

Actualizar un Tercero [BETA]

Funcionalidad en desarrollo y pruebas: este servicio está disponible en el ambiente de pruebas y todavía no debe usarse en producción. El funcionamiento, los campos y los mensajes de error pueden cambiar antes del despliegue definitivo.

Este servicio hace un **upsert por documento**: si el tercero existe, actualiza solo los campos que envías; si no existe, lo crea. Está pensado para que tu aplicación pueda propagar cambios sin conocer el `tercero_id` de nuby.

La regla más importante: **lo que no envías se conserva**. Esto aplica a los tipos de tercero, a los datos del cónyuge y a cualquier otro campo o JSON. No se borra nada por omisión.

¿Para qué sirve este servicio?

Úsalo para **mantener el tercero de nuby alineado con el de tu plataforma** sin guardar ni mapear identificadores internos. Los casos de uso más frecuentes son:

- **Propagar un cambio de datos** del tercero —nombre, dirección, teléfono o correo— desde tu sistema hacia nuby.
- **Actualizar un solo campo** (por ejemplo el `email`) sin reenviar el resto: lo que no envías se conserva.
- **Crear el tercero si el documento todavía no existe**, con el mismo servicio y los obligatorios del alta.
- **Corregir datos sin tocar los tipos ni el cónyuge** que ya estaban registrados.
- **Propagar cambios desde un sistema que solo conoce el documento** del cliente, no el `tercero_id`.

1. El Endpoint (La dirección web)

Método y dirección

Dato	Valor
Método	PUT

Dato	Valor
Ruta	/third-parties
Scope requerido	third-parties:update
Autenticación	Authorization: Bearer <jwt>
Cuerpo de la petición	Objeto JSON con los datos del tercero

PUT https://{{{instancia}}}/service/v2/public/third-parties

¿Qué debo colocar en {{instancia}}?

Es la **dirección web principal** con la que ingresas a tu plataforma, incluyendo la terminación `.nuby.app` o `.arrendasoft.co`. Si entras a nuby escribiendo `inmobiliaria.nuby.app` en el navegador, esa es tu instancia.

Importante: no incluyas `https://` ni barras diagonales (`/`) al final. La dirección se compone así: `https://` + tu instancia + `/service/v2/public/third-parties`.

Encabezados requeridos

Encabezado	¿Obligatorio?	Valor
Authorization	Sí	Bearer <token>, con el token obtenido en el servicio Login .
Content-Type	Sí	application/json: la petición lleva cuerpo JSON.

Permiso requerido (scope): third-parties:update

El cliente OAuth con el que se genera el token debe tener habilitado el permiso `third-parties:update`. Si el token es válido pero al cliente le falta ese permiso, la respuesta es `403` con el código `RECURSO_NO_AUTORIZADO`.

Si aún no tienes un token, consulta la página [Login](#) del libro **API nuby V2**. Allí se explica cómo identificarte y cómo se construye el encabezado `Authorization`.

Requisitos previos de esta operación

- `documento` es obligatorio siempre: es la llave con la que nuby identifica el tercero.
- El `documento` solo admite **letras y números**: sin puntos, guiones ni espacios (`900.123.456` se rechaza).

- Si el tercero **no existe**, el payload debe traer los obligatorios del alta; si falta uno, la respuesta es `422`.
- El cliente OAuth del token debe tener el permiso `third-parties:update`.

El documento es la clave. No se modifica por este endpoint: un documento distinto identifica otro tercero.

2. La Petición (¿Qué debes enviarnos?)

El cuerpo debe ser un **objeto JSON**. En la raíz viaja la referencia de idempotencia y, al mismo nivel, los datos del tercero.

Obligatorios en el alta

`referencia_externa`, `documento`, `tipo_doc_id`, `nombre1`, `direccion1`, `municipio_id` y al menos un tipo (`tipo_tercero_id` o `tipos[]`).

Obligatorios en la edición

`referencia_externa` y `documento`. El resto es opcional: solo se actualiza lo que envías.

Campos del cuerpo

Campo	Tipo	Requerido	Descripción
<code>referencia_externa</code>	texto	Sí	Identificador único de tu operación: es la llave de idempotencia. No puede venir vacía.
<code>documento</code>	texto	Sí	Documento o NIT del tercero. Es la llave de identidad y no se modifica por este endpoint. Solo letras y números.
<code>tipo_doc_id</code>	entero	Solo en el alta	Identificador del tipo de documento en la maestra de nuby. Debe ser mayor que cero y existir en el sistema.
<code>nombre1</code>	texto	Solo en el alta	Primer nombre, o la razón social si es una empresa. nuby lo guarda en mayúsculas.
<code>direccion1</code>	texto	Solo en el alta	Dirección principal del tercero.

