

IP2Location™ Web Service

~~ Bringing Geography to the Internet ~~

| | |
|--|----|
| 1. Overview | 3 |
| 1.1 Overview of the IP2Location Web Service | 4 |
| 1.2 Front End of IP2Location Web Service | 4 |
| 1.2.1 Integration with In-house System | 5 |
| 1.2.2 Direct Access to Hosted Web Service | 6 |
| 1.3 Back End of IP2Location Web Service | 7 |
| 1.3.1 IP Address to IP Number Conversion | 7 |
| 1.3.2 Record Matching | 8 |
| 1.4 Process Flow Overview | 8 |
| 2. Implementation | 9 |
| 2.1 Implementation Details | 10 |
| 2.2 Basic Parameters - Input | 12 |
| 2.3 Basic Parameters - Output | 12 |
| 2.4 List of possible value for MESSAGE field | 13 |
| 2.5 Structure of the Request and Response | 13 |
| 2.5.1 Request by SOAP | 13 |
| 2.5.2 Response by SOAP | 13 |
| 2.5.3 Request by HTTP-GET | 14 |
| 2.5.4 Response by HTTP-GET | 14 |
| 2.5.5 Request by HTTP-POST | 14 |
| 2.5.6 Response by HTTP-POST | 14 |
| 3. Design Information | 16 |
| 3.1 Placement of IP2Location Web Service | 16 |
| Appendix I : ISO3166 Country Code | 17 |
| Appendix II : Sample Code | 23 |

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 IP2Location Web Services.

For more information, please visit <http://www.fraudlabs.com> or contact your Ip2Location representative:

Email : sales@ip2location.com

1.1 Overview of the IP2Location Web Service

IP2Location delivers scalable web services solutions that help you to determine geographical and other information about your online visitors in real time. We are able to identify user's geographical location, such as country, region, city, latitude, longitude, ISP and domain name using proprietary IP address lookup database and technology without invading Internet user's privacy.

IP2Location Web Service is hosted, programmable XML Web Service that allows exchange of data between systems. Customers can integrate our web service regardless of their web server and other business solutions. This XML web service is used under authorized license to IP2Location.net, the leading geolocation service provider.

IP2Location Web Service is very easy to set up and use. You may understand your visitors better by geographical location within a minute of time. From the information that the web service return to you, you may also control contents distribution by region for digital rights management, or even filter access from countries that you do not do business with to avoid legal issues.

Key Features Include:

- Pinpoints the exact location of customers, with country, region, city, latitude and longitude, Internet Service Providers and domain name using IP address
- Covers all IP address range
- Provides a complete XML-based Web Services API
- Contains sample code examples for ease integration

1.2 Front End of IP2Location 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 IP2Location 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/ip2locationsamplecodes.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 - VB.NET:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientVB.zip>
 - ii. Microsoft ASP.Net - C#:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientCSharp.zip>
 - iii. Microsoft ASP 3.0:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientAsp.zip>
 - iv. Java/ Apache:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientJava.zip>
 - v. PHP:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientPHP.zip>
 - vi. Python:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientPython.zip>
 - vii. Perl:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientPerl.zip>
 - viii. ColdFusion MX:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientColdFusion.zip>
 - ix. Access 2000:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientAccess.zip>
 - x. Excel 2000:
<http://www.fraudlabs.com/samplecode/IP2LocationWebServiceClientExcel.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 IP2Location™ Geolocation, 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 IP2Location™ Geolocation, 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

1.2.2. Direct Access to Hosted Web Service

- I. To use our web service directly, please log on to:
<http://ws.fraudlabs.com/ip2locationwebservice.asmx>
- II. Click on "IP2Location"
- 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 IP2Location™ Geolocation, 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 IP2Location™ Geolocation, 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 you fill in the particular field

V. Click on “Invoke” and result is provided

1.3 Back End of IP2Location 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 IP2Location Web Service.

The process works as follows:

1. User submits an IP address that to be analyzed
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 determine the IP address
4. Returns and display the geographical information

1.3.1. IP Address to IP Number Conversion

If the provided IP address is 161.132.13.1, then the IP number is 2709785857. - *How to get this conversion done?*

In general, the formula to convert an IP Address to IP Number is given as below:

Assume that the IP Address is in this format: A.B.C.D.

$$\text{IP Number} = A * (256 * 256 * 256) + B * (256 * 256) + C * (256) + D$$

So for the IP address 161.132.13.1, the IP number is:

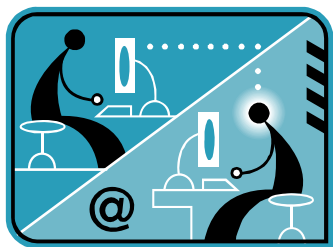
$$\begin{aligned} \text{IP Number} &= 161 * (256 * 256 * 256) + 132 * (256 * 256) + 13 * \\ &\quad (256) + 1 \\ &= 2709785857 \end{aligned}$$

1.3.2. Record Matching

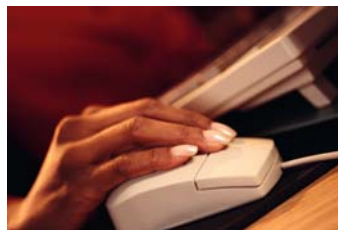
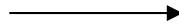
First, we convert the search IP Address to IP Number. Search a record that matches the range condition. It will only give us one match per query. The country information is attached to country fields of the record in our database.

IP From <= IP Number <= IP To

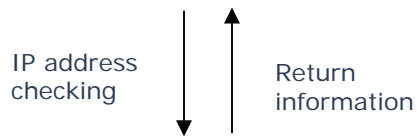
1.4 Process Flow Overview



Step 1
Visitors visit websites



Step 2
Visitor's IP address
captured by merchant



Step 3
IP2Location Web Service
(Geographical information retrieval)

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 IP2Location Web Service.

