

Request Validation

Request validation is used to ensure that the incoming request message is properly formatted and contains the proper attributes. You can set up request validators in an API's [OpenAPI](#) definition file and then import the [OpenAPI](#) definitions into API Gateway. You can also set them up in the API Gateway console or by calling the API Gateway REST API, AWS CLI, or one of the AWS SDKs.

The [x-amazon-apigateway-request-validators](#) object allows you to define custom validators for your API, for this there are three extensions that can be used:

- [x-amazon-apigateway-request-validators](#) - Defines the supported request validators for the containing API as a map between a validator name and the associated request validation rules. This extension applies to a REST API.
- [x-amazon-apigateway-request-validators.requestValidator](#) - Specifies the validation rules of a request validator as part of the x-amazon-apigateway-request-validators object map definition.
- [x-amazon-apigateway-request-validator](#) - Specifies a request validator, by referencing a request_validator_name of the x-amazon-apigateway-request-validators object map, to enable request validation on the containing API or a method. The value of this extension is a JSON string.

The [x-amazon-apigateway-request-validator](#) extension can be specified at the API level or at the method level. The API-level validator applies to all of the methods unless it is overridden by the method-level validator.

Create and configure request validators for your first API

In this section, we will configure three request validators, being:

- All** - This validator is for both body and parameters validation.
- Body** - This is the body-only validator.
- Params** - This is the parameters-only validator.

We also will configure the extension [x-amazon-apigateway-request-validator](#) to use the **Body** validator.

- Go back to the AWS Cloud9 console.
- Add a new [x-amazon-apigateway-request-validators](#) and [x-amazon-apigateway-request-validator](#) to your file `openapi.yaml`.

Review the code and then **copy/paste** it into the `openapi.yaml` file

```
1  openapi: "3.0.1"
2  info:
3    title: "module-2-my-first-api"
4    description: "First API with IaC example"
5    version: "1.0"
6
7  x-amazon-apigateway-request-validators:
8    All:
9      validateRequestParameters: true
10     validateRequestBody: true
11     Body:
12       validateRequestParameters: false
13       validateRequestBody: true
14     Params:
15       validateRequestParameters: true
16       validateRequestBody: false
17
18  x-amazon-apigateway-request-validator: Body
19
20  paths:
21
22    /pricemeter:
23      post:
24        consumes:
25          - application/json
26        produces:
27          - application/json
28        responses:
29          "200":
30            description: "200 response"
31
32        x-amazon-apigateway-integration:
33          httpMethod: "POST"
34          credentials:
35            Fn::GetAtt: [LambdaExecutionRole, Arn]
36          uri:
37            Fn::Sub: "arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/${CostCalculator.Arn}/invocations"
38          responses:
39            default:
40              statusCode: "200"
41              responseTemplates:
42                application/json: |
43                  #set($allParams = $input.params())
44                  {
45                    "body-json": $input.json('$'),
46                    "params": {
47                      #foreach($type in $allParams.keySet())
48                        #set($params = $allParams.get($type))
49                        "type": {
50                          #foreach($paramName in $params.keySet())
51                            "fieldName": "${util.escapeJavaScript($params.get($paramName))}"
52                            #if($foreach.hasNext),end
53                          #end
54                        }
55                      #if($foreach.hasNext),end
56                    },
57                    "stage-variables": {
58                      #foreach($key in $stageVariables.keySet())
59                        "key": "${util.escapeJavaScript($stageVariables.get($key))}"
60                        #if($foreach.hasNext),end
61                      #end
62                    },
63                    "context": {
64                      "account-id": "${context.identity.accountId}",
65                      "api-id": "${context.apiId}",
66                      "api-key": "${context.identity.apiKey}",
67                      "authorizer-principal-id": "${context.authorizer.principalId}",
68                      "caller": "${context.identity.caller}",
69                      "cognito-authentication-provider": "${context.identity.cognitoAuthenticationProvider}",
70                      "cognito-authentication-type": "${context.identity.cognitoAuthenticationType}",
71                      "cognito-identity-id": "${context.identity.cognitoIdentityId}",
72                      "cognito-identity-pool-id": "${context.identity.cognitoIdentityPoolId}",
73                      "http-method": "${context.httpMethod}",
74                      "stage": "${context.stage}",
75                      "source-ip": "${context.identity.sourceIp}",
76                      "user": "${context.identity.user}",
77                      "user-agent": "${context.identity.userAgent}",
78                      "user-arn": "${context.identity.userArn}",
79                      "request-id": "${context.requestId}",
80                      "resource-id": "${context.resourceId}",
81                      "resource-path": "${context.resourcePath}",
82                    }
83                  }
84              requestTemplates:
85                application/json:
86                  #set($inputRoot = $input.path("$"))
87                  {
88                    "price": "$inputRoot.price",
89                    "size": "$inputRoot.size",
90                    "unit": "$inputRoot.unit",
91                    "downPaymentAmount": $inputRoot.downPayment
92                  }
93                  passthroughBehavior: "when_no_match"
94                  type: "aws"
95
96        components:
97          schemas:
98            costCalculatorRequest:
99              type: object
100              properties:
101                price:
102                  type: number
103                size:
104                  type: number
105                unit:
106                  type: string
107                downPayment:
108                  type: number
```

