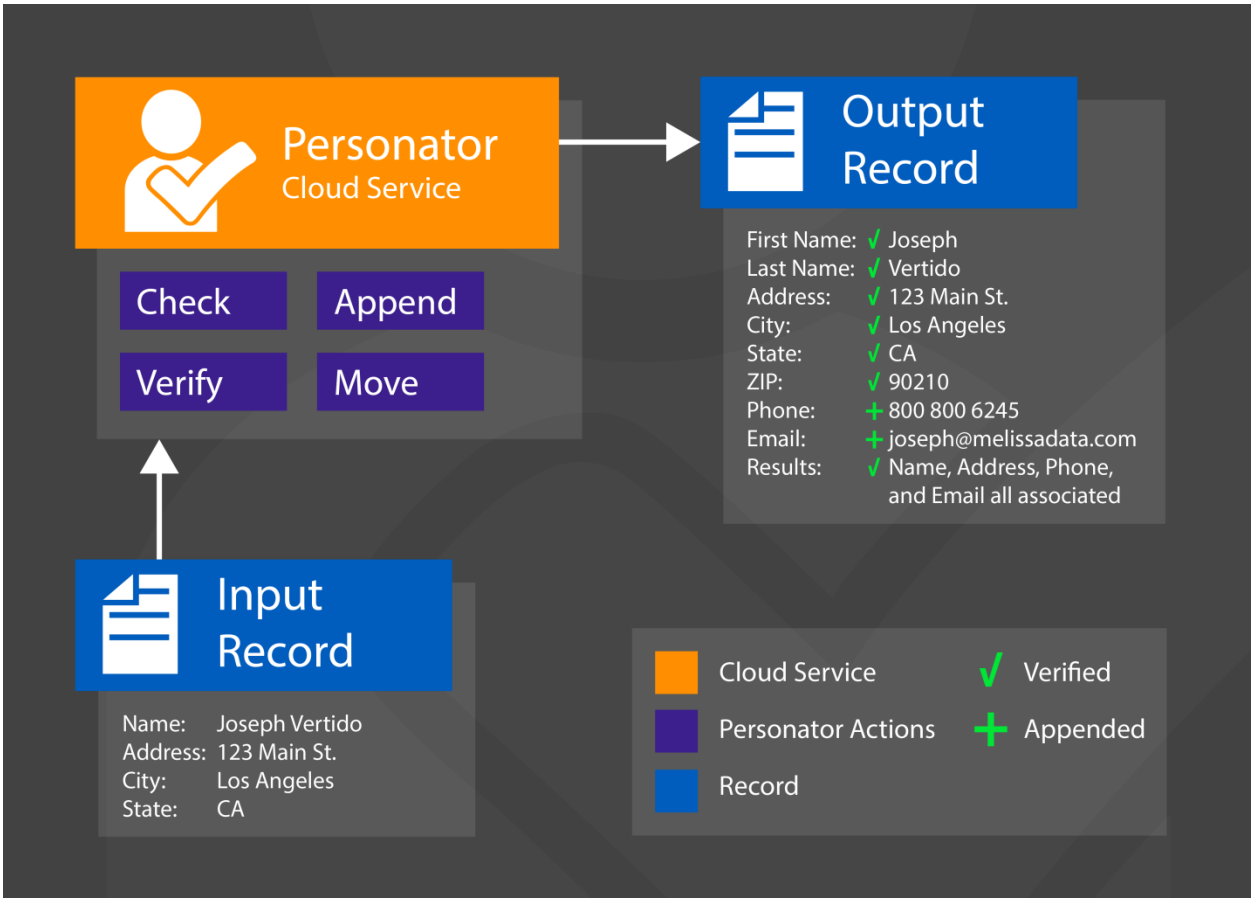
MELISSA DATA®
Your Partner in Data Quality

Personator Web Service: Programmer's Quick Start

Overview

The Personator Web Service features real-time Correction, Verification, Move and Appending of your contact records, including Name, Company Name, Address, Phone and Email. This allows for standardizing data, checking if each domain is valid, verifying relationships between domains, appending missing information, and keeping records up to date, drastically improving the quality of data.



The Personator Web Service features 4 different Actions, each of which is a separate level of subscription:

Check

The Check Action cleans and standardizes dirty contact data and also validates whether the Names are valid, Addresses are deliverable, Phones are callable and emails are existing.