The implementation section covers the following topics:

- IP2Location Function

For those with solid SOAP programming knowledge, integration basics are included below. POST all requests to:

<http://ws.fraudlabs.com/ip2locationwebservice.asmx>

For more information about IP2Location Web Service implementation, please visit <http://www.fraudlabs.com> or contact your IP2Location representative:

Email : sales@ip2location.com

2.1 Implementation Details

This section was created for those who wish to develop applications that make use of IP2Location Web Service.

SOAP – IP2Location Function Request

```
POST /ws.fraudlabs.com_non_ssl/ip2locationwebservice.asmx HTTP/1.1
Host: ws.fraudlabs.com
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://ws.fraudlabs.com/IP2Location"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2Location xmlns="http://ws.fraudlabs.com/">
      <IP>string</IP>
      <LICENSE>string</LICENSE>
    </IP2Location>
  </soap:Body>
</soap:Envelope>
```

SOAP – IP2Location Function Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2LocationResponse xmlns="http://ws.fraudlabs.com/">
      <IP2LocationResult>
        <COUNTRYCODE>string</COUNTRYCODE>
        <COUNTRYNAME>string</COUNTRYNAME>
        <REGION>string</REGION>
        <CITY>string</CITY>
        <LATITUDE>float</LATITUDE>
        <LONGITUDE>float</LONGITUDE>
        <ZIPCODE>string</ZIPCODE>
        <ISPNAME>string</ISPNAME>
        <DOMAINNAME>string</DOMAINNAME>
        <CREDITSAVAILABLE>string</CREDITSAVAILABLE>
        <MESSAGE>string</MESSAGE>
      </IP2LocationResult>
    </IP2LocationResponse>
```

```
</soap:Body>
</soap:Envelope>
```

HTTP GET- IP2Location Function Request

```
GET
/ws.fraudlabs.com_non_ssl/ip2locationwebservice.asmx/IP2Location?IP=string&LICENSE=string HTTP/1.1
Host: ws.fraudlabs.com
```

HTTP GET- IP2Location Function Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2LOCATION xmlns="http://ws.fraudlabs.com/">
  <COUNTRYCODE>string</COUNTRYCODE>
  <COUNTRYNAME>string</COUNTRYNAME>
  <REGION>string</REGION>
  <CITY>string</CITY>
  <LATITUDE>float</LATITUDE>
  <LONGITUDE>float</LONGITUDE>
  <ZIPCODE>string</ZIPCODE>
  <ISPNAME>string</ISPNAME>
  <DOMAINNAME>string</DOMAINNAME>
  <CREDITSAVAILABLE>string</CREDITSAVAILABLE>
  <MESSAGE>string</MESSAGE>
</IP2LOCATION>
```

HTTP POST- IP2Location Function Request

```
POST /ws.fraudlabs.com_non_ssl/ip2locationwebservice.asmx/IP2Location
HTTP/1.1
Host: ws.fraudlabs.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length

IP=string&LICENSE=string
```

HTTP POST- IP2Location Function Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2LOCATION xmlns="http://ws.fraudlabs.com/">
```

```

<COUNTRYCODE>string</COUNTRYCODE>
<COUNTRYNAME>string</COUNTRYNAME>
<REGION>string</REGION>
<CITY>string</CITY>
<LATITUDE>float</LATITUDE>
<LONGITUDE>float</LONGITUDE>
<ZIPCODE>string</ZIPCODE>
<ISPNAME>string</ISPNAME>
<DOMAINNAME>string</DOMAINNAME>
<CREDITSAVAILABLE>string</CREDITSAVAILABLE>
<MESSAGE>string</MESSAGE>
</IP2LOCATION>
    
```

A WSDL is available at:

<http://ws.fraudlabs.com/ip2locationwebservice.asmx?wsdl>

A description of the FraudLabs operation is available at:

<http://ws.fraudlabs.com/ip2locationwebservice.asmx?op=IP2Location>

2.2 Basic Parameters - Input

| Field | Format | Description |
|---------|----------|---|
| IP | Required | IP address of Internet visitor |
| LICENSE | Required | License key for free license and premium users. |

2.3 Basic Parameters - Output

| Field | Format | Description |
|------------------|---------|--|
| COUNTRYCODE | String | ISO-3166 two character country name of the IP address |
| COUNTRYNAME | String | Country name of the IP address |
| REGION | String | Region name of the IP address. |
| CITY | String | City name of the IP address. |
| LATITUDE | Float | Latitude of the IP address. |
| LONGITUDE | Float | Longitude of the IP address. |
| ZIPCODE | String | Zip code of the IP address |
| ISPNAME | String | ISP of the IP address. |
| DOMAINNAME | String | Domain name of the IP address |
| 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 |

2.4 List of possible value for MESSAGE field

| IP2Location Web Service Error Message |
|--|
| • Invalid license key |
| • Invalid IP address |
| • IP address and License Key cannot be blank |
| • No credit available |
| • License key was expired |

2.5 Structure of the Request and Response

2.5.1. Request by SOAP

A valid request posted from user should look something like this:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2Location xmlns="http://ws.fraudlabs.com/">
      <IP>161.132.13.1</IP>
      <LICENSE>xx-xxxx-xxxx</LICENSE>
    </IP2Location>
  </soap:Body>
</soap:Envelope>
```

