



API REFERENCE · API V1

Send SMS

Send one message to one mobile number. The reply comes back in milliseconds with a `messageId`; delivery is reported afterwards.

Updated 2026-10-10 · generated 2026-10-10 · smsray.com/docs/send-sms

1 Send an SMS

POST /sms/send <https://api.smsray.com/api/sms/v1/sms/send>

Queue one SMS to one mobile number, as free text or from an approved template. The cost is reserved from your workspace balance at accept time and the message goes straight into delivery.

HEADERS

`x-api-key` string **required**

Your secret key, `ls_live_` followed by 48 hex characters.

`content-type` string **required**

`application/json`

`Idempotency-Key` string **optional**

1–200 printable ASCII characters. Safe retries for 24 h. See [Idempotency](#).

BODY (JSON)

`to` string **required**

Recipient. **E.164 is recommended** (+9779801234567); the national format is accepted for your workspace's country (for a Nepal workspace 98XXXXXXX, 97... or 96..., with or without 977). Must match `^\+?\d{7,15}$`. See [Numbers & countries](#).

`text` string **optional**

Free text. Send **exactly one** of `text` or `templateId`. At least 1 character, at most 6 segments (GSM-7: 918 characters, UCS-2: 402 code units). When your workspace requires templates (the default) the text must match one of your approved templates of the same `type`.

`templateId` string **optional**

An approved template of your workspace. SMSRay renders it with `variables`. Send exactly one of `text` or `templateId`.

`variables` object **optional**

Values for the template's `{{variables}}`, as strings or numbers. Every variable is required; each value is 1–100 characters with no line break.

`type` string **optional**

`transactional` (default), `otp` or `promotional`. Must be one of your workspace's `allowedTypes`; `promotional` is available on Enterprise only. With `templateId`, the template's type is used when omitted.

senderId string **optional**

Stored as from on the message. Not validated in v1 — sender-ID approval is not available yet.

EXAMPLE REQUESTS**cURL**

```
curl https://api.smsray.com/api/sms/v1/sms/send \
-H "x-api-key: $SMSRAY_API_KEY" \
-H "content-type: application/json" \
-H "Idempotency-Key: order-1042-shipped" \
-d '{ "to": "+9779801234567", "text": "Your order #1042 has shipped." }'
```

Node.js

```
const res = await fetch("https://api.smsray.com/api/sms/v1/sms/send", {
  method: "POST",
  headers: {
    "x-api-key": process.env.SMSRAY_API_KEY,
    "content-type": "application/json",
    "Idempotency-Key": "order-1042-shipped",
  },
  body: JSON.stringify({ to: "+9779801234567", text: "Your order #1042 has shipped." }),
});
const { messageId, status, segments } = await res.json();
// status is "queued"; track it with GET /sms/messages/:id or a webhook
```

Python

```
import os, requests

r = requests.post(
    "https://api.smsray.com/api/sms/v1/sms/send",
    headers={
        "x-api-key": os.environ["SMSRAY_API_KEY"],
        "Idempotency-Key": "order-1042-shipped",
    },
    json={"to": "+9779801234567", "text": "Your order #1042 has shipped."},
    timeout=10,
)
print(r.json()["messageId"])
```

PHP

```
$ch = curl_init("https://api.smsray.com/api/sms/v1/sms/send");
curl_setopt_array($ch, [
    CURLOPT_POST => true,
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_HTTPHEADER => [
        "x-api-key: " . getenv("SMSRAY_API_KEY"),
        "content-type: application/json",
        "Idempotency-Key: order-1042-shipped",
    ],
    CURLOPT_POSTFIELDS => json_encode([
        "to" => "+9779801234567",
        "text" => "Your order #1042 has shipped.",
    ]),
]);
$res = json_decode(curl_exec($ch), true);
```

```
echo $res["messageId"];
```

