Documentation API REST

Version 1.0, avril 2025

NAMEBAY Société Anonyme Monégasque – chez CATS 28 bd Princesse Charlotte - MC 98000 MONACO
Tél. : 377 97 70 61 64 - Fax : 377 97 97 21 13 - e-mail : contact@namebay.com – http://www.namebay.com
Capital social 150 000 Euros - CODE NIS N°6203 Z 08913 - RCI N°99 S 03660 - TVA intracommunautaire FR 36000052679

Table des matières

Noms de domaine

Commande et renouvellement

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```


The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Check availability

Check the availability of a domain with the domain availability api.

Example:

GET [base_url]/api/Domain/[domain]/Availability

Return:


```
{
   "Forbidden": false,
   "Reserved": false,
   "ForbiddenReason": null,
   "ReservedReason": null,
   "Name": "[domain]",
   "Status": "AvailablePerDomainCheck",
   "Notices": "none",
   "AvailabilityResponse": {
      "code": 1,
      "notices": 1
   },
   "Catchable": false,
   "CatchableTld": false,
   "Error": null
}
```

Ordering domains

Order a new domain

Order a domain using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the domain for.

When ordering a new domain, the period is the number of years and the quantity is always 1.

The produitID depends on the tld. Product ids can be found using the product api.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:

```
{
   "IpAddress": "::1",
   "Items": [
      {
         "ProduitID": 9105,
         "Period": 1,
         "Quantity": 1,
         "Confirm": true,
         "Attributes": {
            "domain": "[domain]",
            "ns1": "dns1.namebay.com",
            "ns2": "dns2.namebay.com",
            "registrant_handle": "[user_handle]"
         }
      }
   ]
}
```

Return:


```
{
   "Panier": null,
   "OriginalPanierId": 0,
   "Transaction": {
      "TransactionID": "NBAY-RESTAPI-6abf88005032025042139",
      "Total_HT": 0.0000,
      "Total_TTC": 0.0000,
      "PanierID": 0,
      "OpDetails": [
         {
            "Id": 37.0,
            "ProduitID": 9105,
            "Period": 1.0,
            "Quantity": 1,
            "Pu": 0.0000,
            "IsUpdate": false,
            "NewQuantity": null,
            "PeriodLeft": null,
            "OldProduitID": null,
            "OldPu": null,
            "Total_HT": 0.0000,
            "OpDetailPacks": [
               {
                  "Id": 30.0,
                  "DetailId": 37.0,
                  "ProduitID": 9105,
                  "Pu": 0.0000,
```

```
            "CartItems": [
              {
                "Id": 30.0,
                "DetailPackId": 30.0,
                "Name": "[domain]",
                "Handle": null,
                "Class": "domain",
                "Attributes": {
                  "admin_id": "1111",
                  "billing_id": "1111",
                  "ns1": "dns1.namebay.com",
                  "ns2": "dns2.namebay.com",
                  "registrant_id": "1111",
                  "tech_id": "1111",
                  "tld_id": "31"
                }
              }
            ]
          }
        ]
      }
    ],
    "AutoRenew": false,
    "ModePaiementID": 3,
    "OpState": 1,
    "Errors": [],
    "IpAddress": "::1",
    "Currency": "EUR"
```

```
  }
}
```


Get the domain handle

Get the domain handle using the OwnedProduct api.

GET [base_url]/api/OwnedProduct?user=[user_handle]&object_class=domain&object_name=[domain]

Return:


```
[
  {
    "Handle": "[object_handle]",
    "ParentHandle": null,
    "Name": "[domain]",
    "Class": "domain",
    "StatusID": 1,
    "Status": "REGISTERED",
    "CreDate": "2025-02-21T00:00:00",
    "ExpDate": "2026-02-21T00:00:00",
    "TransferInDate": null,
    "ProductId": 9105,
    "Quantity": 1,
    "ContactTypes": [
      "O",
      "A",
      "T",
      "B",
      "R"
    ],
    "Products": [
```

```
        {
          "Id": 9105,
          "Name": " Enregistrement de nom de domaine  + Présence locale eu ",
          "Type": "Domaine",
          "SubType": "Creation",
          "TransactionId": "NBAY-RESTAPI-6abf88005032025042139",
          "Active": false
        }
      ],
      "Children": null,
      "Error": null
    }
]
```

Renew a domain

Use the panier api to renew existing domains.

When renewing a domain you will need to use the product id of the renewal which is different to the product for registration. As with the original order the period is the number of years and the quantity is always 1. The object_handle of the domain to renew is supplied in the attributes.

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
   "IpAddress": "::1",
   "Items": [
      {
         "ProduitID": 9205,
         "Period": 1,
         "Quantity": 1,
         "Confirm": true,
         "Attributes": {
            "domain": "[domain]",
            "object_handle": "[object_handle]"
         }
      }
   ]
}
```


Return:


```
{
   "Panier": null,
   "OriginalPanierId": 0,
   "Transaction": {
```

```
"TransactionID": "NBAY-RESTAPI-dd3717607032025102908",
"Total_HT": 17.0700,
"Total_TTC": 20.4840,
"PanierID": 0,
"OpDetails": [
  {
    "Id": 30.0,
    "ProduitID": 9205,
    "Period": 1.0,
    "Quantity": 1,
    "Pu": 17.0700,
    "IsUpdate": false,
    "NewQuantity": null,
    "PeriodLeft": null,
    "OldProduitID": null,
    "OldPu": null,
    "Total_HT": 17.0700,
    "OpDetailPacks": [
      {
        "Id": 24.0,
        "DetailId": 30.0,
        "ProduitID": 9205,
        "Pu": 17.0700,
        "CartItems": [
          {
            "Id": 24.0,
            "DetailPackId": 24.0,
            "Name": "[domain]",
```

```
                    "Handle": "[object_handle]",
                    "Class": "domain",
                    "Attributes": {}
                }
            ]
        }
    ]
  }
],
"AutoRenew": false,
"ModePaiementID": 3,
"OpState": 1,
"Errors": [],
"IpAddress": "::1",
"Currency": "EUR"
}
}
```


Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Domain

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedProduct

https://rest.namebay.com/help#Product

Transfert de domaine

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```


The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Check transferability

Check the transferability of a domain with the domain transferable api. User_handle is either your own user handle or the handle of the user you are ordering the certificate for. You will need the auth code of the domain you want to transfer.

Example:

GET
[base_url]/api/Domain/[domain]/Transferable?user=[usere_handle]&authcode=[auth_code]

Return:

{

"IsTransferable": **true**,

"InTransfer": **false**,

"ValidDate": **true**,

"Status": "ok",

"Handle": **null**,

"Name": "[domain]",

"Direction": "In",

"Created": "2020-12-08T00:00:00",

"Message": "This domain is transferable"

}

Ordering a transfer

Order a domain transfer using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the domain for.

When ordering a domain transfer, the period is the number of years and the quantity is always 1.

The produitID depends on the tld. Product ids can be found using the product api.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 33,
      "Period": 1,
      "Quantity": 1,
      "Confirm": true,
      "Attributes": {
        "domain": "[domain]",
        "modify_ns": false,
        "auth_info": "[auth_code]"
      }
    }
  ]
}
```


Return:


```
{
"Panier": null,
```

```
"OriginalPanierId": 0,
"Transaction": {
  "TransactionID": "NBAY-RESTAPI-8b8f10025032025052144",
  "Total_HT": 0.0000,
  "Total_TTC": 0.0000,
  "PanierID": 0,
  "OpDetails": [
    {
      "Id": 31.0,
      "ProduitID": 33,
      "Period": 1.0,
      "Quantity": 1,
      "Pu": 0.0000,
      "IsUpdate": false,
      "NewQuantity": null,
      "PeriodLeft": null,
      "OldProduitID": null,
      "OldPu": null,
      "Total_HT": 0.0000,
      "OpDetailPacks": [
        {
          "Id": 25.0,
          "DetailId": 31.0,
          "ProduitID": 33,
          "Pu": 0.0000,
          "CartItems": [
            {
              "Id": 25.0,
```

```
                "DetailPackId": 25.0,
                "Name": "[domain]",
                "Handle": null,
                "Class": "domain",
                "Attributes": {
                  "admin_id": "111222333",
                  "auth_info": "[auth_code]",
                  "billing_id": "111222333",
                  "modify_ns": "false",
                  "registrant_id": "111222333",
                  "tech_id": "111222333",
                  "tld_id": "5"
                }
              }
            ]
          }
        ]
      }
    ],
    "AutoRenew": false,
    "ModePaiementID": 3,
    "OpState": 1,
    "Errors": [],
    "IpAddress": "::1",
    "Currency": "EUR"
  }
}
```

Check progress

You can check the progress of the transfer with the message polling api or with the transaction api.

There will be 2 messages added to the polling queue, domainTransferredInWaitingForRegistrarAnswer when the current registrar has been notified and domainTransferredIn when the transfer is complete. See Namebay message polling for details on how to use message polling.

You can also track the progress of a ticket with the transcation api using the transaction id of the order.

