

TOC

TOC	1
LAB : Creating and using Data Types	1
Step 1 – Understand Data Types used in our use case	2
Step 2 – Create Error DataType Fragment	4
Step 3 –Define Error Datatype in root RAML file and use it to describe response body	7
Step 4 – Create Address and Registration Datatype Fragments	8
Step 5 –Define Registration Datatype in root RAML file and use it to describe request/response body.....	9
Step 6 – Create Room and Hotel DataType Fragments.....	11
Step 7 –Define Hotel Datatype in root RAML file and use it to describe request/response body.....	12
Step 8 – Create Booking DataType Fragment	14
Step 9–Define Booking Datatype in root RAML file and use it to describe request/response body	15
Step 10 – Create Status Datatype Fragment	16
Step 11 –Define Status Datatype in root RAML file and use it to describe response body	17
End of Exercise!!.....	18

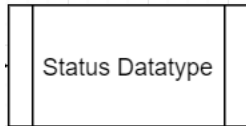
LAB : Creating and using Data Types

In this lab you will -

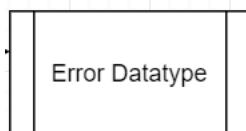
- Create DataType Fragments
- In root RAML, define data types using DataType Fragments
- Use the data types to describe the request body and response body

Step 1 – Understand Data Types used in our use case

Datatype used in Response body of DELETE method (200 HTTP status code):-



Datatype used in the Response Body (Error) of all methods (404/500 HTTP status code of all resources) :-



Registration, Hotel and Booking DataType –

Registration datatype is used to represent the registration of Traveller

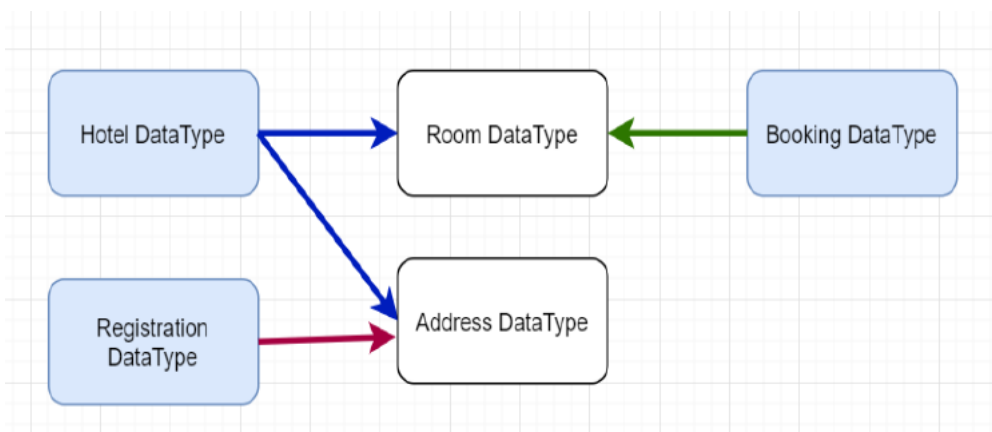
Hotel datatype is used to represent a hotel.

Booking datatype is used to represent the booking done by traveller in a hotel.

Resources in Root RAML file uses Registration, Hotel and Booking but these datatypes use other datatypes also.

Registration, Hotel and Booking DataType references other datatypes :-

Datatypes in blue are the main datatypes used in the request and response body of resources in root RAML . Datatypes marked in white are the ones referenced by Main datatypes:-



Trainer : Ruchi Saini

Hotel Datatype uses Room Datatype to define what kind of rooms are available in Hotel. Hotel Datatype uses Address datatype to represent the address of Hotel.

Registration Datatype uses Address datatype to represent the address of traveller who has registered.

Booking Datatype uses Room Datatype to represent the room booked by traveller.