Go

```
package main

import (
    "bytes"
    "encoding/json"
    "fmt"
    "net/http"
    "os"
)

func main() {
    body, _ := json.Marshal(map[string]string{
        "to": "+9779801234567",
        "text": "Your order #1042 has shipped.",
    })
    req, _ := http.NewRequest("POST", "https://api.smsray.com/api/sms/v1/sms/send",
» bytes.NewReader(body))
    req.Header.Set("x-api-key", os.Getenv("SMSRAY_API_KEY"))
    req.Header.Set("content-type", "application/json")
    req.Header.Set("Idempotency-Key", "order-1042-shipped")

    res, err := http.DefaultClient.Do(req)
    if err != nil {
        panic(err)
    }
    defer res.Body.Close()

    var out struct {
        MessageID string `json:"messageId"`
        Status     string `json:"status"`
    }
    json.NewDecoder(res.Body).Decode(&out)
    fmt.Println(out.MessageID, out.Status) // status is "queued"
}
```

RESPONSES

200

```
{
  "success": true,
  "messageId": "2b1f6c1e-8d4f-4c55-9a51-0d2b0b7f3a11",
  "status": "queued",
  "segments": 1,
  "cost": 0.5,
  "encoding": "GSM7",
  "from": "SMSRay"
}
```

400 · Validation error

```
{
  "error": "Invalid request",
  "code": "invalid_request",
  "details": [
    { "path": ["to"], "message": "Required", "code": "invalid_type" }
  ]
}
```

403 · No matching template

```
{
  "error": "This message doesn't match an approved template. Submit the template for approval in
» your SMSRay dashboard, or send with an approved templateId.",
  "code": "template_required"
}
```

422 · Country not supported yet

```
{
  "error": "We don't deliver to India yet. This workspace can send to: Nepal.",
  "code": "destination_not_supported",
  "details": { "country": "IN", "allowed": ["NP"] }
}
```

RESPONSE FIELDS

messageId string (uuid) **required**

Use it with GET /sms/messages/:id and to match webhook events.

status string **required**

Always queued in this reply.

segments integer **required**

How many SMS parts the text needs.

cost number **required**

Your rate for type × segments, reserved from the balance.

encoding string **required**

GSM7 or UCS2.

from string **required**

The sender recorded for the message: your **senderId** if given, otherwise the default sender.

ERRORS

HTTP	CODE	WHEN
400	invalid_request	Validation failed (see details), both or neither of text and templateId were sent, or the text needs more than 6 segments.
400	invalid_number	The number can't receive SMS — for example a landline, or a number that fails its country's mobile rules.
400	invalid_variables	With templateId : a variable is missing, unknown, empty, longer than 100 characters or contains a line break.
402	insufficient_balance	Your workspace balance is lower than the cost of this message.

HTTP	CODE	WHEN
401	unauthorized	The x-api-key header is missing ("Missing x-api-key") or the key is unknown ("Invalid API key").
403	client_disabled	The API client that owns this key has been disabled in the dashboard.
403	forbidden	This message type is not allowed on your plan or workspace.
403	template_required	The text doesn't match an approved template of the same type, or templateId is not an approved template of this workspace.
403	workspace_pending	Your workspace has not been activated yet, so it cannot send.
403	workspace_suspended	Your workspace is suspended.
409	idempotency_conflict	The same Idempotency-Key was used with a different body, or the first request is still running (Retry-After: 1).
422	destination_not_supported	A valid number in a country your workspace can't send to yet. details is { country, allowed }; show "We don't deliver to <country> yet".
429	rate_limited	You exceeded your per-client request rate (default 20 requests per second). Honour Retry-After.
503	unavailable	Delivery is temporarily unavailable. Nothing was charged; retry with backoff.
500	server_error	Something went wrong on our side. The body never contains a stack trace.

NOTES

- The reply always says queued. Follow the message with GET /sms/messages/:id or, better, a message.status webhook. No webhook is sent for the initial queued.
- Every custom message must use a template approved by SMSRay. Approval protects deliverability and keeps spam off the networks; we aim to review new templates within one business day. OTPs sent with POST /sms/otp/send use SMSRay's built-in template, so OTP works on day one.
- Messages are delivered over the recipient's mobile network, whichever one they are on, with automatic retries. If delivery finally fails, the message ends as failed and the reserved cost is refunded. undelivered is not refunded.
- Numbers are stored in E.164, so to in lookups and webhooks is always +<country code><number>.
- Send an Idempotency-Key on every send so network retries never double-send or double-bill.
- On a key in test mode the message is accepted and recorded with status test, but never sent and never charged.

The web page is always the current version: smsray.com/docs/send-sms

Full API reference PDF: smsray.com/docs/pdf · Your keys and live examples: app.smsray.com/developers