Example:

GET [base_url]/api/Transaction/[transaction_id]/Ticket

Return:


```
[
  {
    "ObjectHandle": "[object_handle]",
    "ObjectName": "[domain]",
    "ClassName": "domain",
    "TicketState": "New",
    "CreateDate": "2025-03-26T16:15:45.85",
    "ExpDate": null,
    "ProductId": 53,
    "Period": 1.0,
    "Quantity": 1,
    "ObjectStatus": 10,
    "Id": 14.0,
    "OpType": "T",
    "TldId": 5,
    "Remark": null,
    "PriorityLevel": 0,
    "RequestDate": "2025-03-26T17:13:25.157",
```

"ModDate": "2025-03-26T17:13:25.157",

"LastProcessingDate": **null**,

"NextProcessingDate": "2025-03-26T17:13:25.157",

"LastProcessingResultId": **null**,

"Version": 2,

"ClientStateId": **null**,

"Extensions": {

"#TicketVersion": "2.0",

"AdminHandle": "[user_handle]",

"BillingHandle": "[user_handle]",

"OwnerHandle": "[user_handle]",

"TechHandle": "[user_handle]",

"ResellerHandle": "[user_handle]",

"DomainName": "[domain]",

"CurrentRegistrar": "Gandi SAS",

"CurrentAdminMail": "UNKOWN@UNKOWN.FR",

"AuthCode": "[auth_code]",

"modifyNs": "False",

"ns1": "DNS1.NAMEBAY.COM",

"ns2": "DNS2.NAMEBAY.COM",

"admin_id": "111222333",

"auth_info": "[auth_code]",

"billing_id": "111222333",

"modify_ns": "false",

"registrant_id": "111222333",

"tech_id": "111222333",

"tld_id": "5"

},

"Logs": **null**

}

]

Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Domain

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#Product

https://rest.namebay.com/help#Poll

https://rest.namebay.com/help#Transaction

Transfert de registrant

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:

{

"access_token": "[token_string]",

"token_type": "bearer",

"expires_in": 1209599,

"userName": "[logname]",

"audience": "namebay_1",

".issued": "Tue, 02 Aug 2022 08:20:38 GMT",

".expires": "Tue, 16 Aug 2022 08:20:38 GMT"

}

The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Check transferability

Check the transferability of a domain with the domain transferable api. User_handle is either your own user handle or the handle of the user you are ordering the certificate for. You will need the auth code of the domain you want to transfer.

Example:

GET
[base_url]/api/Domain/[domain]/Transferable?user=[usere_handle]&authcode=[auth_code]

Return:

{

"IsTransferable": **true**,

"InTransfer": **false**,

"ValidDate": **true**,

"Status": "ok",

"Handle": **null**,

"Name": "[domain]",

"Direction": "In",

"Created": "2020-12-08T00:00:00",

"Message": "This domain is transferable"

}

Ordering a transfer

Order a domain registrant transfer using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the domain for.

When ordering a registrant transfer, the period is the number of years and the quantity is always 1.

The produitID depends on the tld. Product ids can be found using the product api.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 12438,
      "Period": 1,
      "Quantity": 1,
      "Confirm": true,
      "Attributes": {
        "domain": "[domain]",
        "object_handle": "[object_handle]",
        "registrant_handle": "[user_handle]"
      }
    }
  ]
}
```


Return:


```
{
  "Panier": null,
```

```
"OriginalPanierId": 0,
"Transaction": {
  "TransactionID": "NBAY-RESTAPI-aa8c15b28032025101624",
  "Total_HT": 0.0000,
  "Total_TTC": 0.0000,
  "PanierID": 0,
  "OpDetails": [
    {
      "Id": 78,
      "ProduitID": 12438,
      "Period": 1,
      "Quantity": 1,
      "Pu": 0.0000,
      "IsUpdate": false,
      "NewQuantity": null,
      "PeriodLeft": null,
      "OldProduitID": null,
      "OldPu": null,
      "Total_HT": 0.0000,
      "OpDetailPacks": [
        {
          "Id": 69,
          "DetailId": 78,
          "ProduitID": 12438,
          "Pu": 0.0000,
          "CartItems": [
            {
              "Id": 69,
```

```
                    "DetailPackId": 69,
                    "Name": "[domain]",
                    "Handle": "[object_handle]",
                    "Class": "domain",
                    "Attributes": {
                       "registrantHandle": "[user_handle]"
                    }
                 }
              ]
           }
        ]
     }
  ],
  "AutoRenew": false,
  "ModePaiementID": 3,
  "OpState": 1,
  "UserHandle": null,
  "UserId": 111222333,
  "Errors": [],
  "IpAddress": "::1",
  "Currency": "EUR"
 }
}
```

Check progress

You can check the progress of the transfer with the message polling api or with the transaction api.

There will be 3 messages added to the polling queue, domainOwnerChangeWaitingForPreviousOwnerApproval when the current owner has been notified domainOwnerChangeWaitingForNewOwnerApproval when the new owner has been notified and domainOwnerChanged when the transfer is complete. See Namebay message polling for details on how to use message polling.

You can also track the progress of a ticket with the transcation api using the transaction id of the order.

Example:

GET [base_url]/api/Transaction/[transaction_id]/Ticket

Return:


```
[
  {
    "ObjectHandle": "[object_handle]",
    "ObjectName": "[domain]",
    "ClassName": "domain",
    "TicketState": "New",
    "CreateDate": "2025-03-28T11:37:55.107",
    "ExpDate": null,
    "ProductId": 12438,
    "Period": 1.0,
    "Quantity": 1,
    "ObjectStatus": 10,
    "Id": 15.0,
    "OpType": "P",
    "TldId": 5,
    "Remark": null,
    "PriorityLevel": 0,
```

```
    "RequestDate": "2025-03-28T11:37:55.133",
    "ModDate": "2025-03-28T11:37:55.133",
    "LastProcessingDate": null,
    "NextProcessingDate": "2025-03-28T11:37:55.133",
    "LastProcessingResultId": null,
    "Version": 2,
    "ClientStateId": null,
    "Extensions": {
      "#TicketVersion": "2.0",
      "AuthCode": "",
      "AdminHandle": "[user_handle]",
      "BillingHandle": "[user_handle]",
      "OwnerHandle": "[user_handle]",
      "TechHandle": "[user_handle]",
      "ResellerHandle": "[user_handle]",
      "DomainName": "[domain]",
      "ns1": "DNS1.NAMEBAY.COM",
      "ns2": "DNS2.NAMEBAY.COM"
    },
    "Logs": null
  }
]
```


Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Domain

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#Product

https://rest.namebay.com/help#Poll

https://rest.namebay.com/help#Transaction

Gérer les zones DNS

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```


The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Managing zones

Create a record

Create a record using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are creating the record for.

Example:

POST [base_url]/api/Zone/[zone_handle]/Record?user=[user_handle]

Payload:


```
{
   "host": "mail",
   "data": "1.1.1.1",
   "type": "A"
}
```


Return:


```
{
   "Id": 13,
   "ZoneId": 335354,
   "Host": "mail",
   "Data": "1.1.1.1",
   "Type": "A",
   "Weight": null,
   "DateCreated": "2025-04-16T11:02:27.03",
   "NaptrPreference": null,
   "NaptrFlag": null,
   "NaptrRegExp": null
}
```

Update a record

Update a record using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are updating the record for. Id is the id of record to update.

Example:

PUT [base_url]/api/Zone/[zone_handle]/Record/[id]?user=[user_handle]

Payload:


```
{
   "host": "mail",
   "data": "2.2.2.2",
   "type": "A"
}
```


Return:


```
{
   "Id": 13,
   "ZoneId": 335354,
   "Host": "mail",
   "Data": "2.2.2.2",
   "Type": "A",
   "Weight": null,
   "DateCreated": "2025-04-16T11:19:50.593",
   "NaptrPreference": null,
   "NaptrFlag": null,
   "NaptrRegExp": null
}
```

Delete a record

Delete a record using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are deleting the record for. Id is the id of record to delete.

Example:

DELETE [base_url]/api/Zone/[zone_handle]/Record/[id]?user=[user_handle]

Update the dns sec status of a zone

Update the dns sec status using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are updating the record for.

Example:

PUT [base_url]/api/Zone/[zone_handle]/DnsSec?user=[user_handle]

Payload:

{

"isSec": **true**

}

Return:

{

"Zone": "nd250220.fr",

"IsSec": **true**,

"IsSign": **false**,

"SignDate": **null**

}

Create ns keys for external zones

For zones that are not installed at namebay but where the domain is, you can add dns sec information using the zone nskey api.

Create a nskey using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are creating the record for.

Example:

POST [base_url]/api/Zone/[zone_handle]/NsKey?user=[user_handle]

Payload:


```
{
   "keytag": "1111",
   "algorithm": 8,
   "digesttype": 2,
   "digest": "[digest]"
}
```


Return:


```
{
   "Id": 3,
   "ZoneName": "[zone_name]",
   "KeyTag": "1111",
   "Algorithm": 8,
   "DigestType": 2,
   "Digest": "[digest]",
   "DateInsert": "2025-04-16T13:34:02.1733333",
   "Active": true,
   "NbNs": false
}
```


For some tlds (e.g. .eu, .be and .tel) you need also to provide the ns public key.

Example:

POST [base_url]/api/Zone/[zone_handle]/NsPubKey?user=[user_handle]

Payload:


```
{
   "keytag": "1111",
```

"pubkey": "[public_key]"

}

Return:

{

"Id": 3,

"ZoneName": "[zone_name]",

"KeyTag": "1111",

"PubKey": "[public_key]",

"DateInsert": "2025-04-16T13:43:52.0566667",

"NbNs": **false**

}

Import zones

You can import a new zone using the bid file contents and a list of name servers using the zone import api. User_handle is either your own user handle or the handle of the user you are creating the zone for.

Example:

POST [base_url]/api/Zone/Import?user=[user_handle]

Payload:

{

"name": "[domain]",

"binddata": "[bin_file_contents]",

"nslist": [

"[name_server1]",

"[name_server2]"

]

}

Return:


```
{
   "Handle": "[zone_handle]",
   "Name": "[domain]",
   "SoaExpire": 1209600,
   "SoaMinimum": 3600,
   "SoaPrimaryNs": "[name_server1]",
   "SoaSerial": "Apr 16 2025 11:37AM",
   "SoaRefresh": 10800,
   "SoaRespPerson": "[email_address]",
   "SoaRetry": 600,
   "Ttl": 36000
}
```


You can also replace a zone by providing a zone handle with the PUT zone import api.

Example:

PUT [base_url]/api/Zone/Import/[zone_handle]?user=[user_handle]

Payload:


```
{
   "name": "[domain]",
   "binddata": "[bin_file_contents]",
   "nslist": [
      "[name_server1]",
      "[name_server2]"
   ]
}
```

Return:


```
{
   "Handle": "[zone_handle]",
   "Name": "[domain]",
   "SoaExpire": 1209600,
   "SoaMinimum": 3600,
   "SoaPrimaryNs": "[name_server1]",
   "SoaSerial": "Apr 16 2025 11:37AM",
   "SoaRefresh": 10800,
   "SoaRespPerson": "[email_address]",
   "SoaRetry": 600,
   "Ttl": 36000
}
```


Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedProduct

https://rest.namebay.com/help#Product

https://rest.namebay.com/help#Zone

Certificats SSL

Commande et validation des certificats en utilisant META (http) validation

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```


The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Ordering certificates

Order a new certificate

Order a certificate using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the certificate for.

When ordering a new certificate, the period is the number of years and the quantity is always 1.

The validation method can be meta, email or dns.

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
   "Id": 0,
   "UserId": 0,
   "UserHandle": "[use_handle]",
         "IpAddress": "::1",
   "Items": [
      {
         "ProduitID": 901,
         "Period": 1,
         "Quantity": 1,
         "Confirm": true,
         "Attributes": {
            "domain": "[domain]",
            "validationMethod": "Meta",
            "shaOption": "sha2",
            "csr": "[csr]"
         }
      }
   ]
}
```

Return:


```
{
   "Panier": null,
   "OriginalPanierId": 384,
   "Transaction": {
      "TransactionID": "NBAY-API-268e3ff02082022112344",
      "Total_HT": 35.0000,
      "Total_TTC": 42.0000,
      "PanierID": 0,
      "OpDetails": [
         {
            "Id": 3861296.0,
            "ProduitID": 901,
            "Period": 1.0,
            "Quantity": 1,
            "Pu": 35.0000,
            "IsUpdate": false,
            "NewQuantity": null,
            "PeriodLeft": null,
            "OldProduitID": null,
            "OldPu": null,
            "Total_HT": 35.0000,
            "OpDetailPacks": [
               {
                  "Id": 1107285.0,
                  "DetailId": 3861296.0,
                  "ProduitID": 901,
```

```
            "Pu": 35.0000,
            "CartItems": [
              {
                "Id": 1070052.0,
                "DetailPackId": 1107285.0,
                "Name": "[domain]",
                "Handle": null,
                "Class": "certificate",
                "Attributes": {
                  "billing_id": "1192599",
                  "contact1_id": "1192599",
                  "contact2_id": "1192599",
                  "csr": "[csr]",
                  "shaOption": "sha2",
                  "validationMethod": "Meta",
                  "admin_id": "1192599"
                }
              }
            ]
          }
        ]
      }
    ],
    "ModePaiementID": 3,
    "OpState": 1,
    "UserHandle": null,
    "UserId": 1192599.0,
    "Errors": [],
```

```
    "IpAddress": "::1",
    "Currency": "EUR"
  }
}
```


Get the certificate handle

Get the certificate handle using the OwnedProduct api.

GET [base_url]/api/OwnedProduct?user=[user_handle]&object_class=certificate&object_name=[domain]

Return:


```
[
  {
    "Handle": "[object_handle]",
    "Name": "[domain]",
    "Class": "certificate",
    "StatusID": 1,
    "Status": "REGISTERED",
    "CreDate": "2022-07-29T18:35:55",
    "ExpDate": "2023-08-30T18:35:54",
    "TransferInDate": null,
    "ProductId": 901,
    "Quantity": 1,
    "ContactTypes": [
      "A",
      "B"
    ],
    "Products": [
      {
        "Id": 901,
```

```
            "Name": "certificat Alpha SSL",
            "Type": "SSL",
            "SubType": "Creation",
            "TransactionId": "NBAY-API-10babbe29072022104733"
        }
    ],
    "Children": null,
    "Error": null
  }
]
```


Renew a certificate

Use the panier api to renew existing certifiactes.

When renewing a certificate, the period is the number of years and the quantity is always 0. The object_handle of the certificate to renew is supplied in the attributes and IsUpdate is true.

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "Id": 0,
  "UserId": 0,
  "UserHandle": "[user_handle]",
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 901,
      "Period": 1,
      "Quantity": 0,
      "Confirm": true,
```

```
        "IsUpdate": true,
        "Attributes": {
          "domain": "[domain]",
          "csr": "[csr]",
          "object_handle": "[object_handle]"
        }
      }
  ]
}
```


Return:


```
{
  "Panier": null,
  "OriginalPanierId": 387,
  "Transaction": {
    "TransactionID": "NBAY-API-bf754e202082022120353",
    "Total_HT": 35.0000,
    "Total_TTC": 42.0000,
    "PanierID": 0,
    "OpDetails": [
      {
        "Id": 3861297.0,
        "ProduitID": 901,
        "Period": 1.0,
        "Quantity": 0,
        "Pu": 35.0000,
        "IsUpdate": true,
        "NewQuantity": 1,
```

```
        "PeriodLeft": 1.013698630136,
        "OldProduitID": 901,
        "OldPu": 35.0000,
        "Total_HT": 35.0000,
        "OpDetailPacks": [
          {
            "Id": 1107286.0,
            "DetailId": 3861297.0,
            "ProduitID": 901,
            "Pu": 35.0000,
            "CartItems": [
              {
                "Id": 1070053.0,
                "DetailPackId": 1107286.0,
                "Name": "[domain]",
                "Handle": "[object_handle]",
                "Class": "certificate",
                "Attributes": {
                  "csr": "[csr]"
                }
              }
            ]
          }
        ]
      }
    ],
    "ModePaiementID": 3,
    "OpState": 1,
```

"UserHandle": **null**,

"UserId": 1192599.0,

"Errors": [],

"IpAddress": "::1",

"Currency": "EUR"

}

}

Validation

Once the certificate order is complete you can then proceed with the validation.

Validation method meta:

This is done by obtaining the metatag for the certificate which you then place in a well known location, make a request to validate the certificate and then obtain the issued certificate.

Validation method dns:

If the ns is hosted by namebay, the validation will be automated.

If the ns is hosted elsewhere you will need to obtain the dnstxt for the certificate which you then place in TXT record for the zone. The certificate will be validated automatically once the record has been placed.

Validation method email:

For email validation you will receive an email with instructions to validate the certificate.

Get the certificate details

Get the certificate details, including the meta tag, with the certificate api.

GET [base_url]/api/Certificate/[object_handle]?user=[user_handle]

Return:

{

"Handle": "[object_hanle]",

"ValidationMethod": "Meta",

"OrderId": "CEAP220706636856",

"TicketId": 3348153.0,

"DnsTxt": **null**,

"MetaTag": "[mettag]",

"VerificationUrlList": [

"http://[domain]"

],

"WellknownUrlList": [

"http://[domain]/.well-known/pki-validation/gsdv.txt"

],

"VerificationFQDNList": **null**,

"Status": "PENDING",

"TicketState": " WaitingUrlMetaCheck",

"Error": **null**

}

Validation method meta:

Save the metatag in a text file at the location given in the WellknownUrlList.

If the domain address begins with www. Then you can validate both www.domain and domain if you wish. You will need to save the file in both WellknownUrlLists to do this.

Verify the certificate

This is only necessary when using the validation method meta.

Once you have placed the metatag in the specified location you can then call the validate method of the certificate api with the url of the domain to validate in the VerificationUrlList.

If the domain address begins with www. Then you can validate both www.domain and domain by providing both urls in VerificationUrlList.

PUT [base_url]/api/Certificate/[object_handle]/Validate?user=[user_handle]

Payload:

{

"VerificationUrlList": [

"http://[domain]"

```
    ]
}
```


Return:


```
{
    "Handle": "[object_handle]",
    "ValidationMethod": "Meta",
    "OrderId": "CEAP220706636856",
    "TicketId": 3348153.0,
    "DnsTxt": null,
    "MetaTag": "[metatag]",
    "VerificationUrlList": [
        "http://[domain]"
    ],
    "WellknownUrlList": [
        "http://[domain]/.well-known/pki-validation/gsdv.txt"
    ],
    "VerificationFQDNList": null,
    "Status": "PENDING",
    "TicketState": " Waiting Certificate",
    "Error": null
}
```

Get the certificate

Once validated the certificate will be issued and sent via email to the admin contact and can be obtained via the certificate api.

GET [base_url]/api/Certificate/[object_handle]/Certificate?user=[user_handle]

Return:

{

"Handle": "[object_handle]",

"Certificate": "[certificate]",

"RegistrationDate": "2022-07-12T13:00:53.643",

"RevocationDate": **null**,

"Revoked": **false**

}

Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Certificate

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedPorduct

Commande et renouvellement de certificats wildcard

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```


The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Ordering certificates

Order a new certificate

Order a certificate using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the certificate for.

When ordering a new certificate, the period is the number of years and the quantity is always 1.

The wildcard option is a sperate product but must be ordered in the same panier as the certificate.

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "Id": 0,
  "UserId": 0,
  "UserHandle": "[user_handle]",
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 901,
      "Period": 1,
      "Quantity": 1,
      "Confirm": true,
      "Attributes": {
        "domain": "*.[domain]",
        "validationMethod": "email",
        "shaOption": "sha2",
        "csr": "[csr]"
      }
    },
    {
```

```
        "ProduitID": 910,
        "Period": 1,
        "Quantity": 1,
        "Confirm": true,
        "Attributes": {
          "domain": "*.[domain]"
        }
      }
  ]
}
```


