



# SIMWOOD

STRAIGHT-TALKING, **FORWARD-THINKING**, TELECOMS

## Customer API | v3.11.4

Updated 2018-05-02

The latest version of this document can always be found at  
<http://mirror.simwood.com/pdfs/APIv3.pdf>

# Simwood Customer API

<b>Overview</b>	<b>3</b>
<b>Document Conventions</b>	<b>3</b>
<b>Technical</b>	<b>4</b>
<i>Architecture</i>	5
<b>Basic Operations</b>	<b>6</b>
<i>Basic GET Requests</i>	6
<i>JSON Output Format</i>	6
<i>Authenticating Requests</i>	7
<i>PUT and DELETE requests</i>	8
<i>POST requests / Reports</i>	9
<b>API Endpoints</b>	<b>11</b>
<b>Tools</b>	<b>12</b>
<b>Account Management</b>	<b>13</b>
<i>Credit Account Management</i>	14
<i>Prepay Account Management</i>	15
<i>Termination Rate Functions</i>	17
<i>General Accounting Reports / CDRs</i>	19
<i>Summary Reports (Instant)</i>	20
<i>Event Notifications</i>	21
<b>Voice Termination</b>	<b>22</b>
<i>Account Limits (and Dynamic Channel Limits)</i>	22
<i>Adjusting your Channel Allocation</i>	23
<i>Channel Statistics</i>	24
<i>Real-Time Calls in Progress</i>	25
<i>Voice CDRs (Inline Response)</i>	26
<i>Rejected Calls</i>	27
<b>Outbound Voice Trunks</b>	<b>28</b>
<i>Trunk Management</i>	28
<i>Trunk Balances</i>	30
<i>Per-Trunk Realtime Call Information</i>	31
<i>Trunk IP Functions</i>	32
<i>Outbound Destination ACLs</i>	33
<i>IDA Outbound Management</i>	34
<b>Inbound Numbering</b>	<b>35</b>
<i>Number Allocation</i>	35
<i>Number Routing Configuration</i>	37
<i>Inbound Trunk Configuration</i>	50
<i>Mobile Number Inbound SMS Configuration</i>	51
<i>Number Lookup</i>	53
<b>Number Porting</b>	<b>54</b>
<i>Geographic Number Porting ("GNP")</i>	55
<i>Mobile Number Porting ("MNP")</i>	60
<b>Fax and SMS Messaging</b>	<b>62</b>
<i>Inbound Fax Retrieval</i>	62
<i>Outbound SMS</i>	63
<i>Outbound FAX</i>	64
<b>HTTP POST (Inbound Events)</b>	<b>65</b>
<i>Received Fax</i>	65
<i>Received SMS</i>	66
<i>Received SMS (Beta)</i>	67
<i>Outbound Fax Reports</i>	68
<i>Outbound SMS Delivery Reports (DLRs)</i>	69
<b>Change Log / Document History</b>	<b>70</b>

# Overview

## What is it?

The Simwood Customer API ("Application Programming Interface") is a way for your own back-office systems and web sites to seamlessly integrate with Simwood and manage your wholesale telephony account and services.

## What can I do with it?

The API is the preferred way to configure your Simwood services. Everything in the portal can also be configured via the API (and in some cases the API offers some additional functionality)

Our portal is based on the API, and everything you can do in the portal you can do through the API in your own code. We've deliberately made it this way. The only exception to this is the authentication elements of the portal, as using your API key to log in on the web would be cumbersome.

# Document Conventions

Some features of the API are marked with additional labels, explained below;

**BETA**

This is a new feature or addition to the API or underlying stack which should be considered in **BETA**. It has been tested internally before being made available but it's possible bugs may still be present and we would welcome your feedback should you find any unexpected behaviour.

**NB** The parameter format of **BETA** endpoints can change at any time and without notice. Whilst we endeavour to keep this to a minimum we strongly recommend unattended scripts do not rely on BETA functions of the API or you test for the expected response format.

\$

A single \$ symbol indicates that calls to this API endpoint will result in a one-off charge in accordance with our standard rates which can be found at <https://simwood.com/rates> (e.g. Porting Fees) or within the API itself (e.g. Gold Number activation charges)

\$..

A \$.. symbol indicates that calls to this API endpoint can result in the creation or alteration of a recurring charge. e.g. Number Rental.

The charges applicable to these will likely depend on your account type, for more information consult your account documentation, published price lists, or contact the Operations Desk.

# Technical

## How does this differ to previous API versions?

In brief;

Version 1 was SOAP based and arguably overly complicated for most customer requirements.

Version 2 was more XML-RPC based.

Version 3, this version, is close to a REST-based API, notably;

- Every 'resource' has a static URL.
- Multiple HTTP methods can be used against a resource where appropriate i.e. GET, PUT, POST, DELETE
- Authentication is Basic Authentication, secured by HTTPS (SSL/TLS).
- Error codes are mapped to HTTP errors and delivered as both HTTP response headers and body text or a body JSON response.
- Output from the API is in JSON format.
- Input is generally by request parameters or JSON objects (for PUT and POST requests)

There are also some fundamental changes to the architecture to ensure one user cannot impact others. This is a combination of caching, automatic request rate limiting, duplicate request elimination, and out of band reporting.

Whereas previously all queries were responded to real-time as requested, some are now considered 'reports' which are fed off to a cluster of report servers operating against a cluster of database servers. This enables the client to return immediately and either poll for results, or be notified when results are available.

More often than not this is immediate but a customer requesting a year's CDRs every second cannot cause disruption! Of course, trivial queries are still returned immediately.

## What about non-JSON responses?

Our API historically supported multiple response formats; PHP Serialized, XML, Plain Text and the ubiquitous JSON. Of these JSON has been the most popular, with very few customers electing to use the other formats.

As we've built upon our API, and added more advanced number routing, and other configuration options, that are designed to be expressed as a JSON object we've deprecated the other format options.

Any endpoints that previously responded in other formats will continue to do so, to ensure we don't break any existing customer implementations, and the "JSON-only" approach will only be used for new API endpoints.

**Future versions (beyond v3) of the API will be JSON only.**

## Architecture

Some of our customers follow how we do things with interest - this page is for them. If you don't care, you can safely skip it.

The best way to keep up to date with the latest developments behind the scenes is via our blog; <http://blog.simwood.com/>

### Web

The v3 API, like its predecessors, is predominantly written in PHP served by nginx. nginx takes care of basic HTTPS functions but all header generation is done in code. This way the API can be RESTful in its responses. There is no state maintained in the web-app.

### Databases

As before, many queries are made against MySQL although we have dedicated servers for the API's use. These are built for queries and are slaves of masters. We can build additional query servers sideways as required. Writes are, of course, made against the masters. Increasingly, API requests are served directly from our underlying REDIS ram-based data stores where possible.

### Middle-ware

This API introduces multiple layers in between the above for performance and scale:

#### Beanstalkd Queues

We use Beanstalkd a lot for internal messaging. In this API where a request is non-trivial and involves other processes, we'll simply queue a job.

#### REDIS

REDIS replaced memcached as our caching data store, and is used for much more as it supports a number of advanced data structures facilitating vastly improved monitoring and real-time information.

See <http://blog.simwood.com/2013/06/real-time-big-data/> for more information.

#### App Servers

We have numerous daemons running monitoring specific queues (pipes in beanstalkd language). One of the beautiful features of beanstalkd is that it is blocking, i.e. 10 daemons can watch the same pipe and code will block waiting for a response.

One, and one only, will be given the job when there is one. This avoids polling - meaning this is extremely efficient and very scalable as we can just spin up as many daemons as required. Further, release of a job to a daemon is sub-millisecond giving the queueing the performance of direct requests. Most complicated jobs involve multiple daemons working through a sequence of queues.

# Basic Operations

The Simwood API can be found at the following URI;

<https://api.simwood.com/v3/>

**PLEASE NOTE THAT TLS v1.1 OR HIGHER IS MANDATORY FOR ALL REQUESTS**

## Basic GET Requests

A number of end-points in the API can be accessed with a simple **GET** request and will therefore work in a browser. Some do not require authentication and are a good place to start and to test connectivity to the API.

Two simple examples are:

<https://api.simwood.com/v3/tools/myip> - will return your IP address as seen by the API  
<https://api.simwood.com/v3/tools/time> - will return a timestamp from the API

## JSON Output Format

You'll see that in both above cases the output is JSON, e.g:

```
{"timestamp":1388748052,"rfc":"Fri, 03 Jan 2014 11:20:52 +0000"}
```

JSON ("JavaScript Object Notation") is used as it's a lightweight format designed for exchange of data structures which has an additional advantage of being relatively human readable. All modern languages such as PHP, ASP, .net, Node, Perl etc can handle JSON structures and convert to/from their native objects or arrays easily making it the ideal choice for REST APIs.

## "Pretty" JSON

JSON is designed to be machine-readable and as such is sent with minimal whitespace by default however when developing it is often useful to see the output in a more human-friendly format, you can achieve this simply by appending "?pretty=true" to any endpoint e.g <https://api.simwood.com/v3/tools/time?pretty=true> would look like this;

```
{  
  "timestamp":1388748052,  
  "rfc":"Fri, 03 Jan 2014 11:20:52 +0000"  
}
```

## Authenticating Requests

Most commands are linked to an account and therefore require authentication.

We use standard Basic Authentication, i.e. your client makes a request, we respond with a 401 response code, your client replies with the API username and password included.

Your API username and password were provided at the time of creating your Simwood account, if you do not have these please contact our support team and we'll be happy to provide you with them.

Note that, at present;

- Your API username and password is *not* the same as your portal login details
- Your API password is designed to be used programatically, and is typically not memorable
- Your API details allow full access to your account, and must be protected accordingly.
- You cannot, currently, change your API username or password - although our support team are happy to do so if required (e.g. if your details are lost or you have reason to believe they are compromised)

If you pull up an example URL in your browser it'll do this for you. For your code, different development languages will tackle this in different ways but most will 'just deal with it' automatically for you.

cURL for example, will just take the username and password as parameters, e.g.:

```
curl --user name:password https://api.simwood.com/v3/...
```

PHP's cURL implementation is very similar in that you'd set CURLOPT\_USERPWD with [curl\\_setopt](#).

An example authenticated GET request is:

```
https://api.simwood.com/v3/accounts/{ACCOUNT}/prepay/balance
```

*This example URL will not work directly in a web browser as the {ACCOUNT} placeholder in the above will need to be replaced with account ID (typically a six digit number).*

Which would return the following

```
[{"balance": "12.32", "currency": "GBP"}]
```

Adding **?pretty=true** to the end of the URL would give you the same information in the following format;

```
[
  {
    "balance": "12.32",
    "currency": "GBP"
  }
]
```

Both are treated equally by a JSON parser, and are syntactically valid, however the 'pretty-printed' version may be useful for debugging.

## PUT and DELETE requests

In the spirit of being REST-ful, certain URLs can be acted upon with different methods. The URL does not change, only the HTTP method used against it. One example of this is number configuration, for example;

**https://api.simwood.com/v3/numbers/{ACCOUNT}/allocated/{NUMBER}**

*Again, this will not work as {ACCOUNT} and {NUMBER} need replacing with real data. {NUMBER} is the e164 representation of the number, e.g. 442031234567.*

**GET** is the default method, accessing the above end-point with a **GET** will return the current configuration of that number. **PUT** will create the end-point, i.e. allocate the number to your account if it does not exist already. **DELETE** will remove the end-point, i.e. de-configure and remove the number from your account.

**PUT** is context sensitive and specific uses will be described in greater detail within different commands. Briefly though, whereas:

`https://api.simwood.com/v3/numbers/{ACCOUNT}/allocated/{NUMBER}`

will create a number end-point, i.e. allocate the number but

`https://api.simwood.com/v3/numbers/{ACCOUNT}/allocated/{NUMBER}/config`

will configure a number, i.e. write the sent config to it.

**DELETE** works more uniformly but only at the level it operates, e.g.

`https://api.simwood.com/v3/numbers/{ACCOUNT}/allocated/{NUMBER}`

will de-configure and remove a number but

`https://api.simwood.com/v3/numbers/{ACCOUNT}/allocated/{NUMBER}/config`

will just remove the configuration, leaving the number on your account

Setting the method is language specific and **PUT** and **DELETE** cannot be easily replicated in a browser. Almost all environments/browsers will default to **GET** so the chances are if it doesn't work as you expect you are using the **GET** method.

For development languages which do not permit all HTTP methods to be used you can pass a hidden parameter named `_method` in an HTTP POST or GET and set the method to use in there. This will override the actual HTTP method used. For example, in an HTTP form you may use:

```
<input type="hidden" name="_method" id="_method" value="put" />
```

**All modern languages allow HTTP methods to be used correctly, and we would strongly encourage this in preference to setting a `_method` parameter.**



## POST requests / Reports

**POST** is very similar to PUT but is used where one is not updating a specific resource, i.e. requesting reports, or when creating a new resource

For example, to request a summary of calls and charges for the current day the end-point would be:

```
https://api.simwood.com/v3/accounts/{ACCOUNT}/reports/voice/summary/day/in
```

You would use POST for this request as whilst you are sending a request to the server and the end-point is fixed, what that end-point represents and what you get back are variable. POST is used for all report type requests.

The output from such a POST request (which requests a report) will typically be small, i.e.:

```
{ "mode": "vsind", "date": "2012-01-28", "type": "voice_summary", "account": "ACCOUNT", "format": "json", "hash": "4e591630fedf4aa149db9874fb33fe23", "link": "\/v3\/files\/ACCOUNT\/4e591630fedf4aa149db9874fb33fe23" }
```

You will note that this is not in fact a summary of today's charges. It is a hash that uniquely identifies that report and a link to it. As the return suggests, you can retrieve the actual results with a GET request to

```
https://api.simwood.com/v3/files/{ACCOUNT}/4e591630fedf4aa149db9874fb33fe23.
```

### But don't worry...

Whilst it is important to understand the effect a given request method will have, we are somewhat relaxed in our interpretation of them where possible. DELETE is always DELETE with no alternative but we will accept a POST instead of a PUT and a GET instead of a POST where possible. You should not assume we will, and should adhere to the RESTful way, but this flexibility is there if you wish to test behaviour in a browser for example.

### Report De-duplication

As described above, when you request a report the return will just contain a link to the results. That link will be unique to that specific report, i.e. in the above example the hash for today's summary will be the same with every request, but only every request **today**. Tomorrow will generate a different hash. If you make multiple requests for the same report before you retrieve it, duplicates will simply be ignored and the report will only be run once.

However, once a report has been run, the results await your collection within the next 5 minutes. Further requests with the same hash will continue to be ignored as long as that report exists. Once the report is retrieved it will be deleted. Only after a report has been deleted (either by retrieval or expiry), will an identical report request result in a new report.

**NB** reports expire after 5 minutes, any attempt to access the report after 5 minutes will result in a 404 error, you should therefore build your application to poll the report URL multiple times as soon as possible after submitting your request or use the callback URL detailed below.