Data Types used by Resources are as follows :-

Resource Name	Resource Path	Methods (Http Status Code)	DataType
Registrations	/registrations	GET (with 200 Response Body)	Registration[]
		POST – In request body	Registration
		For GET (404 Error Response Body) and POST (500 Error Response Body)	Error
Registration	/registrations/{registrationId}	GET (with 200 Response Body)	Registration
		PATCH – In request body	Registration
		DELETE (with 200 Response Body)	Status
		For GET and DELETE (404 Error Response Body) For PATCH (500 Response Body)	Error
Hotels	/hotels	GET (with 200 Response Body)	Hotel[]
		POST – In request body	Hotel
		For GET (404 Error Response Body) and POST (500 Error Response Body)	Error
Hotel	/hotels/{hotelId}	GET (with 200 Response Body)	Hotel
		PATCH – In request body	Hotel
		DELETE (with 200 Response Body)	Status
		For GET and DELETE (404 Error Response Body) For PATCH (500 Response Body)	Error
	/hotels/{hotelId} /bookings	GET (with 200 Response Body)	Booking[]
Bookings	/bookings	GET (with 200 Response Body)	Booking[]
		POST – In request body	Booking
		For GET (404 Error Response Body) and POST (500 Error Response Body)	Error

Trainer : Ruchi Saini

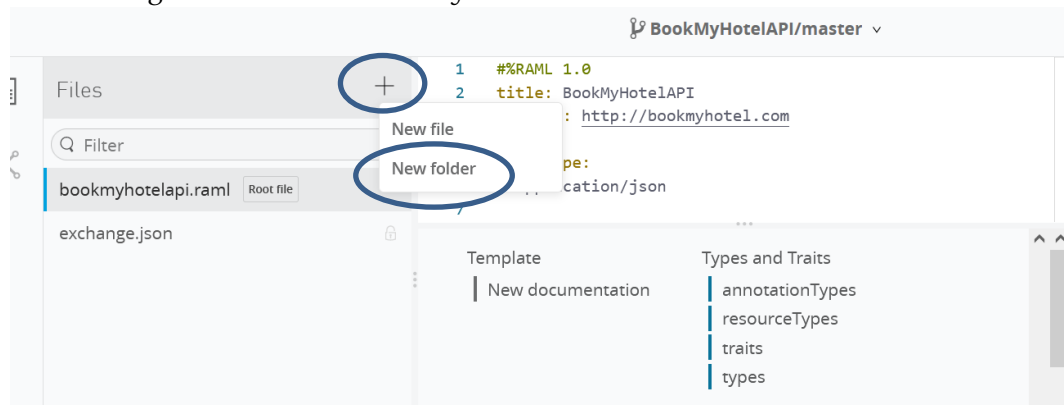
Booking	/bookings/{bookingId}	GET (with 200 Response Body)	Booking
		PATCH – In request body	Booking
		DELETE (with 200 Response Body)	Status
		For GET and DELETE (404 Error Response Body) For PATCH (500 Response Body)	Error

We will create datatype fragments for the above mentioned datatypes and we will then use them to define the request and response bodies of our Resource methods.

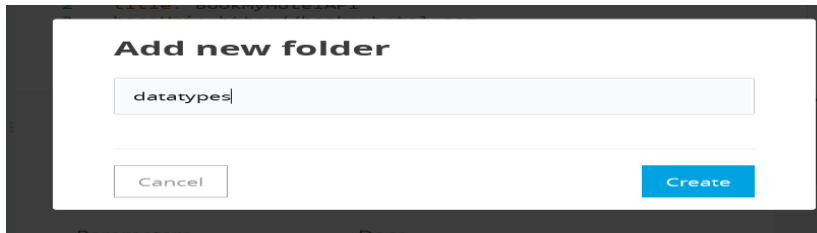
Congratulations, you have completed this step!!

Step 2 – Create Error DataType Fragment

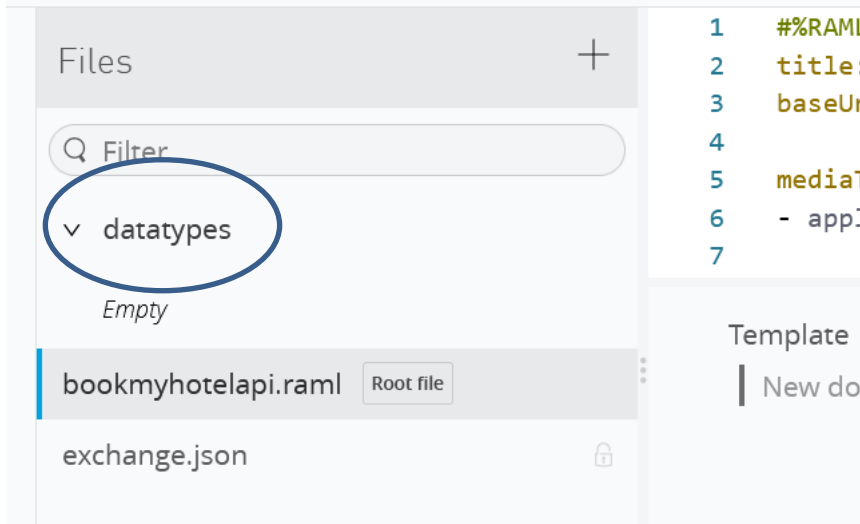
1. Click on + sign and then Create Folder from the Menu as shown below :-