Verify

The Verify Action looks at the entire contact record as a whole and determines whether the name, address, phone and email are associated with each other.

Append

The Append Action allows for populating missing or bad contact information in your data. If your data is missing a Phone Number for example, we can append that missing information based on the other contact domains provided (address, name, and email).

Move

The Move Action retrieves the latest Address for an Individual. If a person has moved or had multiple moves within the last 12 years, Personator will be able to provide the latest address from any of the previous addresses.

WEB SERVICE STRUCTURE

Request Hierarchy	Response Hierarchy
Request Actions Columns CustomerID Options TransmissionReference Records RequestRecord AddressLine1 AddressLine2 City CompanyName … RequestRecord … RequestRecord …	Response TotalRecords TransmissionReference TransmissionResults Records ResponseRecord AddressExtras AddressKey AddressLine1 AddressLine2 City … ResponseRecord … Responserecord …

MELISSA DATA®
Your Partner in Data Quality

PROPER IMPLEMENTATION

STEP1: SET YOUR ACTIONS

Request
- **Actions**
- Columns
- CustomerID
- Options
- TransmissionReference
- Records
  - RequestRecord
    - AddressLine1
    - AddressLine2
    - City
    - CompanyName
    - …

Select one or more of the following in the Actions Input, delimited by a comma

Action	Description
Check	Enables the Correction and Verification of Addresses, Names, Phones and Emails
Verify	Verifies if the Address, Names, Phones and Emails are associated with each other. These are identified through the returned Result Codes.
Append	Enables appending of the Address, Name, Phone and Email
Move	Retrieves the latest address of an individual

Eg. <Actions>**Check,Append,Move**</Actions>

STEP 2: SET YOUR COLUMNS

Request
- Actions
- **Columns**
- CustomerID
- Options
- TransmissionReference
- Records
  - RequestRecord
    - AddressLine1
    - AddressLine2
    - City
    - CompanyName
    - …

MELISSA DATA®
Your Partner in Data Quality

Select one or more of the following in the Columns Input, delimited by a comma.

You can either select the Group Name or Individual Column Names.

Eg.

Individual Columns (Static):

<Columns>**NameFirst,NameLast,Suite,Plus4**</Columns>

Group Columns (Dynamic):

<Columns>**GrpGeoCode**</Columns>

IMPORTANT: Select Individual Columns in order to maintain a static output structure. Selecting Group Columns will dynamically include new output columns added to the Web Service in the Response Structure. If your client is made to handle the dynamic addition of new columns, then use Group Names. Otherwise, use Individual Column Names to ensure a consistent static output.

IMPORTANT: Please take note that Personator is a continuously growing product. New Columns are expected to be added in the future. When consuming the Web Service in SOAP, please make sure that the SOAP libraries used to generate proxy classes are able to handle dynamic additions of new columns in the WSDL without re-referencing or re-consuming the SOAP Endpoint.

Selecting columns or groups in the Columns Input will result in that column or group of columns to appear in the Response Record.

By default, Personator will always return the following Properties in the Response:

Column
CompanyName
RecordExtras
Reserved
NameFull
AddressLine1
AddressLine2
City
State
PostalCode
AddressKey
AddressExtras
EmailAddress
PhoneNumber

MELISSA DATA®
Your Partner in Data Quality

The following are the Individual Column Names and their Groups

Individual Column Name	Group Name
Suite	(No default group)
PrivateMailBox	
Plus4	
Gender	GrpNameDetails
Gender2	
Salutation	
NamePrefix	
NameFirst	
NameMiddle	
NameLast	
NameSuffix	
NamePrefix2	
NameFirst2	
NameMiddle2	
NameLast2	
NameSuffix2	
CityAbbreviation	GrpAddressDetails
StateName	
AddressTypeCode	
CountryCode	
CountryName	
DeliveryIndicator	
UrbanizationName	
CarrierRoute	
DeliveryPointCode	
DeliveryPointCheckDigit	
UTC	
AddressHouseNumber	GrpParsedAddress
AddressPreDirection	
AddressStreetName	
AddressStreetSuffix	
AddressPostDirection	
AddressSuiteName	
AddressSuiteNumber	
AddressPrivateMailboxName	