Campo	Tipo	Requerido	Descripción
<code>municipio_id</code>	entero	Solo en el alta	Identificador del municipio. Debe ser mayor que cero y existir en la maestra de municipios.
<code>tipo_tercero_id</code> o <code>tipos[]</code>	entero / arreglo o cadena	Solo en el alta (al menos uno)	Rol o roles del tercero. Se fusionan, se eliminan duplicados y se descartan valores no numéricos o <code><= 0</code> . Si no se envían en la edición, se conservan los tipos actuales y, si no queda ninguno, la respuesta es <code>422</code> .
<code>persona</code>	entero	No	<code>1</code> persona natural (no admite NIT/NOP); <code>2</code> persona jurídica (exige NIT/NOP). Si se omite, la coherencia documento↔persona no se valida.
<code>nombre2</code>	texto	No	Segundo nombre. Se almacena en mayúsculas. Si no se envía, se conserva el valor actual.
<code>apellido1</code>	texto	No	Primer apellido. Se almacena en mayúsculas. Si no se envía, se conserva el valor actual.
<code>apellido2</code>	texto	No	Segundo apellido. Se almacena en mayúsculas. Si no se envía, se conserva el valor actual.
<code>sexo</code>	texto	No	Si no se envía, se conserva el valor actual.
<code>fecha_nacimiento</code>	fecha	No	Si no se envía, se conserva el valor actual.
<code>dv</code>	texto	No	Dígito de verificación del documento. Se acepta si viene y se conserva si no.
<code>tratamiento_id</code>	entero	No	Identificador de la maestra de tratamiento. Se valida contra el catálogo.
<code>direccion2</code>	texto	No	Dirección secundaria. Si no se envía, se conserva el valor actual.

Campo	Tipo	Requerido	Descripción
telefono1, telefono2	texto	No	Teléfonos del tercero. Si no se envían, se conservan.
fax	texto	No	Si no se envía, se conserva el valor actual.
email	texto	No	Correo principal. El servicio no valida su formato.
email_cumpleanos	texto	No	Correo alterno. Si no se envía, se conserva el valor actual.
codigo_postal	texto	No	Si no se envía, se conserva el valor actual.
tipo_persona_id	entero	No	Identificador de la maestra de tipo de persona. Se valida contra el catálogo.
regimen_id	entero	No	Régimen tributario. Se valida contra el catálogo.
resp_fiscal_id	entero	No	Responsabilidad fiscal. Se valida contra el catálogo.
autoretenedor	indicador	No	Marca de autorretenedor. Si no se envía, se conserva el valor actual.
autore_res	indicador	No	Si no se envía, se conserva el valor actual.
gran_contribuyente	indicador	No	Marca de gran contribuyente. Si no se envía, se conserva el valor actual.
gran_contr_res	indicador	No	Si no se envía, se conserva el valor actual.
aplica_documento_soporte	indicador	No	Marca de documento soporte. Si no se envía, se conserva el valor actual.
banco_id	entero	No	Banco del tercero. Se valida contra el catálogo.
intTipoCuentaB	entero	No	Tipo de cuenta bancaria. Si no se envía, se conserva el valor actual.
no_cuenta	texto	No	Número de cuenta bancaria. Si no se envía, se conserva el valor actual.
representante	texto	No	Si no se envía, se conserva el valor actual.

Campo	Tipo	Requerido	Descripción
<code>representante_legal</code>	texto	No	Si no se envía, se conserva el valor actual.
<code>certificado</code>	texto / indicador	No	Si no se envía, se conserva el valor actual.
<code>CY_*</code> (campos del cónyuge)	según el campo	No	Sin ningún <code>CY_*</code> se conserva <code>data_conyuge</code> ; si envías algunos, solo esos cambian.
<code>data_conyuge</code>	objeto JSON	No	Datos del cónyuge. Si no llega ningún <code>CY_*</code> , se conserva completo.
<code>data_propiedades</code>	objeto JSON	No	Si no se envía, se conserva el JSON anterior.
<code>por_retencion</code>	objeto JSON	No	Si no se envía, se conserva el JSON anterior.
<code>nombre_aux</code>	texto	No	Se calcula en el alta si falta y se conserva en la edición.