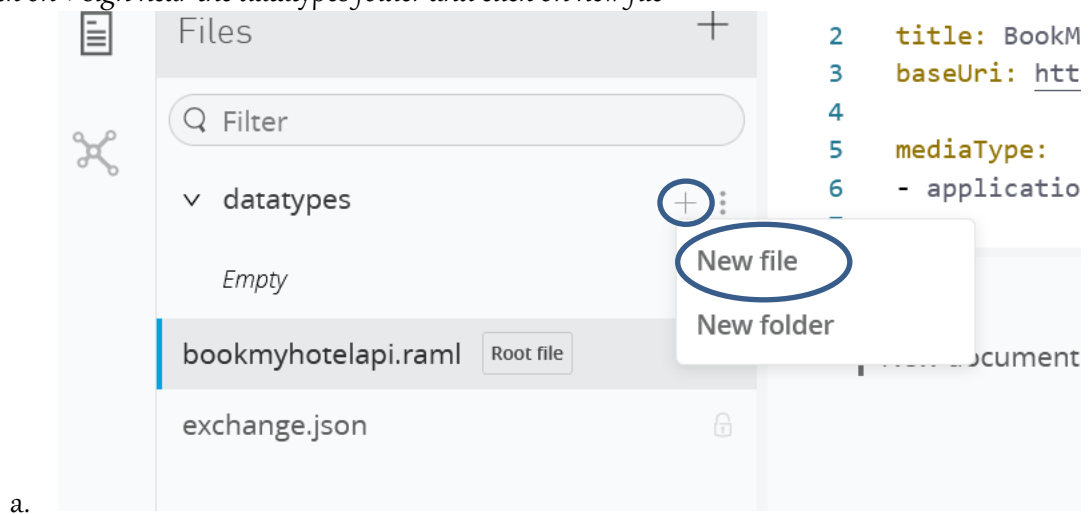
2. In the dialogue box, specify the name of the folder as datatypes and then click on Create button



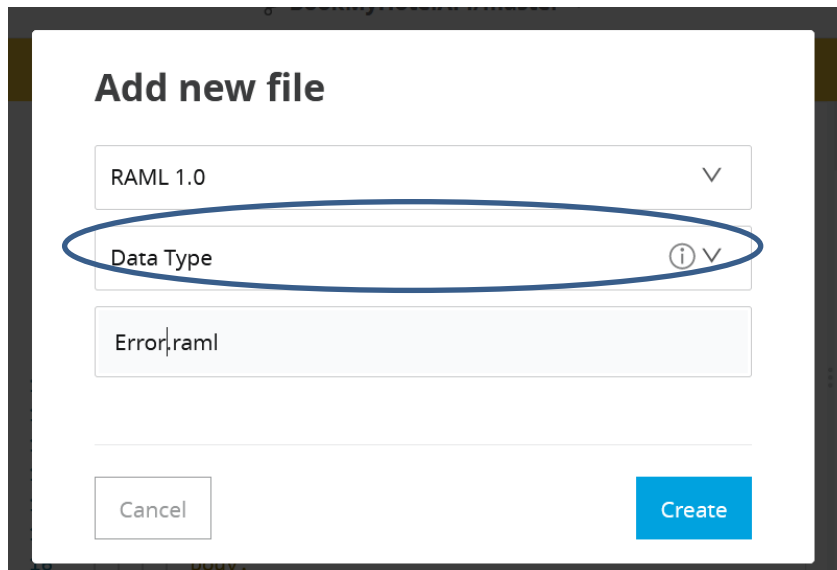
3. In the left hand side you can see the datatypes folder created and it is empty as of now -