AddressPrivateMailboxRange	
AddressRouteService	
AddressLockBox	
AddressDeliveryInstallation	
Latitude	GrpGeocode
Longitude	
CountyName	GrpCensus
CountyFIPS	
CensusTract	
CensusBlock	
PlaceCode	
PlaceName	
CBSACode	
CBSATitle	
CBSALevel	
CBSADivisionCode	
CBSADivisionTitle	
CBSADivisionLevel	
CongressionalDistrict	
DomainName	GrpParsedEmail
MailboxName	
TopLevelDomain	
AreaCode	GrpParsedPhone
NewAreaCode	
PhonePrefix	
PhoneSuffix	
PhoneExtension	
DateOfBirth	GrpDemographicBasic
HouseholdIncome	
LengthOfResidence	
PresenceOfChildren	
MaritalStatus	
DateOfDeath	
DemographicsGender	
OwnRent	(No default group)
Occupation	

MELISSA DATA®
Your Partner in Data Quality

STEP 3: SET YOUR CUSTOMER ID

Request
- Actions
- Columns
- **CustomerID**
- Options
- TransmissionReference
- Records
  - RequestRecord
    - AddressLine1
    - AddressLine2
    - City
    - CompanyName
    - …

Eg. <CustomerID>**123456789**</CustomerID>

STEP 4: SET YOUR OPTIONS

Request
- Actions
- Columns
- CustomerID
- **Options**
- TransmissionReference
- Records
  - RequestRecord
    - AddressLine1
    - AddressLine2
    - City
    - CompanyName
    - …

Options are set using a colon:

Option1 : Setting

Multiple options are delimited by semicolons:

Option1 : Setting ; Option2 : Setting ; Option3 : Setting

Eg. <Options>**CentricHint:Auto;Append:Always**</Options>

STEP 5: INPUT YOUR RECORD/S

Request
- Actions
- Columns
- CustomerID
- Options
- TransmissionReference
- Records
  - **RequestRecord**
    - **AddressLine1**
    - **AddressLine2**
    - **City**
    - **CompanyName**
    - **Country**
    - **EmailAddress**
    - **FirstName**
    - **FreeForm**
    - **FullName**
    - **LastLine**
    - **LastName**
    - **PhoneNumber**
    - **PostalCode**
    - **RecordID**
    - **State**
  - **RequestRecord**
    - **…**
  - **RequestRecord**
    - **…**

The SOAP/XML/JSON protocols allow for sending up to 100 records in a single request. The REST protocol allows for sending only 1 record at a time.

You can submit records as either individual components or as a single line input using FreeForm.

Individual Components Input

Use the following Inputs for Individual Address, Name, Company, Phone and Email components:

Input Property	Description
AddressLine1	Street address. Suites can be included at the end of AddressLine1.
AddressLine2	Street address. This is the continuation of AddressLine1. Suites can also be included in AddressLine2.
City	City
CompanyName	Company
Country	Country

EmailAddress	Email Address
FirstName	First Name
FullName	Full Name
LastLine	City + State + Zip in a single line
LastName	Last Name
PhoneNumber	Phone Number
PostalCode	Zip/Postal Code
RecordID	Record Idenitifier
State	State

Free Form Input

Use the FreeForm Input if your data contains unparsed components in a single string:

Input Property	Description
FreeForm	Address, Phone, Email, Name, and/or Company as a single string Input.

Eg. "22383 Avenida Empresa, 92688, joseph@melissadata.com, 8008006245"

WEB SERVICE RESPONSE

Fields Returned by the Service

Output	Description
TotalRecords	Total number of <ResponseRecord> returned. Maximum of 100.
TransmissionReference	Echoes the Transmission Reference set in the Input
TransmissionResults	Returns GE** or SE01 for Internal Web Service Exceptions or Transmission Errors
Version	Personator Version Number
Records	The <Records> node can contain an array of multiple <ResponseRecords> up to a maximum of 100.
ResponseRecords	Contains the set of default output columns plus all other Individual Columns or Groups specified in the <Columns> Input.

MELISSA DATA®
Your Partner in Data Quality

License String

You should have been provided an encrypted and unique license string or Customer ID from Melissa Data. This is necessary for including with each request to the Personator web service. This value should be put into the CustomerID element in each web service request.

Each action is a separate subscription level and will need to be activated in your account by your representative accordingly.

If you do not have a license string, please contact your Melissa Data sales representative at 1-800-MELISSA (1-800-635-4772).