Hashes are intended to be generated sensibly. For example, a given report for today will always give the same hash. A different report for today or the same report for tomorrow will give different hashes. Reports on different numbers will generate different hashes. Further, de-duplication will only apply where previous results have not been retrieved.

The intention here is to protect the system from the coder who wishes to request a year's CDRs every second. He can, but all requests other than the first will be ignored until he retrieves the results. This rather extreme example is actually real and was a consequence of the way the v2 API paginated results and required clients to step through them. More than a few customer implementations reached the end of the results and went into a race condition.

## Report Retrieval

As shown above, the return from a report request will give a hash and an encoded URL. The hash can be used to access the results directly, even from another system, using a GET request to, for example:

```
https://api.simwood.com/v3/files/{ACCOUNT}/4e591630fedf4aa149db9874fb33fe23
```

This URL can be polled at a sensible interval as it will simply return a 404 Not Found until the report exists. Once the report does exist, it can be downloaded only once and then will be automatically deleted.

To negate the need for polling, clients may instead wish to specify a 'callback' URL with their request. This should be POSTed as a variable called 'callback'. The response to the request will be identical as without it but on successful generation of the report we will make a POST to the URL provided. This will contain a single HTTP POST variable call 'payload' which will contain the report hash in JSON format, e.g.:

```
{"app": "reports", "id": "76e7a8102f93c636785ea8432c72e07a", "data": null}
```

The client should then GET the report as usual within a maximum of 5 minutes, after which it may expire.

## Dates in Reports

Most reports can be run for a range or at a specific date. Where omitted they will generally default to today. Dates are always expressed in MySQL format (YYYY-MM-DD hh:mm:ss), e.g. 2012-01-01 14:30:00

## File (Report Output) Handling

Unlike many APIs the Simwood API is asynchronous when requesting some complex reports to improve performance (this is discussed in more detail above) and some POST requests will result in a small response containing a hash and 'link' to a file e.g.

```
{
  "mode": "vsind",
  "date": "2012-01-28",
  "type": "voice_summary",
  "account": "{ACCOUNT}",
  "format": "json",
  "hash": "4e591630fedf4aa149db9874fb33fe23",
  "link": "/v3/files/{ACCOUNT}/4e591630fedf4aa149db9874fb33fe23"
}
```

The functions below are used to retrieve these;

### **/v3/files/{ACCOUNT}**

**GET**

Lists uncollected files on the account

```
{
  "ff98d8d8fdf9dfd178e72b6e6ba207ff" {
    "name": "ff98d8d8fdf9dfd178e72b6e6ba207ff",
    "content_type": "application/json",
    "length": 410,
    "uri": "/v3/files/ACCOUNT/ff98d8d8fdf9dfd178e72b6e6ba207ff"
  }
}
```

### **/v3/files/{ACCOUNT}/{HASH}**

**GET**

Retrieve specific file on account, where HASH is the hash returned when the report was requested. File will be deleted after retrieval.

# API Endpoints

If the above has all made sense, you should need little more than a list of the available end-points and the HTTP methods they support to get going. This follows and you'll quickly observe they are hierarchical and hopefully consistent. The method shown indicates behaviour as described earlier.

**Each endpoint is documented below in the following format;**

---

**/v3/URL**

<b>GET</b>	Explains what happens when the GET method is used
<b>POST</b>	Explains what happens when the POST method is used. Will also explain what <i>parameter</i> is used for
	<i>parameter</i> An example GET / POST or PUT parameter
	<i>anotherparam</i> Another parameter to be sent by POST

**Please Note** - Where example responses are shown in this documentation they may be reformatted to be more easily human-readable, the actual response will have escaped slashes (e.g. / replaced with \) and not include any excess white space.

**The following conventions are used in describing the URL or other parameters**

- {ACCOUNT}**                      Where a word is capitalised and enclosed by curly braces { } it must be replaced with the appropriate information e.g. {ACCOUNT} or {IP}
- [ on | off ]**                      Where two or more words are separated by the pipe character | and enclosed within square brackets [ ] these are 'either or' options.  
e.g. a url with the form /latest/[1|5|10] allows you to specify any of the following 3 URLs;  
    /latest/1  
    /latest/5  
    /latest/10  
Arbitrary values (e.g. /latest/15) are not supported
- URLs**                              The URLs are shown without the leading https://api.simwood.com/ which must be inserted before the /v3/ when making any API call.
- paramname***                      Parameters are shown in *italics*, these are passed by GET, POST or PUT in the request and do not form part of the URI (except in the case of the GET request, when they are part of the query string after the ? mark)
- paramname[]***                      Parameters with square brackets at the end are different and can be thought of as Array Parameters. These can be passed multiple times but even if only one item is being included you *must* include the [] on the end. For compatibility with some languages (e.g. PHP with Curl) an integer value can be between the square brackets e.g the following two examples are equivalent;

?param[]=apple&param[]=orange&param[]=pear  
?param[0]=apple&param[1]=orange&param[2]=pear

# Tools

The following tools are made available without authentication to help integrate with the Simwood API.

## IP Address

Your IP address, as seen by the Simwood API service

---

### **/v3/tools/myip**

<b>GET</b>	Return your external IP address, as seen by the Simwood API
------------	---

## Time

The current server timestamp

---

### **/v3/tools/time**

<b>GET</b>	Returns the current timestamp
------------	-------------------------------

## Explain

This tool is provided to help debug requests to the Simwood API. It accepts any request method (GET, PUT, POST, DELETE etc) and returns a human-readable report of the information submitted.

---

### **/v3/tools/explain**

<b>ANY</b>	Returns human-readable report of the request received
------------	---

```
Simwood API - Explain Tool
=====
Query ID: swAPI547df1c0985db
Timestamp: 2014-12-02 17:26:03
HTTP Request Method: POST
API Request Method: POST
SSL Used: Yes (OK)
== Query String =====
Your query string contained 2 elements;
  test => 1
  pretty => true
== HTTP Post Vars =====
No HTTP request body request elements
== JSON Request =====
Valid JSON (Decoded Below)
-----
{
  "date_start": "2014-11-25 00:00:00",
  "date_end": "2014-11-26 23:59:00",
  "limit": 50,
  "start": 0,
  "pretty": true,
  "debug": false,
  "filter": {
    "trunk": "920000-L001"
  }
}
== Raw Request =====
{"date_start":"2014-11-25 00:00:00","date_end":"2014-11-26 23:59:00","limit":
50,"start":0,"pretty":true,"debug":false,"filter":{"trunk":"920000-L001"}}
=====
Simwood API v3 http://mirror.simwood.com/pdfs/APIv3.pdf
```

---

**NB** As this is intended to be a human-readable report for debugging purposes the format may change at any time, and without notice, and should not be relied upon.

# Account Management

## Account Type

Your account will be one of the following four types; **developer**, **startup**, **virtual\_interconnect**, or **managed\_interconnect**, each have different commercials but are functionally identical. All new accounts start off as 'Developer', we encourage you to move to 'Start-Up' for production use, and to consider 'Virtual Interconnect' or 'Managed Interconnect' as your requirements evolve.

The differences between these account types can be found in the Simwood Product Brochure in our document library at <http://mirror.simwood.com/pdfs/>

### /v3/accounts/{ACCOUNT}/type

<b>GET</b>	Get your current account type, and limitations Account type will be one of the following; <b>developer, startup, virtual_interconnect, managed_interconnect</b>			
	<pre>{   "success": true,   "data": {     "account_type": "developer",     "number_limit": 5,     "channel_limit": 5,     "min_prepay": 50   } }</pre>			
<b>PUT</b>	Upgrade or downgrade your account This option is only available to customers of account_type <b>developer</b> or <b>startup</b>			
	<table><tr><td>account_type</td><td>One of; <b>developer</b> <b>startup</b></td><td>downgrade to developer (if startup) upgrade to startup (if developer)</td></tr></table>	account_type	One of; <b>developer</b> <b>startup</b>	downgrade to developer (if startup) upgrade to startup (if developer)
account_type	One of; <b>developer</b> <b>startup</b>	downgrade to developer (if startup) upgrade to startup (if developer)		



### Charges Apply

There is a minimum commitment associated with the Start-Up package, along with other commercial differences. Please see the Simwood Product Brochure for more information

## PLEASE NOTE

You can migrate between developer and startup at any time. By changing your package type to Start-Up please ensure you are aware of the commercial obligations of this package, including the **minimum pre-payment amount** (which differs from the developer package) and the **minimum total spend** per month, which replaces the service charge of the Developer pack.

It is not possible to switch to our **Virtual Interconnect** or **Managed Interconnect** options online either via the API or Portal. Please contact us if you are interested in these packages.

If you are unsure, please contact our Operations Desk via eMail to [team@simwood.com](mailto:team@simwood.com), or call us on 0330 122 3000 to discuss your requirements further.

## Credit Account Management

The current status of all invoices on your account is available through the API.

---

### **/v3/accounts/{ACCOUNT}/credit/invoices/unpaid**

<b>GET</b>	List of unpaid invoices on account (since June 2010)
------------	--

---

### **/v3/accounts/{ACCOUNT}/credit/invoices/paid**

<b>GET</b>	List of paid invoices on account (since June 2010)
------------	--

---

### **/v3/accounts/{ACCOUNT}/credit/invoices/all**

<b>GET</b>	List of invoices on account (since June 2010)
------------	---

```
{
  "Invoices": [
    {
      "InvoiceNumber": "I3010999999",
      "Date": "2014-02-30",
      "DueDate": "2014-02-30",
      "Currency": "GBP",
      "Net": "200.00",
      "VAT": "40.00",
      "Total": "240.00",
      "AmountPaid": "240.00",
      "AmountCredited": "0.00",
      "AmountDue": "0.00"
    }
  ],
  "TotalDue": {
    "GBP": 0
  },
  "TotalOverDue": null
}
```

## PDF Copy Invoices

---

### **/v3/accounts/{ACCOUNT}/credit/invoices/{INVOICE\_NUMBER} [.pdf]**

<b>GET</b>	Get invoice INVOICE_NUMBER as a pdf file. The .pdf suffix is optional.
------------	--

## Prepay Account Management

### **/v3/accounts/{ACCOUNT}/prepay/summary**

**POST**

Requests a report of Summary of account movements.  
If the optional *from\_date* and *to\_date* parameters are not specified will default to today.

*from\_date* Start date of report (in form YYYY-MM-DD)

*to\_date* End date of report (in form YYYY-MM-DD)

### **/v3/accounts/{ACCOUNT}/prepay/prepayments/all**

**GET**

List all account pre-payments

### **/v3/accounts/{ACCOUNT}/prepay/prepayments/latest/[ 1 | 10 ]**

**GET**

List last (1) or last ten (10) pre-payments to account

### **/v3/accounts/{ACCOUNT}/prepay/transfers/all**

**GET**

Transfers between this prepay account and others.

### **/v3/accounts/{ACCOUNT}/prepay/transfers/latest/[ 1 | 10 ]**

**GET**

List last (1) or last ten (10) transfers on the prepay account

## Prepay Account Balance

The API provides tools for checking and protecting your pre-paid balance(s)

### **/v3/accounts/{ACCOUNT}/prepay/balance**

<b>GET</b>	Return balance of account. <pre>[{"balance": "2.46880", "currency": "GBP"}]</pre>
------------	--

## Low Balance Alerts

You can set an amount at which you will receive a notification (configured via the notifications, detailed below) when your balance drops below the specified amount.

### **/v3/accounts/{ACCOUNT}/prepay/balance/alert**

<b>GET</b>	Returns the current level of alert. <pre>[{"account": ACCOUNT, "alert_balance": 20}]</pre>
<b>PUT</b>	Sets the alert balance to the specified <i>alert_balance</i> <i>alert_balance</i> The balance at which an alert should be generated

## Balance Locking

We provide the ability to 'lock' a portion of your balance to make it unavailable to spend. i.e. you can specify the balance at which we'll treat your account as 'out of credit' and therefore kill calls in progress.

This is normally zero but the balance locking facility enables customers to keep large credit balances without risking the entire amount in the event that they, or a customer, suffer a compromise.

### **/v3/accounts/{ACCOUNT}/prepay/balance/locked**

<b>GET</b>	Returns the 'locked' balance, this represents a portion of your pre-paid balance that cannot be consumed - e.g. to protect against unexpected spend <pre>[{"account": ACCOUNT, "balance": 10.4688, "locked": 5, "available": 5.4688}]</pre>
<b>PUT</b>	Sets the protected amount to the specified <i>balance</i> . This amount remains in your account but cannot be spent. <i>balance</i> The amount to be protected.
<b>DELETE</b>	Remove the locked balance, allowing the full amount of your pre-paid balance to be used for calls.

**NB** *The above 'locked balance' function is provided on a 'best-efforts' basis and is not intended as a substitute for securing your own VoIP platform. You remain responsible for all calls made on your account.*



## Termination Rate Functions

### Available Tariffs

Some account types have more than one available tariff or rate deck, these can be viewed as follows;

---

#### **/v3/accounts/{ACCOUNT}/rates/tariffs**

**GET**

Request tariffs available on your account, these are shown together with the prefix required for dynamic rate deck selection (no prefix is required for calls using the default rate deck)

```
{
  "success": true,
  "data": [
    {
      "description": "Legacy GBP Platinum",
      "servicelevel": 1,
      "quality": "Platinum",
      "prefix": 999001,
      "url": "\v3\accounts\929999\rates\csv\Simwood_Platinum.csv"
    },
    {
      "description": "Legacy GBP Gold",
      "servicelevel": 2,
      "quality": "Gold",
      "prefix": 999002,
      "default": true,
      "url": "\v3\accounts\929999\rates\csv\Simwood_Gold.csv"
    }
  ]
}
```

Historic ratecards can be found in the archive; <http://mirror.simwood.com/rates/archive>

### Ratecard Downloads (CSV Format)

Our latest termination rates are also always available in CSV and Excel (XLSX) format from the portal <https://portal.simwood.com/> or in CSV format via the API;

---

#### **/v3/accounts/{ACCOUNT}/rates/csv/[default|silver|platinum|gold|name]**

**GET**

Request the latest ratecard in CSV format

**NB: The options silver, platinum and gold are only for 'legacy' or 'startup' account types with multiple rate decks. Virtual Interconnect and Managed Interconnect customers should use default or the file name provided from the above query**

Historic ratecards can be found in the archive; <http://mirror.simwood.com/rates/archive>

## Destination Lookup

**/v3/accounts/{ACCOUNT}/rate/{NUMBER}**

### GET

Returns the cost of calling the specified number in your account currency. Each rate array is keyed by the relevant deck (1 - platinum, 2 - gold, 3 silver) and shows (p)eak, (o)ffpeak, (w)eekend rates and the (c)onnection charge.

```
{
  "desc": "UK - fm3 - Mobile (T-Mobile)",
  "rates": {
    "1": { "p": "0.01200", "o": "0.01200", "w": "0.01200", "c": "0.0000" },
    "2": { "p": "0.01100", "o": "0.01100", "w": "0.01100", "c": "0.0000" }
  }
}
```

### Rate Checker ( Example )

See the rate checker on <https://www.simwood.com/> for a working example built with this API.

