Creating a JSON schema

JSON Schemas allows us to create a schema for the data. This would allow us to validate the data and ensure that it is complete and has the correct structure. This would also adress one of the drawbacks of noSQL and, potentially, XML databases: ***how to ensure that the data is complete***, meaning that all records contain the same data.

We'll build a schema for an interview-like content for a JSON document that we'll later convert to XML.

The content for the interview object

This is a complete JSON document that matches the schema. I cheated a little with the document as I created it alongside the schema.

It is predicated on ideas about the content and how I want to manipulate it. These ideas will be discussed in detail when we talk about the schema.

```
{
  "id": "group-i"group-interview-001""title": "First Group Interview",
  "type": "interview-group",
  "date": "2022-03-27",
  "location": "Office",
  "interviewer": [
    {
      "name": "Batman",
      "email": "bm@gothamn.com"
    }
  ],
  "interviewee": [
    {
      "name": "Scarecrow",
      "email": "drCrane@arkham.com"
    }
  ],
  "audio": {
```

```
    "format": "mp3",
    "url": "https://example.com/audio/clip001.mp3",
    "duration": 45,
    "duration_unit": "minutes"
  },
  "transcript": {
    "url": "https://example.com/transcripts/clip001.html",
    "content": "the textual content of the transcript"
  }
}
```

First Pass: Getting the basics down

With a working document ready to go, we'll look at the first version of the schema and discuss some of the decisions about them.

The first block defines metadata for the schema. This is nont visible to the user and not essential for the schema to work but it helps developers looking at the schema to better understand what it is and how it works.

the `$schema` property defines the version of the JSON schema specification that the document conforms to. Different drafts/versions of the specification change how some elements are defined and how they are validated so it's important to keep this value up and, if necessary, update it when new versions introduce breaking changes.

```
{
  "title": "JSON schem"JSON schema for interviews""description": "JSON sch
  "$schema": "https://json-schema.org/draft/2020-12/schema",
```

Following the metadata, the schema defines the structure of the document. The `type` property defines the type of the document, a JSON object.

It then starts to define the properties of the document, these is the content of conforming documents.

The `$schema` property in this context can be used to reference this schema in conformant documents.

`id` and `title` are simple properties that are strings. Validation will fail otherwise.

```
"type": "object""object""properties": {
  "$schema": {
    "type": "string"
  },
  "id": {
    "type": "string",
    "description": "Unique identifier for the interview"
  },
  "title": {
    "type": "string",
    "description": "Title of the interview"
  },
```

The `type` property, in this context, indicates the type of interview the object represents. Rather than let everyone entering data for a given interview, we set up an enumeration of the types of interviews that we want to support. We also set a default value in case we forget to enter it.

```
  "type": {
    "type": "string",
    "string""description": "Type of interview",
    "default": "interview",
    "enum"    "interview",
      "interview-one-on-one",
      "interview-group",
      "interview-one-on-one-video",
      "interview-g"interview",
      "interview-one-on-one-audio",
      "interview-group-audio"
    ]
  },
```

`date` holds a string representation of the date the interview was held, in the format `yyyy-mm-dd`.

`location` is where the interview took place as a string

```
    "date": {
      "type": "string",
      "string""description"of the interview",
      "format": "date"": " },
    "location": "location""location": {
      "type"cription": "Locati": ""description": "Location of the intervie
    },
```

`interviewer` and `interviewee` are arrays of objects, each represeting a person conducting the interview (`interviewer`) or a person who was interviewed (`interviewee`).

Both of these properties define lower (`minItems`) and upper (`maxItems`) limits on the number of items in the array.

Both of these properties use a `$ref` keyword to reference to an object defined elsewhere in the schema via a `$defs` object.

```
    "interviewers": {
      "type": "array",
      "array""description": "Interviewer(s) of the interview",
      "minItems": 1,
      "maxItems": 5,
      "items"     "$ref": "#/$d": "person"
      }
    },
    "interviewees": {
      ""interviewees": {
      "type"tion": "Subject(s": ""description": "Subject(s) of the interv
      "minItems": 1,
      "maxItems" "$ref": "#": ""items": {
        "$ref": "#/$defs/person"
      }
    },
```

We use `audio` to hold information about the audio recording of the interview.

The `type` property defines the media format for the interview file.

The `url` defines the location of the file using a URL.

We use `duration` and `duration_unit` to define the length of the audio recording.

Rather than use a string to define the duration we build it from the two properties. This will make it easier to create queries asking for interviews of a certain duration.

If we use an `audio` object then all the properties are required.

```
    "audio": {
      "type": "object",
      "object""properties": {
        "format"      "type": "strin": "          "description": "Format (
        },
        "url": {
          "type": "s": ""url": {
          "type"n": "URL of the audio ": ""description"        "duration"
          "type": "numb": ""duration" "description": "Dur": ": "number",
          "description"     "duration_unit": {
          "type": "string"": ""duration_unit""duration_unit": {
          "type": "string",
          "description"
        "format",
        "url",
        "duration"format""required": [
        "format",
        "url",
        "duration",
        "duration_unit"
      ]
    },
```

The transcript object holds information about the transcript produced by running the audio file through a speech-to-text engine and then reviewed by the interviewers.

