscentralerna där den skrivs ut, kuverteras och lämnas för utdelning.

4. Följesedel och faktura sänds till er.

1.3 Testrutiner

Implementeringen av Webbservices måste alltid testas och godkännas av Posten.

Beskrivning av testförfarandet:

1. Posten och testansvarig hos er gör tidsplan och testscenarios.

2. Testerna sker i delmoment.

- inloggning och kommunikation mot testmiljö för Webbservice/PWS
- uppladdning av zip-arkivet mot testmiljön.
- Kontroll av ingående filer.
- Kontroll av layout, typografi och kvalitet på brev av Posten.

Adress filen för test ska innehålla några fiktiva mottagare.

När testerna är avslutade och godkända av Posten är ni redo för att skicka försändelser mot produktionsmiljön.

4 Beskrivning av eBREV Webbservices

4.1 Kommunikationsöversikt

Posten publicerar webservices för att underlätta kommunikation mot tjänsten eBREV Webb. För eBrevWebb kallas denna service PWS (Posten Web Service).

5 Filstruktur på zip-arkiv

För att skicka in data till eBREV Webb via webservice-tjänsten skickas ett komprimerat filarkiv i formatet zip. Zip-arkivet ska innehålla tre alternativt två stycken filer:

- Brevfil (PDF eller Word doc)
- Konfigurationsfil (cfg) med eller utan adresslista.
- Adresslista (txt) om adresser ej återfinns i konfigurationsfilen.

Zip fil måste vara komprimerad på ett sådant sätt att det går att öppna med WinZip. Den får inte vara skyddad med lösenord eller liknande.

5.1 Brevfil

Brevfilen ska vara i Microsoft Word eller PDF-format.

För inskickande av framförallt PDF dokument finns ett antal begränsningar och regler. Dessa återfinns på http://www.posten.se/c/ebrevwebb_hjalp.

Word dokument får endast innehålla bilder (gif, tiff, jpg, bmp), word grafik och text.

5.2 Adresslista

Listan på mottagare kan levereras på två olika sätt:

- Som en del av cfg filen (att föredra, se nästa kapitel).
- Som en separat tab avgränsad text fil.

Oavsett metod så tillämpas reglerna nedan.

Adressfilen ska vara en tabbavgränsad textfil namngiven med ändelsen ".txt", t ex. "adresser.txt".
Filen skall vara sparad i ANSI/ISO-8859-1 format. UTF-8 godkänns inte.

Bokstaven 'Å' får ej användas i adresslistor

- Endast en mottagare fylls i per rad.
- Max 41 tecken i varje fält (om ej annan nämns nedan).
- Max 5000 adressater.

Följande tabbavgränsade fält ska finnas med

Name Address2 Address3 address4 Zip city country
Namn Adressrad Adressrad Adressrad Postnr Postort Landskod

Fältnamn Förklaring.

Namn (mottagare)	max 41 tecken – obligatoriskt värde
Adressrad1	max 41 tecken
Adressrad2	max 41 tecken
Adressrad3	max 41 tecken
Postnr	max 12 tecken – obligatoriskt värde
Postort	max 41 tecken – obligatoriskt värde
Landskod	max 2 tecken – obligatoriskt värde, SE för Sverige

Med giroinfo:

pgbg	PG = postgiro, BG = bankgiro – obligatoriskt värde
gironumber	beroende på typ, max 20 tecken - obligatoriskt värde
betalningsmottagare	betalningsmottagare, max 30 tecken - obligatoriskt värde
belopp	belopp i kronor, ören - obligatoriskt värde
ocr	ocr nummer.
text1-text6	fritextfält som visas i meddelandefältet, max 30 tecken.

Kontakta Posten support för adressfilsmallar.

5.2.1 Utlandsförsändelser

För att skicka brev till utländska mottagare krävs att en korrekt landskod anges.

Landskoden skall följa ISO 3166 standard.

Zip och City är obligatoriska värden oavsett vilket land som försändelsen skall till, adressfälten 1-3 är frivilliga och kan innehålla max 41 tecken.

Debitering sker enligt gällande portotabell.

5.3 .Konfigurationsfil eBREV-WebbD.cfg

Konfigurationsfilen är unik för eBREV webbservices. XML filen innehåller data som avsändare, mottagare, uppdragstyp och liknande.