View the JavaScript Console for examples of the request / responses

## General Accounting Reports / CDRs

**`/v3/accounts/{ACCOUNT}/reports/voice/summary/day/[in|out]`**

**POST**

Request a summary of [incoming] or [outgoing] voice charges

*date*

Optionally specify date in YYYY-MM-DD format to request report as at *date*, otherwise defaults to current.

**`/v3/accounts/{ACCOUNT}/reports/voice/summary/day/[in|out]`**

**POST**

Request a summary of [incoming] or [outgoing] voice charges

*date*

Optionally specify date in YYYY-MM-DD format to request report as at *date*, otherwise defaults to current.

**`/v3/accounts/{ACCOUNT}/reports/voice/cdr/day`**

**POST**

Request daily CDR report

*date*

Optionally specify date in YYYY-MM-DD format to request report as at *date*, otherwise defaults to current.

**`/v3/accounts/{ACCOUNT}/reports/voice/cdr/latest/[10|100|1000|10000]`**

**POST**

Request report of last [ 10 | 100 | 1,000 | 10,000 ] Voice CDRs

**`/v3/accounts/{ACCOUNT}/reports/sms/cdr/day`**

**POST**

Request daily SMS CDR report

*date*

Optionally specify date in YYYY-MM-DD format to request report as at *date*, otherwise defaults to current.

**`/v3/accounts/{ACCOUNT}/reports/sms/cdr/latest/[10|100|1000|10000]`**

**POST**

Request report of last [ 10 | 100 | 1,000 | 10,000 ] SMS CDRs

**`/v3/accounts/{ACCOUNT}/reports/admin/cdr/day`**

**POST**

Request report of non-call related charges

*date*

Optionally specify date in YYYY-MM-DD format to request report as at *date*, otherwise defaults to current.

## Summary Reports (Instant)

The new Summary Reports are, unlike the CDRs above, not asynchronous, the response is inline.

**NB These are intended to be indicative only and are not, at this time, suitable for billing purposes.**

`/v3/accounts/{ACCOUNT}/summary/([in|out])/KEY}`

<b>GET / POST</b>	Request an inbound or outbound summary report by {KEY} (see below). If 'in' or 'out' is omitted from the URI bi-directional traffic will be shown.
	The parameters below can be provided in the query string (with the exception of the 'filter' parameter) or, preferred, as a JSON object in the request
<i>date_start</i>	Optionally specify date/time in YYYY-MM-DD HH:ii:ss format to request report from <i>date_start</i> , otherwise defaults to start of current day. <b>NB Dates must be specified in GMT</b>
<i>date_end</i>	Optionally specify date/time in YYYY-MM-DD HH:ii:ss format to request report from <i>date_end</i> , otherwise defaults to now.
<i>limit</i>	Optionally specify limit of results to return, default 9999
<i>sort</i>	Key to sort by
<i>filter</i>	Array of "search_key" => "value" e.g. {"trunk": "930000-TEST"} would generate a report only for the trunk named "930000-TEST"

### Summary Report Keys

The following keys are available for summary reports;

<b>destid</b>	Summary by Destination ID
<b>iso</b>	Summary by ISO Country Code (of destination)
<b>codec</b>	Summary by Codec
<b>tag</b>	Summary by Tag (as set with "X-simwood-tag" header)
<b>trunk</b>	Summary by Trunk

### Example Summary Report

The report below was generated based on a key of 'trunk' with a sort of 'calls';

```
[
  {
    "trunk": "930000-ACME",
    "calls": 378,
    "acd": 1.6,
    "minutes": 603.32,
    "charges": 8.32
  },
  {
    "trunk": "930000-WIDGETINC",
    "calls": 2856,
    "acd": 1.56,
    "minutes": 4458.47,
    "charges": 23.01
  }
]
```

## Event Notifications

### **/v3/accounts/{ACCOUNT}/notifications**

<b>GET</b>	List active notifications on your account
	<code>["blocked_calls","prepay_debit"]</code>

### **/v3/accounts/{ACCOUNT}/notifications/available**

<b>GET</b>	List available notification TYPES
------------	-----------------------------------

### **/v3/accounts/{ACCOUNT}/notifications/{TYPE}**

<b>GET</b>	List configured recipients for notifications of {TYPE}
<b>DELETE</b>	Delete all configured notifications of {TYPE}

### **/v3/accounts/{ACCOUNT}/notifications/{TYPE}/{METHOD}**

<b>GET</b>	Shows all configured recipients of notifications of {TYPE} using {METHOD}  METHOD is one of email, http or sms*
<b>POST</b>	Add a new notification recipient to receive notifications of {TYPE} using {METHOD}. Returns a hash corresponding to this recipient  <i>destination</i> URL, eMail address or Mobile Number (in E164 format) of the recipient.

### **/v3/accounts/{ACCOUNT}/notifications/{TYPE}/{METHOD}/{HASH}**

<b>GET</b>	Shows the information on this recipient
<b>DELETE</b>	Deletes this recipient

The {HASH} referred to above can be generated locally and is simply an md5()’d version of the notification address. This is used to avoid potential url encoding issues.

\*SMS notification requires credit balance, sent messages will be deducted from your usual credit.

### **/v3/accounts/{ACCOUNT}/notifications/history**

<b>GET</b>	Retrieve a history of recent (last 60 days) notifications on your account All parameters below are optional, by default will return all notifications for the last 60 days.  Returns a JSON Array of Objects, each Object contains a “data” attribute which contains an Object containing all variables in the original notification.  Please note if the original message contained a password it will be redacted and replaced with ##### as these are not stored.
	<i>class</i> Class of notification ( e.g. <i>trunk</i> or <i>billing</i> )
	<i>date_start</i> Start date (no more than 60 days ago)
	<i>date_end</i> End date (YYYY-MM-DD HH:ii:ss)

# Voice Termination

## Account Limits (and Dynamic Channel Limits)

**/v3/voice/{ACCOUNT}/limits**

**GET**

Shows limits in effect on an account including any Dynamic Channel Limits

```
{
  "channel_allocation": 30,
  "channel_allocation_adjustable": false,
  "limit_concurrent_in": 10,
  "limit_concurrent_out": 20,
  "limit_concurrent_out_per_number": 10,
  "limit_concurrent_out_international": 3,
  "limit_concurrent_out_hotspot": 2,
  "limit_rate_out": "10/1s",
  "limit_rate_out_international": null,
  "limit_rate_out_hotspot": "2/12h",
  "dynamic": {
    "balance": "15.52680",
    "limit_concurrent_out": 3,
    "limit_rate_out": "3/1s"
  }
}
```

### Dynamic Channel Limits

*Applies only to accounts without a dedicated channel allocation*

Any dynamic channel limits shown in the "dynamic" block take precedence over the usual account limits.

e.g. in the above example due to the low balance (£15) there is a concurrent channel limit of **3 channels** and a rate limit of **3 calls per second**.

### Channel Allocation (Inbound/Outbound)

*Applies only to accounts with a dedicated channel allocation*

For customers with a dedicated self-managed channel allocation, **channel\_allocation\_adjustable** is *true*.

You can divide this allocation between inbound and outbound as described below.

### Other Limits

These limits are set **per-account** by Simwood and are based on your traffic.

We impose these limits to protect and manage our network utilisation.

If you require more channels or a higher rate of calls per second then please contact the Operations Desk.

## Adjusting your Channel Allocation

Customers with a dedicated channel allocation can manage this allocation, splitting channels between inbound and outbound as required.

---

### **/v3/voice/{ACCOUNT}/limits**

<b>PUT</b>	Update channel limits				
	<i>JSON request</i>	<table><tr><td>limit_concurrent_in</td><td>Inbound channel limit</td></tr><tr><td>limit_concurrent_out</td><td>Outbound channel limit</td></tr></table> <p><b>NB</b> the above must, together, total the value of <b>channel_allocation</b></p> <pre>{   "limit_concurrent_in": 10,   "limit_concurrent_out": 20 }</pre>	limit_concurrent_in	Inbound channel limit	limit_concurrent_out
limit_concurrent_in	Inbound channel limit				
limit_concurrent_out	Outbound channel limit				
	<i>JSON response</i>	<pre>{   "success": true   "data": {     "limit_concurrent_in": 10,     "limit_concurrent_out": 20   } }</pre>			

## Channel Statistics

### `/v3/voice/{ACCOUNT}/channels/current`

GET

Current channel utilisation

**NB: this is returned as an array (with one member) for compatibility with the `/channels/history` function detailed below**

```
[
  {
    "datetime": "2014-02-30 12:34:35",
    "channels": 22,
    "channels_in": 10,
    "channels_out": 12
  }
]
```

### `/v3/voice/{ACCOUNT}/channels/history`

GET

Recent (around 24h) channel utilisation samples

The *channels* count shows the peak number of channels in use between the previous *datetime* timestamp and the current one.

*interval* Optional interval for samples in the following form;  
**1m** - One Minute  
**5m** - Five Minutes (Default)  
**1h** - Hourly

```
[
  {
    "datetime": "2014-02-30 09:29:52",
    "channels": 24,
    "channels_in": 10,
    "channels_out": 14
  },
  {
    "datetime": "2014-02-30 09:28:52",
    "channels": 22,
    "channels_in": 10,
    "channels_out": 12
  },
  {
    "datetime": "2014-02-30 09:27:52",
    "channels": 22,
    "channels_in": 10,
    "channels_out": 12
  },
  {
    "datetime": "2014-02-30 09:26:52",
    "channels": 20,
    "channels_in": 10,
    "channels_out": 10
  },
  {
    "datetime": "2014-02-30 09:25:52",
    "channels": 17,
    "channels_in": 9,
    "channels_out": 8
  },
  ...
]
```



## Real-Time Calls in Progress

**/v3/voice/{ACCOUNT}/inprogress/current**

**GET**

Number and value of calls in progress relative to account balance. Useful for fraud monitoring.

```
{
  "datetime": "2014-02-30 12:34:35",
  "total": 1.164,
  "callcount": 107,
  "balance": 24.406,
  "percent_available": 0.25,
  "approx_seconds_remaining": 56838,
  "calls": {
    "1739": {
      "location": "UK - Fixed",
      "country": "GB",
      "total": 0.85,
      "callcount": 85
    },
    "2303": {
      "location": "UK - Mobile - T-Mobile",
      "country": "GB",
      "total": 0.244,
      "callcount": 19
    },
    "277": {
      "location": "Mexico - Mexico City",
      "country": "MX",
      "total": 0.04,
      "callcount": 2
    },
    "2761": {
      "location": "Spain - Mobile - Telefonica",
      "country": "ES",
      "total": 0.02,
      "callcount": 1
    }
  },
  "countries": {
    "MX": {
      "total": 0.04,
      "callcount": 2
    },
    "ES": {
      "total": 0.02,
      "callcount": 1
    },
    "GB": {
      "total": 1.094,
      "callcount": 104
    }
  }
}
```

The above example shows many calls in progress to the UK, along with two to Mexico (MX) and one to Spain (ES) - calls are grouped both by destination (keyed by a unique numeric identifier for that destination) in the "calls" group and by country in "countries" - this makes it simple to alert, for example, if there are any more than a predefined number of calls to any country you don't expect.

See <http://blog.simwood.com/2014/01/2-quick-scripts-to-help-you-sleep-easier/> for some examples of how this data can be used to help protect you against fraud and monitor your VoIP traffic.

## Voice CDRs (Inline Response)

In addition to the CDR Reports (above) you can retrieve any CDRs from the last three months with a simple inline response (rather than polling for a report)

### `/v3/voice/{ACCOUNT}/cdr`

<b>GET / POST</b>	Both GET and POST are supported. The request can either be made as a GET query string or JSON POST body. At present, the <i>filter</i> attribute is only supported where requested by POST with a JSON body.
<i>date_start</i>	Optionally specify date/time in YYYY-MM-DD HH:ii:ss format to request report from <i>date_start</i> , otherwise defaults to start of current day.
<b>NB Dates must be specified in GMT</b>	
<i>date_end</i>	Optionally specify date/time in YYYY-MM-DD HH:ii:ss format to request report from <i>date_end</i> , otherwise defaults to now.
<i>limit</i>	Optionally specify limit of results to return, default 500
<i>start</i>	Optional offset to start from (for pagination)
<i>filter</i>	Object of "search_key" => "value" e.g. {"trunk": "930000-TEST"} would return CDRs only for the trunk named "930000-TEST".
	At present you can filter only on the following values, more will be added in the future; <b>from, to, toISO, trunk, tag</b>

### `/v3/voice/{ACCOUNT}/cdr/{YYYY-MM-DD}`

<b>GET / POST</b>	As above, a shorthand way of retrieving CDRs for a particular day.
<i>limit</i>	Optionally specify limit of results to return, default 500
<i>start</i>	Optional offset to start from (for pagination)
<i>filter</i>	<i>As above</i>

### `/v3/voice/{ACCOUNT}/cdr/{YYYY-MM-DD}/{REFERENCE}`

<b>GET</b>	Retrieve more information, where available, on a specific call using the <b>reference</b> value returned from the above CDRs (or the CDR Reporting)
------------	---

This function is temporarily unavailable as of version v3.10.5

## Rejected Calls

When calls are rejected by Simwood for any reason (e.g. exceeding a channel limit, invalid CLI information, or because they are on a blacklist or your own do not call list) no CDR is generated and, therefore, they cannot be retrieved with the above endpoints.

These rejections are notified using the notification endpoint you have configured (e.g. by eMail or HTTP post)

Additionally, the most recent rejections can be retrieved from the following API endpoints.

---

### **/v3/voice/{ACCOUNT}/rejected**

**GET**

Get all available rejected calls

```
{
  "success": true,
  "data": [
    {
      "calldate": "2016-02-08 07:00:04",
      "reason": "maxcost",
      "from": "441632496000",
      "to": "447700900123",
      "message": "Call to 447700900123 will exceed 0.1 ( 0.25450 )",
      "source_ip": "10.0.0.0",
      "trunk": "9XXXXX-L001",
      "notified": false
    }
  ]
}
```

---

### **/v3/voice/{ACCOUNT}/rejected/{REASON}**

**GET**

As above, but an optional *REASON* is specified to return only rejections of the specified reason (e.g. "maxcost" or "cli")

**NB** Rejection data should be considered ephemeral. In any event the above endpoints will never return more than data from the current day and the preceding day.

# Outbound Voice Trunks

## Trunk Management

Trunks should follow the naming ACCOUNT-**{ID}** e.g. if your account is 930004 and the trunk for ACME Products you could name the trunk 930004-ACMEPRODUCTS.

The "L001" trunk is your default **IP-Authenticated** trunk and cannot be renamed, additionally there are some features (Trunk Balances, Realtime Trunk Calls in Progress etc) that are not available on the L001 trunk.