2.5.2. Response by SOAP

After posting a valid request, IP2Location will return several parameters. An example of a response returned by IP2Location is shown below:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <IP2LocationResponse xmlns="http://ws.fraudlabs.com/">
      <IP2LocationResult>
        <COUNTRYCODE>PE</COUNTRYCODE>
        <COUNTRYNAME>PERU</COUNTRYNAME>
        <REGION>-</REGION>
        <CITY>-</CITY>
        <LATITUDE>-12.05</LATITUDE>
        <LONGITUDE>-77.05</LONGITUDE>
        <ZIPCODE>-</ZIPCODE>
        <ISPNAME>RED CIENTIFICA PERUANA</ISPNAME>
        <DOMAINNAME>RCP.NET.PE</DOMAINNAME>
        <CREDITSAVAILABLE>88</CREDITSAVAILABLE>
        <MESSAGE />
      </IP2LocationResult>
    </IP2LocationResponse>
  </soap:Body>
</soap:Envelope>
```

```
</IP2LocationResult>
</IP2LocationResponse>
</soap:Body>
</soap:Envelope>
```

2.5.3. Request by HTTP-GET

A valid request posted by user:

```
GET
/ws.fraudlabs.com_non_ssl/ip2locationwebservice.asmx/IP2Location?IP=161
.132.13.1&LICENSE=XX-XXXX-XXXX HTTP/1.1
Host: ws.fraudlabs.com
```

2.5.4. Response by HTTP-GET

An example of a response returned by IP2Location is shown below:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2LOCATION xmlns="http://ws.fraudlabs.com/">
  <COUNTRYCODE>PE</COUNTRYCODE>
  <COUNTRYNAME>PERU</COUNTRYNAME>
  <REGION>-</REGION>
  <CITY>-</CITY>
  <LATITUDE>-12.05</LATITUDE>
  <LONGITUDE>-77.05</LONGITUDE>
  <ZIPCODE>-</ZIPCODE>
  <ISPNAME>RED CIENTIFICA PERUANA</ISPNAME>
  <DOMAINNAME>RCP.NET.PE</DOMAINNAME>
  <CREDITSAVAILABLE>88</CREDITSAVAILABLE>
  <MESSAGE />
</IP2LOCATION>
```

2.5.5. Request by HTTP-POST

A valid request posted by user:

```
POST /ws.fraudlabs.com_non_ssl/ip2locationwebservice.asmx/IP2Location
HTTP/1.1
Host: ws.fraudlabs.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length

IP=161.132.13.1&LICENSE=XX-XXXX-XXXX
```

