

MailBoxValidator™ Email Validation Web Service

1. Overview	3
1.1 Overview of the MailBoxValidator Web Service _____	4
1.2 Front End of MailBoxValidator Email Validation Web Service __	5
1.2.1.Integration with In-house System _____	5
1.3 Back End of MailBoxValidator Email Validation Web Service ____	7
1.4 Process Flow Overview _____	7
2. Implementation _____	8
2.1 Basic Parameters - Input _____	9
2.2 Basic Parameters – Output _____	9
2.3 List of levels passed _____	9
2.4 List of possible value for MESSAGE field _____	9
3. Design Information _____	11
3.1 Placement of MailBoxValidator Email Validation Web Service _	11
Appendix I : Sample Code _____	12

1. Overview

This documentation provides a basic understanding and information to help you get started with our products. Look over this documentation to gain a high-level understanding of the process flow that underlies the MailBoxValidator Email Validation Web Service.

For more information, please visit <http://www.fraudlabs.com> or contact your MailBoxValidator representative:

Email: sales@fraudlabs.com

1.1 Overview of the MailBoxValidator Web Service

MailBoxValidator delivers scalable web services solutions that help you check if an email address is a valid one. It finds invalid email addresses without actually sending an email and make sure you keep valid email addresses of your customers.

MailBoxValidator Email Validation Web Service is hosted, programmable XML Web Service that allows exchange of data between systems. It is hosted on redundant servers and 24x7x365 monitoring. MailBoxValidator is highly scalable email validation solution and it can be easily integrated into your website to check addresses in the real time.

Key Features Include:

- Validates the incorrect email address formats and syntax
- Validates DNS domain by checking MX records
- SMTP checking
- Provides a complete XML-based Web Services API
- Contains sample code examples for ease integration

1.2 Front End of MailBoxValidator Email Validation Web Service

The general idea is that front end is responsible for collection input from the user and conforms to some specification that the back end can use. Front end of MailBoxValidator Email Validation Web Service is rather simple to understand. As we are using platform independent XML format to exchange data between systems, you may either integrate our web service to your in-house system, or direct access to our hosted web service.

1.2.1. Integration with In-house System

- I. Our sample codes in different languages are available at: http://www.fraudlabs.com/mailboxvalidator_samplecodes.aspx . Log on and download sample codes that you need (For sample codes in different languages please refer to Appendix II). Below are links to get sample codes:
 - i. Microsoft ASP.NET (v1.1) - VB.NET:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientvb.zip
 - ii. Microsoft ASP.Net (v1.1) - C#:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientcsharp.zip
 - iii. Microsoft ASP.NET (v2.0) - VB.NET:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientvb2005.zip
 - iv. Microsoft ASP.Net (v2.0) - C#:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientcsharp2005.zip
 - v. Java:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientjava.zip
 - vi. PHP:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientphp.zip
 - vii. Python:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientpython.zip
 - viii. Perl:
http://www.fraudlabs.com/samplecode/mailboxvalidator_w_ebserviceclientperl.zip
 - ix. ColdFusion 9:

http://www.fraudlabs.com/samplecode/mailboxvalidator_wbserviceclientcoldfusion.zip

- II. Please go through 'readme' file that we provide together with the sample codes for more set up information
- III. In order to use our service, you need to get your own license key – *How?*
 - i. log on to <http://www.fraudlabs.com>
 - ii. sign up as our registered member
 - iii. check your email to complete user activation
 - iv. get license key :
 - a) Free License Account:
 - 1) at the left menu bar, under category of MailBoxValidator™ Email Validation, click "free license"
 - 2) view Terms of Use (*please note that you must agree with our Terms of Use before proceed)
 - 3) click on "Get Free License Now"
 - 4) the license key will be sent to your email
 - b) Premium Subscription:
 - 1) at the left menu bar, under category of MailBoxValidator™ Email Validation, click "subscribe now"
 - 2) fill in required field in the secure payment form
 - 3) click on "make payment"
 - 4) the license key will be sent to your email after your payment is confirmed by our online payment merchant
- IV. Now fill in the required field
- V. Click on "Submit" and result is provided

