TGH

Making Integrations Simpler

# Mastering RAML 1.0: A Comprehensive Guide To Data Types, Security Schemes, & Traits


**Author**

## Aniket Pal


## Contents



©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

# Fragments in RAML

- RAML stands for Restful API modeling language. It is used to write the specification of Restful APIs, which is understandable by both humans and machines.
- Fragments are the reusable logic that can be maintained through every API in API specification.
- There are two types of fragments:
  - **Local fragments**: These fragments are not published to exchange and are used in a single project.
  - **Global fragments**: These fragments are published to exchange and can be added to multiple projects as a dependency.
- The Fragments in the RAML1.0 are:
  - DataType.
  - SecuritySchemes
  - Traits.
  - Examples.
  - ResourceTypes.
  - Library.
  - Annotation Types
  - Extensions.
  - Overlays.
- In this blog, we'll learn about some of the fragments, which are Data types, Security Schemes, and Traits.

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

# Data Types

- RAML 1.0 introduces data types, which provide a concise and powerful way to describe the data in an API.
- It is defined using the keyword **type**.
- RAML 1.0 has support for the following data types:
  - String:
    - It is a type of datatype that has a set of characters that can include alphabets, numbers, and symbols inside double quotes.
    - We can provide a range for this datatype using the keyword:
      - minLength: It is used to specify the minimum length of a string datatype.
      - maxLength: It is used to specify the maximum length of a string datatype.
    - Example:

```
type: string
minLength: 5
maxLength: 15
example: "Alexa"
```

  - Number:
    - It is a type of datatype that accepts only the numeric value.
    - It can be restricted to a specific range using the following keyword:
      - Minimum: It is used to specify the minimum number.
      - Maximum: It is used to specify the maximum number.
    - Example:

```
type: number
minimum: 3
maximum: 10
example: 7.3
```

  - Boolean:
    - It is a type of datatype that has only two values i.e. true or false.
    - Example:

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

```
type: boolean
example: true
```

- Object:
  - It is a type of datatype i.e. in object format which can contain several attributes.
  - Example:

```
type: object
properties:
  Id:
    type: integer
    minimum: 1
    maximum: 100
    example: 12
  Name:
    type: string
    minLength: 5
    maxLength: 15
    example: "Allen"
```

- Array:
  - It is a type of datatype i.e. in array format which can be a number array, string array, or collection of objects.
  - It uses the **items** keyword to define the type of array.
  - The size of the array can be restricted using the keywords:
    - minItems: It specifies the minimum number of elements in the array.
    - maxItems: It specifies the maximum number of elements in the array.
  - Example:

```
type: array
items: object
minItems: 1
maxItems: 6
example: [
        {
```

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

```
    "Id": 101,
    "Name": "Alex"
  },
  {
    "Id": 102,
    "Name": "Allen"
  }
]
```

- Integer:
  - It is a type of datatype that accepts only integer values.
  - Examples:

```
type: integer
minimum: 1
maximum: 100
example: 23
```

- File:
  - It is a type of datatype that accepts only files.
  - It has a limit of 8.3 MB.
  - Example:

```
post:
  body:
    application/json:
      type: file
```

- Date-only:
  - It is a type of datatype that accepts only the date, with no implications about time or offset.
  - Example:

```
type: date-only
example: 2024-09-21
```

- Time-only:

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

- It is a type of datatype that accepts only the time, with no implications about date or offset.
- Example:

```
type: time-only
example: 12:30:00
```

- DateTime:
  - It is a type of datatype that accepts the datetime
  - Example:

```
type: datetime
example: 2016-02-28T16:41:41.090Z
```

- DateTime-only:
  - It is a type of datatype that accepts only the datetime, with no implications about offset.
  - Example:

```
type: datetime-only
example: 2015-07-04T21:00:00
```

- Any:
  - It is a type of datatype that can accept any type of datatype.
  - Example:

```
type: any
```

- Nil:
  - It is a type of datatype that accepts only nil.
  - Example:

```
type: nil
```

- To externalize the datatype, we need to create a separate file of type datatype and refer it in the main RAML file.

External Datatype file

```
#%RAML 1.0 DataType
```

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

```
    type: string
    minLength: 5
    maxLength: 15
    example: "Allen"
```

Main RAML file:

```
    types:
      name: !include /DataTypes/nameDataType.raml

    /students:
      get:
    queryParameters:
      name:
       required: true
        responses:
        200:
         body:
          application/json:
            example: !include Examples/getStudentsResponse.json
```

In the main RAML file, we'll create a placeholder and we'll make of that placeholder to refer to the datatype. In this case, we are making use of a name as a placeholder.