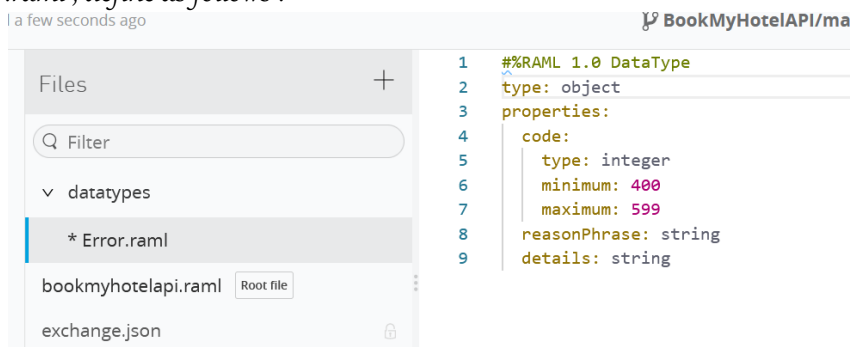
4. Click on + sign near the datatypes folder and click on new file



5. In the Add New File dialogue box, select **Data Type** from the drop down and specify the name of as **Error.raml**



- a.
 - b. This will create Error datatype fragment (Error.raml)
6. In Error.raml , define as follows :-



- a.
- b. The first line is added by default and shows the fragment identifier.
- c. Under the fragment identifier, Specify type as object
- d. Then Specify properties facet
- e. Under properties facet, specify all properties i.e. code, reasonPhrase and details
- f. The property **code** is integer and denotes the http status code of error. For the property **code**, we are also specifying **minimum** and **maximum** facets. Check above how the **type**, **minimum** and **maximum** facets are defined for the **code** property
- g. The property **reasonPhrase** is string and denotes the reason of error
- h. The property **details** is string and denotes the error message in detail.

Step 3 – Define Error Datatype in root RAML file and use it to describe response body

1. In the base/root RAML file bookmyhotelapi.raml , define Error datatype as follows -

```
1  #%RAML 1.0
2  title: BookMyHotelAPI
3  baseUrl: http://bookmyhotel.com
4
5  mediaType:
6  - application/json
7
8  types:
9  |   Error: !include datatypes/Error.raml
10
11 /registrations:
12 |   get:
```

- a. Here we have specified **types**:
- b. Under it , specify the keyname: value
- c. Keyname/ datatype is specified as **Error**
- d. Value has the include tag with location of Error datatype fragment's raml file. The include tag takes the argument which is the location of External file, parses it and sets the value using external file data.

2. For registrations resource (/registrations) GET method, describe the **response body** for 404

HTTP status code using **Error** Datatype as shown below -

```
- /registrations:
  !
  get:
  |   headers:
  |   |   Accept?:
  |   |   responses:
  |   |   200:
  |   |   |   headers:
  |   |   |   |   Content-Type:
  |   |   |   body:
  |   |   404:
  |   |   |   headers:
  |   |   |   |   Content-Type:
  |   |   |   body:
  |   |   |   Error
```

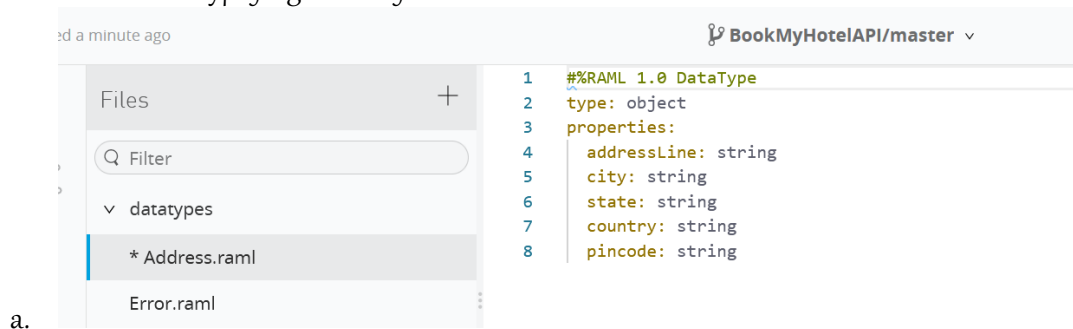
3. In the response of each resource/method combination, we have used either 404 Error HTTP status code or 500 Error HTTP status code.