Return:


```
{
  "Panier": null,
  "OriginalPanierId": 389,
  "Transaction": {
    "TransactionID": "NBAY-API-af83d2203082022095100",
    "Total_HT": 149.0000,
    "Total_TTC": 178.8000,
    "PanierID": 0,
    "OpDetails": [
      {
        "Id": 3861298.0,
        "ProduitID": 901,
        "Period": 1.0,
        "Quantity": 1,
        "Pu": 35.0000,
```

```
"IsUpdate": false,
"NewQuantity": null,
"PeriodLeft": null,
"OldProduitID": null,
"OldPu": null,
"Total_HT": 35.0000,
"OpDetailPacks": [
  {
    "Id": 1107287.0,
    "DetailId": 3861298.0,
    "ProduitID": 901,
    "Pu": 35.0000,
    "CartItems": [
      {
        "Id": 1070054.0,
        "DetailPackId": 1107287.0,
        "Name": "*.[mydomain]",
        "Handle": null,
        "Class": "certificate",
        "Attributes": {
          "admin_id": "1192599",
          "billing_id": "1192599",
          "shaOption": "sha2",
          "contact1_id": "1192599",
          "contact2_id": "1192599",
          "csr": "[csr]",
          "validationMethod": "email"
        }
```

```
            }
          ]
        }
      ]
    },
    {
      "Id": 3861299.0,
      "ProduitID": 910,
      "Period": 1.0,
      "Quantity": 1,
      "Pu": 114.0000,
      "IsUpdate": false,
      "NewQuantity": null,
      "PeriodLeft": null,
      "OldProduitID": null,
      "OldPu": null,
      "Total_HT": 114.0000,
      "OpDetailPacks": [
        {
          "Id": 1107288.0,
          "DetailId": 3861299.0,
          "ProduitID": 910,
          "Pu": 114.0000,
          "CartItems": [
            {
              "Id": 1070055.0,
              "DetailPackId": 1107288.0,
              "Name": "*.[mydomain]",
```

```
                    "Handle": null,
                    "Class": "certificate",
                    "Attributes": {}
                }
            ]
        }
    ]
  }
 ],
 "ModePaiementID": 3,
 "OpState": 1,
 "UserHandle": null,
 "UserId": 1192599.0,
 "Errors": [],
 "IpAddress": "::1",
 "Currency": "EUR"
 }
}
```

Get the certificate handle

Get the certificate handle using the OwnedProduct api.

GET
[base_url]/api/OwnedProduct?user=[user_handle]&object_class=certificate&object_name=[domain]

Return:


```
[
  {
    "Handle": "[object_handle]",
    "Name": "[domain]",
    "Class": "certificate",
    "StatusID": 1,
    "Status": "REGISTERED",
    "CreDate": "2022-07-29T18:35:55",
    "ExpDate": "2023-08-30T18:35:54",
    "TransferInDate": null,
    "ProductId": 901,
    "Quantity": 1,
    "ContactTypes": [
      "A",
      "B"
    ],
    "Products": [
      {
        "Id": 901,
        "Name": "certificat Alpha SSL",
        "Type": "SSL",
        "SubType": "Creation",
        "TransactionId": "NBAY-API-10babbe29072022104733"
```

```
        }
      ],
      "Children": null,
      "Error": null
    }
]
```


Renew a certificate

Use the panier api to renew existing certifiactes.

When renewing a certificate, the period is the number of years and the quantity is always 0. The object_handle of the certificate to renew is supplied in the attributes and IsUpdate is true.

The wildcard option is a sperate product but must be ordered in the same panier as the certificate.

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "Id": 0,
  "UserId": 0,
  "UserHandle": "[user_handle]",
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 901,
      "Period": 1,
      "Quantity": 0,
      "Confirm": true,
      "IsUpdate": true,
      "Attributes": {
```

```
            "domain": "*.[domain]",
            "csr": "[csr]",
            "object_handle": "[object_handle]"
        }
    },
    {
        "ProduitID": 910,
        "Period": 1,
        "Quantity": 1,
        "Confirm": true,
        "Attributes": {
            "domain": "*.[domain]",
        }
    }
  ]
}
```


Return:


```
{
  "Panier": null,
  "OriginalPanierId": 402,
  "Transaction": {
    "TransactionID": "NBAY-API-58b3ad705082022122739",
    "Total_HT": 149.0000,
    "Total_TTC": 178.8000,
    "PanierID": 0,
    "OpDetails": [
      {
```

"Id": 3861310.0,

"ProduitID": 901,

"Period": 1.0,

"Quantity": 0,

"Pu": 35.0000,

"IsUpdate": **true**,

"NewQuantity": 1,

"PeriodLeft": 0.021917808219,

"OldProduitID": 901,

"OldPu": 35.0000,

"Total_HT": 35.0000,

"OpDetailPacks": [

{

"Id": 1107299.0,

"DetailId": 3861310.0,

"ProduitID": 901,

"Pu": 35.0000,

"CartItems": [

{

"Id": 1070066.0,

"DetailPackId": 1107299.0,

"Name": "*.[domain]",

"Handle": "[object_handle]",

"Class": "certificate",

"Attributes": {

"csr": "[csr]"

}

}

```
            ]
          }
        ]
      },
      {
        "Id": 3861311.0,
        "ProduitID": 910,
        "Period": 1.0,
        "Quantity": 1,
        "Pu": 114.0000,
        "IsUpdate": false,
        "NewQuantity": null,
        "PeriodLeft": null,
        "OldProduitID": null,
        "OldPu": null,
        "Total_HT": 114.0000,
        "OpDetailPacks": [
          {
            "Id": 1107300.0,
            "DetailId": 3861311.0,
            "ProduitID": 910,
            "Pu": 114.0000,
            "CartItems": [
              {
                "Id": 1070067.0,
                "DetailPackId": 1107300.0,
                "Name": "*.[domain]",
                "Handle": null,
```

```
                    "Class": "certificate",
                    "Attributes": {}
                }
            ]
        }
    ]
  },
  "ModePaiementID": 3,
  "OpState": 1,
  "UserHandle": null,
  "UserId": 1192599.0,
  "Errors": [],
  "IpAddress": "::1",
  "Currency": "EUR"
 }
}
```


Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Certificate

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedPorduct

Hébergements

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```


The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Ordering hosting

Order a new hosting

Order a hosting using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the hosting for.

When ordering a new hosting, the period is the number of years and the quantity is always 1.

Product ids can be found using the product api.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:

```
{
   "IpAddress": "::1",
   "Items": [
      {
         "ProduitID": 24441,
         "Period": 1,
         "Quantity": 1,
         "Confirm": true,
         "Attributes": {
            "domain": "[domain]"
         }
      }
   ]
}
```

Return:

{

"Panier": **null**,

"OriginalPanierId": 0,

"Transaction": {

"TransactionID": "NBAY-RESTAPI-a86a9e802042025042254",

"Total_HT": 0.0000,

"Total_TTC": 0.0000,

"PanierID": 0,

"OpDetails": [

{

"Id": 36.0,

"ProduitID": 24441,

"Period": 1.0,

"Quantity": 1,

"Pu": 0.0000,

"IsUpdate": **false**,

"NewQuantity": **null**,

"PeriodLeft": **null**,

"OldProduitID": **null**,

"OldPu": **null**,

"Total_HT": 0.0000,

"OpDetailPacks": [

{

"Id": 30.0,

"DetailId": 36.0,

"ProduitID": 24441,

"Pu": 0.0000,

"CartItems": [

{

```
                    "Id": 30.0,
                    "DetailPackId": 30.0,
                    "Name": "[domain]",
                    "Handle": null,
                    "Class": "hosting",
                    "Attributes": {
                      "admin_id": "111222333",
                      "billing_id": "111222333"
                    }
                  }
                ]
              }
            ]
          }
        ],
        "AutoRenew": false,
        "ModePaiementID": 3,
        "OpState": 1,
        "Errors": [],
        "IpAddress": "::1",
        "Currency": "EUR"
    }
}
```

Get the hosting handle

Get the hosting handle using the OwnedProduct api.

GET [base_url]/api/OwnedProduct?user=[user_handle]&object_class=hosting&object_name=[domain]

Return:


```
[
  {
    "Handle": "[object_handle]",
    "ParentHandle": null,
    "Name": "[domain]",
    "Class": "hosting",
    "StatusID": 1,
    "Status": "REGISTERED",
    "CreDate": "2025-02-25T15:09:05.787",
    "ExpDate": "2026-02-25T15:09:05.763",
    "TransferInDate": null,
    "ProductId": 24441,
    "Quantity": 1,
    "ContactTypes": [
      "A",
      "B",
      "R"
    ],
    "Products": [
      {
        "Id": 24441,
        "Name": "formule hebergement - Linux20",
        "Type": "Hebergement",
```