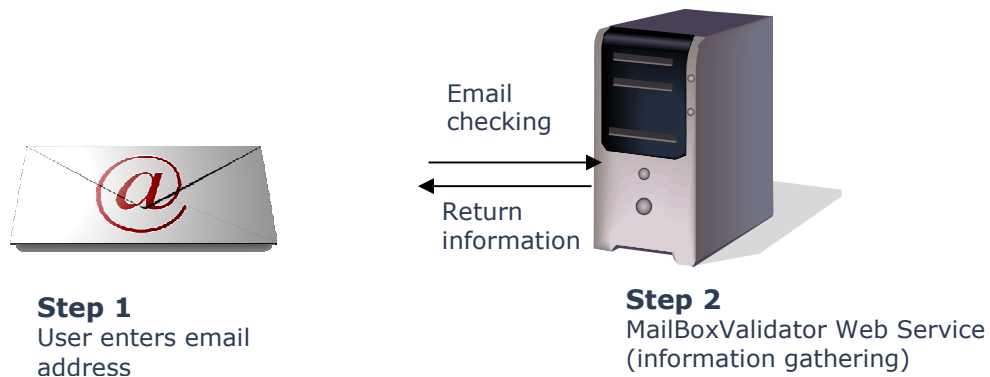
1.3 Back End of MailBoxValidator Email Validation Web Service

Back end is the part that processes the input from the front end. The process is seamless to the end-user and works by interaction with a SOAP API through MailBoxValidator Email Validation Web Service.

The process works as follows:

1. User enters email that to be validate
2. System verify the license key before proceed
3. If the license key is valid, it will proceed and check the credits availability. If the remaining credits still available, it will start to retrieve related information from database
4. Returns and display the precise information

1.4 Process Flow Overview



2. Implementation

This section provides basic information of the process of integrating the web service into your website. Look over this section to gain a high-level understanding of requesting MailBoxValidator Email Validation Web Service.

A WSDL is available at:

<http://v1.fraudlabs.com/mailboxvalidatorwebservice.asmx?wsdl>

For more information about MailBoxValidator Email Validation Web Service implementation, please visit <http://www.fraudlabs.com> or contact your MailBoxValidator representative:

Email: sales@fraudlabs.com

2.1 Basic Parameters - Input

Field	Format	Description
EMAIL	Required	Email address to be verified.
LICENSE	Required	License key for free license and premium users.

2.2 Basic Parameters – Output

Field	Format	Description
IS_SYNTAX	string	Returns True or False to determine if the syntax is correct
IS_DOMAIN	string	Returns True or False to determine if the domain name can be reached
IS_SMTP	string	Returns True or False to determine if the SMTP server is working
IS_LEVEL	string	Returns number of levels of tests passed. If level is 3, means mail server exists but the particular email address could not be verified. If the level is 4, means the email address exists.
CREDITSAVAILABLE	integer	Number of queries remaining in your account, can be used to alert you when you may need to add more queries to your account.
MESSAGE	string	Web Service Message Response

2.3 List of levels passed

Levels	Description
Level 1	The email address syntax is correct.
Level 2	The domain of the email address can be reached.
Level 3	Successfully connected to the SMTP server.
Level 4	The email address has been verified.

* Please note that mail servers that uses SSL/TLS or require authentication may not return an accurate result.

2.4 List of possible value for MESSAGE field

MailBoxValidator Web Service Error Message
<ul style="list-style-type: none">• Error: Domain Checking Failed
<ul style="list-style-type: none">• Error: Syntax checking failed
<ul style="list-style-type: none">• Error: Domain checking failed, connection attempt failed

- | |
|---|
| <ul style="list-style-type: none">• No credit available |
| <ul style="list-style-type: none">• License key was expired |

3. Design Information

This section provides suggestion regarding the placing of MailBoxValidator Email Validation Web Service into Web pages. Look over this section to get a general idea for adding this service to your website.