4. Describe the response body using **Error** datatype for 404/500 Http status code of all resources/methods in the same way

Congratulations, you have completed this step!!

Step 4 – Create Address and Registration Datatype Fragments

1. Click on + sign near the datatypes folder and click on new file
2. In the Add New File dialogue box, select **DataType** from the drop down and specify the name as **Address.raml**
3. Define Address Datatype fragment as follows :-



4. Specify **type** as object and all its **properties** – addressLine, city, state, country and pincode.
5. The datatype of each property is string
6. Now in the next few steps, lets create Registration datatype fragment.
7. Click on + sign near the datatypes folder and click on new file
8. In the Add New File dialogue box, select **DataType** from the drop down and specify the name as **Registration.raml**
9. Define Registration Datatype fragment as follows :-


```
1  #%RAML 1.0 DataType
2  type: object
3  properties:
4    registrationId: string
5    customerName: string
6    title: string
7    address: !include Address.raml
8    paymentType: string
9
```

10. Registration Datatype is of type object. The **properties** facet is used to define its properties.
11. As per this datatype , when a traveller is registered then he has the **registrationId**. His **name**, **title**, **address** and **paymentType** are recorded
12. The properties - registrationId, customerName , title and paymentType are of type String.
13. The value of address property is defined using the previously created Address Datatype fragment.

Congratulations, you have completed this step!!

Step 5 –Define Registration Datatype in root RAML file and use it to describe request/response body

1. In the base/root RAML file bookmyhotelapi.raml , define Registration datatype as follows -

```

#%RAML 1.0
title: BookMyHotelAPI
baseUri: http://bookmyhotel.com

mediaType:
- application/json

types:
  Registration: !include datatypes/Registration.raml
  Error: !include datatypes/Error.raml
```

2. For registrations resource (/registrations) **GET method** – **200** HTTP status code, describe the **response body** as **Registration[]** as shown below =

a.

```
/registrations:
  get:
    headers:
      Accept?:
    responses:
      200:
        headers:
          Content-Type:
        body:
          Registration[]
      404:
        headers:
          Content-Type:
        body:
          Error
```

3. For registrations resource (/registrations) **POST method** – describe the **request body** as **Registration** as shown below =

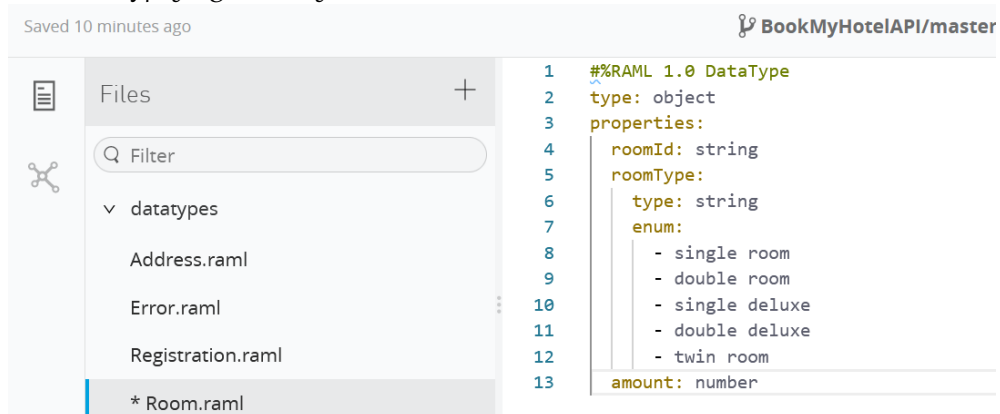
```
post:
  headers:
    Content-Type:
  body:
    Registration
  responses:
    201:
      headers:
        Location:
    500:
      headers:
        Content-Type:
      body:
        Error
```

- a.
4. For Registration resource (/registrations/{registrationId}) **GET** method, Describe the response body for 200 HTTP status code using **Registration** datatype
5. For Registration resource (/registrations/{registrationId}) **PATCH** method, Describe the request body using **Registration** datatype

Congratulations, you have completed this step!!