```
            "SubType": "Creation",
            "TransactionId": "NBAY-RESTAPI-59c7e2125022025120512",
            "Active": false
        }
    ],
    "Children": null,
    "Error": null
}
]
```

Renew a hosting

Use the panier api to renew existing hostings.

When renewing a hosting, the period is the number of years. When renewing only the quantity should be set to 0. The object_handle of the hosting to renew is supplied in the attributes and IsUpdate is true.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 24441,
      "Period": 1,
      "Quantity": 0,
      "Confirm": true,
      "IsUpdate": true,
      "Attributes": {
        "domain": "[domain]",
        "object_handle": "[object_handle]"
      }
    }
  ]
}
```


Return:


```
{
  "Panier": null,
  "OriginalPanierId": 0,
```

```
"Transaction": {
   "TransactionID": "NBAY-RESTAPI-be00116020420250448 21",
   "Total_HT": 0.0000,
   "Total_TTC": 0.0000,
   "PanierID": 0,
   "OpDetails": [
      {
         "Id": 86.0,
         "ProduitID": 24441,
         "Period": 1.0,
         "Quantity": 0,
         "Pu": 0.0000,
         "IsUpdate": true,
         "NewQuantity": 1,
         "PeriodLeft": 0.901369863014,
         "OldProduitID": 24441,
         "OldPu": 0.0000,
         "Total_HT": 0.0000,
         "OpDetailPacks": [
            {
               "Id": 77.0,
               "DetailId": 86.0,
               "ProduitID": 24441,
               "Pu": 0.0000,
               "CartItems": [
                  {
                     "Id": 77.0,
                     "DetailPackId": 77.0,
```

```
                    "Name": "[domain]",
                    "Handle": "[object_handle]",
                    "Class": "hosting",
                    "Attributes": {}
                }
            ]
        }
    ]
  }
 ],
 "AutoRenew": false,
 "ModePaiementID": 3,
 "OpState": 1,
 "Errors": [],
 "IpAddress": "::1",
 "Currency": "EUR"
 }
}
```

Managing hosting

Create an ftp user

Create an ftp user using the hosting user api. Hosting_handle is the handle of your hosting. User_handle is either your own user handle or the handle of the user you are creating the ftp user for.

You will need to provide a dir and a password. The dir must be either the hosting dir or a sub directory of the hosting dir. A user_name will be generated and returned in the response. A user_login is also returned in the response. For linux systems the user_name and user_login are the same but on windows the user_login is a combination of host and user_name hence the 2 fields.

Example:

POST [base_url]/api/Hosting/[hosting_handle]/User?user=[user_handle]

Payload:

```
{
   "Pwd": "[password]",
   "FullName": "[name]",
   "Description": "[description]",
   "Dir": "[dir]"
}
```

Return:

```
{
   "Id": [user_id],
   "HostingHandle": "[hosting_handle]",
   "Host": "[domain]",
   "UserName": "[user_name]",
   "Login": "[user_login]",
   "Pwd": null,
   "FullName": "[name]",
```

"Description": "[description]",

"Dir": "[dir]",

"Uid": 28089,

"Gid": 28089,

"DateCre": "2025-02-28T11:39:09.413",

"Etat": 0,

"IsParent": **false**,

"IsFpUser": **false**,

"IsUnixUser": **false**,

"IsFtpUser": **true**

}

Update an ftp user

You can modify the password of an ftp user with the hosting user api. Hosting_handle is the handle of your hosting. User_id is the id of the ftp user you want to update. User_handle is either your own user handle or the handle of the user you are updating the ftp user for.

When the hosting is deployed there will be an admin user created whose UserName is the same as the hosting handle. You will need to specify the password for this user.

Example:

PUT [base_url]/api/Hosting/[hosting_handle]/User/[user_id]?user=[user_handle]

Payload:

{

"Pwd": "[new_password]"

}

Return:

{

"Id": [user_id],

"HostingHandle": "[hosting_handle]",

"Host": "[domain]",

"UserName": "[user_name]",

"Login": "[user_login]",

"Pwd": **null**,

"FullName": "[name]",

"Description": "[description]",

"Dir": "[dir]",

"Uid": 28089,

"Gid": 28089,

"DateCre": "2025-02-28T11:39:09.413",

"Etat": 0,

"IsParent": **false**,

"IsFpUser": **false**,

"IsUnixUser": **false**,

"IsFtpUser": **true**

}

Delete an ftp user

Delete an ftp user using the hosting user api. Hosting_handle is the handle of your hosting. User_id is the id of the ftp user you want to delete. User_handle is either your own user handle or the handle of the user you are deleting the ftp user for.

Example:

DELETE [base_url]/api/Hosting/[hosting_handle]/User/[user_id]?user=[user_handle]

Create a database

Create a database using the hosting db api. Hosting_handle is the handle of your hosting. User_handle is either your own user handle or the handle of the user you are creating the database for. The db_name, db_user and password are returned in the response.

Example:

POST [base_url]/api/Hosting/[hosting_handle]/Db?user=[user_handle]

Payload:

{
"Type": "mysql5"
}

Return:

{
"Name": "[db_name]",
"HostingHandle": "[hosting_handle]",
"User": "[db_user]",
"Pwd": "[password]",
"Type": "mysql5",
"Size": 0,
"MaxSize": 100,
"ServerId": 128,
"DateCre": "2025-02-26T15:58:17.777",
"DateReq": "2025-02-26T15:58:17.777",
"Etat": "ok",
"UserLang": "E"
}

Update a database

You can modify the password of an db user with the hosting db api. Hosting_handle is the handle of your hosting. Db_name is the name of the database you want to update. User_handle is either your own user handle or the handle of the user you are updating the db for.

Example:

PUT [base_url]/api/Hosting/[hosting_handle]/Db/[db_name]?user=[user_handle]

Payload:

{

"Pwd": "[password]"

}

Return:

{

"Name": "[db_name]",

"HostingHandle": "[hosting_handle]",

"User": "[db_user]",

"Pwd": null,

"Type": "mysql5",

"Size": 0,

"MaxSize": 100,

"ServerId": 128,

"DateCre": "2025-02-26T15:58:17.777",

"DateReq": "2025-02-26T15:58:17.777",

"Etat": "ok",

"UserLang": "E"

}

Delete a database

Delete a database using the hosting db api. Hosting_handle is the handle of your hosting. Db_name is the name of the database you want to delete. User_handle is either your own user handle or the handle of the user you are deleting the db for.

Example:

DELETE [base_url]/api/Hosting/[hosting_handle]/Db/[user_id]?user=[user_handle]

Create additional sites

Create multiple sites on a hosting using the hosting site api. Hosting_handle is the handle of your hosting. User_handle is either your own user handle or the handle of the user you are creating the site for.

You need to provide a domain that exists and points to the ip address of your hosting.

Example:

POST [base_url]/api/Hosting/[hosting_handle]/Site?user=[user_handle]

Payload:


```
{
   "Domain": "[domain]"


}
```


Return:


```
{
   "Id": 1,
   "HostingHandle": "[object_handle]",
   "PhpVersion": null,
   "Domain": "[domain]",
   "Status": 1,
   "DateModified": "2025-04-08T10:42:48.35",
   "DateCreated": "2025-04-08T10:42:48.35"
}
```


Create a wordpress site

Create a wordpress on a hosting using the hosting wordpress api. Hosting_handle is the handle of your hosting. User_handle is either your own user handle or the handle of the user you are creating the wordpress for.

You can create a wordpress site on the root of your hosting, in a sub directory with a virtual directory, on an alias or on one of the additional sites if you have one.

The simplest way to create a wordpress is on the root. You need only provide the Title. A database will be created and the credentials for both the database and the wordpress admin user will be returned in the response.

N.B. the passwords are only returned on creation, they are not returned by GET requests.

Example:

POST [base_url]/api/Hosting/[hosting_handle]/Wordpress?user=[user_handle]

```
{
   "Title":"Test wp"
}
```

Return:


```
{
   "Id": 2,
   "Site": null,
   "Alias": null,
   "HostingHandle": "[hosting_handle]",
   "Db": {
      "Name": "[db_name]",
      "HostingHandle": "[hosting_handle]",
      "User": "[db_user]",
      "Pwd": "[db_password]",
      "Type": "mysql5",
      "Size": 0,
      "MaxSize": 100,
      "ServerId": 128,
      "DateCre": "2025-04-09T14:06:13.27",
      "DateReq": "2025-04-09T14:06:13.27",
      "Etat": "ok",
      "UserLang": "E"
   },
   "Dir": null,
   "Title": "Test wp",
```

"AdminUser": "wp_admin",

"AdminEmail": "[user_email]",

"AdminPassword": "[admin_password]"

}

Update wordpress admin user

Update a wordpress admin user on a hosting using the hosting wordpress api. Hosting_handle is the handle of your hosting. User_handle is either your own user handle or the handle of the user you are updating the wordpress for. Wordpress_id is the id of the wordpress you are updating.

You can update the email address, the password or both. You must supply the AdminUser in the payload.