2.5.6. Response by HTTP-POST

An example of a response returned by IP2Location is shown below:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<IP2LOCATION xmlns="http://ws.fraudlabs.com/">
  <COUNTRYCODE>PE</COUNTRYCODE>
  <COUNTRYNAME>PERU</COUNTRYNAME>
  <REGION>-</REGION>
  <CITY>-</CITY>
  <LATITUDE>-12.05</LATITUDE>
  <LONGITUDE>-77.05</LONGITUDE>
  <ZIPCODE>-</ZIPCODE>
  <ISPNAME>RED CIENTIFICA PERUANA</ISPNAME>
  <DOMAINNAME>RCP.NET.PE</DOMAINNAME>
  <CREDITSAVAILABLE>88</CREDITSAVAILABLE>
  <MESSAGE />
</IP2LOCATION>
```

3. Design Information

This section provides suggestion regarding the placing of IP2Location 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 IP2Location Web Service

As IP2Location web service provides you very informative details regarding identification of online visitors' location by using IP address, this service can be placed on any forms on your website, offering a seamless integration. In fact, you may already be using this kind of service without realizing it.

Description: Visitors geographical information is needed when visitor attempts to access to certain websites, especially when it involves online transactions.

Level of security: High

Site Type(s):

- e-Commerce Solutions
- Internet Retailers
- Internet Security Specialist
- Fraud Screening and Web Analytics
- Software Developers

Benefits:

- Understand your visitors better by geographical location
- Control contents distribution by region for digital rights management
- Prevent password sharing and abuse of service for information security
- Redirect web pages based on geographical region for load balancing
- Web log statistics and analysis to understand your visitors better by geographical location
- Reduce credit card fraud
- Auto-selection of country on forms
- Filter access from countries you do not do business with to avoid legal issues
- Save advertisement costs by GEO targeting for increased sales and click-through

Appendix I : ISO3166 Country Code

This table lists all valid ISO3166 two characters country codes that returns from component API query and describe the country names behind these country codes.

| Country Code | Country Name |
|--------------|---------------------------------------|
| AD | ANDORRA |
| AE | UNITED ARAB EMIRATES |
| AF | AFGHANISTAN |
| AG | ANTIGUA AND BARBUDA |
| AI | ANGUILLA |
| AL | ALBANIA |
| AM | ARMENIA |
| AN | NETHERLANDS ANTILLES |
| AO | ANGOLA |
| AP | ASIA PACIFIC |
| AQ | ANTARCTICA |
| AR | ARGENTINA |
| AS | AMERICAN SAMOA |
| AT | AUSTRIA |
| AU | AUSTRALIA |
| AW | ARUBA |
| AZ | AZERBAIJAN |
| BA | BOSNIA AND HERZEGOWINA |
| BB | BARBADOS |
| BD | BANGLADESH |
| BE | BELGIUM |
| BF | BURKINA FASO |
| BG | BULGARIA |
| BH | BAHRAIN |
| BI | BURUNDI |
| BJ | BENIN |
| BM | BERMUDA |
| BN | BRUNEI DARUSSALAM |
| BO | BOLIVIA |
| BR | BRAZIL |
| BS | BAHAMAS |
| BT | BHUTAN |
| BV | BOUVET ISLAND |
| BW | BOTSWANA |
| BY | BELARUS |
| BZ | BELIZE |
| CA | CANADA |
| CC | COCOS (KEELING) ISLANDS |
| CD | CONGO, THE DEMOCRATIC REPUBLIC OF THE |
| CF | CENTRAL AFRICAN REPUBLIC |
| CG | CONGO |

| Country Code | Country Code |
|--------------|---------------------------------|
| CH | SWITZERLAND |
| CI | COTE D'IVOIRE |
| CK | COOK ISLANDS |
| CL | CHILE |
| CM | CAMEROON |
| CN | CHINA |
| CO | COLOMBIA |
| CR | COSTA RICA |
| CS | CZECHOSLOVAKIA (FORMER) |
| CU | CUBA |
| CV | CAPE VERDE |
| CX | CHRISTMAS ISLAND |
| CY | CYPRUS |
| CZ | CZECH REPUBLIC |
| DE | GERMANY |
| DJ | DJIBOUTI |
| DK | DENMARK |
| DM | DOMINICA |
| DO | DOMINICAN REPUBLIC |
| DZ | ALGERIA |
| EC | ECUADOR |
| EE | ESTONIA |
| EG | EGYPT |
| EH | WESTERN SAHARA |
| ER | ERITREA |
| ES | SPAIN |
| ET | ETHIOPIA |
| EU | EUROPEAN UNION |
| FI | FINLAND |
| FJ | FIJI |
| FK | FALKLAND ISLANDS (MALVINAS) |
| FM | MICRONESIA, FEDERATED STATES OF |
| FO | FAROE ISLANDS |
| FR | FRANCE |
| FX | FRANCE, METROPOLITAN |
| GA | GABON |
| GB | GREAT BRITAIN |
| GD | GRENADA |
| GE | GEORGIA |
| GF | FRENCH GUIANA |
| GH | GHANA |
| GI | GIBRALTAR |
| GL | GREENLAND |
| GM | GAMBIA |
| GN | GUINEA |
| GP | GADELOUPE |
| GQ | EQUATORIAL GUINEA |
| GR | GREECE |

| Country Code | Country Code |
|--------------|--|
| GS | SOUTH GEORGIA & SOUTH SANDWICH ISLANDS |
| GT | GUATEMALA |
| GU | GUAM |
| GW | GUINEA-BISSAU |
| GY | GUYANA |
| HK | HONG KONG |
| HM | HEARD ISLAND AND MCDONALD ISLANDS |
| HN | HONDURAS |
| HR | CROATIA |
| HT | HAITI |
| HU | HUNGARY |
| ID | INDONESIA |
| IE | IRELAND |
| IL | ISRAEL |
| IN | INDIA |
| IO | BRITISH INDIAN OCEAN TERRITORY |
| IQ | IRAQ |
| IR | IRAN, ISLAMIC REPUBLIC OF |
| IS | ICELAND |
| IT | ITALY |
| JM | JAMAICA |
| JO | JORDAN |
| JP | JAPAN |
| KE | KENYA |
| KG | KYRGYZSTAN |
| KH | CAMBODIA |
| KI | KIRIBATI |
| KM | COMOROS |
| KN | SAINT KITTS AND NEVIS |
| KP | KOREA, DEMOCRATIC PEOPLE'S REPUBLIC OF |
| KR | KOREA, REPUBLIC OF |
| KW | KUWAIT |
| KY | CAYMAN ISLANDS |
| KZ | KAZAKSTAN |
| LA | LAO PEOPLE'S DEMOCRATIC REPUBLIC |
| LB | LEBANON |
| LC | SAINT LUCIA |
| LI | LIECHTENSTEIN |
| LK | SRI LANKA |
| LR | LIBERIA |
| LS | LESOTHO |
| LT | LITHUANIA |
| LU | LUXEMBOURG |
| LV | LATVIA |
| LY | LIBYAN ARAB JAMAHIRIYA |
| MA | MOROCCO |
| MC | MONACO |
| MD | MOLDOVA, REPUBLIC OF |

| Country Code | Country Code |
|--------------|---------------------------------|
| MG | MADAGASCAR |
| MH | MARSHALL ISLANDS |
| MK | MACEDONIA, THE FORMER YUGOSLAV |
| ML | MALI |
| MM | MYANMAR |
| MN | MONGOLIA |
| MO | MACAU |
| MP | NORTHERN MARIANA ISLANDS |
| MQ | MARTINIQUE |
| MR | MAURITANIA |
| MS | MONTSERRAT |
| MT | MALTA |
| MU | MAURITIUS |
| MV | MALDIVES |
| MW | MALAWI |
| MX | MEXICO |
| MY | MALAYSIA |
| MZ | MOZAMBIQUE |
| NA | NAMIBIA |
| NC | NEW CALEDONIA |
| NE | NIGER |
| NF | NORFOLK ISLAND |
| NG | NIGERIA |
| NI | NICARAGUA |
| NL | NETHERLANDS |
| NO | NORWAY |
| NP | NEPAL |
| NR | NAURU |
| NU | NIUE |
| NZ | NEW ZEALAND |
| OM | OMAN |
| PA | PANAMA |
| PE | PERU |
| PF | FRENCH POLYNESIA |
| PG | PAPUA NEW GUINEA |
| PH | PHILIPPINES |
| PK | PAKISTAN |
| PL | POLAND |
| PM | SAINT PIERRE AND MIQUELON |
| PN | PITCAIRN |
| PR | PUERTO RICO |
| PS | PALESTINIAN TERRITORY, OCCUPIED |
| PT | PORTUGAL |
| PW | PALAU |
| PY | PARAGUAY |
| QA | QATAR |
| RE | REUNION |
| RO | ROMANIA |

| Country Code | Country Code |
|--------------|--------------------------------------|
| RU | RUSSIAN FEDERATION |
| RW | RWANDA |
| SA | SAUDI ARABIA |
| SB | SOLOMON ISLANDS |
| SC | SEYCHELLES |
| SD | SUDAN |
| SE | SWEDEN |
| SG | SINGAPORE |
| SH | SAINT HELENA |
| SI | SLOVENIA |
| SJ | SVALBARD AND JAN MAYEN |
| SK | SLOVAKIA |
| SL | SIERRA LEONE |
| SM | SAN MARINO |
| SN | SENEGAL |
| SO | SOMALIA |
| SR | SURINAME |
| ST | SAO TOME AND PRINCIPE |
| SU | RUSSIAN FEDERATION |
| SV | EL SALVADOR |
| SY | SYRIAN ARAB REPUBLIC |
| SZ | SWAZILAND |
| TC | TURKS AND CAICOS ISLANDS |
| TD | CHAD |
| TF | FRENCH SOUTHERN TERRITORIES |
| TG | TOGO |
| TH | THAILAND |
| TJ | TAJIKISTAN |
| TK | TOKELAU |
| TM | TURKMENISTAN |
| TN | TUNISIA |
| TO | TONGA |
| TP | EAST TIMOR |
| TR | TURKEY |
| TT | TRINIDAD AND TOBAGO |
| TV | TUVALU |
| TW | TAIWAN, PROVINCE OF CHINA |
| TZ | TANZANIA, UNITED REPUBLIC OF |
| UA | UKRAINE |
| UG | UGANDA |
| UK | UNITED KINGDOM |
| UM | UNITED STATES MINOR OUTLYING ISLANDS |
| US | UNITED STATES |
| UY | URUGUAY |
| UZ | UZBEKISTAN |
| VA | HOLY SEE (VATICAN CITY STATE) |
| VC | SAINT VINCENT AND THE GRENADINES |
| VE | VENEZUELA |

| Country Code | Country Code |
|--------------|-------------------------|
| VG | VIRGIN ISLANDS, BRITISH |
| VI | VIRGIN ISLANDS, U.S. |
| VN | VIET NAM |
| VU | VANUATU |
| WF | WALLIS AND FUTUNA |
| WS | SAMOA |
| YE | YEMEN |
| YT | MAYOTTE |
| YU | YUGOSLAVIA |
| ZA | SOUTH AFRICA |
| ZM | ZAMBIA |
| ZW | ZIMBABWE |

Appendix II : Sample Code

IP2Location 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/ip2locationsamplecodes.aspx>

i. ASP.NET – VB.NET (SOAP)