Step 6 – Create Room and Hotel DataType Fragments

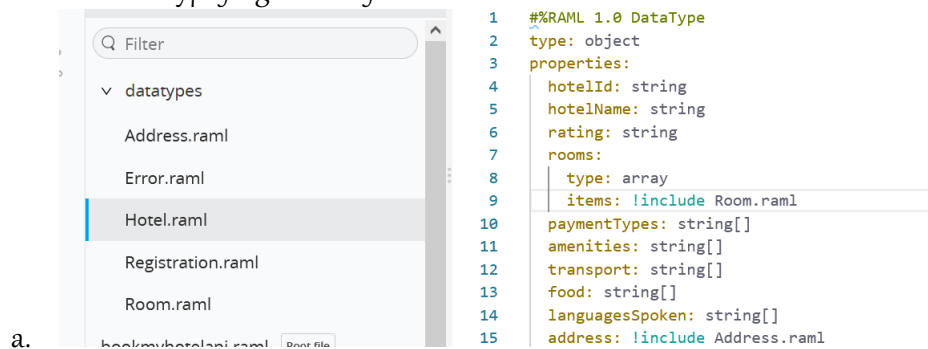
1. Click on + sign near the datatypes folder and click on new file
2. In the Add New File dialogue box, select **DataType** from the drop down and specify the name as **Room.raml**
3. Define Room Datatype fragment as follows :-

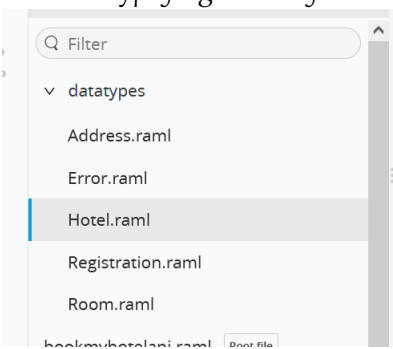


```
1  #%RAML 1.0 DataType
2  type: object
3  properties:
4    roomId: string
5    roomType:
6      type: string
7      enum:
8        - single room
9        - double room
10       - single deluxe
11       - double deluxe
12       - twin room
13  amount: number
```

- a.
4. Specify Type as Object. This datatype represents Hotel **Room**.
5. The Room is defined using roomId, roomType and amount(price associated with this room)
6. Property roomId is of type string and property amount is of type number.

7. Property **roomType** is of type string and **enum** facet is specified for this property. The enum facet is used to define possible values.
8. Lets create **Hotel datatype fragment** in next few steps
9. Click on + sign near the datatypes folder and click on new file
10. In the Add New File dialogue box, select **Data Type** from the drop down and specify the name as **Hotel.raml**
11. Define Hotel Datatype fragment as follows :-



- a. 
- b. This consists of hotel details like hotelId, hotelName, rating etc.
- c. The rooms property defines the types of room available in hotel. Notice it is of type array and items facet is used to include the Room datatype fragment
- d. The properties like paymentTypes, amenities, transport and food are string array.
- e. The address property specifies the address of hotel. This is set using Address datatype fragment.

Congratulations, you have completed this step!!

Step 7 – Define Hotel Datatype in root RAML file and use it to describe request/response body

1. In the base/root RAML file bookmyhotelapi.raml , define Hotel datatype as follows -

```
1  #%RAML 1.0
2  title: BookMyHotelAPI
3  baseUrl: http://bookmyhotel.com
4
5  mediaType:
6  - application/json
7
8  types:
9  Hotel: !include datatypes/Hotel.raml
10 Registration: !include datatypes/Registration.raml
11 Error: !include datatypes/Error.raml
12
13
14 /registrations:
15   get:
16     headers:
```

2. For Hotels resource (/hotels) **GET** method – **200** HTTP status code, describe the **response body** as

Hotel[] as shown below :

a.

```
responses:
  200:
    headers:
      Content-Type:
    body:
      Hotel[]
  404:
    headers:
      Content-Type:
    body:
      Error
```

3. For hotels resource (/hotels) **POST** method – describe the **request body** as **Hotel** as shown below

:-

```
post:
  headers:
    Content-Type:
  body:
    Hotel
  responses:
    201:
      headers:
        Location:
    500:
      headers:
        Content-Type:
```

a.

4. For Hotel resource (/hotels/{hotelId}) **GET** method, Describe the response body for 200 HTTP status code using Hotel datatype
5. For Hotel resource (/hotels/{hotelId}) **PATCH** method, Describe the request body using Hotel datatype

Congratulations, you have completed this step!!