Example:

POST [base_url]/api/Hosting/[hosting_handle]/Wordpress/[wordpress_id]?user=[user_handle]

{

"AdminUser": "wp_admin",

"AdminEmail": "[user_email]",

"AdminPassword": "[admin_password]"

}

Return:

{

"Id": 2,

"Site": **null**,

"Alias": **null**,

"HostingHandle": "[hosting_handle]",

"Db": {

"Name": "[db_name]",

```
        "HostingHandle": "[hosting_handle]",
        "User": "[db_user]",
        "Pwd": null,
        "Type": "mysql5",
        "Size": 0,
        "MaxSize": 100,
        "ServerId": 128,
        "DateCre": "2025-04-09T14:06:13.27",
        "DateReq": "2025-04-09T14:06:13.27",
        "Etat": "ok",
        "UserLang": "E"
    },
    "Dir": null,
    "Title": "Test wp",
    "AdminUser": "wp_admin",
    "AdminEmail": "[user_email]",
    "AdminPassword": null
}
```


Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedProduct

https://rest.namebay.com/help#Product

https://rest.namebay.com/help#Hosting

E-mails

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```


The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Ordering email service

Order a new email service

Order an email service using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the service for.

When ordering a new email service, the period is the number of years and the quantity is the number of instances i.e. the number of email addresses required. The number of instances can be increased at a later date with an update order on the existing service, similar to a renewal (see below).

Product ids can be found using the product api.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:

```
{
   "IpAddress": "::1",
   "Items": [
      {
         "ProduitID": 1040,
         "Period": 2,
         "Quantity": 20,
         "Confirm": true,
         "Attributes": {
            "domain": "[domain]"
         }
      }
   ]
}
```

Return:

{

```
"Panier": null,
"OriginalPanierId": 0,
"Transaction": {
  "TransactionID": "NBAY-RESTAPI-39ac9e601042025101808",
  "Total_HT": 0.0000,
  "Total_TTC": 0.0000,
  "PanierID": 0,
  "OpDetails": [
    {
      "Id": 33.0,
      "ProduitID": 1040,
      "Period": 2.0,
      "Quantity": 20,
      "Pu": 0.0000,
      "IsUpdate": false,
      "NewQuantity": null,
      "PeriodLeft": null,
      "OldProduitID": null,
      "OldPu": null,
      "Total_HT": 0.0000,
      "OpDetailPacks": [
        {
          "Id": 27.0,
          "DetailId": 33.0,
          "ProduitID": 1040,
          "Pu": 0.0000,
          "CartItems": [
            {
```

```
                    "Id": 27.0,
                    "DetailPackId": 27.0,
                    "Name": "[domain]",
                    "Handle": null,
                    "Class": "mail",
                    "Attributes": {
                      "admin_id": "111222333",
                      "billing_id": "111222333"
                    }
                  }
                ]
              }
            ]
          }
        ],
        "AutoRenew": false,
        "ModePaiementID": 3,
        "OpState": 1,
        "Errors": [],
        "IpAddress": "::1",
        "Currency": "EUR"
    }
}
```

Get the service handle

Get the service handle using the OwnedProduct api.

GET [base_url]/api/OwnedProduct?user=[user_handle]&object_class=mail&object_name=[domain]

Return:


```
[
  {
    "Handle": "[object_handle]",
    "ParentHandle": null,
    "Name": "[domain]",
    "Class": "mail",
    "StatusID": 1,
    "Status": "REGISTERED",
    "CreDate": "2025-03-05T14:23:11.877",
    "ExpDate": "2027-03-05T00:00:00",
    "TransferInDate": null,
    "ProductId": 1040,
    "Quantity": 20,
    "ContactTypes": [
      "R"
    ],
    "Products": [
      {
        "Id": 1040,
        "Name": "Compte Mail",
        "Type": "Mail",
        "SubType": "Creation",
        "TransactionId": "NONE",
```

```
            "Active": false
          }
        ],
        "Children": null,
        "Error": null
      }
]
```


Renew an email service

Use the panier api to renew existing email services.

When renewing an email service, the period is the number of years. When renewing only, the quantity should be set to 0. The object_handle of the product to renew is supplied in the attributes and IsUpdate is true.

You can also use the quantity to increase the number of instances. The quantity in the command is the number of additional instances to add to the service. This can be done without extending the period by setting the period to 0.

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 1040,
      "Period": 1,
      "Quantity": 0,
      "IsUpdate": true,
      "Confirm": true,
      "Attributes": {
        "domain": "[domain]",
```

```
            "object_handle": "[object_handle]"
          }
        }
    ]
}
```


Return:


```
{
    "Panier": null,
    "OriginalPanierId": 0,
    "Transaction": {
        "TransactionID": "NBAY-RESTAPI-18d2abb01042025111324",
        "Total_HT": 0.0000,
        "Total_TTC": 0.0000,
        "PanierID": 0,
        "OpDetails": [
            {
                "Id": 34.0,
                "ProduitID": 1040,
                "Period": 1.0,
                "Quantity": 0,
                "Pu": 0.0000,
                "IsUpdate": true,
                "NewQuantity": 20,
                "PeriodLeft": 2,
                "OldProduitID": 1040,
                "OldPu": null,
                "Total_HT": 0.0000,
```

```
        "OpDetailPacks": [
          {
            "Id": 28.0,
            "DetailId": 34.0,
            "ProduitID": 1040,
            "Pu": 0.0000,
            "CartItems": [
              {
                "Id": 28.0,
                "DetailPackId": 28.0,
                "Name": "[domain]",
                "Handle": "[object_handle]",
                "Class": "mail",
                "Attributes": {}
              }
            ]
          }
        ]
      }
    ],
    "AutoRenew": false,
    "ModePaiementID": 3,
    "OpState": 1,
    "Errors": [],
    "IpAddress": "::1",
    "Currency": "EUR"
  }
}
```

Managing mail boxes

Create a mail box

Create a mailbox using the mail api. User_handle is either your own user handle or the handle of the user you are creating the mailbox for.

You will need the service object_handle, the email address that you want to create the mailbox for, a user name and a password.

Example:

POST [base_url]/api/Mail?user=[user_handle]

Payload:


```
{
   "ServiceHandle": "[object_handle]",
   "UserType": "pop",
   "UserMailAddress": "[email_address]",
   "UserName": "[user_name]",
   "UserPwd": "[password]"
}
```


Return:


```
{
   "ServiceHandle": "[object_handle]",
   "ObjectHandle": "[mailbox_handle]",
   "DomHandle": "[domain_handle]",
   "UserType": "pop",
   "UserName": "[user_name]",
   "UserDescription": null,
   "UserMailAddress": "[email_address]",
   "IsCatchAll": false,
   "IsRedir": false,
```

"Forward": **null**,

"Alias": **null**,

"Etat": 1,

"State": "Completed",

"CreDate": "0001-01-01T00:00:00",

"Errors": []

}

Update a mail box

Update a mailbox using the mail api. Mailbox_handle is the handle of the mailbox you want to update, user_handle is either your own user handle or the handle of the user you are creating the mailbox for.

Example:

PUT [base_url]/api/Mail/[mailbox_handle]?user=[user_handle]

Payload:

{

"UserPwd": "[new_password]"

}

Return:

{

"ServiceHandle": "[object_handle]",

"ObjectHandle": "[mailbox_handle]",

"DomHandle": "[domain_handle]",

"UserType": "pop",

"UserName": "[user_name]",

"UserDescription": **null**,

"UserMailAddress": "[email_address]",

"IsCatchAll": **false**,

"IsRedir": **false**,

"Forward": **null**,

"Alias": **null**,

"Etat": 2,

"State": "Completed",

"CreDate": "0001-01-01T00:00:00",

"Errors": []

}

Delete a mail box

Delete a mailbox using the mail api. Mailbox_handle is the handle of the mailbox you want to delete, user_handle is either your own user handle or the handle of the user you are creating the mailbox for.

Example:

DELETE [base_url]/api/Mail/[mailbox_handle]?user=[user_handle]

Return:

{

"ServiceHandle": "[object_handle]",

"ObjectHandle": "[mailbox_handle]",

"DomHandle": "[domain_handle]",

"UserType": "pop",

"UserName": "[user_name]",

"UserDescription": **null**,

"UserMailAddress": "[email_address]",

"IsCatchAll": **false**,

"IsRedir": **false**,

"Forward": **null**,

"Alias": **null**,

"Etat": 3,

"State": "Completed",

"CreDate": "0001-01-01T00:00:00",

"Errors": []

}

Set the autoresponder

Set the autoresponder using the mail autoresponder api. The PUT method will create the autoresponder if it does not exist and update the autoresponder if it does. Mailbox_handle is the handle of the mailbox you want to delete, user_handle is either your own user handle or the handle of the user you are creating the mailbox for.

Example:

PUT [base_url]/api/Mail/[mailbox_handle]/Autoresponder?user=[user_handle]

Payload:

{

"start": "2025-03-30",

"finish": "2025-04-04",

"active": **true**,

"subject": "test",

"message": "test mesg"

}

Return:

{

"ObjectHandle": "[mailbox_handle]",

"Active": **true**,

"MailAddress": "[email_address]",

"Forward": **null**,

"Message": "test mesg",

"Subject": "test",

"Start": "2025-03-30T00:00:00+01:00",

"Finish": "2025-04-04T00:00:00+02:00",

"Errors": []

}

Remove the autoresponder

Set the autoresponder to inactive using the PUT method of the mail autoresponder api. Mailbox_handle is the handle of the mailbox you want to delete, user_handle is either your own user handle or the handle of the user you are creating the mailbox for. You can also delete the autoresponder using the DELETE method.

Example:

PUT [base_url]/api/Mail/[mailbox_handle]/Autoresponder?user=[user_handle]

Payload:

{

"active": **false**

}

Return:

{

"ObjectHandle": "[mailbox_handle]",

"Active": **false**,

"MailAddress": "[email_address]",

"Forward": **null**,

"Message": "test mesg",

"Subject": "test",

"Start": "2025-03-30T00:00:00+01:00",

"Finish": "2025-04-04T00:00:00+02:00",

"Errors": []

}

Or:

DELETE [base_url]/api/Mail/[mailbox_handle]/Autoresponder?user=[user_handle]

Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedProduct

https://rest.namebay.com/help#Product

https://rest.namebay.com/help#Mail

Redirection URL

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:

{

"access_token": "[token_string]",

"token_type": "bearer",

"expires_in": 1209599,

"userName": "[logname]",

"audience": "namebay_1",

".issued": "Tue, 02 Aug 2022 08:20:38 GMT",

".expires": "Tue, 16 Aug 2022 08:20:38 GMT"

}

The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Ordering redirection service

Order a new redirection service

Order a redirection service using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the service for.

When ordering a new redirection service, the period is the number of years and the quantity is the number of instances i.e. the number of redirections required. The number of instances can be increased at a later date with an update order on the existing service, similar to a renewal (see below).

Product ids can be found using the product api.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:

{

"IpAddress": "::1",

"Items": [

{

"ProduitID": 1036,

"Period": 1,

"Quantity": 1,

"Confirm": **true**,

"Attributes": {

"domain": "[domain]"

}

