Simplenote API2

Authentication

Simplenote uses a simple token based authentication system. All methods will require a valid authentication token to process.

Obtaining a token

Call the /api/login method supplying two fields, email and password. These should be the same as what you created your account with in the Simplenote application. The server will respond by attempting to set a cookie named 'auth' with the token as the value. Also, the HTTP response body will contain the token.

HTTP POST to the following URL:

```
https://simple-note.appspot.com/api/login
```

The body of the request should contain:

```
email=[email address]&password=[password]
```

The entire request body should be encoded in base64.

Using the token

Subsequent calls to the API methods will check that either the 'auth' cookie is present with a valid token or that the token is supplied as a query parameter (&auth=[token]). Once issued, the token is valid for 24 hours. The email address should also be passed with each request either as a query parameter or as a cookie.

Security

The Simplenote API allows either regular HTTP or encrypted HTTPS requests, except for the login method which can only be accessed through HTTPS.

Working with notes

Each note has a unique string associated with it, using this *key*, individual notes can be modified, created, retrieved, or deleted. Once a note has been created it resides at the note URL and can be modified using that URL (/api2/data/[note key]).

Note object:

```
{
      key : (string, note identifier, created by server),
      deleted : (bool, whether or not note is in trash),
      modifydate : (last modified date, in seconds since epoch),
      createdate : (note created date, in seconds since epoch),
      syncnum : (integer, number set by server, track note changes),
      version : (integer, number set by server, track note content
changes),
      minversion : (integer, number set by server, minimum version
available for note),
      sharekey : (string, shared note identifier),
      publishkey : (string, published note identifier),
      systemtags : [ (Array of strings, some set by server) ],
      tags : [ (Array of strings) ],
      content : (string, data content)
}
```

Properties in green can be set by the client. Other properties are returned only by the server.

*New properties may be returned in the note object without notice. Please make sure your code can handle this! We'll make sure to announce any changes for existing properties though.

systemtags currently can only contain 'pinned' and 'unread'. If the 'pinned' system tag is present, that note should be sorted at the top of the list. If the 'unread' system tag is present, it should be removed when that note is read by the user. To remove a system tag just do an update of the note with the new *systemtags* array.

sharekey, and *publishkey*, can currently only be generated by the iPad/iPhone apps, we'll be exposing this functionality soon via the API.

tags currently should not contain any spaces. [We'll be adding a more robust tags API]

Creating a note

To create a note, send a POST request to the following URL:

```
https://simple-note.appspot.com/api2/data?auth=[auth_token]&email=[email]
```

The body of the POST request should be the new note object. At minimum, the *content* property must be supplied. This will create a note and return a note object with a new note identifier (in the *key* property).

Retrieving a note

To retrieve a note, send a GET request to the note URL:

```
https://simple-note.appspot.com/api2/data/[note_key]?auth=[auth_token]&email=[email]
```

The *key* parameter must be supplied, *auth* may be omitted if the 'auth' cookie is present, likewise for *email*.

The response body will contain a note object.

Updating a note

To update a note, a POST request is made to the note URL:

```
https://simple-note.appspot.com/api2/data/[note_key]?auth=[auth_token]&email=[email]
```

The body should contain a partial note object. The minimum properties to send should be the *content*. To support merging, last *version* number received from the server for this note should be sent. If *version* is omitted, the new contents will automatically overwrite the current note contents, even if changes have been made on other clients.

The response will have the updated note object, including the new *version* and *syncnumber* properties assigned by the server. If no changes have been made by other clients since the last update, then *content* will not be included in the response note object. If there have been other changes then *content* will be returned and the client should update their local store accordingly.

modifydate, and *createdate* can be set by the client. If not provided, the server will use the current time for these values. Sorting in the Simplenote clients are done based on these values and should be sent especially if the change did not happen at the time of updating the note on the server.

Deleting a note

The latest Simplenote includes a Trash feature. Deleting a note now means to send a note to the Trash, where it can also be restored. This is done using the *deleted* property of the note object. To delete a note, just update a note and set the *deleted* property to 1.   Restoring a note would mean to update and set the *deleted* property back to 0.