---

### **/v3/voice/{ACCOUNT}/outbound**

<b>GET</b>	List all active outbound trunks.
------------	----------------------------------

---

### **/v3/voice/{ACCOUNT}/outbound/{TRUNK}**

<b>GET</b>	Request information on specified {TRUNK}
<b>PUT</b>	Create new trunk {TRUNK} <b>NB for an 'auth' trunk the SIP password will be returned ONLY as a response to this API call and cannot be retrieved later. If you forget the password it can be reset as described below (Trunk Password Functions)</b>
	<b>or (if additional optional configuration parameters specified, see overleaf)</b> Update existing {TRUNK} settings such as channel and rate limits
	type                      When creating a new trunk can be one of;  <b>ip</b> create an IP-authenticated trunk <b>auth</b> create a username/pass authenticated trunk  <i>for compatibility, if not specified defaults to 'auth'</i>
<b>DELETE</b>	Delete trunk {TRUNK} <b>NB the default trunk {ACCOUNT}-L001 cannot be deleted</b>

Newly created trunks are available for use immediately although, at times, may not show in the outbound trunk list for a short time.

## Trunk Options

/v3/voice/{ACCOUNT}/outbound/{TRUNK}

<b>PUT</b>	Update existing {TRUNK} settings such as channel and rate limits <b>or (if trunk does not exist) create new {TRUNK} (see previous page)</b>
<b>The options below are only to be used when updating an existing trunk</b>	
<i>enabled</i> <i>[optional]</i>	Enable [1] or Disable [0] this trunk. NB this takes precedence over in/out below
<i>enabled_in</i> <i>[optional]</i>	Enable [1] or Disable [0] inbound calls on this trunk (when being used via SIP registration)
<i>enabled_out</i> <i>[optional]</i>	Enable [1] or Disable [0] outbound calls on this trunk.
<i>quality</i> <i>[optional]</i>	<b>Only for 'legacy' or 'startup' account types</b> Platinum [1], Gold [2] or Silver [3] rate deck
<i>limit_concurrent_*</i> <i>[optional]</i> <i>The * above to be replaced by one of the parameters opposite</i>	Concurrent channel limit for the class of calls indicated as an integer value or [null] (do not impose a limit) <b>out</b> <b>out_international_hotspot</b> <b>in</b> <b>out_international</b> <b>out_per_number</b>
<i>limit_rate_*</i> <i>[optional]</i>	Rate limit for each of the above outbound categories in the form calls/duration (or [null] where no limit is to be imposed) e.g. <b>5/10s</b> 5 calls per 10 seconds <b>100/12h</b> 100 calls in 12 hours <b>NB The timeframe must be one of [ 12h   10s   1s ]</b>
<i>cli_default</i> <i>[optional]</i>	Default CLI to be presented when no valid CLI is provided or cli_force_default is set - must be in E.164 format (e.g. 441632960123)
<i>cli_force_default</i> <i>[optional]</i>	Set to [1] to force above value to be used on all calls. [0] allows customer-specified CLI
<i>max_cpm</i> <i>[optional]</i>	Maximum cost per minute (in your billing currency) of the B leg of the call. Using this can help ensure a customer trunk cannot make calls to expensive destinations
<i>max_cpc</i> <i>[optional]</i>	To be used in conjunction with the above, sets a maximum connection cost per call.
<i>max_cost</i> <i>[optional]</i>	Sets a maximum cost per call, i.e. when the cost of the call reaches (approximately) this amount it will be cleared
<i>max_dur</i> <i>[optional]</i>	Sets a maximum duration per call, when the call reaches this duration, it will be cleared.
<i>emergency_enabled</i> <i>[optional]</i>	Set to [1] to enable emergency calls (requires account activation)

**NB** If you update a trunk and one or more parameters are invalid, the update *will* succeed with the valid parameters, please check the output returned and ensure the trunk is configured as you expect.  
Please note that all trunk settings are updated immediately, therefore disabling (e.g. by setting enabled = 0) is an effective way to block any further calls being made on a trunk. It will **not** stop calls currently in progress.

## Trunk Balances

Each trunk can have its own balance, this allows you to manage the spend of individual customers. You can also view "inprogress" information for a trunk which will show the value of calls in progress. Where the value of calls in progress exceeds the trunk balance, calls in progress will be ended.

---

**/v3/voice/{ACCOUNT}/outbound/{TRUNK}/balance**

<b>GET</b>	Get balance for {TRUNK}	<pre>{   "success": true,   "data": {     "trunk": "920123-ACME",     "balance": 39.544   } }</pre>
<b>PUT</b>	Set or adjust the balance for {TRUNK}	
	<i>balance</i>	Sets the balance of the trunk to <i>balance</i>
	<b>- or -</b>	
	<i>adjust</i>	Adjust the trunk balance by the amount shown, use negative amounts to decrement the balance.
<b>DELETE</b>	Remove the balance from trunk {TRUNK}	
	<b>NB</b>	<b>The trunk will function as before, without it's own balance, so will be limited only by your account balance or balance lock</b>
		<b>This is NOT EQUIVALENT TO setting a balance to 0 which would prevent further calls being made (i.e. Balance exhausted)</b>

The trunk balance feature is provided for convenience and is not a substitute for your own billing and credit control procedures. Simwood will not be liable for any calls made on a trunk when its trunk-specific balance is depleted for any reason.

**Set balance on trunk 920123-ACME to £200**

```
{"balance": 200}
```

We strongly recommend that "balance" is used to set an initial balance only, and thereafter the level is maintained using the "adjust" function.

**Adjust balance on trunk 920123-ACME by £50 (e.g. if Customer has topped up)**

```
{"adjust": 50}
```

**Adjust balance on trunk 920123-ACME by £20 (e.g. to deduct supplementary services)**

```
{"adjust": -20}
```

**NB** Your account primary -L001 IP trunk does not support Trunk Balances.

## Per-Trunk Realtime Call Information

As with your account you can also view realtime "in progress" information on a per-trunk basis.

---

**/v3/voice/{ACCOUNT}/outbound/{TRUNK}/inprogress**

**GET**

Provides information on the calls currently in progress on the specified TRUNK

```
{
  "datetime": "2014-02-30 12:34:35",
  "total": 1.164,
  "callcount": 107,
  "balance": 24.406,
  "percent_available": 0.25,
  "approx_seconds_remaining": 56838,
}
```

Where a trunk has a balance the three additional elements will be present which have the same meaning as in your account snapshot but pertain only to the TRUNK specified;

**balance, percent\_available, and approx\_seconds\_remaining**

These are omitted where a trunk has no balance, but you can still view the current value of calls in progress.

**NB** Your account primary -L001 IP trunk does not support this feature.

## Trunk IP Functions

The following applies only to trunks using IP-Based authentication, your primary trunk {{ACCOUNT}}-L001 is an example of one such trunk which does not require a SIP username and password.

Please ensure you only add IP addresses that you control and DELETE any that are no longer required.

Please note that that this functionality should **NOT** be used to update an account with dynamic IP addresses - such installations, where unavoidable, should use authenticated SIP trunks as described above.

---

### **/v3/voice/{ACCOUNT}/outbound/{TRUNK}/acl**

<b>GET</b>	Request a list of IP address authorised on ip-based {TRUNK}
------------	---

---

### **/v3/voice/{ACCOUNT}/outbound/{TRUNK}/acl/{IP}**

<b>PUT</b>	Add {IP} to ip-based {TRUNK}
------------	------------------------------

<b>DELETE</b>	Remove {IP} from ip-based {TRUNK}
---------------	-----------------------------------

## Trunk Password Functions

The following applies only to authenticated trunks (those using a username and password) or being used for SIP registration.

---

### **/v3/voice/{ACCOUNT}/outbound/{TRUNK}/password\_reset**

<b>POST</b>	Request and return a new password for this trunk.
-------------	---

**NB The old password will be disabled immediately, any devices configured to use this trunk will need reconfigured to continue to make outbound calls.**

```
{
  "updated": true,
  "trunk": "{TRUNK}",
  "user": "{TRUNK}",
  "pass": "e5d5aca5e39251bdc19554d3"
}
```

**NB** It is not possible to specify a password for a trunk, they are automatically generated.

Likewise, the Operations Desk **cannot** recover a forgotten password, the only facility they have is to reset the password using the above functionality. Please keep your password(s) safe.



## Outbound Destination ACLs

Not to be confused with IP ACLs (to determine which IPs can make outbound calls on a particular trunk) Destination ACLs allow you to limit access on a per-account or per-trunk basis to certain destinations.

The ACL is specified in a JSON-encoded object as follows;

allow	Array	Allowed Prefixes	e.g. [441,442,443,448]
deny	Array	Denied Prefixes	e.g. [44870]

This would be encoded in JSON as follows;

```
{"allow": [441, 442, 443, 448], "deny": [44870]}
```

This example would allow calls to all UK Geographic, 03 Numbers and all 08 Numbers except 0870.

Please note

- (1) Longest prefix matching is used, so 44870 in the **deny** list will block even if 448 is allowed.
- (2) If *only allow* is specified, this is treated as a whitelist. all other destinations will be denied.
- (3) If *only deny* is specified, this is treated as a blacklist. all other destinations will be allowed.
- (4) Account-level blocks will override any trunk settings.  
(e.g. a trunk cannot call a destination blocked at the account level)
- (5) Trunk level blocks will override any account-level allows.  
(e.g. you may deny certain trunks access to destinations that are otherwise allowed)

### **/v3/voice/{ACCOUNT}/outbound/destinationacl** **/v3/voice/{ACCOUNT}/outbound/{TRUNK}/destinationacl**

<b>GET</b>	Retrieve active ACL on your account or trunk as specified
<b>PUT</b>	Replace active ACL with the JSON object PUT.  <b><i>The preferred method of doing this is to send the new ACL as the body of the PUT request, however if your implementation does not support this you may send the entire string in a single HTTP form encoded variable 'payload'</i></b>
<b>DELETE</b>	Remove the ACL associated with the account or trunk  <b><i>Please note that the default is to allow access to ALL destinations without restriction - please ensure this is what you want.</i></b>

Calls rejected due to failing a destination ACL rule will have the following X-Reason headers set in the SIP response to the initial INVITE;

X-Reason: 447700900123 matches trunk do not route 447

X-Reason: 449098790000 matches customer do not route 449

Either one of these may have (cached) appended where the number has been blocked more than once in the last 60s, in such case the prefix may not be shown.

**NB** The above 'destination acl' function is provided on a 'best-efforts' basis and is not intended as a substitute for securing your own VoIP platform. You remain responsible for all calls made on your account.

## IDA Outbound Management

You can access your outbound SIP rates from any\* standard BT landline using our shared IDA code **12940**. Access is restricted to authorised CLIs.

IDA users are managed much like trunks (see above), each number added automatically creates a 'trunk' in the form {ACCOUNT}-IDA01xxxxxxxx, you will see these identifiers as the trunk in your CDRs and the same settings can be applied as to trunks (see above)

---

### /v3/voice/{ACCOUNT}/ida

<b>GET</b>	Retrieve active IDAs on your account
------------	--------------------------------------

---

### /v3/voice/{ACCOUNT}/ida/{CLI}

<b>GET</b>	Retrieve the details associated with this IDA User
------------	--

<b>PUT</b>	Create a new IDA user with the specified {CLI}  <b>or (if additional parameters are supplied)</b> update an existing IDA user (see above "Trunk Management" for an example of what parameters are available)
------------	---

<b>DELETE</b>	Delete the IDA user with the specified {CLI}
---------------	--

Users making calls using the IDA service should dial the full number prefixed with 12940 e.g. to call 029 2120 2120 you would dial **1294002921202120**

1. At present you cannot associate more than one CLI with a single IDA 'trunk'
2. If another Simwood customer has enabled a CLI for the IDA service you will not be able to associate the same CLI with your own account.

\* Some landlines may not permit IDA calls

### IDA for Virtual Interconnect ("Hosted IDA")

Virtual Interconnect - Inbound customers using their own IDA codes should not use the above functionality.

Please contact us for more information on IDA for Virtual Interconnect.

# Inbound Numbering

## Number Allocation

### Number Ranges

**/v3/numbers/{ACCOUNT}/ranges**

<b>GET</b>	Retrieves a list of all available number ranges, including descriptions. This is intended for customers to populate, for example, a drop down to allow customers to select an area
	<pre>[   {     "id": "96fb9c60495aa3d4e05848b8e9fc4535",     "prefix": "1209255",     "sabc": "1209",     "description": "Geographic - Redruth",     "chargeband": "geo"   },   {     "id": "aead59d8024038d74b1109ac12642fb9",     "prefix": "1237418",     "sabc": "1237",     "description": "Geographic - Bideford",     "chargeband": "geo"   },   ... ]</pre>

### Available Numbers

**/v3/numbers/{ACCOUNT}/available/[all|gold|standard]/[1|10|100]**

<b>GET</b>	Returns 1,10 or 100 numbers available for allocation matching the <i>pattern</i> specified.
	One of all, gold, or standard should be specified in the URL; <b>all</b> returns all available numbers matching <i>pattern</i> <b>gold</b> returns only gold numbers matching <i>pattern</i> <b>standard</b> returns only non-gold numbers matching <i>pattern</i>
	<i>pattern</i> Search for numbers matching specified <i>pattern</i> (can use wildcards e.g *203*)
	<pre>[{"country_code": "44",   "number": "1134032330",   "recommended_gold_premium": "0",   "wholesale_gold_premium": "0",   "block": "03dd542cafcecf43fc06024ee6099311424c71cf",   "bill_class": "Carrier",   "SMS": "0"}]</pre>

**/v3/numbers/{ACCOUNT}/available/consecutive/[10|20|30|40|50|60|..100]**

<b>POST</b>	Request a report of 10,20,30,40,50,60,70,80,90 or 100 consecutive numbers available for allocation matching the <i>pattern</i> specified.
	<i>pattern</i> Search for numbers matching specified <i>pattern</i> (can use wildcards e.g *203*)

Please note the above options [e.g. 1|10|100] are the only options, arbitrary values [e.g. /25] are not supported.

**NB** Some number types, e.g. OTT Mobile Numbers are only available as gold numbers at this time.

## Allocated Numbering

### `/v3/numbers/{ACCOUNT}/allocated/all`

<b>POST</b>	Request a report of all current allocated numbers on account.
	<i>pattern</i> Optionally specify to include only those numbers that match the specified <i>pattern</i> (can use wildcards e.g *203*)
	<i>key</i> Only return those numbers that match the specified <i>key</i> in their metadata (see Advanced Routing below) <b>NB Keys are case-insensitive, wildcards not supported.</b>

### `/v3/numbers/{ACCOUNT}/allocated/[ 10 | 100 | 1000 | 10000 ]`

<b>POST</b>	Request a report of the first [ 10   100   1,000   10,000 ] numbers that match the optional <i>pattern</i> .
	<i>pattern</i> Optionally specify to include only those numbers that match the specified <i>pattern</i> (can use wildcards e.g *203*)
	<i>key</i> Only return those numbers that match the specified <i>key</i> in their metadata (see Advanced Routing below) <b>NB Keys are case-insensitive, wildcards not supported.</b>

Please note the above options (e.g. 1|10|100) are the only options, arbitrary values (e.g. /25) are not supported.