On line 7 to 16 we are defining the **All**, **Body** and **Params** validators.

On line 18 we are defining that the entire API paths will use the validator **Body**.

Important
Remember that you can override the API validator method by specifying the [x-amazon-apigateway-request-validator](#) inside the method.

- Deploy the project again using SAM to verify that the **Request Validator** was added to your API Method Request.

```
1 sam build && sam deploy
```

- Check if the stack was successfully updated and open the [Amazon API Gateway console](#).
- Choose your REST API named, `module-2-my-first-api`.
- In **Resources** click in `/pricemeter` POST
- Click in **Method Request** and check that the **Request Validator** is configured to use the validator **Body**.

Create resource

/pricemeter POST

Method request

Integration request

Integration response

Method response

Test

Method request settings

Authorization: NONE

Request validator: Body

API key required: False

SDK operation name: Generated based on method and path

The next step is to map the `costCalculatorRequest` model to the **Request Body**. After configuring the **Request Body** you should be able to validate the request body against the model that you defined previously.

Configure the Request Body for request validation

- Go back to the AWS Cloud9 console.
- Add a new [requestBody](#) object to your file `openapi.yaml`.

Review the code and then **copy/paste** it into the `openapi.yaml` file

```
1  openapi: "3.0.1"
2  info:
3    title: "module-2-my-first-api"
4    description: "First API with IaC example"
5    version: "1.0"
6
7  x-amazon-apigateway-request-validators:
8    All:
9      validateRequestParameters: true
10     validateRequestBody: true
11     Body:
12       validateRequestParameters: false
13       validateRequestBody: true
14     Params:
15       validateRequestParameters: true
16       validateRequestBody: false
17
18  x-amazon-apigateway-request-validator: Body
19
20  paths:
21
22    /pricemeter:
23      post:
24        consumes:
25          - application/json
26        produces:
27          - application/json
28        responses:
29          "200":
30            description: "200 response"
31
32        requestBody:
33          required: true
34          content:
35            application/json:
36              schema:
37                $ref: "#/components/schemas/costCalculatorRequest"
38
39        x-amazon-apigateway-integration:
40          httpMethod: "POST"
41          credentials:
42            Fn::GetAtt: [LambdaExecutionRole, Arn]
43          uri:
44            Fn::Sub: "arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/${CostCalculator.Arn}/invocations"
45          responses:
46            default:
47              statusCode: "200"
48              responseTemplates:
49                application/json: |
50                  #set($allParams = $input.params())
51                  {
52                    "body-json": $input.json('$'),
53                    "params": {
54                      #foreach($type in $allParams.keySet())
55                        #set($params = $allParams.get($type))
56                        "type": {
57                          #foreach($paramName in $params.keySet())
58                            "fieldName": "${util.escapeJavaScript($params.get($paramName))}"
59                            #if($foreach.hasNext),end
60                          #end
61                        }
62                      #if($foreach.hasNext),end
63                    },
64                    "stage-variables": {
65                      #foreach($key in $stageVariables.keySet())
66                        "key": "${util.escapeJavaScript($stageVariables.get($key))}"
67                        #if($foreach.hasNext),end
68                      #end
69                    },
70                    "context": {
71                      "account-id": "${context.identity.accountId}",
72                      "api-id": "${context.apiId}",
73                      "api-key": "${context.identity.apiKey}",
74                      "authorizer-principal-id": "${context.authorizer.principalId}",
75                      "caller": "${context.identity.caller}",
76                      "cognito-authentication-provider": "${context.identity.cognitoAuthenticationProvider}",
77                      "cognito-authentication-type": "${context.identity.cognitoAuthenticationType}",
78                      "cognito-identity-id": "${context.identity.cognitoIdentityId}",
79                      "cognito-identity-pool-id": "${context.identity.cognitoIdentityPoolId}",
80                      "http-method": "${context.httpMethod}",
81                      "stage": "${context.stage}",
82                      "source-ip": "${context.identity.sourceIp}",
83                      "user": "${context.identity.user}",
84                      "user-agent": "${context.identity.userAgent}",
85                      "user-arn": "${context.identity.userArn}",
86                      "request-id": "${context.requestId}",
87                      "resource-id": "${context.resourceId}",
88                      "resource-path": "${context.resourcePath}",
89                    }
90                  }
91              requestTemplates:
92                application/json:
93                  #set($inputRoot = $input.path("$"))
94                  {
95                    "price": "$inputRoot.price",
96                    "size": "$inputRoot.size",
97                    "unit": "$inputRoot.unit",
98                    "downPaymentAmount": $inputRoot.downPayment
99                  }
100                  passthroughBehavior: "when_no_match"
101                  type: "aws"
102
103        components:
104          schemas:
105            costCalculatorRequest:
106              type: object
107              properties:
108                price:
109                  type: number
110                size:
111                  type: number
112                unit:
113                  type: string
114                downPayment:
115                  type: number
```