Filen är en XML-struktur och teckenkoden ska vara UTF-8
Exempel återfinns i kapitel 9.

Definitioner på XML taggar

<SenderData>

Tag	Beskrivning
<UserID>	Värdet i fältet AnvändarID ska vara det som valts vid registrering
<BPN>	Värdet som ni tilldelats enligt välkomstbrevet. Det är ett numeriskt värde om tio positioner inkl två inledande nollor (ex. 0022233311).
<Email>	Kontakt adress som används vid eventuella störningar.

<PricelInfo> alt <ProductOptions>

Tag	Beskrivning
<ColorType>	'1' för färg "0" för svartvitt utskrift
<DeliveryType>	A' för 1:a klassbrev; 'B' för ekonomibrev
<DocumentId>	1 = Med försättsblad (dvs mottagare och avsändare skrivs på ett eget ark) 2 = Giro (endast B-Kort och endast via Posten.se) 3 = Giro (information ligger i bifogad adressfil) 4 = Utan försättsblad (mottagare och avsändare skrivs på dokumentets förstasida)

<SenderAddress>

Avsändaradressen i breven dit obeställbar post returneras. Max 41 tecken per adressrad.

Tag	Beskrivning
<addressLine1>	Avsändarens namn
<addressLine2>	Adressrad 2 (gata)
<addressLine3>	Adressrad 3
<addressLine4>	Adressrad 4
<Zipcode>	Max 8 tecken
<Country>	Endast SE som avsändare

<Attachments>

Det finns tre stycken attachment typer; PDF, DOC och Adresslista. PDF och DOC pekar ut brevfilen och Adresslista pekar ut en tab separerad adressfil (då sådan används). Om du väljer att ange mottagarna i cfg filen behövs inte denna sektion.

Tag	Beskrivning
<Type>	Filty. DOC, PDF alt Adresslista
<Name>	Filens namn
<Email>	Kontakt adress som används vid eventuella störningar.

Ex.

```
<Attachments>
  <Type>PDF</Type>
  <Name>pdf.pdf</Name>
</Attachments>
<Attachments>
  <Type>Adresslista</Type>
  <Name>listan.txt</Name>
</Attachments>
```

<Recipients>

Denna array innehåller mottagaradresser. När denna array återfinns i cfg filen behövs ingen fristående adressfil. Fältens innehåll och begränsningar beskrivs i kapitlet 5.2

```
<Recipients>
<Letter>
<File>DOCUMENT.PDF</File>
<ZipCode>806 36</ZipCode>
<City>GÄVLE</City>
<Country>SE</Country>
<RecipientName>Gunilla Åberg</RecipientName>
<AddressLine1/>
<AddressLine2>Ängsullsvägen 6</AddressLine2>
<AddressLine3/>
</Letter>
</Recipients>
```

Med giro info skall även nedanstående ingå i Recipient taggen:

```
<GiroInfo>
<GiroType>PG</GiroType>
<GiroNumber>12345</GiroNumber>
<Amount>100,20</Amount>
<SenderName>robert</SenderName>
<OCR>848484848488</OCR>
<RefText1>Detta är en fritext</RefText1>
<RefText2/>
<RefText3/>
<RefText4/>
<RefText5/>
<RefText6/>
</GiroInfo>
```

6 API

Autentisering

För HTTP Basic anges användare och lösenord som en base64-kodad HTTP-header enligt RFC-1945.

Exceptions och felkoder

Metoden `sendWithAddressing` och `send` kastar exceptions när försändelsen inte är godkänd eller vid driftsstörningar, dessa fel måste hanteras av klientapplikationen.

En beskrivning av felkoderna återfinns i kapitel 6.2

Returvärden

Det returvärde (märke) som returneras från "send" metoderna skall alltid sparas av kund. Det bör kopplas ihop med det data som du skickat in till Posten i en databas eller liknande. Detta märke används av Posten servicedesk för att referera till er försändelse. Detta märke skall även användas vid reklamation.