Step 8 – Create Booking DataType Fragment

1. Click on + sign near the datatypes folder and click on new file
2. In the Add New File dialogue box, select **DataType** from the drop down and specify the name as **Booking.raml**
3. Define Booking Datatype fragment as follows :-

```
1  #%RAML 1.0 DataType
2  type: object
3  properties:
4    bookingId: string
5    bookingStartDate: date-only
6    bookingEndDate: date-only
7    amenitiesSelected: string[]
8    transportSelected: string[]
9    food: string
10   room: !include Room.raml
11
```

a.

4. Booking Datatype represents the booking of a traveller.
5. Booking consist of bookingId, booking start and end date, what amenities are selected , what transport is selected, what type of food traveller wants and which room he has booked.

6. The Booking Datatype is of type object. Its properties are of type string, date-only , string[]. The room property has value which is set using Room Datatype fragment created in one of the steps above.

Congratulations, you have completed this step!!

Step 9—Define Booking Datatype in root RAML file and use it to describe request/response body

1. In the base/root RAML file bookmyhotelapi.raml , define Booking datatype as follows -

```
1  #%RAML 1.0
2  title: BookMyHotelAPI
3  baseUrl: http://bookmyhotel.com
4
5  mediaType:
6  - application/json
7
8  types:
9    Booking: !include datatypes/Booking.raml
10   Hotel: !include datatypes/Hotel.raml
11   Registration: !include datatypes/Registration.raml
12   Error: !include datatypes/Error.raml
13
14
15  /registrations:
16    get:
17      headers:
18        Accept?:
```

2. For (/hotels/{hotelId}/bookings) GET method – 200 HTTP status code, describe the **response body** as **Booking[]** as shown below :-

a. 

```

/ bookings:
  get:
    headers:
      Accept?:
    responses:
      200:
        headers:
          Content-Type:
        body:
          Booking[]
      404:
        headers:
          Content-Type:
        body:
          Error

```

3. For Bookings resource (/bookings) **GET** method – **200** HTTP status code, describe the **response body** as **Booking[]**
4. For Bookings resource (/bookings) **POST** method – describe the **request body** as **Booking**
5. For Booking resource (/bookings/{bookingId}) **GET** method, Describe the **response body** for **200** HTTP status code using **Booking** datatype
6. For Booking resource (/bookings/{bookingId}) **PATCH** method, Describe the **request body** using **Booking** datatype

Congratulations, you have completed this step!!

Step 10 – Create Status Datatype Fragment

1. Click on + sign near the datatypes folder and click on new file
2. In the Add New File dialogue box, select **DataType** from the drop down and specify the name as **Status.raml**
3. Define Status Datatype fragment as follows :-


```
1  %%RAML 1.0 DataType
2  type: object
3  properties:
4    message: string
5
```

a.

4. The status message can be returned in the response body to the consumer wherever this datatype is used.

Congratulations, you have completed this step!!

Step 11 – Define Status Datatype in root RAML file and use it to describe response body

1. In the base/root RAML file bookmyhotelapi.raml , define Status datatype as follows -

```
1  %%RAML 1.0
2  title: BookMyHotelAPI
3  baseUri: http://bookmyhotel.com
4
5  mediaType:
6    - application/json
7
8  types:
9    Status: !include datatypes/Status.raml
10   Booking: !include datatypes/Booking.raml
11   Hotel: !include datatypes/Hotel.raml
12   Registration: !include datatypes/Registration.raml
13   Error: !include datatypes/Error.raml
14
15
16 /registrations:
17   get:
18     headers:
19       Accept?:
20     responses:
21       200:
```

2. For all nested resources Registration(/registrations/{registrationId}), Hotel (/hotels/{hotelId}), Booking (/bookings/{bookingId}) **DELETE method** – 200 HTTP status code, describe the **response body** as **Status** as shown below :-

a.

```
delete:
  headers:
    Accept?:
  responses:
    200:
      headers:
        Content-Type:
      body:
        Status
    404:
      headers:
        Content-Type:
      body:
        Error
```

Congratulations, you have completed this step!!

Congratulations!! You have completed this exercise.

End of Exercise!!