### `/v3/numbers/{ACCOUNT}/allocated/{NUMBER}`

<b>GET</b>	Return configuration information on allocated {NUMBER}
<b>PUT</b>	Allocate an available {NUMBER} to the account
<b>DELETE</b>	De-configure and irrevocably remove {NUMBER} from account



#### **Gold Number Activation Fee**

Where {NUMBER} is a Gold Number an activation fee will be charged



#### **Number Rentals**

There is an ongoing monthly fee for geographic numbers, please see <https://simwood.com/rates> for more information

## Last Call

### `/v3/numbers/{ACCOUNT}/allocated/{NUMBER}/lastcall`

<b>GET</b>	Returns a JSON object describing the most recent call to this number
	<pre>{   "success": true,   "data": {     "calldate": "2017-05-01 12:34:01",     "cli": "07700900123",     "disposition": "NORMAL_CLEARING",     "billsec": "64",     "duration": "68"   } }</pre>

## Number Routing Configuration

Introduced in May 2014, the following is the preferred way of configuring a number, the previous method is still detailed in this document but is deprecated and will be removed in a future revision of the API.

When a new configuration is provided this will take precedence over any existing configuration.

---

### **/v3/numbers/{ACCOUNT}/allocated/{NUMBER}/config**

<b>GET</b>	<p>Return configuration information on allocated {NUMBER}</p> <p><b>See Configuration Syntax (New) below</b></p> <pre>{   "routing": {     "default": [       {         "type": "sip",         "endpoint": "44163296000@pbx.simwood.com"       },       {         "type": "reg",         "user": "930XXX-SIPUSER"       }     ],     [       {         "type": "pstn",         "number": "447700900123"       }     ]   ] }</pre>
<b>PUT</b>	<p>Replace active configuration for {NUMBER} with the JSON object PUT.</p> <p><b><i>The preferred method of doing this is to send the new routing configuration as the body of the PUT request, however if your implementation does not support this you may send the entire string in a single HTTP form encoded variable 'payload'</i></b></p> <pre>{"success": true}</pre> <p>If any error(s) occurred whilst validating the configuration these will be shown;</p> <pre>{   "success": false,   "errors": [     "Message #1",     "Message #2"   ] }</pre>
<b>DELETE</b>	<p>De-configure the configuration of {NUMBER}</p> <p><b>NB</b> if configuration is still present on the /voice endpoint, it will be used</p>

Similarly, it is now possible to set a default configuration which will be used for all numbers on your account where no other configuration exists - this is ideal for customers who send all calls to a SIP URI and handle onward routing themselves

---

### **/v3/numbers/{ACCOUNT}/default/config**

<b>GET</b>	Return default number configuration for {ACCOUNT}
<b>PUT</b>	Replace active default configuration with the JSON object PUT.

## Number Configuration Syntax - Advanced

Numbers are configured using a JSON object which is described below, offering increased flexibility over the previous route configuration.

An example (annotated) configuration is below, and the full list of options can be found overleaf.

<pre>{   "options": {     ...   },   "rules": {     "officehours": [       {         "dow": [1,2,3,4,5],         "time": [0900,1700]       }     ]   },   "routing": {     "officehours": [       [         {           "type": "reg",           "user": "930XXX-SIPUSER",           "timeout": 30         },         {           "type": "sip",           "endpoint": "%e164@pbx.mycompany.com",           "timeout": 30         }       ]     ],     [       {         "type": "pstn",         "number": "447700900123"       }     ]   ],   "default": [     [       {         "type": "pstn",         "number": "447700900123"       }     ]   ] }, "meta": {   ... } }</pre>	<p>Per number configuration options (detailed below) apply to the number at all times</p> <p>Rules define times of day that routing blocks (below) apply, outwith the rules given - or if the rules section is omitted entirely - the <b>default</b> routing block will be used.</p> <p>This block will run during the <b>officehours</b> time block defined above (Mon-Fri 9am-5pm) and will call the SIP endpoint shown and the user on the registration proxy simultaneously.</p> <p>After the timeout above (30s) we will try the PSTN number provided</p> <p>The <b>default</b> routing block will be used outwith the office hours specified above. So calls outside of normal office hours will be forwarded directly to the PSTN number shown.</p> <p>The <b>meta</b> block contains arbitrary metadata that you want to associate with the number</p> <p>See below for more information.</p>
---	---

## Number Configuration Syntax

### "options"

The following options can be set on a per-number basis, they apply to the entire configuration at all times

<i>enabled</i>	<i>[true/false]</i>	Allows number to be disabled [false] without removing the configuration.	Default [true]
<i>block_payphone</i>	<i>[true/false]</i>	Prevents inbound calls originating from payphones	Default [false]
<i>acr</i>	<i>[true/false]</i>	Apply ACR to this number. ACR Prevents calls originating from Withheld numbers reaching this number. Withheld callers will be diverted to a recorded announcement.	Default [false]
<i>icr</i>	<i>true</i> <i>486</i> <i>404</i> <i>false</i>	Apply ICR to this number. Simwood ICR Prevents calls where no ANI or valid CLI is present reaching this number. Withheld numbers are allowed as long as a valid number is present.  The behaviour of this parameter varies as follows;  <b>true</b> plays recorded message <b>486</b> returns busy signal <b>404</b> returns unknown number <b>false</b> disables ICR	Default [false]
<i>trunk</i>	<i>9XXXXX-TRUNK</i>	Associate a trunk with this number for billing purposes, shown in CDRs.  <b>NB</b> This does NOT result in the number being routed to a registration-based SIP trunk, you would need to use the appropriate 'reg' routing block  You can also configure this using the endpoint <code>/v3/numbers/[ACCOUNT]/allocated/{NUMBER}/trunk</code>	Default [9XXXXX-L001] or your account default inbound trunk if different.



### Chargeable Options

Some options may incur additional monthly fees. Please see <https://simwood.com/rates> for full information

## "rules"

The 'rules' parameter in the JSON object should be an array of named objects (named using the characters a-z and the \_ character only) each of which defines a time period using the following parameters;

<i>dow</i>	<i>Array of values [1..7]</i>	The days of week this rule is active (according to ISO 8601) e.g. <b>1 = Monday, 7 = Sunday.</b> <b>NB if one day must still be an array. e.g. [3] not 3</b>
<i>time</i>	<i>Array [start,end]</i>	Array of two values, denoting the start and end time this rule applies in 24h format (e.g. [0900,1700] would represent 9am - 5pm. The leading 0 can be omitted)
<i>day</i>	<i>Array of values [1..31]</i>	The days of month this rule is active <b>NB if one day must still be an array. e.g. [25] not 25</b>
<i>month</i>	<i>Array of values [1..12]</i>	The months this rule is active e.g. <b>1 = January, 12 = December.</b> <b>NB if one day must still be an array. e.g. [3] not 3</b>

## Example

<b>Office Hours</b>	Mon - Fri 0900-1700	<pre>"rules": {   "officehours": [     {       "dow": [1,2,3,4,5],       "time": [900,1700]     }   ] }</pre>
<b>Weekends</b>	Sat / Sun (All Day)	<pre>"rules": {   "weekend": [     {       "dow": [6,7]     }   ] }</pre>
<b>Christmas / New Year</b>	Dec 25th December 26th January 1st January 2nd	<pre>"rules": {   "christmasholiday": [     {       "month": [12],       "day": [25,26]     },{       "month": [1],       "day": [1,2]     }   ] }</pre>



## "routing"

Much like the rules above the 'routing' parameter is an array of objects named corresponding to the rules. There is also the special 'default' rule, which applies when there are no rules specified or outwith the times specified in the rules.

Routing blocks (described below) can be arranged to allow dialling in parallel or in sequence, or a combination of both as shown below;

Ring A & B simultaneously, Then C	Ring A, B and C simultaneously
<pre>"routing": {   "officehours": [     [       {         BLOCK A       },       {         BLOCK B       }     ],     [       {         BLOCK C       }     ]   ] }</pre>	<pre>"routing": {   "officehours": [     [       {         BLOCK A       },       {         BLOCK B       },       {         BLOCK C       }     ]   ] }</pre>
Ring A, then B, then C	
<pre>"routing": {   "officehours": [     [       {         BLOCK A       },     ],     [       {         BLOCK B       }     ],     [       {         BLOCK C       }     ]   ] }</pre>	

## "meta"

The 'meta' parameter in the JSON object allows you to store your own arbitrary data in the Simwood database associated with a number and which can be easily accessed via the API.

The **key** parameter can be used to search for number(s) matching a specified key, the rest of the object is freeform up to a size limit of around 512 bytes.

<i>key</i>	<i>String [40 chars]</i>	A (non-unique) key for this number which you can use to search for matching numbers, most commonly customers use this to store a representation of their own customer account ID or username.
------------	--------------------------	---

### **NB Keys are treated as case-insensitive**

..	..	The rest of the <b>meta</b> parameter can be composed of ANY valid JSON structure, some example uses are given below.
----	----	---

## Examples

<p>In this example the key is used to store the customer account ID.</p> <p>The friendlyName is used to allow the customer to assign a memorable name to the number in the customer interface.</p> <p>The lastUpdated parameter is used to show when a number was last updated.</p>	<pre>"meta": {   "key": "403010",   "friendlyName": "Main office number",   "lastUpdated": "2014-02-30 01:20:30" }</pre>
<p>In this example the key is used to store the customer eMail address and a password. This could be used to build a simple portal for managing forwarding of one number (e.g. a personal number redirection service)</p> <p><b>NB This illustrates a potential use only - please do NOT store clear text passwords in this manner.</b></p>	<pre>"meta": {   "key": "customer@example.com",   "pass": "secretWord" }</pre>
<p>This shows how this could be used to store a note associated with a number that hasn't been configured yet.</p>	<pre>"meta": {   "notes": "This is reserved for Brian             at Example Corp and is not             yet in use",   "reserved": true }</pre>

The flexibility offered by the "meta" block enables customers to build a full service (e.g. offering number translation services, fax to eMail or similar) without requiring a local database.

**NB** Meta attributes are intended for API use only and are NOT displayed in the portal. These attributes may be overwritten if a number is later configured using the portal.

## Routing Block

Each routing block is defined by a minimum of a 'type' and some optional parameters delay and timeout;

<i>parameter</i>	<i>values</i>	
<b>type</b>	<i>sip</i>	SIP endpoint
	<i>reg</i>	Registered SIP user
	<i>pstn</i>	PSTN Destination (i.e. voice call forwarding)
	<i>fax</i>	Receive as Fax to eMail or HTTP <b>NB fax cannot be used in conjunction with any other endpoint</b>
	<i>busy</i>	Busy tone
<b>delay</b>	<i>[1...n]</i>	Delay before executing this leg
<b>timeout</b>	<i>[1...n]</i>	Timeout for answer from this leg <i>Not supported by <b>busy</b> or <b>fax</b> endpoints</i>

**NB** A number can be configured for either voice *or* fax routing  
It is not possible to simultaneously use the same number for voice and fax.

## Routing Block Additional Parameters

Depending on the type of endpoint selected above there are some additional endpoint-specific mandatory and optional parameters;

<i>type</i>	<i>parameter</i>	
<b>sip</b>	endpoint	<p>SIP Endpoint (user@host.com with optional :port and ;transport parameters e.g. %164@example.com:5060;transport=tcp) the following substitutions will take place;</p> <p><b>%e164</b> will be replaced with the full number in <b>E164</b> format  <b>%ukn</b> will be replaced with the number in <b>UK National</b> format</p>
	sdes	<p>One of <b>optional</b>, <b>required</b>, or <b>none</b>. If optional we will offer SDES encryption for the audio on inbound calls. If set to required, the call will not complete without successful negotiation of SDES. If not specified, or set to none, SDES will not be offered by default.</p>
	opus	<p>One of <b>never</b>, <b>always</b>, or <b>only</b>. Override the default handling of Opus offers in INVITEs to your platform;</p> <p><b>never</b> will suppress the Opus codec from the SDP, which may be useful for UDP destinations to prevent long INVITEs becoming fragmented as Opus does typically increase the SDP payload by around 220 bytes.</p> <p><b>always</b> will always offer the Opus codec, together with the usual codec offering of G.722 and G.711, in the SDP.</p> <p><b>only</b> will offer the Opus codec only, excluding all other codecs. This is ideal if your platform uses Opus and you can support Opus for all calls, and still keeps the SDP to a size that is unlikely to result in fragmentation</p>
	zone	<p>One of our zone names [<b>man</b>, <b>slo</b>, <b>lon</b>, <b>ny</b> or <b>sj</b>].</p> <p>If present, calls originating in the specified zone will be passed to this endpoint, overriding any equal priority endpoints without a zone specified.</p> <p>This can be used together with multiple sip endpoints to ensure that calls are routed in the most efficient way depending on where they ingress the Simwood network. e.g. calls arriving in Manchester, will route to your own local proxy.</p> <p>This will be particularly useful to customers with direct connections to Simwood, or hosting their own equipment on-net.</p> <p><b>NB if a call originates from any zone which matches an endpoint with a specific zone set, then any other endpoints within that group will be disregarded.</b></p> <p><b>If the call originates from an zone which does not match a specific endpoint, any endpoint without a zone parameter will be used.</b></p> <p><b>There must always be one endpoint at the same level with no zone specified.</b></p>

<i>type</i>	<i>parameter</i>	
<b>reg</b>	<i>user</i>	SIP registration user (e.g. 9xxxxx-USERNAME)
	<i>sdes</i>	One of <b>optional</b> or <b>required</b> . Defaults to <b>optional</b> , we will offer SDES encryption for the audio on inbound calls to registered endpoints. If set to <b>required</b> , the call will not complete without successful negotiation of SDES.
	<i>opus</i>	One of <b>never</b> , <b>always</b> , or <b>only</b> . Override the default handling of Opus offers in INVITEs to your platform;  <b>never</b> will suppress the Opus codec from the SDP, which may be useful for UDP destinations to prevent long INVITEs becoming fragmented as Opus does typically increase the SDP payload by around 220 bytes.  <b>always</b> will always offer the Opus codec, together with the usual codec offering of G.722 and G.711, in the SDP.  <b>only</b> will offer the Opus codec only, excluding all other codecs. This is ideal if your platform uses Opus and you can support Opus for all calls, and still keeps the SDP to a size that is unlikely to result in fragmentation
	<b>NB</b> if the user is not registered at the time the call is received this will be skipped entirely, we advise you use another destination (or 'busy') in addition to a <b>reg</b> endpoint.	
<b>pstn</b>	<i>number</i>	Destination number in e164 format (e.g. 447700900123) <b>NB the pstn divert function is intended for voice calls only.</b>
	<i>maxcpm</i>	Maximum cost per minute (in your billing currency) of the B leg of the call, intended for use with NTS services to limit exposure to expensive destinations or to ensure that the destination number cost is covered by the revenue share from the inbound leg. (e.g. 0.05)
	<i>maxcpc</i>	To be used in conjunction with the above, sets a maximum call cost (e.g. connection cost) <b>NB does not limit call duration to a cost 'limit'</b>
	<i>cli</i>	CLI to present when forwarding the call, if not specified will present the callers CLI where it is available.
	<i>trunk</i>	The trunk to be associated (for billing and CDR purposes) with the outbound (B-leg) (e.g. 9xxxxx-TRUNKNAME)
<b>fax</b>	<i>method</i>	One of <b>http</b> or <b>mail</b>
	<i>endpoint</i>	Destination eMail address or HTTP POST URI e.g. user@host.com http://www.yourdomain.com/cgi/inboundfax.php
<b>busy</b>	no parameters available	

**NB** The 'pstn' endpoint is intended for **voice calls only**.  
We cannot guarantee successful transmission of faxes or data on calls forwarded to the PSTN using this functionality.