6.1 Metoder

6.1.1 sendWithAddressing

```
String sendWithAddressing( byte[] data,  
                          String tfsMsgType,  
                          String tfsSender,  
                          String tfsReceiver )
```

Returvärde:

Sträng innehållande statuskod och transaktionsreferens (ex. 200 PWS.xxxx.xxxx.xxxx).
Beskrivning: Sänd data som ett argument i form av en minnesbuffert samt adresseringsparametrar. `tfsMsgType` samt `tfsReceiver` är i normala fall "ebrevwebb", `tfsSender` är kundunik id.

6.1.2 Ping

```
String ping()
```

Beskrivning

Kontrollerar webbservicens status. Endast för test.

Vad noga med att inte anropa denna metod mellan varje anrop till `sendxxx` då detta förlänger processtiden för dina inskick.

Returvärde

En sträng. "pong".

6.1.3 Send

Send ersätts av sendWithAddressing och skall ej användas annat än för bakåtkompatibilitet.

Synopsis: String send(byte[] data)

Returvärde: Sträng innehållande statuskod och transaktionsreferens (märke).

Beskrivning: Sänd data som ett argument i form av en minnesbuffert. Adressering sker med SOAP-Headers.

Headers: TFSSnd avsändaradress

TFSRec mottagaradress

TFSMsgType meddelandetyyp

SOAP Headers

För korrekt hantering i eBREV Webb via Webservices krävs utöver själva meddelandet vissa metadata som styr meddelandeflödet, t ex avsändare och mottagare, meddelandetyyp och önskade kvittenser.

Dessa uppgifter anges normalt direkt i meddelandet på något av de headerformat som specificerats av Posten men i PWS finns även möjlighet att ange dessa som SOAP Headers.

Det finns även möjlighet att ange egendefinierade attribut kring meddelandet som dock inte blir

Teknisk Handbok eBREV Webb, Web Services V1.2_040614 9

synliga för mottagaren. Dessa attribut registreras dock i TFS och förblir associerade med ursprungsmeddelandet och synliga när tex beställda kvittenser erhålls. De kan innehålla information som bara är av intresse för avsändaren tex avsändarens interna meddelande-id.

Headers i förfrågan

TFSSnd (obligatorisk)

Avsändarens adress. Adressen måste tillsammans med eventuell kvalificerare vara registrerad i TFS och knuten till ett avtal med Posten.

Adressen erhålls av Posten.

TFSRec (obligatorisk)

Mottagarens adress. Måste också vara kopplad till ett avtal med Posten.

Vid användning av eBrevWebb via Webservice ska adressen vara EBREVWEBB.

TFSMsgType (obligatorisk)

Meddelandetyyp enligt Posten. I denna tjänst ska det stå EBREVWEBB

6.2 Exceptions och felkoder

När PWS inte accepterar en försändelse kastar den ett s.k. "exception".
Exempel av utskrift av ett felmeddelande från ett exception;

Errorcode: (300) Invalid zip. Too few files, found: 2, expected: 3

I felmeddelandet återfinns en felkod som beskrivs nedan

Kod	Beskrivning
200	OK
300	Ej godkänd zip fil.
301	Ej godkänd cfg fil
302	Konfigurationsfil saknas
303	Felaktigt dokument
304	Dokument (doc/pdf) saknas
305	Adresslista saknas
306	Ogiltig adresslista
307	Konverterings fel kontakta support.
308	Kommunikationsfel ,kontakta support.
309	Felaktig teckenuppsättning (character set). Adresslistan skall vara ANSI (ISO-8859-1). Config skall vara UTF-8.
310	XML Parsning misslyckades, felaktig konfig fil
312	Dokumentnamned i CFG stämmer ej överens med filens namn
313	Zip filen är för stor (MAX 3MB)
314	Dokumentet är för stort (MAX 3MB)
315	SOAP header saknas (endast om du använder "send". Byt till "sendWithAdressing" metoden istället.
316	Konverterings fel, kontakta support.
317	För många mottagare i adressfilen
318	Ogiltigt PDF. Vanligast förekommande när dokumentet skapats av Ghostscript i kombination med TTF fonter. Kontrollera vilket dokument som skapat PDF'en samt att fonterna (typsnitten) som används är giltiga. Fonter som har namn av typen xxxx+TTE är ej giltiga.
319	Felaktigt antal sidor i dokumentet. Min 1 Max 6 (12 för b-post).