```
Private Sub Ip2LocationWebService()  
    Dim x_IP2Location As New IP2LocationWebService.Ip2LocationWebService  
    Dim oIP2Location As New IP2LOCATION  
    Try  
        oIP2Location = x_IP2Location.IP2Location( _  
            Me.txtIP.Text, _  
            Me.txtLicense.Text)  
        Me.txtResult.Text = "Country Code: " & oIP2Location.COUNTRYCODE  
& vbNewLine  
        Me.txtResult.Text += "Country Name: " & oIP2Location.COUNTRYNAME  
& vbNewLine  
        Me.txtResult.Text += "Region Name: " & oIP2Location.REGION &  
vbNewLine  
        Me.txtResult.Text += "City: " & oIP2Location.CITY & vbNewLine  
        Me.txtResult.Text += "Latitude: " & oIP2Location.LATITUDE &  
vbNewLine  
        Me.txtResult.Text += "Longitude: " & oIP2Location.LONGITUDE &  
vbNewLine  
        Me.txtResult.Text += "Zip Code: " & oIP2Location.ZIPCODE &  
vbNewLine  
        Me.txtResult.Text += "ISP Name: " & oIP2Location.ISPNAME &  
vbNewLine  
        Me.txtResult.Text += "Domain Name: " & oIP2Location.DOMAINNAME &  
vbNewLine  
        Me.txtResult.Text += "Credits Available: " &  
oIP2Location.CREDITSAVAILABLE & vbNewLine  
        Me.txtResult.Text += "Message: " & oIP2Location.MESSAGE &  
vbNewLine  
    Catch ex As Exception  
        Response.Write(ex.Message)  
    End Try  
End Sub
```

ii. ASP.NET – VB.NET (HTML)

```
Private Sub Lookup()  
    Dim x_IP2Location As New IP2LocationWebService.Ip2LocationWebService  
    Dim oIP2Location As New IP2LOCATION  
    Try  
        ' create the request URL  
        Dim x_builder As String  
        ' add the host element of the URL  
        x_builder =  
"http://ws.fraudlabs.com/Ip2LocationWebService.asmx/IP2Location?" & _
```

```
        "IP=" & Me.txtIP.Text & _
        "&LICENSE=" & Me.txtLicense.Text
        ' create the web client and obtain the response data
        ' as a byte array
    Dim x_web_client As WebClient = New WebClient
    Dim x_response() As Byte =
x_web_client.DownloadData(x_builder.ToString())
        ' process the string result to obtain a validation result
        processResultString(Encoding.Default.GetString(x_response))
    Catch ex As Exception
        Response.Write(ex.Message)
    End Try
End Sub

Private Sub processResultString(ByVal p_result As String)
    Dim indexOpen As Integer
    Dim indexClose As Integer
    Dim strParam As String

    strParam = "<COUNTRYCODE>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</COUNTRYCODE>")
    Me.txtResult.Text = "Country Code: "
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)