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

# Security Schemes

- These are used to define the security mechanism that will be applied to all resources of the API or some specific resources of the API.
- It is created using the keyword **securitySchemes** and referred to using the keyword **securedBy**.
- It is described by using the keyword **describedBy.**
- **Type** keyword is used to specify the type of security applied.
- RAML1.0, supports the following policies at the specification level:
    - Basic Authentication.
    - OAuth1.0
    - OAuth2.0
    - Custom policy.
    - Digest Authentication.
    - Pass through.
- Example:

```
securitySchemes:
  basic:
    type: Basic Authentication
    describedBy:
      headers:
        username:
          type: string
          minLength: 5
          maxLength: 15
          example: "username"
        password:
          type: string
          minLength: 5
          maxLength: 15
          example: "password"
/students:
  get:
```

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

```
    securedBy: [basic]
      responses:
        200:
    body:
      application/json:
          example: !include Examples/getStudentsResponse.json
```

- **basic** is the name for the security schemes.

- To externalize the security schemes, we need to create a separate file of type security scheme and refer to the main file.

External security schemes:

```
#%RAML 1.0 SecurityScheme

type: Basic Authentication
describedBy:
  headers:
    username:
      type: string
      minLength: 5
      maxLength: 15
      example: "username"
    password:
      type: string
      minLength: 5
      maxLength: 15
      example: "password"
```

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

Main RAML file:

```
securitySchemes:
  basicAuth: !include /Security Schemes/basicAuthsecurityScheme.raml

/students:
  get:
    securedBy: [basicAuth]
    responses:
      200:
        body:
          application/json:
            example: !include Examples/getStudentsResponse.json
```

In the main RAML file, we need to create a placeholder i.e. basicAuth in this case, and in the resource, we'll use the same placeholder after the securedBy keyword to apply the policy on the resource.

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

# Traits

- Traits are the set of behaviors that can be applied to different parts of API.
- It is created using the keyword **traits** and referred to using the keyword **is**.
- Example:

```
traits:
  qpTrait:
    queryParameters:
      dept:
        type: string
        required: true
        example: "CSE"

/students:
  get:
    is: [qpTrait]
    responses:
      200:
        body:
          application/json:
            example: !include Examples/getStudentsResponse.json
```

  - **qpTrait** is the name of the trait, using which it will be referred to in the resource.
- To externalize the Trait, we have to create a separate file type of Trait and that needed to be referred in the main file.

External Trait file

```
#%RAML 1.0 Trait

responses:
  500:
    body:
```

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

```
    application/body:
      example: ../Examples/500_errorResponse.json

  400:
    body:
      application/json:
        example: ../Examples/400_errorResponse.json

  404:
    body:
      application/json:
        example: ../Examples/404_errorResponse.json

  405:
    body:
      application/json:
        example: ../Examples/405_errorResponse.json
```

Main RAML file

```
traits:
  error: !include /Traits/errorTrait.raml

/students:
  get:
    is: [error]
    responses:
      200:
        body:
          application/json:
            example: !include Examples/getStudentsResponse.json
```

The above external trait is an error trait that has all the error responses. In the main RAML file, we have to create a placeholder that will be used to refer to the resource.

©TGH Software Solutions Pvt. Ltd.

No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent

TGH

Making Integrations Simpler

# TGH Software Solutions Pvt. Ltd.

*www.techygeekhub.com*

At TGH, we specialize in driving digital transformation through seamless Integration Technologies.

Operating as an INTEGRATION FACTORY, we serve as a one-stop shop for all your integration needs. Our expert team is well-versed in enterprise software and legacy system integration, along with leading iPaaS technologies like Boomi, MuleSoft, Workato, OIC, and more.

We're committed to enhancing business processes and solving problems through our integration expertise.


**Email address**

connect@techygeekhub.com

**Phone number**

+ 011-40071137

+ 91-8810610395

**Our offices**

Noida Office

iThum
Plot No -40, Tower A,
Office No: 810,
Sector-62, Noida,
Uttar Pradesh, 201301

Hyderabad Office

Plot no: 6/3, 5th Floor,
Techno Pearl Building,
HUDA Techno Enclave,
HITEC City, Hyderabad,
Telangana 500081