On line 31 we are defining the `requestBody` object to the resource `/pricemeter`, method `POST`.

On line 36 we are referencing the schema that contains the definition for the `costCalculatorRequest` that was previously defined in this module.

- Deploy the project again using SAM to verify that the **Request Body** was mapped to your API Method Request.

```
1 sam build && sam deploy
```

Create resource

/pricemeter POST

Method request

Integration request

Integration response

Method response

Test

Method request settings

Authorization: NONE

Request validator: Body

API key required: False

SDK operation name: Generated based on method and path

Request paths (0)

URL query string parameters (0)

HTTP request headers (0)

Request body (1)

Content type: application/json

costCalculatorRequest

Test the API

Go back to the API Gateway console and select the `POST` integration method to test out the API.

- Click on the **Test** option to provide a sample request message.
- Copy and Paste the following JSON Sample in the **Request Body** section

```
{
  "price": "400000",
  "size": "1000",
  "unit": "sqft",
  "downPayment": "20"
}
```

Create resource

/pricemeter POST

Headers

header[Value1]

header[Value2]

Client certificate

No client certificates have been generated.

Request body

```
1 {
2   "price": "400000",
3   "size": "1000",
4   "unit": "sqft",
5   "downPayment": "20"
6 }
7
```

Test

- Click on **Test**

Notice that this time, the message failed with a message of `Invalid request body`.

Create resource

/pricemeter POST

pricemeter - POST method test results

Request: pricemeter

Latency: 5

Status: 400

Response body

```
{
  "message": "Invalid request body"
}
```

Response headers

```
{
  "x-amzn-ErrorType": "BadRequestException"
}
```

Log

```
Tue Sep 19 21:00:35 UTC 2023 : Starting execution for request: 4d4b571-dfeb-4c8-b06-899c32507c49
Tue Sep 19 21:00:35 UTC 2023 : HTTP Method: POST, Resource Path: /pricemeter
Tue Sep 19 21:00:35 UTC 2023 : Method request path: {}
Tue Sep 19 21:00:35 UTC 2023 : Method request query string: {}
Tue Sep 19 21:00:35 UTC 2023 : Method request headers: {}
Tue Sep 19 21:00:35 UTC 2023 : Method request body before transformations: {
  "price": "400000",
  "size": "1000",
  "unit": "sqft",
  "downPayment": "20"
}
Tue Sep 19 21:00:35 UTC 2023 : Request body does not match model schema for content type application/json: [Instance type (string) does not match any allowed primitive type (allowed: ["integer", "number"])]
Tue Sep 19 21:00:35 UTC 2023 : Method completed with status: 400
```

The request failed validation because `price`, `size` and `downPayment` are defined as **number** in our model/schema. Remove the string quotes (") from price, size, and downPayment and try your request again.

- Click on the **Test** icon to provide a sample request message.
- Copy and Paste the following JSON Sample in the **Request Body** section

```
{
  "price": 400000,
  "size": 1000,
  "unit": "sqft",
  "downPayment": 20
}
```

- Click on **Test**

You should see that the normal response is returned by the service.

Select your cookie preferences

We use essential cookies and similar tools that are necessary to provide our site and services. We use performance cookies to collect anonymous statistics, so we can understand how customers use our site and make improvements. Essential cookies cannot be deactivated, but you can choose "Customize" or "Decline" to decline performance cookies.

If you agree, AWS and approved third parties will also use cookies to provide useful site features, remember your preferences, and display relevant content, including relevant advertising. To accept or decline all non-essential cookies, choose "Accept" or "Decline." To make more detailed choices, choose "Customize."

Accept

Decline

Customize