    End If
    strParam = "<COUNTRYNAME>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</COUNTRYNAME>")
    Me.txtResult.Text += vbNewLine & "Country Name: "
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If

    strParam = "<REGION>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</REGION>")
    Me.txtResult.Text += vbNewLine & "Region: "
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If

    strParam = "<CITY>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
    indexClose = p_result.IndexOf("</CITY>")
    Me.txtResult.Text += vbNewLine & "City: "
    If indexClose <> -1 Then
        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If

    strParam = "<LATITUDE>"
    indexOpen = p_result.IndexOf(strParam) + strParam.Length
```

```
indexClose = p_result.IndexOf("</LATITUDE>")
Me.txtResult.Text += vbNewLine & "Latitude: "
If indexClose <> -1 Then
    Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
End If

strParam = "<LONGITUDE>"
indexOpen = p_result.IndexOf(strParam) + strParam.Length
indexClose = p_result.IndexOf("</LONGITUDE>")
Me.txtResult.Text += vbNewLine & "Longitude: "
If indexClose <> -1 Then
    Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
End If

strParam = "<ZIPCODE>"
indexOpen = p_result.IndexOf(strParam) + strParam.Length
indexClose = p_result.IndexOf("</ZIPCODE>")
Me.txtResult.Text += vbNewLine & "Zip Code: "
If indexClose <> -1 Then
    Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
End If

strParam = "<ISPNAME>"
indexOpen = p_result.IndexOf(strParam) + strParam.Length
indexClose = p_result.IndexOf("</ISPNAME>")
Me.txtResult.Text += vbNewLine & "ISP Name: "
If indexClose <> -1 Then
    Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
End If

strParam = "<DOMAINNAME>"
indexOpen = p_result.IndexOf(strParam) + strParam.Length
indexClose = p_result.IndexOf("</DOMAINNAME>")
Me.txtResult.Text += vbNewLine & "Domain Name: "
If indexClose <> -1 Then
    Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
End If

strParam = "<CREDITSAVAILABLE>"
indexOpen = p_result.IndexOf(strParam) + strParam.Length
indexClose = p_result.IndexOf("</CREDITSAVAILABLE>")
Me.txtResult.Text += vbNewLine & "Credits Available: "
If indexClose <> -1 Then
    Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
End If

strParam = "<MESSAGE>"
indexOpen = p_result.IndexOf(strParam) + strParam.Length
indexClose = p_result.IndexOf("</MESSAGE>")
Me.txtResult.Text += vbNewLine & "Message: "
If indexClose <> -1 Then
```

```

        Me.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen)
    End If
End Sub

```

iii. ASP.NET – C#.NET (SOAP)

```

private void Ip2LocationWebService()
{
    Ip2LocationWebService.Ip2LocationWebService x_Ip2Location = new
Ip2LocationWebService.Ip2LocationWebService();
    IP2LOCATION oIp2Location = new IP2LOCATION();
    try
    {
        oIp2Location = x_Ip2Location.IP2Location(
            this.txtIP.Text,
            this.txtLicense.Text);
        this.txtResult.Text = "Country Code: " + oIp2Location.COUNTRYCODE
+ "\n";
        this.txtResult.Text += "Country Name: " +
oIp2Location.COUNTRYNAME + "\n";
        this.txtResult.Text += "Region Name: " + oIp2Location.REGION +
"\n";
        this.txtResult.Text += "City: " + oIp2Location.CITY + "\n";
        this.txtResult.Text += "Latitude: " + oIp2Location.LATITUDE +
"\n";
        this.txtResult.Text += "Longitude: " + oIp2Location.LONGITUDE +
"\n";
        this.txtResult.Text += "Zip Code: " + oIp2Location.ZIPCODE +
"\n";
        this.txtResult.Text += "ISP Name: " + oIp2Location.ISPNAME +
"\n";
        this.txtResult.Text += "Domain Name: " + oIp2Location.DOMAINNAME
+ "\n";
        this.txtResult.Text += "Credits Available: " +
oIp2Location.CREDITSAVAILABLE + "\n";
        this.txtResult.Text += "Message: " + oIp2Location.MESSAGE + "\n";
    }
    catch (Exception ex)
    {
        Response.Write(ex.Message);
    }
}

```

iv. ASP.NET – C#.NET (HTML)

```

private void Lookup()
{
    Ip2LocationWebService.Ip2LocationWebService x_Ip2Location = new
Ip2LocationWebServiceClientCSharp.Ip2LocationWebService.Ip2LocationWebS
ervice();
    IP2LOCATION oIp2Location = new IP2LOCATION();
    try
    {

```

```
        //create the request URL
        string x_builder;
        //add the host element of the URL
        x_builder =
"http://ws.fraudlabs.com/Ip2LocationWebService.asmx/IP2Location?";
        x_builder += "IP=" + this.txtIP.Text;
        x_builder += "&LICENSE=" + this.txtLicense.Text;
        // create the web client and obtain the response data
        // as a byte array
        WebClient x_web_client = new WebClient();
        byte [] x_response =
x_web_client.DownloadData(x_builder.ToString());
        // process the string result to obtain a validation result

        processResultString(Encoding.Default.GetString(x_response));
    }
    catch (Exception ex)
    {
        Response.Write(ex.Message);
    }
}

private void processResultString(String p_result)
{
    int indexOpen;
    int indexClose;
    string strParam;

    strParam = "<COUNTRYCODE>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</COUNTRYCODE>");
    this.txtResult.Text = "Country Code: ";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
    }

    strParam = "<COUNTRYNAME>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</COUNTRYNAME>");
    this.txtResult.Text += "\rCountry Name: ";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
    }

    strParam = "<REGION>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</REGION>");
    this.txtResult.Text += "\rRegion Name: ";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
    }
}
```