Campos que se ignoran: los campos fuera de esta lista (por ejemplo `tercero_id`, `creador` o `password_hash`) nunca se escriben y no generan error.

Preservación

- Sin `tipo_tercero_id` / `tipos[]`: se conservan los tipos existentes.
- Sin `CY_*`: se conserva el cónyuge; si envías algunos, solo esos cambian.
- Sin un campo o JSON: se conserva el valor anterior.
- `nombre_aux` se calcula en el alta si falta y se conserva en la edición.

Ejemplos de cuerpo

Actualizar solo el email de un tercero que ya existe:

```
{
  "referencia_externa": "TP-2026-0001-EDIT-1",
  "documento": "900123456",
  "email": "nuevo@acme.co"
}
```

Crear si no existe, con los obligatorios del alta:

```
{
  "referencia_externa": "TP-2026-0002",
  "documento": "900654321",
  "tipo_doc_id": 1,
  "nombre1": "ACME SAS",
  "tipo_tercero_id": 1,
  "direccion1": "CL 1 #2-3",
  "municipio_id": 1
}
```

3. La Respuesta (¿Qué te entregaremos?)

Si el tercero existe y el payload es válido, la respuesta es `200` con el identificador del tercero y los indicadores de la operación:

```
{
  "success": true,
  "status": 200,
  "message": "Tercero actualizado.",
  "body": {
    "tercero_id": 123,
    "documento": "900123456",
    "created": false,
    "updated": true,
    "idempotent_replay": false
  }
}
```

Estructura principal de la respuesta

Campo	Tipo	Descripción
<code>success</code>	booleano	<code>true</code> cuando la operación se completó.
<code>status</code>	entero	Replica el código HTTP de la respuesta: <code>200</code> si actualizó, <code>201</code> si creó.
<code>message</code>	texto	Mensaje legible del resultado.
<code>body</code>	objeto	Identificador del tercero e indicadores de la operación. En un error, este campo no existe.

Las respuestas de escriturano traen `pagination` ni `filters_applied`: esos campos pertenecen a las consultas de lectura.

Campos del cuerpo (`body`)

Campo	Tipo	Descripción
<code>tercero_id</code>	entero	Identificador interno del tercero en nuby.
<code>documento</code>	texto	Documento con el que se resolvió el tercero: el mismo que enviaste.
<code>created</code>	booleano	<code>true</code> si el tercero se creó en esta llamada.
<code>updated</code>	booleano	<code>true</code> si el tercero existía y se aplicaron los campos enviados.
<code>idempotent_replay</code>	booleano	<code>true</code> cuando la respuesta viene de un reintento con la misma <code>referencia_externa</code> .

Escenarios alternativos

Si el documento no existía y el payload trae los obligatorios, responde `201` con `created: true, updated: false`. El `body` queda así:

```
{
  "tercero_id": 124,
  "documento": "900654321",
  "created": true,
  "updated": false,
  "idempotent_replay": false
}
```

`idempotent_replay: true` indica que la respuesta viene de un reintento con la misma `referencia_externa`:

```
{
  "tercero_id": 123,
  "documento": "900123456",
  "created": false,
  "updated": true,
  "idempotent_replay": true
}
```

```
}
```

4. Idempotencia y reintentos

Toda escritura exige `referencia_externa`. La clave lógica es la combinación de tu origen, el recurso `tercero` y la referencia; el hash del payload es SHA-256 sobre una forma canónica del JSON, así que el **orden de las llaves no altera la comparación**.

Situación	Qué responde nuby
Reintento con el mismo payload	La respuesta original con <code>idempotent_replay: true</code> .
Otra solicitud con la misma referencia en curso	<code>409 IDEMPOTENCY_IN_PROGRESS</code> .
La misma referencia con un payload distinto	<code>409 IDEMPOTENCY_PAYLOAD_MISMATCH</code> .
La solicitud anterior terminó en <code>422</code>	La referencia se libera : puedes reintentar con el mismo valor y el payload corregido.
La solicitud anterior terminó en <code>500</code>	Se permite reintentar, incluso con datos corregidos.
No se pudo registrar la solicitud de idempotencia	<code>500 IDEMPOTENCY_STORAGE_ERROR</code> .