To delete a note from the trash (permanent delete), make a DELETE request to the note URL:

```
https://simple-note.appspot.com/api2/data/[note key]?auth=[auth token]&email=[email]
```

Getting the note index

To make it simple to synchronize your notes with other applications, the API provides an easy way to get an index of your notes. This method will return a JSON object with a count of the number of note objects and an array of the note objects themselves. The note objects are the same as retrieved by a normal GET call on a note, with the *content* omitted. Since there is a maximum number of notes that can be returned in one call, if there are more notes in the index, then the *mark* parameter will be set and it should be used in the next index call to continue retrieving notes. For example, if the user has 300 notes, this would require at least 3 calls to retrieve all notes.

To get the index, make a GET request to the following URL:

```
https://simple-note.appspot.com/api2/index?length=[number of notes]&mark=[bookmark key]&since=[time value]&auth=[auth token]&email=[email]
```

The *auth* parameter is optional and is needed only if the ‘auth’ cookie is not present. The *email* parameter is also optional and is needed only if the ‘email’ cookie is not present. The *length* parameter determines the maximum number of note objects to return (maximum is 100). If there are more note objects, then the response will include a *mark* parameter. This is a note key and should be used as the value of the *mark* parameter in a subsequent call to retrieve the next set of note results.

The *since* parameter is a time value (like *modifydate* or *createdate*) and instead of returning all notes will return all notes that have been changed since the time specified.

Sample Index Response:

```
{
 "count": 1,
 "data":
      [
      {"modifydate": "1285591393.044700",
       "tags": [],
       "deleted": 0,
       "createdate": "1285591393.044700",
       "systemtags": [],
       "version": 1,
       "syncnum": 1,
       "key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA",
       "minversion": 1}
      ]
}
```

Examples

1. Get index

```
GET /api2/index
```

Response:

```
{"count": 0, "data": []}
```

2. Create a note

```
{"content" : "New note!"}
POST /api2/data
```

Response:

```
{"modifydate": "1285591393.044700", "tags": [], "deleted": 0,
"createdate": "1285591393.044700", "systemtags": [], "version": 1,
"syncnum": 1, "key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA",
"minversion": 1}
```

3. Get index again should show newly created note

```
GET /api2/index
```

Response:

```
{"count": 1, "data": [{"modifydate": "1285591393.044700", "tags": [],
"deleted": 0, "createdate": "1285591393.044700", "systemtags": [],
"version": 1, "syncnum": 1, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA", "minversion": 1}]}
```

4. Retrieve note using note url

```
GET /api2/data/agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA
```

Response:

```
{"modifydate": "1285591393.044700", "tags": [], "deleted": 0,
"createdate": "1285591393.044700", "systemtags": [], "content": "New
note!", "version": 1, "syncnum": 1, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA", "minversion": 1}
```

5. Update note with new *content*, server returns new *version* and new *syncnum*

```
{"content" : "New note! with change", "version" : 1}
POST /api2/data/agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA
```

Response:

```
{"modifydate": "1285591647.688616", "tags": [], "deleted": 0,
"createdate": "1285591393.044700", "systemtags": [], "version": 2,
"syncnum": 2, "key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA",
"minversion": 1}
```

6. Send note to trash by setting *deleted* to 1, only *syncnum* is changed because *content* was not modified

```
{"deleted" : 1}
POST /api2/data/agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA
```

Response:

```
{"modifydate": "1285591687.123246", "tags": [], "deleted": 1,
"createdate": "1285591393.044700", "systemtags": [], "version": 2,
"syncnum": 3, "key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA",
"minversion": 1}
```

7. Permanently remove note from trash, only works if note is deleted

```
DELETE /api2/data/agtzaW1wbGUtbm90ZXIKCxIETm90ZRhsDA
```

Response:
Nothing returned, status code 200

8. Create 4 new notes (not shown), then retrieve index contains all 4 notes

```
GET /api2/index
```

Response:

```
{"count": 4, "data": [{"modifydate": "1285617794.872000", "tags": [],
"deleted": 0, "createdate": "1285617793.685000", "systemtags": [],
"version": 1, "syncnum": 1, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhvDA", "minversion": 1}, {"modifydate":
"1285617791.608000", "tags": [], "deleted": 0, "createdate":
"1285617790.018000", "systemtags": [], "version": 1, "syncnum": 1,
"key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhwDA", "minversion": 1},
{"modifydate": "1285617788.655000", "tags": [], "deleted": 0,
"createdate": "1285617787.044000", "systemtags": [], "version": 2,
"syncnum": 2, "key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhtDA",
"minversion": 1}, {"modifydate": "1285617783.647000", "tags": [],
"deleted": 0, "createdate": "1285617780.555000", "systemtags": [],
"version": 1, "syncnum": 1, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhuDA", "minversion": 1}]}
```