```
strParam = "<CITY>";
indexOpen = p_result.IndexOf(strParam) + strParam.Length;
indexClose = p_result.IndexOf("</CITY>");
this.txtResult.Text += "\rCity: ";
if (indexClose != -1)
{
    this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
}

strParam = "<LATITUDE>";
indexOpen = p_result.IndexOf(strParam) + strParam.Length;
indexClose = p_result.IndexOf("</LATITUDE>");
this.txtResult.Text += "\rLatitude: ";
if (indexClose != -1)
{
    this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
}

strParam = "<LONGITUDE>";
indexOpen = p_result.IndexOf(strParam) + strParam.Length;
indexClose = p_result.IndexOf("</LONGITUDE>");
this.txtResult.Text += "\rLongitude: ";
if (indexClose != -1)
{
    this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
}

strParam = "<ZIPCODE>";
indexOpen = p_result.IndexOf(strParam) + strParam.Length;
indexClose = p_result.IndexOf("</ZIPCODE>");
this.txtResult.Text += "\rZip Code: ";
if (indexClose != -1)
{
    this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
}

strParam = "<ISPNAME>";
indexOpen = p_result.IndexOf(strParam) + strParam.Length;
indexClose = p_result.IndexOf("</ISPNAME>");
this.txtResult.Text += "\rISP Name: ";
if (indexClose != -1)
{
    this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
}

strParam = "<DOMAINNAME>";
indexOpen = p_result.IndexOf(strParam) + strParam.Length;
indexClose = p_result.IndexOf("</DOMAINNAME>");
this.txtResult.Text += "\rDomain Name: ";
if (indexClose != -1)
{
```

```

    this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
}

    strParam = "<CREDITSAVAILABLE>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</CREDITSAVAILABLE>");
    this.txtResult.Text += "\rCredits Available:";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
    }

    strParam = "<MESSAGE>";
    indexOpen = p_result.IndexOf(strParam) + strParam.Length;
    indexClose = p_result.IndexOf("</MESSAGE>");
    this.txtResult.Text += "\rMessage: ";
    if (indexClose != -1)
    {
        this.txtResult.Text += p_result.Substring(indexOpen, indexClose -
indexOpen);
    }
}

```

v. ASP 3.0

```

Sub sendRequest(strIP, strLICENSE)
    dim SoapBody
    if((strIP <> "") AND (strLICENSE <> "") AND (strLICENSE <> "<Your
License Key>")) then
        SoapBody = xmlSoap(strIP, strLICENSE)
    end if
%>

<table border="0" cellspacing="1" cellpadding="3" rules="rows">
<%
    if SoapBody = "" then
%><tr><td><b>Empty Soap Body Response</b></td></tr><%
    else
        dim xml
        on error resume next
        set xml = Server.CreateObject("Microsoft.XMLDOM")
        if(err.number = 0) then
            xml.async = False
            xml.loadxml(SoapBody)
            dim oNode : set oNode =
xml.selectSingleNode("soap:Envelope/soap:Body/" & SoapAction &
"Response/" & SoapAction & "Result")
            if (oNode.selectSingleNode("Error").nodeTypedValue = "") then
%><tr>
            <th colspan="2" align="left"><b>Result</b></th></tr>
<%
            if (TypeName(oNode.selectSingleNode("COUNTRYCODE")) =
"IXMLDOMElement") then

```

```
%><tr>
  <td align="left">Country Code: </td>
  <td><%=oNode.selectSingleNode("COUNTRYCODE").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("COUNTRYNAME")) =
"IXMLDOMEElement") then %><tr>
  <td align="left">Country Name: </td>
  <td><%=oNode.selectSingleNode("COUNTRYNAME").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("REGION")) = "IXMLDOMEElement")
then %><tr>
  <td align="left">Region Name: </td>
  <td><%=oNode.selectSingleNode("REGION").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("CITY")) = "IXMLDOMEElement") then
%><tr>
  <td align="left">City: </td>
  <td><%=oNode.selectSingleNode("CITY").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("LATITUDE")) = "IXMLDOMEElement")
then %><tr>
  <td align="left">Latitude: </td>
  <td><%=oNode.selectSingleNode("LATITUDE").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("LONGITUDE")) = "IXMLDOMEElement")
then %><tr>
  <td align="left">Longitude: </td>
  <td><%=oNode.selectSingleNode("LONGITUDE").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("ZIPCODE")) = "IXMLDOMEElement")
then %><tr>
  <td align="left">Zip Code: </td>
  <td><%=oNode.selectSingleNode("ZIPCODE").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("ISPNAME")) = "IXMLDOMEElement")
then %><tr>
  <td align="left">ISP Name: </td>
  <td><%=oNode.selectSingleNode("ISPNAME").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("DOMAINNAME")) =
"IXMLDOMEElement") then %><tr>
```

```
<td align="left">Domain Name: </td>
<td><%=oNode.selectSingleNode("DOMAINNAME").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("CREDITSAVAILABLE")) =
"IXMLDOMEElement") then %><tr>
<td align="left">Credits Available: </td>
<td><%=oNode.selectSingleNode("CREDITSAVAILABLE").nodeTypedValue%></td>
</tr>
<% end if %>

<% if(TypeName(oNode.selectSingleNode("MESSAGE")) = "IXMLDOMEElement")
then %><tr>
<td align="left">Message: </td>
<td><%=oNode.selectSingleNode("MESSAGE").nodeTypedValue%></td>
</tr>
<% end if %>

<% else ' error element %><tr>
<td align="left" colspan="2"><b>Error</b></td></tr>
<tr>
<td align="left">Description:</td>
<td><%=oNode.selectSingleNode("MESSAGE").nodeTypedValue%></td>
</tr>