7 Kundstöd

Postens Service Desk har öppet 07.00–21.00 vardagar på 020-55 77 00.

Frågor om priser, tilläggstjänster etc. besvaras av er säljare på Posten eller av Posten Kundtjänst Företag (020-23 22 20, www.posten.se).

Posten Kundtjänst Företag

105 00 Stockholm

Tel 020-23 22 20. Fax 08-28 81 77

kundtjanst.foretag@posten.se

Du hittar också information på www.posten.se

8 FAQ

När kan kunden skicka in sina brev?

Tjänsten är öppen dygnet runt med undantag för service.

Vart vänder kunden sig när han har kommunikationsproblem?

Man vänder sig till Postens Service Desk på telefon 020-55 77 00

Vilka SOAP-standarder stöds?

PWS stödjer de inofficiella W3C-standarderna:

SOAP 1.1, <http://www.w3.org/TR/soap>

WSDL 1.1, <http://www.w3.org/TR/wsdl>

Vilka implementationer av SOAP är kompatibla med PWS?

PWS bygger på Apache Axis 1.1. Kompatibilitet med andra SOAP-implementationer framgår av diverse interoperabilitetstester, t ex <http://www.whitemesa.com/interop.htm>.

I strävan att bibehålla kompatibilitet har i möjligaste mån komplexa datatyper undvikits i parametrar och returvärden.

9 Exempel

9.1 Konfigurationsfil med separat adressfil

OBS ! teckenkodning i UTF-8
Filnamn=eBREV-WebbD.cfg

Exempel med fristående adresslista:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\xml\config.xsd">
  <SenderData>
    <UserId>xxxx</UserId>
    <CustomerId>yyyy</CustomerId>
    <BPN>1234567</BPN>
    <DocumentId>4</DocumentId>
  </SenderData>
  <PriceInfo>
    <ColorType>0</ColorType>
    <DeliveryType>B</DeliveryType>
  </PriceInfo>
  <senderAddress>
    <addressLine1>Företaget AB</addressLine1>
    <addressLine2>Box 123</addressLine2>
    <addressLine3></addressLine3>
    <addressLine4></addressLine4>
    <zipCode>12345</zipCode>
    <city>Stockholm</city>
    <country>SE</country>
  </senderAddress>
  <Attachments>
    <Type>DOC</Type>
    <Name>doc.doc</Name>
  </Attachments>
  <Attachments>
    <Type>Adresslista</Type>
    <Name>addresses.txt</Name>
  </Attachments>
</Config>
```

9.2 Konfigurationsfil med adresslista och giro information

Denna typ av config är att föredra framför den där mottagaradresserna levereras i en separat text fil.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Config>
  <SenderData>
    <BPN>0020000000</BPN>
    <UserId>myuser</UserId>
    <Email>kale@mycompany.com</Email>
  </SenderData>
  <ProductOptions>
    <ColorType>0</ColorType>
    <DeliveryType>B</DeliveryType>
    <DocumentId>3</DocumentId>
  </ProductOptions>
  <SenderAddress>
    <AddressLine1></AddressLine1>
    <AddressLine2></AddressLine2>
    <AddressLine3></AddressLine3>
    <AddressLine4></AddressLine4>
    <ZipCode>12345</ZipCode>
    <City>Mycity</City>
    <Country>SE</Country>
  </SenderAddress>
  <Recipients>
  <Letter>
    <File>DOCUMENT.PDF</File>
    <ZipCode>75435</ZipCode>
    <City>Uppsala</City>
    <Country>SE</Country>
    <RecipientName>Kalle Kanin</RecipientName>
    <AddressLine1>Gatan 123</AddressLine1>
    <AddressLine2 />
    <AddressLine3 />
    <GiroInfo>
      <GiroType>PG</GiroType>
      <GiroNumber>123-123</GiroNumber>
      <Amount>100,01</Amount>
      <PaymentReceiver>My Company</PaymentReceiver>
      <OCR>848484848488</OCR>
      <RefText1>Fritext</RefText1>
      <RefText2></RefText2>
      <RefText3></RefText3>
      <RefText4></RefText4>
      <RefText5></RefText5>
      <RefText6></RefText6>
    </GiroInfo>
  </Letter>
</Recipients>
</Config>
```

9.2 Kodexempel

Kodexemplen som anropar PWS tjänsten i detta avsnitt är skrivna i Java (nyttjar produkten Axis från Apache; <http://ws.apache.org/axis/>) samt i C# (MS Visual Studio .NET).