## Number Configuration Worked Examples

### Call user@host.com over SIP for 20s, then try 447700900123 with custom CLI

```
{
  "routing": {
    "default": [
      [
        { "type": "sip", "endpoint": "user@host.com", "timeout": 20}
      ],
      [
        { "type": "pstn", "number": "447700900123", "cli": "442921202120", "maxcpm": 0.02}
      ]
    ]
  }
}
```

### Forward all calls to 447700900123

```
{
  "routing": {
    "default": [
      [
        {
          "type": "pstn",
          "number": "447700900123"
        }
      ]
    ]
  }
}
```

### Forward all calls to SIP endpoint

```
{
  "routing": {
    "default": [
      [
        {
          "type": "sip",
          "endpoint": "%did@sip.mycompany.com"
        }
      ]
    ]
  }
}
```

The above might seem complicated but corresponds to the following simple PHP example, which illustrates the underlying structure;

### PHP Example of above

```
<?php
// Define the route
$arrayRouteDefinition = Array('type' => 'sip', 'endpoint' => '%did@sip.mycompany.com');
// Add it to the 'default' routing block
$arrayRouting['default'][] = Array($arrayRouteDefinition);
// Add the routing configuration to the config
$arrayConfig['routing'] = $arrayRouting;
// encodedConfig now contains the JSON encoded routing
$encodedConfig = json_encode($arrayConfig);
```

Another advantage of this method is that you can retrieve the configuration as a JSON object via your favourite programming language, edit the required entry in place and PUT the changed configuration back.

## Number Configuration Extended Example

Below is an extended example that meets the following requirements;

During business hours (Monday-Friday, 9am-5pm) connect the call over SIP to our PBX at sip.mycompany.com.

If there is no response within 30 seconds, try the weekday out of hours mobile over the PSTN on 07700900123.

During the weekend (Saturday and Sunday all day) send calls directly to the weekend out of hours mobile on 07700900555

At all other times (so before 9am and after 5pm weekdays) send calls to the weekday out of hours mobile on 07700900123

### JSON of above

```
{
  "rules": {
    "officehours": [
      {
        "dow": [1,2,3,4,5],
        "time": [900,1700]
      }
    ],
    "weekend": [
      {
        "dow": [6,7]
      }
    ]
  },
  "routing": {
    "officehours": [
      [
        {
          "type": "sip",
          "endpoint": "%did@sip.mycompany.com",
          "timeout": 30
        }
      ], [
        {
          "type": "pstn",
          "number": "447700900123"
        }
      ]
    ],
    "weekend": [
      [
        {
          "type": "pstn",
          "number": "447700900555"
        }
      ]
    ],
    "default": [
      [
        {
          "type": "pstn",
          "number": "447700900123"
        }
      ]
    ]
  }
}
```

## Number Fax Routing Configuration

**NB** A number can be configured for either voice *or* fax routing  
It is not possible to simultaneously use the same number for voice and fax.

## Number Fax Configuration Worked Examples

### Receive FAX and forward, by eMail, to user@example.com

```
{
  "routing": {
    "default": [
      [
        {"type": "fax", "method": "mail", "endpoint": "user@example.com"}
      ]
    ]
  }
}
```

### Receive FAX as HTTP POST to http://example.com/api/inbound\_fax

```
{
  "routing": {
    "default": [
      [
        {"type": "fax", "method": "http", "endpoint": "http://example.com/api/inbound_fax"}
      ]
    ]
  }
}
```

As with the voice routing, the above might seem complicated but corresponds to the following PHP example, which illustrates the underlying structure;

### PHP Example of above

```
<?php
// Define the route
$arrayRouteDefinition = Array('type' => 'fax',
                              'method' => 'mail',
                              'endpoint' => 'user@example.com');

// Add it to the 'default' routing block
$arrayRouting['default'][] = Array($arrayRouteDefinition);

// Add the routing configuration to the config
$arrayConfig['routing'] = $arrayRouting;

// encodedConfig now contains the JSON encoded routing
$encodedConfig = json_encode($arrayConfig);
```

Another advantage of this method is that you can retrieve the configuration as a JSON object via your favourite programming language, edit the required entry in place and PUT the changed configuration back.



## Number Configuration - Success

Changes should take effect immediately, and the following simple JSON object will be returned;

---

success	true	Will always be <b>true</b> when the routing has been successfully updated.
---------	------	--

---

## Number Configuration - Errors

In the event of a configuration error a simple JSON object will be returned as follows;

---

success	false	Will always be <b>false</b> where any error was present. Your config will <b>NOT have been changed</b> .
---------	-------	--

errors	Array	This will contain an array of human-readable errors each will include an indication of where the error was in your structure. A non-exhaustive list of these are below.
--------	-------	---

---

**NB** Where any error is present in the configuration the extant configuration will remain in place even if only one element has an error, the entire configuration will be rejected.

---

Setting ' <b>OPTION</b> ' must be <b>X, Y or Z</b>	An option in the "options" section has an invalid value, please select from one of the provided values
--	--

Invalid parameter ' <b>OPTION</b> ' in settings	A parameter in the "options" section is unrecognised. please remove this parameter
---	--

Rule name ' <b>NAME</b> ' is invalid.	Rule names must contain the characters shown only
---------------------------------------	---

[Rule Routing Block] must be an array.	Rules and routing blocks must be arrays, even if they contain only one item.
--	--

Rule ' <b>NAME</b> ' entry <b>X</b> parameter ' <b>PARAM</b> ' is invalid.	The value provided for the parameter in the rule shown is invalid.
--	--

Routing block ' <b>NAME</b> ' does not match any specified rules (or default)	A routing block has been specified that doesn't correspond to any time-dependant rules (or "default") - therefore would never be called. Check your rule names match.
---	---

Unknown section ' <b>NAME</b> ' in configuration.	There is a section that is unrecognised, if you wish to store your own information in a number configuration you may do so but this must be within the 'meta' section (which can contain anything)
---	--

---

## Inbound Trunk Configuration

Inbound numbers can be associated with a trunk for billing reconciliation purposes or to take advantage of some of the trunk controls for inbound traffic.

**NB** This does NOT result in the number being routed to a registration-based SIP trunk, you would need to use the appropriate 'reg' routing block in the Number Configuration shown above.

---

### **/v3/numbers/{ACCOUNT}/allocated/{NUMBER}/trunk**

**GET** Get the trunk currently associated with this number

```
{
  "success": true,
  "data": {
    "trunk": "930XXX-ACMEPRODUCTS"
  }
}
```

**PUT** Associate this number with a trunk

<i>JSON request</i>	trunk	The trunk to associate with this number
---------------------	-------	---

<i>JSON response</i>	{	"success": true,
		"data": {
		"trunk": "930XXX-ACMEPRODUCTS"
		}
	}	

**DELETE** Remove the association with the trunk

## Mobile Number Inbound SMS Configuration

**NB** Only mobile or OTT numbers can be configured for inbound SMS.

UK Mobile Numbers will be able to receive SMS (this includes both OTT numbers and MSISDNs obtained from Simwood, as well as SMS to numbers ported-in) - you can deliver these over HTTP to your own platform.

### **/v3/numbers/{ACCOUNT}/allocated/{NUMBER}/sms**

**GET** Request current inbound SMS configuration for the provided number

```
{
  "success": true,
  "data": {
    "mode": "http",
    "endpoint": "http://api.yourdomain.com/path/to/mtsms.cgi"
  }
}
```

**PUT** Update SMS configuration for the provided mobile number

*JSON request*

<b>mode</b>	One of [ http   http_json ] Both use the HTTP method, however <b>http</b> posts data to your script as application/x-www-form-urlencoded (much like a typical HTML <form> would).
-------------	--

**BETA**

**http\_json** makes an HTTP POST request with Content-type application/json with the body of the request as a JSON document (described below)

Only the beta **http\_json** method supports TLS endpoints.

*endpoint*

**Only where mode is 'http' or 'http\_json'**

An HTTP(S) URL that will receive a POST request for each SMS sent to this number.

**NB** only the beta http\_json method supports TLS endpoints.

```
{
  "mode": "http",
  "endpoint": "http://api.yourdomain.com/path/to/mt"
}
```

*JSON response*

```
{
  "success": true
}
```

**DELETE** Delete SMS configuration for this number

## Number Configuration - 999 Emergency Services

**/v3/numbers/{ACCOUNT}/allocated/{NUMBER}/999**

<b>GET</b>	Return current 999 information for {NUMBER}		
<b>PUT</b>	Replace current 999 details on {NUMBER} <i>Please note maximum field lengths indicated below</i>		
	<i>title</i>	20	<b>Individual End User Only</b> Title (e.g. Mr, Ms, Mrs) <i>Titles that disclose gender are preferred by the Emergency Services.</i>
	<i>forename</i>	20	<b>Individual End User Only</b> Forename or Initials
	<i>name</i>	50	<b>Individual End User</b> Surname
	<i>bussuffix</i>	50	<b>Business End User Only</b> Suffix (Ltd, Plc) [See Note 2]
	<i>premises</i>	60	<b>Mandatory for Individual and Business End Users</b> Identifies premises on a thoroughfare i.e. House Number or Name (e.g. 104, The Lighthouse, Thatched Cottage)
	<i>thoroughfare</i>	55	<b>Mandatory for Individual and Business End Users</b> Street Name (e.g. King Street, Station Road, Beech Avenue)
	<i>locality</i>	30	<b>Mandatory for Individual and Business End Users</b> Village or an area within an Town <i>and</i> Town if possible
	<i>postcode</i>	12	<b>Mandatory for Individual and Business End Users</b> The full current postcode as recognised by Royal Mail's PAF database. This must be in the form Out-code space In code e.g. LS11 5DF, S9 5AD, S60 3ML

**Remember this information is to assist the Emergency Service response and you have a legal obligation to ensure this information is provided fully and accurately to the best of your ability.**

The name and address information should be sufficient to identify the premises or individual promptly in an emergency. This is more important than it matching the 'official' record; for a business entry include *only* the business details, do NOT specify your contact there or primary account holder etc as an individual.

### Note 1 : Business Names

Business names should be chosen that best allow the Emergency Services to identify and locate the business - typically this is the 'name over the door' rather than that of a parent or holding company irrespective of who you address the bill to.

### Note 2 - Business Suffix

Addition to business name (e.g. Ltd or Plc) this can also be used to include a brief description that identifies the function of the business - e.g. "Hospital", "Hotel", "Fuel Storage Depot" provides valuable extra information to the Emergency Services



### Submission Charge

There is a charge for this service.

Please see <https://simwood.com/rates> for full information

## Number Lookup

We provide a simple API endpoint to allow you to look up the rangeholder information for a particular number or number range.

This may be of use to customers looking to port numbers, however it should be noted that if a number has previously been ported the LCP may not be the rangeholder

---

### **/v3/numbers/{ACCOUNT}/lookup/{NUMBER}**

**GET**

Lookup information on the **number** provided  
( the number should be provided in e.164 format e.g. the following result would be obtained from a **GET** request to **/v3/numbers/{ACCOUNT}/443301223000** )

```
{
  "success": true,
  "formatted": "+44(0)330 1223000",
  "data": {
    "code": "3301",
    "prefix": "330122",
    "rh": "Simwood eSMS Limited"
  }
}
```

# Number Porting

Ports can be submitted, and viewed via the API.

There are different endpoints (and data required) depending on the type of port, it is imperative you use the correct endpoint for the type of port requested.

type of number	port type	
UK Geographic Numbers <b>01xxxxxxxxx</b> <b>02xxxxxxxxx</b>	GNP	Geographic Number Porting is outlined in our Geographic Number Portability Guide.  Numbers can be ported from most major fixed-line and VoIP service providers and lead times are subject to the type of port.  You must have authority from the end user to port a number, and evidence of this may be requested.
UK Mobile Numbers <b>07[1-9]xxxxxxxxx</b>	MNP	Mobile Number Portability is managed differently from GNP.  Numbers can be ported from all UK MNOs and MVNOs and the port will typically take 2-3 days. The active mobile service (and SIM) associated with the number will cease when the port completes.  MNP requires a "PAC" (Porting Authorisation Code) to authenticate the porting request. This is provided to the existing subscriber by their Mobile Service Provider. This code must be provided at the time of porting.  PACs are valid for 30 days.
UK Non-Geographic Numbers <b>03xxxxxxxxx</b> <b>08[457]xxxxxxxxx</b>		At present, these numbers cannot be ported via the API.  Some NGNs are portable, please contact the Porting Desk.
UK Premium Rate Numbers <b>070xxxxxxxxx</b> <b>09xxxxxxxxx</b>		At present, these numbers cannot be ported

## Update December 2016

Please note that there are now **two** endpoints listed for each GNP API function as follows

```
/v3/porting/{ACCOUNT}/ports  
/v3/porting/{ACCOUNT}/gnp
```

We strongly recommend use of the /gnp endpoint for consistency with the new porting types, however the /ports endpoint provided historically will remain in service for backward compatibility.

## Geographic Number Porting (“GNP”)

### New GNP [Geographic Number Porting] Submission

**/v3/porting/{ACCOUNT}/ports**  
**/v3/porting/{ACCOUNT}/gnp**

<b>POST</b>	Submit a new Geographic Number port. The following must be submitted as a JSON object, an example is below	
<i>mbn</i>		Main Billing Number (MBN) in UK (01xxxxxxxx) format
<i>orig_ref</i>		For resubmissions only, the original order reference
<i>lcp</i>		Losing Communications Provider (e.g. “BT”)
<i>lcp_cupid</i>		The numeric CUPID of the LCP <i>or hosting network</i> , see the <b>/porting/{ACCOUNT}/lcps</b> endpoint (detailed below)
<i>port_date</i> <i>(Optional)</i>		Porting date (CRD) in form YYYY-MM-DD HH:mm:ss <b>Please see minimum lead times below.</b> If omitted, will port as soon as possible.
<i>contact_email</i>		A contact eMail address for the port (note this must be the address of the submitter <b>NOT your customer</b> )
<i>account_number</i>		The account number with the LCP (if known)
<i>billing_postcode</i>		The postcode associated with the current installation This <b>MUST</b> match the records held by the LCP
<i>type</i>		One of the port types listed below (e.g. “single”)
<i>lines</i>		Number of lines in the existing installation
<i>channels</i>		Number of channels in the existing installation
<i>customer</i>	object*	This must be a JSON object containing <b>each of the elements described below</b> NB This is NOT the same as 999 above.
<i>numbers</i>	array*	An array of “number” objects, see below



#### Submission Charge

There is a charge for this service.