Una referencia por cambio. Una referencia distinta habilita un cambio nuevo sobre el mismo documento; reutilizar la anterior devuelve la respuesta vieja y tu cambio no se aplica.

5. Seguridad y Posibles Errores

Tu pase tiene fecha de caducidad

Por medidas de seguridad, el token que te entregamos **solo dura 1 hora**. Una vez transcurrido ese tiempo, el sistema te bloquea el acceso con un error `401` y tu sistema debe volver a consumir el servicio de **Login** para pedir un pase nuevo.

Esta respuesta la produce el **middleware de autenticación** y no el contrato de la operación, así que usa el envelope heredado (`statusCode` y `error.description`), no `success` ni `error_code`:

Situación	HTTP	Detalle
No se envió el encabezado <code>Authorization</code>	<code>401</code>	<code>JWT Token required.</code>
El encabezado no tiene el formato <code>Bearer {token}</code>	<code>400</code>	<code>JWT Token not send.</code>

Situación	HTTP	Detalle
El token no fue encontrado en el sistema	401	JWT Token not found.
El token expiró	401	JWT Token expired.

Errores del contrato de la operación

Los errores de la operación traen `success: false`, el código HTTP, un `message` legible y un `error_code` estable:

HTTP	error_code	Cuándo
400	VALIDATION_ERROR	El cuerpo no es un objeto JSON.
403	CONSUMIDOR_NO_AUTENTICADO	El token es válido, pero no corresponde a un cliente OAuth habilitado para consumir la API.
403	RECURSO_NO_AUTORIZADO	El consumidor no tiene <code>third-parties:update</code> .
409	TERCERO_IDENTIDAD_CONFLICTIVA	El documento existe con otro <code>tipo_doc_id</code> .
409	IDEMPOTENCY_PAYLOAD_MISMATCH	La misma referencia se usó con otro payload.
409	IDEMPOTENCY_IN_PROGRESS	Otra solicitud con la misma referencia está en proceso.
422	VALIDATION_ERROR	Falta un obligatorio o un valor no cumple las reglas del modelo.
500	INTERNAL_ERROR	Fallo técnico; el detalle queda en el log <code>API_INTEGRACION</code> .

Mensajes de validación frecuentes

- El campo "{Etiqueta}" es obligatorio.
- El campo "{Etiqueta}" debe ser un identificador numérico mayor que cero. (para `tipo_doc_id` y `municipio_id`)
- Debe enviar al menos un tipo de tercero en "Tipo de Tercero" o en `tipos[]`.
- El valor ingresado en el campo "{Etiqueta}" no es válido para el sistema. (maestras)
- Una Persona Natural no puede registrarse con un tipo de documento de tipo NIT.
- Una Persona Jurídica debe registrarse de forma obligatoria con un documento de tipo NIT.

Trata el `409 TERCERO_IDENTIDAD_CONFLICTIVA` como un caso de negocio, no como una falla técnica: ese documento existe en nuby con otro tipo de

documento, y este endpoint no lo cambia. No lo reintentas con el mismo payload.

6. Ejemplos de integración

Aquí tienes ejemplos listos para adaptar a tu plataforma. Todos usan `{{instancia}}`, `Authorization: Bearer`, actualizan solo el `email` de un tercero y tratan el éxito (`200/201`) junto con los rechazos de negocio más frecuentes (`409` y `422`).

cURL

```
TOKEN="TU_TOKEN_AQUI"
INSTANCIA="mi-inmobiliaria.nuby.app"

curl -X PUT "https://${INSTANCIA}/service/v2/public/third-parties" \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer ${TOKEN}" \
  -d '{
    "referencia_externa": "TP-2026-0001-EDIT-1",
    "documento": "900123456",
    "email": "nuevo@acme.co"
  }'
```

PHP

```
<?php
$instancia = 'mi-inmobiliaria.nuby.app'; // Tu dirección web completa
$token     = 'TU_TOKEN_AQUI';           // Token obtenido del servicio Login

$payload = [
    'referencia_externa' => 'TP-2026-0001-EDIT-1',
    'documento'          => '900123456',
    'email'              => 'nuevo@acme.co',
```

```

];

$ch = curl_init("https://{${instancia}}/service/v2/public/third-parties");
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'PUT');
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, [
    'Content-Type: application/json',
    "Authorization: Bearer ${token}",
]);
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($payload));

$respuesta = curl_exec($ch);
$httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
curl_close($ch);

$info = json_decode($respuesta, true);

if ($httpCode === 200 || $httpCode === 201) {
    echo "Tercero {$info['body']['tercero_id']}: {$httpCode}\n";
    echo "Creado: " . var_export($info['body']['created'], true) . "\n";
    echo "Actualizado: " . var_export($info['body']['updated'], true) . "\n";
} elseif ($httpCode === 409 || $httpCode === 422) {
    echo "Rechazo de negocio {$info['error_code']}: {$info['message']}\n";
} else {
    echo "Error HTTP {$httpCode}: {$info['message']}\n";
}