```
      }
    ]
}
```


Return:


```
{
   "Panier": null,
   "OriginalPanierId": 0,
   "Transaction": {
      "TransactionID": "NBAY-RESTAPI-39ac9e601042025101808",
      "Total_HT": 0.0000,
      "Total_TTC": 0.0000,
      "PanierID": 0,
      "OpDetails": [
         {
            "Id": 33.0,
            "ProduitID": 1040,
            "Period": 1.0,
            "Quantity": 1,
            "Pu": 0.0000,
            "IsUpdate": false,
            "NewQuantity": null,
            "PeriodLeft": null,
            "OldProduitID": null,
            "OldPu": null,
            "Total_HT": 0.0000,
            "OpDetailPacks": [
               {
```

```
            "Id": 27.0,
            "DetailId": 33.0,
            "ProduitID": 1036,
            "Pu": 0.0000,
            "CartItems": [
              {
                "Id": 27.0,
                "DetailPackId": 27.0,
                "Name": "[domain]",
                "Handle": null,
                "Class": "service",
                "Attributes": {
                  "admin_id": "111222333",
                  "billing_id": "111222333"
                }
              }
            ]
          }
        ]
      }
    ],
    "AutoRenew": false,
    "ModePaiementID": 3,
    "OpState": 1,
    "Errors": [],
    "IpAddress": "::1",
    "Currency": "EUR"
}
```

}

Get the service handle

Get the service handle using the OwnedProduct api.

GET
[base_url]/api/OwnedProduct?user=[user_handle]&object_class=service&object_name=[domain]

Return:


```
[
  {
    "Handle": "[object_handle]",
    "ParentHandle": null,
    "Name": "[domain]",
    "Class": "service",
    "StatusID": 0,
    "Status": "NONE",
    "CreDate": "2025-04-09T16:55:55.843",
    "ExpDate": "2026-02-20T12:36:42.43",
    "TransferInDate": null,
    "ProductId": 1036,
    "Quantity": 1,
    "ContactTypes": [
      "A",
      "B",
      "R"
    ],
    "Products": [
      {
        "Id": 1036,
        "Name": "Redirection Url",
```

```
            "Type": "UrlRedirect",
            "SubType": "Creation",
            "TransactionId": "NBAY-RESTAPI-3b9de9e09042025042725",
            "Active": false
        }
    ],
    "Children": null,
    "Error": null
  }
]
```


Renew a redirection service

Use the panier api to renew existing redirection services.

When renewing a redirection service, the period is the number of years. When renewing only, the quantity should be set to 0. The object_handle of the product to renew is supplied in the attributes and IsUpdate is true.

You can also use the quantity to increase the number of instances. The quantity in the command is the number of additional instances to add to the service. This can be done without extending the period by setting the period to 0.

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 1036,
      "Period": 1,
      "Quantity": 0,
      "IsUpdate": true,
      "Confirm": true,
```

```
        "Attributes": {
          "domain": "[domain]",
          "object_handle": "[object_handle]"
        }
      }
    ]
}
```


Return:


```
{
  "Panier": null,
  "OriginalPanierId": 0,
  "Transaction": {
    "TransactionID": "NBAY-RESTAPI-18d2abb01042025111324",
    "Total_HT": 0.0000,
    "Total_TTC": 0.0000,
    "PanierID": 0,
    "OpDetails": [
      {
        "Id": 34.0,
        "ProduitID": 1036,
        "Period": 1.0,
        "Quantity": 0,
        "Pu": 0.0000,
        "IsUpdate": true,
        "NewQuantity": 1,
        "PeriodLeft": 1,
        "OldProduitID": 1036,
```

```
        "OldPu": null,
        "Total_HT": 0.0000,
        "OpDetailPacks": [
          {
            "Id": 28.0,
            "DetailId": 34.0,
            "ProduitID": 1036,
            "Pu": 0.0000,
            "CartItems": [
              {
                "Id": 28.0,
                "DetailPackId": 28.0,
                "Name": "[domain]",
                "Handle": "[object_handle]",
                "Class": "service",
                "Attributes": {}
              }
            ]
          }
        ]
      }
    ],
    "AutoRenew": false,
    "ModePaiementID": 3,
    "OpState": 1,
    "Errors": [],
    "IpAddress": "::1",
    "Currency": "EUR"
```

```
    }
}
```

Managing redirections

Create a redirection

Create a redirection using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are creating the redirection for.

You will need a UrlOrigin, this is either the domain or a sub-domain, a UrlDestination and a Type.

Example:

POST [base_url]/api/Zone/[zone_handle]/Redirection?user=[user_handle]

Payload:

```
{
    "UrlOrigin": "[url_origin]",
    "UrlDestination": "[url_destination]",
    "Type": 4,
    "Active": true,
    "Title": "[title]",
    "MetaKeyword": "[meta_keywords]",
    "MetaDescription": "[meta_description]"
}
```

Return:

```
{
    "Handle": "[redirection_handle]",
    "CreatedOn": "2025-04-11T11:03:03.86",
    "HitsYesterday": 0,
    "UrlOrigin": "[url_origin]",
```

"UrlDestination": "[url_destination]",

"Type": 4,

"Active": **true**,

"Title": "[title]",

"MetaKeyword": "[meta_keywords]",

"MetaDescription": "[meta_description]",

"Relative": **false**

}

Update a redirection

Update a redirection using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are creating the redirection for. Redirection_handle is the handle of the redirection to update.

Example:

PUT
[base_url]/api/Zone/[zone_handle]/Redirection/[redirection_handle]?user=[user_handle]

Payload:

{

"UrlOrigin": "[url_origin]",

"UrlDestination": "[url_destination]",

"Type": 4,

"Active": **true**,

"Title": "[title]",

"MetaKeyword": "[meta_keywords]",

"MetaDescription": "[meta_description]"

}

Return:

```
{
   "Handle": "[redirection_handle]",
   "CreatedOn": "2025-04-11T11:03:03.86",
   "HitsYesterday": 0,
   "UrlOrigin": "[url_origin]",
   "UrlDestination": "[url_destination]",
   "Type": 4,
   "Active": true,
   "Title": "[title]",
   "MetaKeyword": "[meta_keywords]",
   "MetaDescription": "[meta_description]",
   "Relative": false
}
```


Delete a redirection

Delete a redirection using the zone api. Zone_handle is the handle of the zone. This will be the same as the domain handle for domains registered with namebay. User_handle is either your own user handle or the handle of the user you are creating the redirection for. Redirection_handle is the handle of the redirection to delete.

Example:

DELETE
[base_url]/api/Zone/[zone_handle]/Redirection/[redirection_handle]?user=[user_handle]

Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedProduct

https://rest.namebay.com/help#Product

https://rest.namebay.com/help#Zone

Namebay message polling

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:


```
{
   "access_token": "[token_string]",
   "token_type": "bearer",
   "expires_in": 1209599,
   "userName": "[logname]",
   "audience": "namebay_1",
   ".issued": "Tue, 02 Aug 2022 08:20:38 GMT",
   ".expires": "Tue, 16 Aug 2022 08:20:38 GMT"
}
```

The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Polling

This api allows the user to get notified for various events such as the completion of an order.

For example, when a hosting creation ticket completes you will have a hostingCreated message.

The id is the id that is used to acknowledge the message, the count is the number of messages in the queue and the Data contains information such as the object_handle and statusId.

GET [base_url]/api/Poll?user=[user_handle]

Return:


```
{
   "Id": 3,
   "Count": 4,
   "Created": "2025-03-11T14:09:01.377",
   "Message": "hostingCreated",
   "Data": {
      "object_handle": "[object_handle]",
      "object_class": "hosting",
      "tickets_id": 13,
      "statusId": 1,
      "logDate": "2025-03-11T14:09:01.377",
      "old_object_handle": ""
   }
}
```


Acknowledgement

To acknowledge a message and remove it from the queue you use the acknowledge method. This returns the id of the message and the count for the number of messages left in the queue.

Example:

PUT [base_url]/api/Poll/33/Acknowledge?user=[user_handle]

Return:

{

"Id": 33,

"Count": 4

}

Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Poll

Namebay Rest overview

Authentication

Obtain a token from the authentication endpoint

POST [auth_url]/token

Payload:

username=[logname]&password=[password]&grant_type=password

Return:

{

"access_token": "[token_string]",

"token_type": "bearer",

"expires_in": 1209599,

"userName": "[logname]",

"audience": "namebay_1",

".issued": "Tue, 02 Aug 2022 08:20:38 GMT",

".expires": "Tue, 16 Aug 2022 08:20:38 GMT"

}

The access_token is then supplied in the Authorization header of subsequent api calls as a bearer token.