Please see <https://simwood.com/rates> for full information

#### Port Types

Each port will be one of the following types, if unsure please contact the LCP before submission.

Subsequent ports must be used where the number is already ported (i.e. the LCP is not the rangeholder)

Please observe the lead times in the Number Portability Guide when specifying a port\_date.

<b>single</b>	Single Line - 4 Working Days (14 if > 10 lines porting in same installation/time)
<b>multi</b>	Multi Line - 7-17 Working Days (dependant on number of lines/numbers, see guide)
<b>sub_single</b>	Single Line - 7 Working Days (17 if > 10 lines porting in same installation/time)
<b>sub_multi</b>	Multi Line - 10-25 Working Days (dependant on number of lines/numbers, see guide)

The above is provided for guidance only, the Number Portability Guide should be consulted to determine lead times for porting requests.

## Number Object

The 'numbers' array in the above should be an array of objects each with the following structure. These must include all numbers, including the MBN (even though it is specified separately)

<i>parameter</i>	<i>values</i>	
<b>number</b>	01xxxxxxxxx 02xxxxxxxxx	Number in UK National format
<b>type</b>	<i>mbn</i>	Main Billing Number (must only be one of this type)
	<i>associated</i>	Associated Number (e.g. another DDI on an ISDN circuit)
	<i>other</i>	Other Number at the same address (but not associated)
<b>action</b>	<i>port</i>	Port this number to Simwood
	<i>retain</i>	Retain service on this number as-is (only for 'other' type)
	<i>drop</i>	Drop this number and cease service on the porting date

## Customer Object

<i>parameter</i>	<i>max length</i>		
<b>title</b>	20	<b>Individual End User Only</b> Title (e.g. Mr, Ms, Mrs) <i>Titles that disclose gender are preferred by the Emergency Services.</i>	
<b>forename</b>	20	<b>Individual End User</b> Forename or Initials	<b>Business End User</b> Forename/Initials of Signatory
<b>name</b>	50	<b>Individual End User</b> Surname	<b>Business End User</b> Surname of LoA Signatory
<b>company</b>	50		<b>Business End User Only</b> Business Name
<b>bussuffix</b>	50		<b>Business End User Only</b> Suffix (Ltd, Plc)
<b>premises</b>	60	<b>Mandatory for Individual and Business End Users</b> Identifies premises on a thoroughfare i.e. House Number or Name (e.g. 104, The Lighthouse, Thatched Cottage)	
<b>thoroughfare</b>	55	<b>Mandatory for Individual and Business End Users</b> Street Name (e.g. King Street, Station Road, Beech Avenue)	
<b>locality</b>	30	<b>Mandatory for Individual and Business End Users</b> Village or an area within an Town <i>and</i> Town if possible	
<b>postcode</b>	12	<b>Mandatory for Individual and Business End Users</b> The full current postcode as recognised by Royal Mail's PAF database. This must be in the form Out-code space In-code e.g. LS11 5DF, S9 5AD, S60 3ML	



## New Port Submission - Full Example

### JSON of example porting request

```
{
  "mbn": "01632960100",
  "contact_email": "simwood.customer@example.com",
  "lcp": "BT",
  "lcp_cupid": "001",
  "account_number": "NA1234B32",
  "billing_postcode": "A12 3BC",
  "type": "multi",
  "lines": 1,
  "channels": 30,
  "customer": {
    "title": "Mr",
    "forename": "John",
    "name": "Doe",
    "loa_initial": "J",
    "loa_surname": "Doe",
    "premises": "123",
    "thoroughfare": "Some Street",
    "locality": "Sometown",
    "postcode": "A12 3BC"
  },
  "numbers": [
    {"number": "01632960100", "type": "mbn", "action": "port"},
    {"number": "01632960101", "type": "associated", "action": "port"},
    {"number": "01632960102", "type": "associated", "action": "port"},
    {"number": "01632960103", "type": "associated", "action": "port"},
    {"number": "01632960104", "type": "associated", "action": "port"},
    {"number": "01632960105", "type": "associated", "action": "port"},
    {"number": "01632960106", "type": "associated", "action": "port"},
    {"number": "01632960290", "type": "other", "action": "retain"}
  ]
}
```

### Response

The response will be a JSON similar to the following;

```
{
  "success": true,
  "ref": 12345,
  "url": "/v3/porting/{ACCOUNT}/ports/12345"
}
```

*The "ref" value corresponds to the ticket that will be used to track the progress of the porting request.*

If any error(s) occurred whilst validating the configuration these will be returned as follows;

```
{
  "success": false,
  "errors": [ "Message #1", "Message #2" ]
}
```

### Port Resubmissions

When resubmitting a previously rejected port, please use the original porting order number in the **orig\_ref** parameter to ensure your port is processed and billed correctly at the reduced rate for resubmissions.

**NB** Failure to provide this will result in the port being processed as a new order and the normal porting fees applicable to the order type will apply.

## View Port List

**/v3/porting/{ACCOUNT}/ports**  
**/v3/porting/{ACCOUNT}/gnp**

**GET**

Get a list of currently outstanding, or recently completed geographic number ports

```
{
  "success": true,
  "data": [
    {
      "ref": "54721"
      "mbn": "01632960100",
      "date": "2015-02-29",
      "crd": "2015-03-14",
      "status_code": "submitted_rh"
      "status": "Submitted to RH"
    },
    {
      ...
    }
  ]
}
```

## View Port Status

**/v3/porting/{ACCOUNT}/ports/{ORDER\_REFERENCE}**  
**/v3/porting/{ACCOUNT}/gnp/{ORDER\_REFERENCE}**

**GET**

Get full detail as submitted, and history, of a port

The *customer* and *numbers* elements will be returned as originally submitted but are omitted here for clarity.

The *events* element contains a chronological history of the port.

```
{
  "success": true,
  "data": {
    "mbn": "01632960100",
    "status": "Porting Request Received",
    "status_code": "rcvd",
    "contact_email": "your.name@example.com",
    "lcp": "BT",
    "rh": "BT",
    "account_number": "NA1234B32",
    "billing_postcode": "A12 3BC",
    "type": "multi",
    "lines": "1",
    "channels": "1",
    "customer": { ... },
    "numbers": [ ... ],
    "events": [
      {
        "date": "2015-02-29 13:33:51",
        "status_code": "submitted_lcp",
        "status": "Submitted to LCP",
      },
      {
        "date": "2015-02-29 14:23:51",
        "status_code": "accepted",
        "status": "Porting Request Accepted",
      },
      {
        "date": "2015-02-29 13:33:51",
        "status_code": "rcvd",
        "status": "Porting Request Received",
        "info": "Order Submitted by API"
      }
    ]
  }
}
```

## View Porting LCPs

To be used when submitting new porting requests, this endpoint provides a list of available LCPs and the corresponding CUPIDs.

This endpoint provides a list of all CPs with which we have established non-geographic porting agreements. It is worth noting, however, that there may be other rangeholders we can port from where their range is hosted by one of the CPs in this list.

---

### /v3/porting/{ACCOUNT}/lcps

**GET**

Get list of LCP (Losing Communications Providers) we can port from and the corresponding CUPIDs

```
{
  "success": true,
  "cps": [
    {
      "cpName": "BT",
      "cupid": "1"
    },
    {
      "cpName": "Telephony Services",
      "cupid": "93"
    },
    {
      "cpName": "Gamma Telecom",
      "cupid": "31"
    }
  ]
}
```

**NB** When providing a CUPID in the API you can use either an integer (e.g. 1) or conventional format of the three digit string value (e.g. 001 for BT)

**When submitting a porting request where a number is hosted by a different network, please provide the CUPID of the hosting network (e.g. an ITSP using Telephony Services for number hosting should have the lcp\_cupid provided as "093")**

## Mobile Number Porting (“MNP”)

### New MNP [Mobile Number Porting] Submission

**/v3/porting/{ACCOUNT}/mnp**

**POST**

Submit a new Mobile Number port.

<i>msisdn</i>	MSISDN (mobile number) in E.164 format e.g. 447700900123
<i>pac</i>	PAC for this number (e.g. ABC1234567)
<i>contact_email</i>	A contact eMail address for the port (note this must be the address of the submitter <b>NOT your customer</b> )
<i>port_date</i> <i>[Optional]</i>	Porting date (CRD) in form YYYY-MM-DD If omitted, will port as soon as possible. (two working days from today)

```
{
  "success": true,
  "data": {
    "pac": "ABC1234567",
    "msisdn": "447700900123",
    "date_port": "2016-12-16",
    "ref": 43001
  }
}
```

The “ref” value returned corresponds to the ticket that will be used to track the progress of the porting request.



#### Submission Charge

There is a charge for this service.

Please see <https://simwood.com/rates> for full information

### View Port List

**/v3/porting/{ACCOUNT}/mnp**

**GET**

Get a list of currently outstanding, or recently completed mobile number ports

```
{
  "success": true,
  "data": [
    {
      "pac": "ABC1234567",
      "msisdn": "447700900123",
      "date_added": "2016-12-06",
      "date_updated": "2016-12-06",
      "date_port": "2016-12-08",
      "status_code": "pending",
      "status": "Pending",
      "ref": 43001
    },
    {
      ...
    }
  ]
}
```

## View Port Status

**/v3/porting/{ACCOUNT}/mnp/{PAC}**

**GET**

Get details of an MNP porting request

```
{
  "success": true,
  "data": {
    "pac": "ABC1234567",
    "msisdn": "447700900123",
    "date_added": "2016-12-06",
    "date_updated": "2016-12-06",
    "date_port": "2016-12-08",
    "status_code": "pending",
    "status": "Pending",
    "ref": 43001
  }
}
```

## MNP Exports

At this time it is not possible to generate a PAC from the API. Please raise a ticket via <https://support.simwood.com/> or by eMail to [team@simwood.com](mailto:team@simwood.com) to request a PAC for an MSISDN you have imported.

# Fax and SMS Messaging

## Inbound Fax Retrieval

Faxes received on your Simwood numbers can be retrieved via the API for a period of seven days from receipt and can be queried via the API

---

**/v3/fax/{ACCOUNT}/inbound/[ {NUMBER} ]**

**GET**

Lists last seven days of inbound faxes (optionally filtered by NUMBER)

```
{
  "success": true,
  "data": [
    {
      "hash": "7e2bb7bb87300ff83c657b04b07b8261"
      "time": "2015-02-30 12:34:56",
      "originator": "07700900123",
      "destination": "443301223000",
      "station": "MYFAX",
      "duration": 31,
      "pages": 1,
      "url": "/v3/.../7e2bb7bb87300ff83c657b04b07b8261.pdf"
    }
  ]
}
```

---

**/v3/fax/{ACCOUNT}/inbound/{NUMBER}/{HASH}**

**GET**

Retrieve specific fax in PDF format, where NUMBER is the destination fax number and HASH is the hash in the HTTP POST request made to your platform, or in the list retrieved above.

**DELETE**

Force deletion of the specified fax HASH received on NUMBER. Faxes will be automatically deleted after seven days if not manually deleted.

---

## Outbound SMS

<b>/v3/messaging/{ACCOUNT}/sms</b>					
<b>POST</b>	Send an SMS Message				
<i>to</i>	Recipient in e164 format (e.g. 447700900123)				
<i>from</i>	Originator number in e.164 format or alpha-numeric (maximum 11 characters if alphanumeric)				
<i>message</i>	Plain text message to send				
<i>flash</i> <i>[optional]</i>	Defaults to 0. If set to 1 message is sent as a 'flash' message (i.e. it will be displayed on the phone screen but not stored in the recipient's inbox, subject to handset and network support)				
<i>replace</i> <i>[optional]</i>	Defaults to 0. If set to 1 message is sent with an instruction to the handset to replace the previous. This must be set to 1 on both the original and replacement message.				
<i>concat</i> <i>[optional]</i>	An integer defining the maximum number of SMS to send to deliver your message where it is too long for the normal 160 plain text character limit of a single SMS. The default value of 1 will truncate your message if it is longer than 160 characters unless you set this parameter to a higher value.				
<i>report</i> <i>[optional]</i>	URL for delivery report. The following placeholders can be used;				
	<table border="1"><tr><td>%id%</td><td>Message ID</td></tr><tr><td>%status%</td><td>Status Code</td></tr></table>	%id%	Message ID	%status%	Status Code
%id%	Message ID				
%status%	Status Code				

If successful the message will be queued immediately and an id returned as follows;

```
[{"id": "02f150a0690171038624cc9d0e89207d"}]
```



### Message Charge

There is a charge for sending SMS messages.  
Please see <https://simwood.com/rates> for full information

## Outbound FAX

### /v3/messaging/{ACCOUNT}/fax

POST	Send an Fax				
<i>to[]</i>	Recipient number in e164 format (e.g.447700900123)  <i>This parameter may be repeated multiple times, the [] are part of the parameter name and must be included.</i>				
<i>from</i>	Originator number in e.164 format.				
<i>file[]</i>	The PDF file(s) to send. Please note valid content types below and ensure this is posted as a file (per RFC1867) (e.g. using an HTML form input type of "file")  <i>This parameter may be repeated multiple times, the [] are part of the parameter name and must be included.</i>				
<i>sendat [optional]</i>	Defaults to immediate, but a future date can be specified in the form YYYY-MM-DD hh:mm:ss				
<i>priority [optional]</i>	Jobs in the queue are processed in order of priority, then by submission date. Billing also depends on priority. 10 is default, less than 10 being more urgent, more than 10 being non-urgent.				
<i>report [optional]</i>	URL for delivery report. The following placeholders can be used;				
	<table><tbody><tr><td>%id%</td><td>Message ID</td></tr><tr><td>%status%</td><td>Status Code</td></tr></tbody></table>	%id%	Message ID	%status%	Status Code
%id%	Message ID				
%status%	Status Code				

If submission is successful, the fax will be queued immediately and response returned as an array containing the *number* and corresponding *id* of the submitted fax for each destination number;

```
[{"number": "441632000123", "id": "b902e8e46b91900af276f52995a3082e"}, {"number": "447700900123", "id": "71cea3179fdd13271a2ac14a366941f8"}]
```

### Fax Content Types

To ensure faxes arrive as intended, all faxes should be sent in PDF format with an application/pdf mime type.



#### Message Charge

There is a charge for sending fax messages.  
Please see <https://simwood.com/rates> for full information



## HTTP POST (Inbound Events)

Certain commands enable us to send data to your system over HTTP(S) in response to events rather than commands. These events are:

- Fax received on a number configured for HTTP POST
- SMS received on a number configured for HTTP POST
- The status of outgoing SMS messages
- The progress of outgoing faxes

In all cases the message will be sent to the URL you specified and will intelligently retry until a HTTP response code of 200 is received, which indicates success. Your application must therefore raise an error code in the HTTP header (e.g. 500) in the event of a problem rather than a plain text message in the body of the response. All responses are discarded.