Sample Request/Response

Sample REST Requests (INSERT UNIQUE CUSTOMER ID)

1. https://personator.melissadata.net/v3/WEB/ContactVerify/doContactVerify?t=&format=json&id=12345678&act=Check,Verify,Append&cols=&opt=&comp=Melissa+Data&a1=22382+Avenida+Empresa&city=Rancho+Santa+Margarita&state=CA&postal=92688
2. https://personator.melissadata.net/v3/WEB/ContactVerify/doContactVerify?t=&format=xml&id=12345678&act=Check,Verify,Append&cols=&opt=&comp=Melissa+Data&a1=22382+Avenida+Empresa&city=Rancho+Santa+Margarita&state=CA&postal=92688

Sample JSON Response


```
{
  "Records":[
    {
      "AddressExtras":" ",
      "AddressKey":"92688211282",
      "AddressLine1":"22382 Avenida Empresa",
      "AddressLine2":" ",
      "City":"Rancho Santa Margarita",
      "CompanyName":"Melissa Data",
      "EmailAddress":" ",
      "NameFull":"",
      "PhoneNumber":"9495895208 ",
      "PostalCode":"92688-2112",
      "RecordExtras":" ",
```

```
        "RecordID":"1",
        "Reserved":" ",
        "Results":"AS01,DA30,PS01,PS08,PS11,VR04,VR07",
        "State":"CA"
      }
   ],
   "TotalRecords":"1",
   "TransmissionReference":" ",
   "TransmissionResults":" ",
   "Version":"3.3.3"
}
```

Sample XML Response


```
<Response xmlns = "http://schemas.datacontract.org/2004/07/WcfServiceMD.mdContactVerify" xmlns:i =
"http://www.w3.org/2001/XMLSchema-instance">
    <Records>
        <ResponseRecord>
            <AddressExtras></AddressExtras>
            <AddressKey>92688211282</AddressKey>
            <AddressLine1>22382 Avenida Empresa</AddressLine1>
            <AddressLine2></AddressLine2>
            <City>Rancho Santa Margarita</City>
            <CompanyName>Melissa Data</CompanyName>
            <EmailAddress></EmailAddress>
            <NameFull/>
            <PhoneNumber>9495895208</PhoneNumber>
            <PostalCode>92688-2112</PostalCode>
            <RecordExtras></RecordExtras>
            <RecordID>1</RecordID>
            <Reserved></Reserved>
            <Results>AS01,DA30,PS01,PS08,PS11,VR04,VR07</Results>
            <State>CA</State>
        </ResponseRecord>
    </Records>
    <TotalRecords>1</TotalRecords>
    <TransmissionReference></TransmissionReference>
    <TransmissionResults></TransmissionResults>
    <Version>3.3.3</Version>
</Response>
```

Single vs Batch

Melissa Data cloud services are capable of both single record real-time processing and batch processing. The difference is simply in the number of records sent in each request. Melissa Data cloud services take an array of records. This array can contain a single record or 100 records. For a real-time process like a Web form entry or a call center application, send in a request with one record. For a batch processing scenario like a database, send requests of up to 100 records until all the records are processed. Note: Make sure each record in the request has a unique Record ID.

Sample Batch XML Response

```
<Response xmlns = "http://schemas.datacontract.org/2004/07/WcfServiceMD.mdContactVerify" xmlns:i =
"http://www.w3.org/2001/XMLSchema-instance">
    <Records>
        <ResponseRecord>
            <AddressExtras></AddressExtras>
            <AddressKey>92688211282</AddressKey>
            <AddressLine1>22382 Avenida Empresa</AddressLine1>
            …
            <RecordID>1</RecordID>
        </ResponseRecord>
        <ResponseRecord>
            <AddressExtras></AddressExtras>
            <AddressKey>92688211282</AddressKey>
            <AddressLine1>22382 Avenida Empresa</AddressLine1>
            …
            <RecordID>2</RecordID>
        </ResponseRecord>
        …
        …
        <ResponseRecord>
            <AddressExtras></AddressExtras>
            <AddressKey>92688211282</AddressKey>
            <AddressLine1>22382 Avenida Empresa</AddressLine1>
            …
            <RecordID>100</RecordID>
        </ResponseRecord>

    </Records>
    <TotalRecords>100</TotalRecords>
    <TransmissionReference></TransmissionReference>
    <TransmissionResults></TransmissionResults>
    <Version>3.3.3</Version>
</Response>
```

MELISSA DATA®
Your Partner in Data Quality

Personator Web Service URLs [Cloud Service Endpoint URLs]

Choosing a Web Service Protocol

The Melissa Data Personator Web Service supports REST, JSON, XML, and SOAP. For the undecided here are some Pros and Cons of one over the other.