<% end if 'error element does not exist
set xml = nothing
else
%><tr>

<td align="left">This objects requires Microsofts XML Parser 3.0 SP2
or greater.</td></tr>
<tr>
<td>Download here: <a
href="http://download.microsoft.com/download/xml/SP/3.20/W9X2KMeXP/EN-
US/msxml3sp2Setup.exe">http://download.microsoft.com/download/xml/SP/3.
20/W9X2KMeXP/EN-US/msxml3sp2Setup.exe</a></td>
</tr>

<% Response.Write("Error: " & err.number)
end if 'DOM object is valid
end if 'soapbody is <> "" empty string
%>

</table>
<%
end sub

function xmlSoap(strIP, strLICENSE)
' Instantiate objects to hold the XML DOM and the HTTP/XML
communication:
' Instantiate object for HTTP/XML communication:
Dim xmlhttp, strSoap
Set xmlhttp = Server.CreateObject("Msxml2.ServerXMLHTTP")

' Build XML document:
```

```

strSoap = "<?xml version=\"1.0\" encoding=\"utf-8\"?>" & _
          "<soap:Envelope
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"
xmlns:soap=\"http://schemas.xmlsoap.org/soap/envelope/\">" & _
          "<soap:Body>" & _
          "<" & SoapAction & " xmlns=\"\" & SoapNamespace & "\">" & _
          "<IP>" & strIP & "</IP>" & _
          "<LICENSE>" & strLICENSE & "</LICENSE>" & _
          "</" & SoapAction & ">" & _
          "</soap:Body>" & _
          "</soap:Envelope>"

'Build custom HTTP header:
xmlhttp.Open "POST", "http://" & SoapServer & SoapPath, False
'False = Do not respond immediately
xmlhttp.setRequestHeader "Man", "POST " & SoapPath & " HTTP/1.1"
xmlhttp.setRequestHeader "Host", SoapServer
xmlhttp.setRequestHeader "Content-Type", "text/xml; charset=utf-8"
xmlhttp.setRequestHeader "SOAPAction", SoapNamespace & SoapAction

'Send it using the header generated above:
xmlhttp.send(strSoap)
if xmlhttp.Status = 200 then
'Response from server was success
xmlSoap = xmlhttp.responseText
'note xmlSoap is the function name hence the return value is a string
Variant
Else
'Response from server failed
xmlSoap = ""
'Tell administrator what went wrong - maybe not users though
Response.Write("Server Error.<br>")
Response.Write("status = " & xmlhttp.status)
Response.Write("<br>" & xmlhttp.statusText & "<br>" &
xmlhttp.responseText)
Response.Write("<br><pre>" & Request.ServerVariables("ALL_HTTP") &
"</pre>")
end If
Set xmlhttp = nothing
end function
%>

```

vi. PHP

```

<?php

// Get your own license key from http://www.fraudlabs.com
$license = "<Enter License Key>";

// Replace these parameters to your values
$ip = "68.142.197.65";

require_once('./lib/nusoap.php');

```

```

// Call the service with this message
$msg = '<IP2Location xmlns="http://ws.fraudlabs.com/">
<IP>'.$ip.'</IP>
<LICENSE>'.$license.'</LICENSE>
</IP2Location>';

// Create a new SOAP object
$soapclient = new
soapclient('http://ws.fraudlabs.com/ip2locationwebservice.asmx?wsdl','w
sdl');
if (!$soapclient->fault) {
    $result = $soapclient->call('IP2Location', array($msg));
    if(!isset($result['Error'])) {
        // No Error. Retrieve results.
        echo "COUNTRYCODE = " . $result["COUNTRYCODE"] . "<br>";
        echo "COUNTRYNAME = " . $result["COUNTRYNAME"] . "<br>";
        echo "REGION = " . $result["REGION"] . "<br>";
        echo "CITY = " . $result["CITY"] . "<br>";
        echo "LATITUDE = " . $result["LATITUDE"] . "<br>";
        echo "LONGITUDE = " . $result["LONGITUDE"] . "<br>";
        echo "ZIPCODE = " . $result["ZIPCODE"] . "<br>";
        echo "ISPNAME = " . $result["ISPNAME"] . "<br>";
        echo "DOMAINNAME = " . $result["DOMAINNAME"] . "<br>";
        echo "CREDITSAVAILABLE = " . $result["CREDITSAVAILABLE"] .
"<br>";
        echo "MESSAGE = " . $result["MESSAGE"] . "<br>"; }
    else {
        // Error
        echo "Error Description = " . $result["Error"]["Desc"] . "<br>";
        echo "Error Number = " . $result["Error"]["Number"] . "<br>";
    }
    else {
        echo "Error = {$soapclient->faultcode}<br>";
        echo "String = {$soapclient->faultstring}";
    } // end if $soapclient->call is success
?>

```

vii. JAVA / APACHE

```

import java.io.*;

//Web Service Client Class
public class Client
{
    //Entry Point to this Application
    public static void main(String[] args)
    {
        try
        {
            //Create a proxy
            ApacheSoapProxy proxy = new ApacheSoapProxy ();

            //Invoke FraudLabsCheck() over SOAP and get the new OID
            //Require Parameter FraudLabsCheck(String IP, String LICENSE)
            String result = proxy.FraudLabs("68.142.197.65", "<Enter License

```

```
Key>");  
  
    //Print out the value  
    System.out.println (result);  
    }  
    catch (java.net.MalformedURLException exception)  
    {  
        exception.printStackTrace ();  
    }  
    catch (org.apache.soap.SOAPException exception)  
    {  
        exception.printStackTrace ();  
    }  
    }  
}
```