### Received Fax

When a fax is received on a number configured to relay them by HTTP POST you will receive an HTTP POST message with a single parameter *payload* containing a JSON-encoded representation of the following;

Description																							
app	'fax_inbound' in this case. This is to facilitate re-use of the same status receiver your side for multiple applications.																						
hash	The unique hash for the fax, used as a reference and to retrieve the fax																						
data	An object containing the following parameters: <table><tbody><tr><td>id</td><td>Unique ID for the fax</td></tr><tr><td>time</td><td>Time fax was received (in form YYYY-MM-DD HH:mm:ss)</td></tr><tr><td>originator</td><td>The CLI of the calling fax machine (if available)</td></tr><tr><td>destination</td><td>The destination number (i.e. the Simwood number that received the fax)</td></tr><tr><td>status_msg</td><td>A human readable status message (e.g. "Ok")</td></tr><tr><td>status_code</td><td>The above in a form which should be quoted on any ticket raised.</td></tr><tr><td>bps</td><td>The speed of fax receipt in bits per second.</td></tr><tr><td>station</td><td>If present, the CSID of the remote fax machine.</td></tr><tr><td>duration</td><td>The billable time (in seconds) used to receive the fax.</td></tr><tr><td>pages</td><td>The number of pages received</td></tr><tr><td>url</td><td>The URL where the fax (in PDF format) can be retrieved, takes the form <i>https://api.simwood.com/v3/files/{ACCOUNT}/fax-NNNNNNNNNNNNNNNNNN</i></td></tr></tbody></table>	id	Unique ID for the fax	time	Time fax was received (in form YYYY-MM-DD HH:mm:ss)	originator	The CLI of the calling fax machine (if available)	destination	The destination number (i.e. the Simwood number that received the fax)	status_msg	A human readable status message (e.g. "Ok")	status_code	The above in a form which should be quoted on any ticket raised.	bps	The speed of fax receipt in bits per second.	station	If present, the CSID of the remote fax machine.	duration	The billable time (in seconds) used to receive the fax.	pages	The number of pages received	url	The URL where the fax (in PDF format) can be retrieved, takes the form <i>https://api.simwood.com/v3/files/{ACCOUNT}/fax-NNNNNNNNNNNNNNNNNN</i>
id	Unique ID for the fax																						
time	Time fax was received (in form YYYY-MM-DD HH:mm:ss)																						
originator	The CLI of the calling fax machine (if available)																						
destination	The destination number (i.e. the Simwood number that received the fax)																						
status_msg	A human readable status message (e.g. "Ok")																						
status_code	The above in a form which should be quoted on any ticket raised.																						
bps	The speed of fax receipt in bits per second.																						
station	If present, the CSID of the remote fax machine.																						
duration	The billable time (in seconds) used to receive the fax.																						
pages	The number of pages received																						
url	The URL where the fax (in PDF format) can be retrieved, takes the form <i>https://api.simwood.com/v3/files/{ACCOUNT}/fax-NNNNNNNNNNNNNNNNNN</i>																						

#### Please note that faxes expire for your own security

You must request the PDF from the URL provided within 5 minutes from successful receipt of the notification

## Received SMS

When an SMS is received on a number configured to relay them by using the **http** method you will receive an HTTP POST message of Content-type **application/x-www-form-urlencoded** with a single parameter *payload* containing a JSON-encoded representation of the following;

Please note your endpoint cannot require TLSv1.1 or higher. The legacy SMS service supports SSLv3 and TLSv1 only (or plain http) for TLS support please use the beta service described below (with type http\_json)

Description	
app	'mvno_inbound_sms' in this case. This is to facilitate re-use of the same status receiver your side for multiple applications.
data	An object containing the following parameters:
id	Unique ID
time	<i>Timestamp of message (in form YYYY-MM-DD HH:mm:ss) Where available from the originating network.</i>
originator	The MSISDN or Originator of the SMS message
destination	The destination number (i.e. the Simwood number that received the SMS)
message	The content of the SMS message
length	The length (characters) of the SMS message.

**For your security we do *not* retain message content after relay to your own platform**

## Received SMS (Beta)

BETA

When an SMS is received on a number configured to relay them by using the **http\_json** method you will receive an HTTP POST message of Content-type **application/json** with the following JSON body;

Your endpoint can support (or require) TLSv1.1, TLSv1.2.  
Please note that SSLv3 and TLSv1 are not supported by this service.

Description											
app	'sms_inbound_beta' in this case. <b>NB This field exists to facilitate re-use of the same status receiver your side for multiple applications. However as this is a beta this is subject to change and it is strongly recommended to use a dedicated URL to receive HTTP requests.</b>										
id	A unique ID representing this request e.g. si_422296075c882529362765a39aa75b19										
data	An object containing the following parameters: <table><tbody><tr><td>time</td><td>Timestamp of message (in form YYYY-MM-DD HH:mm:ss) Where available from the originating network.</td></tr><tr><td>originator</td><td>The MSISDN or Originator of the SMS message</td></tr><tr><td>destination</td><td>The destination number (i.e. the Simwood number that received the SMS)</td></tr><tr><td>message</td><td>The content of the SMS message</td></tr><tr><td>length</td><td>The length (characters) of the SMS message.</td></tr></tbody></table>	time	Timestamp of message (in form YYYY-MM-DD HH:mm:ss) Where available from the originating network.	originator	The MSISDN or Originator of the SMS message	destination	The destination number (i.e. the Simwood number that received the SMS)	message	The content of the SMS message	length	The length (characters) of the SMS message.
time	Timestamp of message (in form YYYY-MM-DD HH:mm:ss) Where available from the originating network.										
originator	The MSISDN or Originator of the SMS message										
destination	The destination number (i.e. the Simwood number that received the SMS)										
message	The content of the SMS message										
length	The length (characters) of the SMS message.										

An example full message is shown below

```
{
  "app": "sms_inbound_beta",
  "id": "si_422296075c882529362765a39aa75b19",
  "data": {
    "originator": "447700900321",
    "destination": "447700900123",
    "length": 12,
    "message": "Hello World."
  }
}
```

**For your security we do *not* retain message content after relay to your own platform**

## Outbound Fax Reports

Where a *report* field was specified for an outgoing fax, we will make a POST to the URL specified. If present, '%id%' in your URL will be replaced by the message ID returned in the response below and '%status%' will be replaced with the status code.

Additionally the POSTed data will contain a single parameter called *payload*, the value of which will be a JSON encoded representation of the following.

Description	
app	'faxsend_status' in this case. This is to facilitate re-use of the same status receiver your side for multiple applications.
id	The unique id for the fax returned when it was submitted.
data	An object containing the following parameters:
dtime	The date/time the status was generated.
status	A numeric indication of status. There will typically be three. <b>1</b> <b>Queued:</b> The fax has been queued to the fax server. For scheduled faxes this will happen at the specified time. <b>5</b> <b>Processing:</b> We have begun conversion and transmission of fax <b>10</b> <b>OK:</b> Fax transmission completed successfully <b>&gt;10</b> Fax transmission may not have completed successfully, please see the status_msg and supplementary fields for more details.
status_msg	A human readable status
station	If present, the CSID of the remote fax machine.
duration	The billable time (in seconds) spent on your fax. Note, as we employ an intelligent retry system this may be the sum of several transmissions.
errors	The number of transmission errors.
retries	The number of retries required.

## Outbound SMS Delivery Reports (DLRs)

Where a *report* field was specified for an outgoing SMS, we will make a POST to the URL specified. If present, '%id%' in your URL will be replaced by the message ID returned in the response below and '%status%' will be replaced with the status code.

Additionally the POSTed data will contain a single parameter called *payload*, the value of which will be a JSON encoded representation of the following.

Field	Description				
app	'sms_send_status' in this case. This is to facilitate re-use of the same status receiver your side for multiple applications.				
id	The unique id for the SMS returned when it was submitted.				
data	An object containing the following parameters: <table border="1"><tbody><tr><td>dttime</td><td>The date/time the status was generated. Note, this is the time the status was generated or received by Simwood. For example, in the case of a delivery report it will not be the time of actual delivery but the time we learned of delivery.</td></tr><tr><td>status</td><td>A numeric indication of status. There will typically be three for SMS.<ol style="list-style-type: none"><li><b>1 Received:</b> This will be raised immediately the SMS is committed to a queue at Simwood. Generally this will be simultaneous with your message submission but messages will be queued between the API and Simwood systems for performance.</li><li><b>2 Submitted:</b> The SMS has left Simwood. Again, this will generally coincide with your submission and status 1 above but queueing at every stage affords scalability and performance enhancement.</li><li><b>11 Delivered:</b> Confirmation received from the handset of delivery. Generally this will be 4 seconds or so after the above for a handset in signal.</li><li><b>12 Rejected:</b> Message was rejected, has not been delivered and will not be retried.</li><li><b>13 Error:</b> There was a syntax error with the message, usually relating to invalid destination address but in some cases disallowed source address.</li><li><b>14 Queued:</b> Message has been buffered for delivery in transit. This usually indicates the phone is off or out of service coverage.</li><li><b>18 SMSCa:</b> Message has been accepted by the SMSC. For networks where delivery reports are unavailable, this is the closest status to a delivery receipt.</li><li><b>26 SMSCr:</b> Message was rejected by the SMSC and will not be retried.</li></ol></td></tr></tbody></table>	dttime	The date/time the status was generated. Note, this is the time the status was generated or received by Simwood. For example, in the case of a delivery report it will not be the time of actual delivery but the time we learned of delivery.	status	A numeric indication of status. There will typically be three for SMS. <ol style="list-style-type: none"><li><b>1 Received:</b> This will be raised immediately the SMS is committed to a queue at Simwood. Generally this will be simultaneous with your message submission but messages will be queued between the API and Simwood systems for performance.</li><li><b>2 Submitted:</b> The SMS has left Simwood. Again, this will generally coincide with your submission and status 1 above but queueing at every stage affords scalability and performance enhancement.</li><li><b>11 Delivered:</b> Confirmation received from the handset of delivery. Generally this will be 4 seconds or so after the above for a handset in signal.</li><li><b>12 Rejected:</b> Message was rejected, has not been delivered and will not be retried.</li><li><b>13 Error:</b> There was a syntax error with the message, usually relating to invalid destination address but in some cases disallowed source address.</li><li><b>14 Queued:</b> Message has been buffered for delivery in transit. This usually indicates the phone is off or out of service coverage.</li><li><b>18 SMSCa:</b> Message has been accepted by the SMSC. For networks where delivery reports are unavailable, this is the closest status to a delivery receipt.</li><li><b>26 SMSCr:</b> Message was rejected by the SMSC and will not be retried.</li></ol>
dttime	The date/time the status was generated. Note, this is the time the status was generated or received by Simwood. For example, in the case of a delivery report it will not be the time of actual delivery but the time we learned of delivery.				
status	A numeric indication of status. There will typically be three for SMS. <ol style="list-style-type: none"><li><b>1 Received:</b> This will be raised immediately the SMS is committed to a queue at Simwood. Generally this will be simultaneous with your message submission but messages will be queued between the API and Simwood systems for performance.</li><li><b>2 Submitted:</b> The SMS has left Simwood. Again, this will generally coincide with your submission and status 1 above but queueing at every stage affords scalability and performance enhancement.</li><li><b>11 Delivered:</b> Confirmation received from the handset of delivery. Generally this will be 4 seconds or so after the above for a handset in signal.</li><li><b>12 Rejected:</b> Message was rejected, has not been delivered and will not be retried.</li><li><b>13 Error:</b> There was a syntax error with the message, usually relating to invalid destination address but in some cases disallowed source address.</li><li><b>14 Queued:</b> Message has been buffered for delivery in transit. This usually indicates the phone is off or out of service coverage.</li><li><b>18 SMSCa:</b> Message has been accepted by the SMSC. For networks where delivery reports are unavailable, this is the closest status to a delivery receipt.</li><li><b>26 SMSCr:</b> Message was rejected by the SMSC and will not be retried.</li></ol>				

## Change Log / Document History

Version	Date	Author	Notes
3.11.4	2015-05-02	RM	New availability zone filters on inbound configuration
3.11.3	2018-04-23	RM	New Inbound SMS Beta Removed deprecated number configuration method
3.11.2	2018-01-18	RM	Additional trunk limit controls
3.11.1	2017-12-20	RM	Changes to codec handling
3.11.0	2017-12-18	RM	Graduation of many "BETA" functions
3.10.7	2017-08-04	RM	Minor changes / corrections
3.10.6	2017-08-02	RM	Correction to porting LCP function
3.10.5	2017-07-20	RM	Additional information for inbound HTTP POST messages
3.10.4	2017-06-26	RM	Minor changes and clarifications
3.10.3	2017-06-19	RM	Improvements to BETA voice stack functions inc SDES support Number porting no longer considered BETA API
3.10.2	2017-06-15	RM	Additional BETA voice stack functions
3.10.1	2017-05-17	RM	Corrections and clarification of numbering endpoints Correction of supported fax formats
3.10.0	2017-03-01	RM	Deprecation of non-HTTPs (SSL/TLS) endpoints
3.9.25	2017-02-26	RM	Minor changes
3.9.24	2016-12-07	RM	MNP Mobile Number Porting Numbering changes (OTT) Document structure changes
3.9.23	2016-11-14	RM	<i>Withdrawn - not released</i>
3.9.22	2016-11-09	RM	Call routing improvements including NTS B-leg trunk
3.9.21	2016-08-04	RM	Trunk improvements
3.9.20	2016-07-14	RM	New trunk features
3.9.19	2015-02-09	RM	Expose recent rejections in API
3.9.18	2015-12-17	RM	Porting changes, additional functionality
3.9.17	2015-12-04	RM	Channel allocations
3.9.16	2015-12-01	RM	Improvements to porting
3.9.15	2015-10-30	RM	Deprecate <i>combined</i> rate endpoints in favour of CSV format Minor changes
3.9.14	2015-10-01	RM	New fax archive functionality
3.9.13	2015-09-26	RM	Minor changes / corrections
3.9.12	2015-09-15	RM	Minor changes / corrections
3.9.11	2015-08-31	RM	Remove deprecated routing functions Updated references for Simwood Mobile
3.9.10	2015-07-22	RM	Revisions to porting
3.9.9	2015-07-02	RM	Added CSV Ratecards Additional minor changes

<b>Version</b>	<b>Date</b>	<b>Author</b>	<b>Notes</b>
<b>3.9.7</b>	2014-12-10	RM	Added notification history Identify chargeable endpoints
<b>3.9.6</b>	2014-12-04	RM	Improvements to Trunk Balances Added per-trunk realtime call information
<b>3.9.5</b>	2014-11-21	RM	Porting submission now supports CRD (port_date) Additional "instant" CDR reports Last call function on numbering Inbound Trunks
<b>3.9.4</b>	2014-11-12	RM	Add porting submission
<b>3.9.3</b>	2014-10-20	RM	Add retrieval of PDF invoices
<b>3.9.2</b>	2014-10-09	RM	Minor changes
<b>3.9.1</b>	2014-08-07	RM	Clarification of Trunk Configuration
<b>3.9</b>	2014-08-06	RM	Initial Release of 3.9