Reference

Initialize transactional flow

Initialize transactional flow

POST https://us-east-1.di-hub.telesign.com/hubv2/model/flow/init-flow

> **NOTE:**
>
> Select regional availability. This feature is currently available for customers with end users in select countries. To find out if this feature is available in your region, contact a Telesign expert. This product is available for full-service accounts only.

This endpoint starts a transactional flow inside the Flow Builder. The request must contain a valid flowId and a dataToReplace object whose keys match the parameters defined when the flow was created.

General Requirements

- **Authentication:** This authentication method is mandatory, to obtain a valid token from the authentication system, post that include it in all API requests.
- **Encoding:** The API only accepts characters in UTF-8 Unicode format.
- **Accepts:** application/json is the expected data format for the API.
- **Responds with:** application/json is the data format for API responses.
- **Required headers:**
  - `Content-Type: application/json`
  - `Authorization: Bearer [your-Access-token]`

How `dataToReplace` works

`dataToReplace` is the payload that feeds the flow with the dynamic values your business logic needs. Each entry is a simple key → value pair. **Key** must be identical (case, spelling, numbers, underscores) to the parameter name configured under *Request parameters*. **Value** is a primitive JSON value (string, number, or boolean).

LANGUAGE

Shell | Node | Ruby | PHP | Python

CREDENTIALS BEARER JWT

Bearer token

cURL Request — Full example Example

```
curl --request POST \
     --url https://us-east-1.di-hub.telesign.com/hubv2/model/fl
     --header 'accept: application/json' \
     --header 'content-type: application/json' \
     --data '
{
  "flowId": "18380c27-1d1c-4f79-a449-a7c8b12ddf62",
  "dataToReplace": {
    "name": "Juan",
    "cellphone": "11234567890",
    "email": "juan.perez@example.com",
    "age": 28,
    "message": "Hello, this is an example"
  }
}
'
```

Try It!

RESPONSE — 400 - RequestValidationFailed Example

```
{
  "data": null,
  "error": {
    "message": "Request validation failed",
    "details": {
      "flowId": {
        "message": "flowId must be a UUID v4"
      },
      "dataToReplace": {
        "message": "dataToReplace must be an object"
      }
    }
  }
}
```

Required versus optional parameters

In two situations, a parameter becomes required and will trigger a `400 – Missing Required Parameters` error if absent:

Field(s)	Situation
Mail field / Phone field	Select **one** existing parameter that holds the user's e-mail or phone number. The Hub flags it as required for nodes such as **PhoneID**.
Advanced → Select required fields	Multi-select listing **all** parameters in the flow. Any parameter you select here is required at runtime; un-selected parameters remain optional.

If a required parameter is missing **or** the key name doesn't match, the response is:

JSON

```
{
  "data": null,
  "error": {
    "message": "Required fields missing:
<parameterName>"
  }
}
```

> 📘 NOTE:
>
> The comparison is case-sensitive (clientId ≠ ClientID). Parameters not flagged in either list above are optional and can be omitted without error.

Naming and value rules

#	Content type	Rule
1	Case-sensitive	`clientId` , `clientID` , and `CLIENTID` are three different parameters.
2	Free-form names	Letters, numbers, and underscores allowed (`segment` ,

#	Content type	Rule
		`Segment1` , `customer_email`). Avoid spaces.
3	Primitive values	Accepts string, number, boolean.
4	One-to-one mapping	Every **required** parameter must appear in `dataToReplace` ; optional parameters may be skipped.

Example 1 – ID lookup

1. Create flow: Add a Request parameter called `idNumber` .
2. Call `/init-flow` with:

```
JSON

{
  "flowId": "…",
  "dataToReplace": {
    "idNumber": "90123456"
  }
}
```

3. Result: The flow forwards `90123456` to a downstream personal-data service.

Example 2 – Customer segment routing

1. Create flow: Add a parameter called `segment` .
2. Call `/init-flow` with:

```
JSON

{
  "flowId": "…",
  "dataToReplace": {
    "segment": "GOLD"
  }
}
```

3. Result: Branching rules route GOLD customers to the premium path.

> 📘 NOTE:
>
> If you rename or delete a parameter in the flow designer, update your `dataToReplace` keys accordingly to avoid the 400 – Missing Required Parameters error.

Log in to see full request history

TIME	STATUS	USER AGENT
Make a request to see history.		

0 Requests This Month

Body Params

flowId string required

18380c27-1d1c-4f

Unique identifier of the flow created in the Transactional Hub.

dataToReplace object required

Dictionary of key → value pairs the flow will consume. Keys must match the names defined in Request parameters (case-sensitive). Values are primitive JSON types.

DATATOREPLACE OBJECT +

Responses

— 200

Success. The request has been successfully processed, and the flow has been initialized.

Error Condition	error.message	Description
Success	—	The request was processed successfully. A `pTracking` UUID is returned so you can trace the transaction.

— 400

Missing Required Parameters. Missing required parameters or invalid body fields. The client must send both `flowId` (UUID v4) and `dataToReplace` (object).

Error Condition	`error.message`	Description
Missing / invalid body fields	Request validation failed	One or more fields are missing or have the wrong type/format. See error.details for a per-field breakdown.
Missing required dynamic parameters	Required fields missing: , …	The flow was configured to expect certain parameters that were not supplied.
Invalid flow status	Invalid Flow	The supplied flowId exists, but its status does not allow this operation.

RESPONSE BODY

object

data object | null

Contains the results of the request. Since this is an error, this value is `null`.

HAS ADDITIONAL FIELDS

error object

Provides details on why the request failed.

message string

A high-level, general description of the problem. It summarizes what went wrong with the overall request without providing specific details.

details object

Provides a specific, detailed report for each field that had an issue. This helps you identify exactly what needs to be fixed.

DETAILS OBJECT +

— 401

Authentication issues. The request failed because no authorization header provided, an incomplete header (e.g. `Bearer` without token), no authorization header provided, unsupported authorization method, invalid signature, or an expired token.

Error Condition	`error.message`	Description
No Authorization header	No authorization header provided	Authentication header is completely absent.
Bad header format	Authorization header format error	Header present but incomplete (e.g. Bearer without the token).
Unsupported scheme	Unsupported authorization method	Scheme is not Bearer or Basic.
Token failed validation	Error validating token	Token could not be parsed or verified.
Token expired / not found	Invalid or expired token	Token is syntactically valid but no longer accepted.

— 404

Flow ID not found. The flowId provided does not exist or does not belong to the authenticated account.

Error Condition	error.message	Description
Unknown flowId	This FlowId was not found	The supplied flowId does not exist.

— 500

Internal Server Error. Unexpected exception on the server side.

Error Condition	error.message	Description
Internal error	Internal Server Error	Unexpected exception on the server side.

Updated about 1 month ago