9. Retrieve index again using *length* = 2, response contains only 2 notes and also *mark* parameter to continue retrieving notes

```
GET /api2/index?length=2
```

Response:

```
{"count": 2, "data": [{"modifydate": "1285617794.872000", "tags": [],
"deleted": 0, "createdate": "1285617793.685000", "systemtags": [],
"version": 1, "syncnum": 1, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhvDA", "minversion": 1}, {"modifydate":
"1285617791.608000", "tags": [], "deleted": 0, "createdate":
"1285617790.018000", "systemtags": [], "version": 1, "syncnum": 1,
"key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhwDA", "minversion": 1}],
"mark": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhtDA"}
```

10. Continue retrieving notes using bookmark *mark*. Response contains remaining 2 notes and *mark* parameter is omitted since there are no more notes.

```
GET /api2/index?length=2&mark=agtzaW1wbGUtbm90ZXIKCxIETm90ZRhtDA
```

Response:

```
{"count": 2, "data": [{"modifydate": "1285617788.655000", "tags": [],
"deleted": 0, "createdate": "1285617787.044000", "systemtags": [],
"version": 2, "syncnum": 2, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhtDA", "minversion": 1}, {"modifydate":
"1285617783.647000", "tags": [], "deleted": 0, "createdate":
"1285617780.555000", "systemtags": [], "version": 1, "syncnum": 1,
"key": "agtzaW1wbGUtbm90ZXIKCxIETm90ZRhuDA", "minversion": 1}]}
```

11. Using same bookmark *mark* from example 10, retrieve index with maximum *length* = 1. Response contains only 1 note and another *mark* to retrieve last remaining note.

```
GET /api2/index?length=1&mark=agtzaW1wbGUtbm90ZXIKCxIETm90ZRhtDA
```

Response:

```
{"count": 1, "data": [{"modifydate": "1285617788.655000", "tags": [],
"deleted": 0, "createdate": "1285617787.044000", "systemtags": [],
"version": 2, "syncnum": 2, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhtDA", "minversion": 1}], "mark":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhuDA"}
```

12. Using bookmark *mark* from example 11, retrieve last remaining note in index, response does not contain *mark* since there are no more notes.

```
GET /api2/index?length=2&mark=agtzaW1wbGUtbm90ZXIKCxIETm90ZRhuDA
```

Response:

```
{"count": 1, "data": [{"modifydate": "1285617783.647000", "tags": [],
"deleted": 0, "createdate": "1285617780.555000", "systemtags": [],
"version": 1, "syncnum": 1, "key":
"agtzaW1wbGUtbm90ZXIKCxIETm90ZRhuDA", "minversion": 1}]}
```

Checking for changes

To check for changes you can use *syncnum* and *version*. *syncnum* will increment whenever there is any change to a note, content change, tag change, etc. *version* will increment whenever the *content* property is changed. You should store both these numbers in your client to track changes and determine when a note needs to be updated or saved.

Psuedo code algorithm for syncing:

```
  For any note changed locally (including new notes):
    Save note to server, update note with response (new syncnum, version, possibly new content)

  Get note index
  For each remote note,
    if remote syncnum > local syncnum,
      Retrieve note, update note with response
    if new note (key is not in local store),
      Retrieve note, update note with response

  For each local note not in index,
    Permanent delete, remove note from local store
```

Response codes

The HTTP response status code should always be checked to be sure that an operation has completed successfully. Successful operations return with a status code of 200. Any other code is an error.

Some common error codes:

400 - Bad Request
401 - Unauthorized
403 - Forbidden
404 - Not Found
500 - Server Error

Best practices for error codes in common API methods:

`https://simple-note.appspot.com/api/login`

Any error code, retry or ask for password

`https://simple-note.appspot.com/api/index`

401 - User invalid, either authorization key expired or user incorrect, retry login
Any other error, retry

`https://simple-note.appspot.com/api/note`

401 - User invalid, either authorization key expired or user incorrect, retry login
404 - Note does not exist, do not retry
Any other error, retry

`https://simple-note.appspot.com/api/delete`

401 - User invalid, either authorization key expired or user incorrect, retry login
404 - Note does not exist, do not retry
Any other error, retry