REST
Pros: REST is lightweight, and relies upon HTTP to do its work. If you don't need a strict API definition, this is the way to go. REST is also format-agnostic so you can use XML or JSON as responses.
Cons: REST can only be used for the sending of single records and doesn't support strict contracts or more involved security. The Response is an XML or JSON document.

XML
Pros: XML allows recordset structures of more than one record at a time and has very good support with most languages and browsers. Supports namespaces
Cons:
Developers need to use tools to serialize/deserialize the XML structure.

JSON
Pros: JSON relies on simple object serialization based on Javascript's object initializers. It is very simple to use with Javascript and easily parsed and understood by developers.
Cons: No support for formal definitions. No namespace support. Not much support in Web Service clients with some platforms.

SOAP
Pros: SOAP (using a WSDL) is a heavy-weight XML standard that is centered around document passing. The advantage with this is that your requests and responses can be very well structured.
Cons: The downside SOAP documents are very verbose, and hard to consume without a SOAP toolkit and generally carry more overhead.

Basic Order of Operations (Pseudo Code)

1. Choose SOAP, XML, JSON, or the RESTFul service
2. Create an instance of the request object.
3. Set your Actions
4. Set additional Columns you want the service to return
5. Populate the request element CustomerID with your Product License
6. Set the options for the web service
7. Add input email addresses to the <Records> array with anywhere from 1 to 100 <RequestRecord> items. (SOAP, XML)
8. Call the method and pass in the request to the service using the SOAP endpoint for SOAP request and the WEB endpoint for XML or JSON requests.
9. Examine and parse the response from the reply object back from the service.
10. Interpret the results.

MELISSA DATA®

Your Partner in Data Quality

Interpreting Results

Melissa Data's Personator Service uses Result Codes to determine the status of a record.

The Melissa Data Cloud Services use the following Results conventions:

Check Result Codes:

1. CLOUD SERVICE ERRORS: SExx
2. CLOUD TRANSMISSION ERRORS: GExx
3. ADDRESS STATUS CODES: ESxx
4. ADDRESS ERROR CODES: Eexx
5. NAME STATUS CODES: NSxx
6. NAME ERROR CODES: NExx
7. PHONE STATUS CODES: PSxx
8. PHONE ERROR CODES: PExx
9. EMAIL STATUS CODES: ESxx
10. EMAIL ERROR CODES: Eexx
11. GEOCODER STATUS CODES: GSxx
12. GEOCODER ERROR CODES: Gexx

The following are some of the commonly used Result Codes to indicate Good Data

Output	Description
Deliverable Address	AS01, AS02 or AS03
Successful Name Parsing	NS01
10-Digit Validated Phone	PS01
Domain Validated Email	ES01
Rooftop Level GeoCode	GS05 or GS06
9-Digit Zip Centroid GeoCode	GS01

Error Codes indicate problems in the data. For example, EE01-EE04 Results Codes specifies exactly what was wrong with the Email Address.

Verify Result Codes

Verify Codes are indicated by VRxx or VSxx.

VRxx is an indicator for a match. For Example, VR01 means that the Person's Name and Address Matched.

VSxx codes may also be included in the list of results. VSxx codes indicate either an error or other additional statuses from the Verification Process. For example, a VS01 indicates that the information was verified to a historical address and a VS00 means that the address was not found in the Personator Verification Database.

Append Result codes

Append Codes are indicated by DAxx.

The appearance of any DAxx code is an indicator that data was appended. For example, a DA40 means that an Email Address has been appended in the record response.

Move Result Codes

Moves are indicated by the AS12 Result Code

Sample Code

Fully working examples are available on the wiki pages:

Click here to go to the Personator Web Service Wiki Page

Wiki Page

A product support Wiki is available for your convenience. In the wiki you will find documentation about the service in more detail.

Click here to go to the Personator Web Service Wiki Page.

Misc. Considerations

Firewall

If you are behind a firewall, you may need to allow specific IP addresses access in order to communicate with the service. For a full list of IP Addresses, see IP Address Information.

Result Codes

The service returns a series of results codes to tell you of the status of your Address, Name, Phone, Email, and any changes or errors found during the verification process.

For a full list of the result codes returned by the Personator Web Service, see Returned Result Codes.