3.1 Placement of MailBoxValidator Email Validation Web Service

As MailBoxValidator Email Validation Web Service provides you very informative details by using Email, this service can be placed on any forms on your website, offering a seamless integration.

Level of security: High

Site Type(s):

- e-Commerce Solutions
- Internet Retailers
- Software Developers

Benefits:

- Get rid of undelivered mails
- Validate correct format / syntax of emails
- Validate domain for DNS and MX lookups
- Validate SMTP if it accepts connection
- Improve data entry speed and accuracy

Appendix I : Sample Code

MailBoxValidator Web Service sample code is available in several different programming languages. Below are some examples, for more different programming languages please log on to:

<http://www.fraudlabs.com/mailboxvalidatorsamplecodes.aspx>

i. ASP.NET – VB.NET (SOAP)

```
Private Sub MailBoxValidatorWebService()  
Dim x_MailBoxValidator As New  
MailBoxValidatorWebService.MailBoxValidatorWebService  
Dim oMailBoxValidator As New  
MailBoxValidatorWebService.MailBoxValidatorOutput  
Dim iMailBoxValidator As New  
MailBoxValidatorWebService.MailBoxValidatorInput  
  
Try  
    iMailBoxValidator.EMAIL = Me.txtEmail.Text  
    iMailBoxValidator.LICENSE = Me.txtLicense.Text  
  
    oMailBoxValidator =  
x_MailBoxValidator.MailBoxValidator(iMailBoxValidator)  
  
    txtResult.Text = "IS_SYNTAX:" & oMailBoxValidator.IS_SYNTAX &  
vbNewLine  
    txtResult.Text += "IS_DOMAIN:" & oMailBoxValidator.IS_DOMAIN &  
vbNewLine  
    txtResult.Text += "IS_SMTP:" & oMailBoxValidator.IS_SMTP &  
vbNewLine  
    txtResult.Text += "IS_LEVEL:" & oMailBoxValidator.IS_LEVEL &  
vbNewLine  
    txtResult.Text += "CREDITSAVAILABLE:" &  
oMailBoxValidator.CREDITSAVAILABLE & vbNewLine  
    txtResult.Text += "MESSAGE:" & oMailBoxValidator.MESSAGE &  
vbNewLine  
Catch ex As Exception  
    Response.Write(ex.Message)  
End Try  
End Sub
```

ii. ASP.NET – C#.NET (SOAP)

```
private void MailBoxValidatorWebService()  
{  
    MailBoxValidatorWebService x_MailBoxValidator = new  
MailBoxValidatorWebService();  
    MailBoxValidatorOutput oMailBoxValidator = new  
MailBoxValidatorOutput();  
    MailBoxValidatorInput iMailBoxValidator = new  
MailBoxValidatorInput();  
}
```

```
try
{
    iMailBoxValidator.EMAIL = this.txtEmail.Text;
    iMailBoxValidator.LICENSE = this.txtLicense.Text;

    oMailBoxValidator =
x_MailBoxValidator.MailBoxValidator(iMailBoxValidator);

    this.txtResult.Text = "IS_SYNTAX:" + oMailBoxValidator.IS_SYNTAX +
"\n";
    this.txtResult.Text += "IS_DOMAIN:" + oMailBoxValidator.IS_DOMAIN +
"\n";
    this.txtResult.Text += "IS_SMTP:" + oMailBoxValidator.IS_SMTP +
"\n";
    this.txtResult.Text += "IS_LEVEL:" + oMailBoxValidator.IS_LEVEL +
"\n";
    this.txtResult.Text += "CREDITSAVAILABLE:" +
oMailBoxValidator.CREDITSAVAILABLE + "\n";
    this.txtResult.Text += "MESSAGE:" + oMailBoxValidator.MESSAGE +
"\n";
}
catch (Exception ex)
{
    Response.Write(ex.Message);
}
}
```