Exemplen måste kompletteras med kundunika parametrar.

Observera att testmiljön och produktionsmiljön använder sig av olika URL'er samt inloggningsinformation.

Observera att dessa exempel inte är kompletta utan skall endast användas som en vägledning.

9.2.1 Sändning med Java (>=1.5)

```
import java.io.ByteArrayOutputStream;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.URL;

import javax.xml.namespace.QName;

import org.apache.axis.client.Call;
import org.apache.axis.client.Service;

public class Pws2Sample {

    public final String NAMESPACE = "http://pws.posten.se";
    public final int BUFSIZE = 8152;

    /**
     * Endpoint make sure it is correct!
     */
    public final String ENDPOINT = "http://localhost:9081/pws/services/PWS";

    /**
     * Authentication. Change here
     */
    public String username = "YOUR_USER";
    public String password = "YOUR_PASSWORD";

    public byte[] getByteBuffer( InputStream is ) throws IOException {
        ByteArrayOutputStream os = new ByteArrayOutputStream();
        byte[] buf = new byte[BUFSIZE];
        int len;

        while( true ) {
            len = is.read( buf, 0, buf.length );
            if( len < 0 )
                break;
            os.write( buf, 0, len );
        }
        return os.toByteArray();
    }

    public void go(String fileName){
        try {
            InputStream is = new FileInputStream( fileName );

            // Replace sender with your own user
            String tfsMsgType = "EBREVWEBB";
            String tfsSender = "Company X";
            String tfsReceiver = "EBREVWEBB";

            Service service = new Service();
```

```

        Call call = (Call) service.createCall();

        call.setTargetEndpointAddress( new URL( ENDPOINT ) );
        call.setOperationName( new QName( NAMESPACE, "sendWithAddressing" ) );
        call.setUsername( username );
        call.setPassword( password );

        String result = (String) call.invoke( new Object[] {getByteBuffer(is), tfsMsgType,
tfsSender, tfsReceiver} );

        // The result contains 200 + mark (e.g. "200 SOAP.xxx.xxx")
        System.out.println(result);
    }
    catch( Exception e ) {
        // Message contains error code and description
        // Consult your manual for more information.
        e.getMessage();
    }
}

}

public static void main( String[] args ) {
    Pws2Sample sample = new Pws2Sample();
    sample.go(args[0]);
}
}
}

```

9.2.2 Sändning med C# (MS Visual Studio .NET 3.5 WCF)

Kodfragment. Koden använder sig av en proxy klass ([SendClient](#)) som genererats av Visual Studio.

```

// Avoids the first (505) error
System.Net.ServicePointManager.Expect100Continue = false;

// Load the zip as a binary
string filename = "c:/mytestfile.zip"
System.IO.FileStream fs = File.OpenRead(filename);
byte[] data = new byte[fs.Length];
fs.Read(data, 0, data.Length);

EndpointAddress ea = new EndpointAddress(endpoint); // <-- Insert the correct URL

// Setup security
BasicHttpBinding basic = new BasicHttpBinding();
basic.Security.Mode = BasicHttpSecurityMode.TransportCredentialOnly;
basic.Security.Transport.ClientCredentialType = HttpClientCredentialType.Basic;

pws2.SendClient client = new pws2.SendClient(basic,ea);
client.ClientCredentials.UserName.UserName = "xxx"; // <-- Change here
client.ClientCredentials.UserName.Password = "yyy"; // <-- Change here

try
{
    // The result contains 200 + mark (e.g. "200 PWS.xxx.xxx")
    // Save this in a database for later reference!
    string result = client.sendWithAddressing(data, "EBREVWEBB", "Your company", "EBREVWEBB");
    // <-- Change TfsSnd (Your Company) here
    lstStatus.Items.Add("Result:" + result);
}
catch (FaultException fe)
{
    // Add proper error handling!
    MessageBox.Show(fe.ToString());
}
catch (Exception eee)
{
    MessageBox.Show(eee.ToString());
}
}

```