Product

This api returns a list of products with their properties.

Amongst the properties are the id which is needed to place an order and a list of attributes that can be supplied when placing the order.

Example:

GET
[base_url]/api/Product?user=[user_handle]&type=domaine&subtype=creation&tldname=com

Return:

[

{

"Pu": 0.0000,

"Promotions": [],

"Id": 771,

"Name": "Enregistrement de nom de domaine .COM",

"ProductType": "Création de Nom de domaine",

"TypeId": 1,

"Type": "Domaine",

"SubType": "Creation",

"Package": false,

"Attributes": [

{

```
    "Name": "domain",
    "Mandatory": true,
    "Validate": "DomainAvailable,OnePerOrder,ValidDomainTld",
    "DefaultAttribute": null,
    "DefaultType": null,
    "Class": "domain",
    "Header": 1,
    "MapTo": null,
    "Conversion": null,
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
    "Name": "admin_handle",
    "Mandatory": true,
    "Validate": null,
    "DefaultAttribute": "handle",
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": "admin_id",
    "Conversion": "UserHandleToId",
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
```

```
    "Name": "tech_handle",
    "Mandatory": true,
    "Validate": null,
    "DefaultAttribute": "handle",
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": "tech_id",
    "Conversion": "UserHandleToId",
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
    "Name": "registrant_handle",
    "Mandatory": true,
    "Validate": null,
    "DefaultAttribute": "handle",
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": "registrant_id",
    "Conversion": "UserHandleToId",
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
```

```
    "Name": "billing_handle",
    "Mandatory": true,
    "Validate": null,
    "DefaultAttribute": "facturationHandle",
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": "billing_id",
    "Conversion": "UserHandleToId",
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
    "Name": "ns1",
    "Mandatory": true,
    "Validate": null,
    "DefaultAttribute": null,
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": null,
    "Conversion": null,
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
```

```
    "Name": "ns2",
    "Mandatory": true,
    "Validate": null,
    "DefaultAttribute": null,
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": null,
    "Conversion": null,
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
    "Name": "ns(n)",
    "Mandatory": false,
    "Validate": null,
    "DefaultAttribute": null,
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": null,
    "Conversion": null,
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
```

```
    "Name": "ip(n)",
    "Mandatory": false,
    "Validate": null,
    "DefaultAttribute": null,
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": null,
    "Conversion": null,
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
    "Name": "tld_id",
    "Mandatory": false,
    "Validate": null,
    "DefaultAttribute": null,
    "DefaultType": null,
    "Class": "domain",
    "Header": 0,
    "MapTo": null,
    "Conversion": null,
    "Mode": 0,
    "MinLength": null,
    "MaxLength": null
  },
  {
```

```
            "Name": "catch",
            "Mandatory": false,
            "Validate": null,
            "DefaultAttribute": null,
            "DefaultType": null,
            "Class": "domain",
            "Header": 0,
            "MapTo": null,
            "Conversion": null,
            "Mode": 0,
            "MinLength": null,
            "MaxLength": null
        }
    ],
    "TrusteeProduct": false,
    "Tlds": [
        {
            "Tld_id": 1,
            "Name": "com",
            "Registry_id": "verisign",
            "Country": "Générique"
        }
    ],
    "ParentProducts": null,
    "ChildProducts": null,
    "GlobalSignProduit": null,
    "DiscountDependances": null,
    "DiscountParents": null,
```

```
    "Discounts": null,
    "Active": true
  }
]
```


Panier

Placing an order

Placing an order for a product is done using the panier api. User_handle is either your own user handle or the handle of the user you are ordering the domain for.

For most of the products such as domain, hosting, certifiacte, the period is the number of years and the quantity is always 1. For services like email the quantity is the number of instances required.

The produitID is the Product id which can be found using the product api.

The attributes depend on the product. Details can be found using the product api.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:


```
{
  "IpAddress": "::1",
  "Items": [
    {
      "ProduitID": 9105,
      "Period": 1,
      "Quantity": 1,
      "Confirm": true,
      "Attributes": {
        "domain": "[domain]",
        "ns1": "dns1.namebay.com",
        "ns2": "dns2.namebay.com",
```

```
            "registrant_handle": "[user_handle]"
          }
        }
    ]
}
```


Return:


```
{
    "Panier": null,
    "OriginalPanierId": 0,
    "Transaction": {
        "TransactionID": "NBAY-RESTAPI-6abf88005032025042139",
        "Total_HT": 0.0000,
        "Total_TTC": 0.0000,
        "PanierID": 0,
        "OpDetails": [
            {
                "Id": 37.0,
                "ProduitID": 9105,
                "Period": 1.0,
                "Quantity": 1,
                "Pu": 0.0000,
                "IsUpdate": false,
                "NewQuantity": null,
                "PeriodLeft": null,
                "OldProduitID": null,
                "OldPu": null,
```

```
"Total_HT": 0.0000,
"OpDetailPacks": [
  {
    "Id": 30.0,
    "DetailId": 37.0,
    "ProduitID": 9105,
    "Pu": 0.0000,
    "CartItems": [
      {
        "Id": 30.0,
        "DetailPackId": 30.0,
        "Name": "[domain]",
        "Handle": null,
        "Class": "domain",
        "Attributes": {
          "admin_id": "1111",
          "billing_id": "1111",
          "ns1": "dns1.namebay.com",
          "ns2": "dns2.namebay.com",
          "registrant_id": "1111",
          "tech_id": "1111",
          "tld_id": "31"
        }
      }
    ]
  }
]
}
```

```
    ],
    "AutoRenew": false,
    "ModePaiementID": 3,
    "OpState": 1,
    "Errors": [],
    "IpAddress": "::1",
    "Currency": "EUR"
  }
}
```


Get the object handle

You can obtain the object handle using the OwnedProduct api.

Example:

GET [base_url]/api/OwnedProduct?user=[user_handle]&object_class=domain&object_name=[domain]

Return:


```
[
  {
    "Handle": "[object_handle]",
    "ParentHandle": null,
    "Name": "[domain]",
    "Class": "domain",
    "StatusID": 1,
    "Status": "REGISTERED",
    "CreDate": "2025-02-21T00:00:00",
    "ExpDate": "2026-02-21T00:00:00",
    "TransferInDate": null,
    "ProductId": 9105,
    "Quantity": 1,
```

```
    "ContactTypes": [
      "O",
      "A",
      "T",
      "B",
      "R"
    ],
    "Products": [
      {
        "Id": 9105,
        "Name": " Enregistrement de nom de domaine  + Présence locale eu ",
        "Type": "Domaine",
        "SubType": "Creation",
        "TransactionId": "NBAY-RESTAPI-6abf88005032025042139",
        "Active": false
      }
    ],
    "Children": null,
    "Error": null
  }
]
```

Renew a product

Use the panier api to renew existing products.

When renewing a domain you will need to use the product id of the renewal which is different to the product for registration. As with the original order the period is the number of years and the quantity is always 1. The object_handle of the domain to renew is supplied in the attributes.

For other products, the period is the number of years. When renewing a product the quantity should be set to 0. The object_handle of the product to renew is supplied in the attributes and IsUpdate is true.

For service products such as email you can also use the quantity to increase the number of instances. The quantity in the command is the number of additional instances to add to the service. This can be done without extending the period by setting the period to 0.

Example:

POST [base_url]/api/Panier?user=[user_handle]

Payload:

```
{
   "IpAddress": "::1",
   "Items": [
      {
         "ProduitID": 9205,
         "Period": 1,
         "Quantity": 1,
         "Confirm": true,
         "Attributes": {
            "domain": "[domain]",
            "object_handle": "[object_handle]"
         }
      }
   ]
}
```

Return:


```
{
   "Panier": null,
   "OriginalPanierId": 0,
   "Transaction": {
      "TransactionID": "NBAY-RESTAPI-dd3717607032025102908",
      "Total_HT": 17.0700,
      "Total_TTC": 20.4840,
      "PanierID": 0,
      "OpDetails": [
         {
            "Id": 30.0,
            "ProduitID": 9205,
            "Period": 1.0,
            "Quantity": 1,
            "Pu": 17.0700,
            "IsUpdate": false,
            "NewQuantity": null,
            "PeriodLeft": null,
            "OldProduitID": null,
            "OldPu": null,
            "Total_HT": 17.0700,
            "OpDetailPacks": [
               {
                  "Id": 24.0,
                  "DetailId": 30.0,
                  "ProduitID": 9205,
```

```
                "Pu": 17.0700,
                "CartItems": [
                  {
                    "Id": 24.0,
                    "DetailPackId": 24.0,
                    "Name": "[domain]",
                    "Handle": "[object_handle]",
                    "Class": "domain",
                    "Attributes": {}
                  }
                ]
              }
            ]
          }
        ],
        "AutoRenew": false,
        "ModePaiementID": 3,
        "OpState": 1,
        "Errors": [],
        "IpAddress": "::1",
        "Currency": "EUR"
    }
}
```

Management

There are also a number of apis available for managing products and services such as domain, hosting, mail etc.

Appendix

Auth_url: https://auth.namebay.com

Base_url: https://rest.namebay.com

Technical documentation:

https://rest.namebay.com/help#Domain

https://rest.namebay.com/help#Panier

https://rest.namebay.com/help#OwnedProduct

https://rest.namebay.com/help#Product