iii. PHP

```
<?php
if (!isset($_POST['submit'])) {} // if page is not submitted to itself
echo the form
else
{
    $email = $_POST["email"];
    $license = $_POST["license"];

    if ($license == "<Enter License Key>" || $license == "")
    {
        echo "license key is a required field." ;
    }
    else
    {
        $wsdl =
"http://v1.fraudlabs.com/mailboxvalidatorwebservice.asmx?wsdl";
        $client = new SoapClient($wsdl);
        $parms = array("EMAIL" => $email,"LICENSE" => $license);

        $result = $client->MailBoxValidator($parms);

        echo "IS_SYNTAX = " . $result->IS_SYNTAX . "<br>";
        echo "IS_DOMAIN = " . $result->IS_DOMAIN . "<br>";
        echo "IS SMTP = " . $result->IS SMTP . "<br>";
    }
}
```

```
    echo "IS_LEVEL = " . $result->IS_LEVEL . "<br>";
    echo "CREDITSAVAILABLE = " . $result->CREDITSAVAILABLE . "<br>";
    echo "MESSAGE = " . $result->MESSAGE . "<br>";
  }
}
?>
```

iv. **JAVA**

```
public synchronized String MailBoxValidator(String strEMAIL, String
strLICENSE) throws SOAPException
{
    String returnValue = "";
    if (this.url_ == null)
    {
        new SOAPException (Constants.FAULT_CODE_CLIENT, "A URL must be
specified through ApacheSoapProxy.setEndPoint(URL)");
    }
    // Get this from the soapAction attribute on the
    // soap:operation element that is found within the SOAP
    // binding information in the WSDL
    this.soapActionUri_ = "http://v1.fraudlabs.com/";
    ApacheMessageBody ourBody = new ApacheMessageBody ();

    // Set the argument
    ourBody.strEMAIL = strEMAIL;
    ourBody.strLICENSE = strLICENSE;
    //Replace the default body with our own
    this.envelope_.setBody (ourBody);
    message_.send (this.getEndPoint(), this.soapActionUri_,
this.envelope_);
    try
    {
        //Since the Body.unmarshall() handler is static, we can't
        //replace the basic machinery easily. Instead, we must obtain and
        parse the message on our own.
        this.soapMessage_ = this.message_.receive();
        XMLReader reader =
(XMLReader)Class.forName("org.apache.xerces.parsers.SAXParser").newInst
ance();
        SAXHandler handler = new SAXHandler();
        String strResult = "";

        handler.setElementToSearchFor("IS_SYNTAX");
        reader.setContentHandler(handler);
        reader.parse( new InputSource (new StringReader
(this.soapMessage_.getContent().toString() ));
        strResult += "IS_SYNTAX: " + handler.getResult() + "\n";

        handler.setElementToSearchFor("IS_DOMAIN");
        reader.setContentHandler(handler);
        reader.parse( new InputSource (new StringReader
(this.soapMessage_.getContent().toString() ));
        strResult += "IS_DOMAIN: " + handler.getResult() + "\n";
    }
}
```

```
        handler.setElementToSearchFor("IS_SMTP");
        reader.setContentHandler(handler);
        reader.parse( new InputSource (new StringReader
(this.soapMessage_.getContent().toString() ));
        strResult += "IS_SMTP: " + handler.getResult() + "\n";

        handler.setElementToSearchFor("IS_LEVEL");
        reader.setContentHandler(handler);
        reader.parse( new InputSource (new StringReader
(this.soapMessage_.getContent().toString() ));
        strResult += "IS_LEVEL: " + handler.getResult() + "\n";

        handler.setElementToSearchFor("CREDITSAVAILABLE");
        reader.setContentHandler(handler);
        reader.parse( new InputSource (new StringReader
(this.soapMessage_.getContent().toString() ));
        strResult += "CREDITSAVAILABLE: " + handler.getResult() + "\n";

        handler.setElementToSearchFor("MESSAGE");
        reader.setContentHandler(handler);
        reader.parse( new InputSource (new StringReader
(this.soapMessage_.getContent().toString() ));
        strResult += "MESSAGE: " + handler.getResult() + "\n";

        returnValue = strResult;
    }
    catch (Exception exception)
    {
        exception.printStackTrace ();
    }
    return returnValue;
}
```