```

Python

```

import requests

# 1. Configura tus datos de acceso
instancia = "mi-inmobiliaria.nuby.app" # Reemplaza con tu dirección web completa
token = "TU_TOKEN_AQUI"                # Token obtenido del servicio Login

# 2. Prepara el cuerpo y envía la petición

```

```

url = f"https://{instancia}/service/v2/public/third-parties"
headers = {
    "Content-Type": "application/json",
    "Authorization": f"Bearer {token}",
}
payload = {
    "referencia_externa": "TP-2026-0001-EDIT-1",
    "documento": "900123456",
    "email": "nuevo@acme.co",
}

respuesta = requests.put(url, json=payload, headers=headers, timeout=30)

# 3. Procesa la respuesta
if respuesta.status_code in (200, 201):
    body = respuesta.json()["body"]
    print(f"Tercero {body['tercero_id']} - creado: {body['created']} - actualizado: {body['updated']}")
elif respuesta.status_code in (409, 422):
    error = respuesta.json()
    print(f"Rechazo de negocio {error['error_code']}: {error['message']}")
else:
    print(f"Error HTTP {respuesta.status_code}: {respuesta.text}")

```

JavaScript (Fetch API)

```

// 1. Configura tus datos de acceso
const instancia = 'mi-inmobiliaria.nuby.app'; // Reemplaza con tu dirección web completa
const token = 'TU_TOKEN_AQUI'; // Token obtenido del servicio Login

// 2. Función que actualiza (o crea) el tercero
async function actualizarTercero(payload) {
    const url = `https://${instancia}/service/v2/public/third-parties`;

    const response = await fetch(url, {
        method: 'PUT',

```

```

    headers: {
      'Content-Type': 'application/json',
      'Authorization': `Bearer ${token}`,
    },
    body: JSON.stringify(payload),
  });

  const info = await response.json().catch(() => ({}));

  if (!response.ok) {
    console.error(`Error HTTP ${response.status} ${info.error_code ?? ''}: ${info.message ?? ''}`);
    return null;
  }

  return info.body;
}

// 3. Cambia solo el email y revisa el resultado
actualizarTercero({
  referencia_externa: 'TP-2026-0001-EDIT-1',
  documento: '900123456',
  email: 'nuevo@acme.co',
}).then((body) => {
  if (!body) {
    return;
  }

  console.log(`Tercero ${body.tercero_id} creado: ${body.created} actualizado: ${body.updated}`);
});

```

7. Buenas prácticas y límites conocidos

Recomendaciones

- **Manda una referencia nueva por cambio.** No reutilices la referencia de un cambio anterior: el replay devuelve la respuesta vieja.

- **No envíes campos vacíos para “borrar”.** Un campo ausente se conserva; para limpiar un valor hay que definir una regla explícita (no cubierta por este endpoint).
- **Manda el tipo de tercero en el alta.** Puede ser cualquiera válido; no hay un mapeo fijo.
- **Verifica el 201/200.** El primero crea, el segundo actualiza; ambos son éxito.
- **No cambies el documento por este endpoint.** Si el documento cambió, es otro tercero.

Límites de este servicio

- **No devuelve el detalle del tercero.** Solo el `tercero_id`, el `documento` y los indicadores de la operación.
- **No borra datos.** Un campo que no envías se conserva; no existe una operación de borrado por este servicio.
- **No completa datos por su cuenta.** Si el tercero no existe y falta un obligatorio, responde `422`.
- **No cambia el tipo de documento.** Si el documento existe con otro `tipo_doc_id`, responde `409 TERCERO_IDENTIDAD_CONFLICTIVA`.
- **No impone un tipo de tercero.** Acepta cualquier `tipo_tercero_id` o `tipos[]` válido.

Revisión #1

Creado el 28 octubre 2026 14:26:41 por Jorge Iván González

Actualizado el 28 octubre 2026 14:27:19 por Jorge Iván González