It has two properties: `url` holds the URL location for the transcript and `content` holds the complete text of the transcribed audio.

If we use a transcript object then both its properties are required.

```
  "transcript": {
    "type": "object",
    "object""properties": {
      "url"      "type": "strin": "          "description": "URL of th"
      },
      "content": {
        "type": "string": "         "description": "Content of the tr":
    "required": [
      "url",
      "content"
 "url""required": [
      "url",
      "content"
    ]
  }
},
```

The `$defs` object hold definitions that are common to more than one object in the schema. The inteview schema defines a `person` object for both the `interviewer` and `interviewee` arrays.

The `name` property is required.

```
"$defs": {
  "person": {
    "type": "object",
    "object""properties": {
      "name"      "type": "strin": "          "description": "Name of t
      "email": {
        "type": "string": "         "description": "Email of the pers":
      "name"
    ]
  }
```

```
    }
  }
  "name""required": [
          "name"
        ]
      }
    }
  }
```

I've validated the schema below using the JSON Schema Validator using the document instance discussed in ***The content for the interview object***.

Refining the schema

After creating the first version of the schema, I realized that there are some additional changes that would be nice to have.

The first one is to make the schema more flexible. Right now it only allows audio interviews, there is no way to record in-person or video interviews.

I'm also wondering if it's worth it to add a project property to the schema. This would allow us to associate the interview with a project and later on we can use the property to filter the list of interviews by the project they were conducted for.

Rename the audio element

The `audio` property works wonderfully for recording data about audio files.

However, it's not very flexible. It's not clear how to add a new records for in-person and video recorded interviews.

I think a better idea would be to rename to `media` and specify more generic properties for the media object. An initial version of the media object may look like this:

The object now has a `kind` property to indicate the type of media. I chose `kind` so we don't overload the meaning of `type`.

The possible values for `kind` are:

- `audio` (default) represents an audio recording
- `video` represents a video recording
- `n/a` represents the lack of a recording, possibly because it's an in-person interview, the recording was removed from the project or is not available for other reasons

We added `kind` to the list of required field for media objects.

```
"media": {
  "type": "object",
  "object""properties": {
    "kind"  "type": "strin": "      "default": "": ""default""default"[
        "audio""audio""enum"deo",
        "n/a"
      ]
    },
    "format": {
      "type": "strin"n/a""format": {
      "type""type": "string"
    }": ""url": {
      "type" "type": "number"
    ": ""duration": {
      "type"  "type": "string"
   ": ""duration_unit"": [
    "kind","kind"format",
    "url",
    "url""required": [
    "kind",
    "format",
    "url",
    "duration",
    "duration_unit"
  ]
}
```

I moved the definition of the media object under `$defs` to make it reusable in other contexts.

Adding new properties

There are a few things that I want to change to the schema or add to the schema.

The first thing to do is to change the root of the schema from `properties` to `interview`. Otherwise validators will not validate the objects against the schema which defeats the purpose.

The next item is to add a `project` attribute as a string representing the project we conducted the interview for. This will enable us to filter interviews by project.

I also added an `apiVersion` attribute to the schema to indicate the version of the schema the document conforms to

The final product

This is the current version of the `interview` schema.

```
{
  "title": "JSON schem"JSON schema for interviews""description": "JSON sch
  "apiVersion": 2,
  "$schema"s://json-schema.org/draft/2020-12/schema",
  "type": "obje": "
  "interview""interview""interview": {
    "$schema": {
      "type""id": {
      "type": "str": ",
      "description": "Unique ide": ""description""description"    "title"
      "type": "string",
      "d": ""title": "Title of the interview"
    },
"Title of the interview""description"g",
      "description": "Project that": ""description""project": {
      "type"d": {
      "type": "string",
 ": " "description": "Type of interview",
      "default":": ""description""kind": {
```

```
      "type": "string",
      "description"one-on-one",
        "interview-group""interview-group": "interview",
      "enum"    "interview-group-video",
        "interview-one-on-one-audio",
        "interview-group-audio"
      ]
    },
    "date": {
      "type": "string",
      "description": "Date of the interview",
      "format": "date"
    },
    "location": "interview-group-video""date""date"scription": "Loc": ": "
      "description""interviewers": {
      "type": "": ""interviewers""format"": "Interviewer(s) o": ""locatio
      "type": "string",
      "description"     "items": {
        "$ref": "#/$defs/": ""interviewers": {
      "type": "array",
      "description"y",
      "description": "Subject(s) of the": ""minItems": 1,
      "maxItems": 5,
      "items": {
        "$ref"
        "$ref": "#/$defs/person"
    ": ""interviewees": {
      "type": "array",
      "description": "Subject(s) of the interview",
      "minItems"   "type": "object",
": "  "properties": {
"properties": {
        "$ref"pe": "string",
          "description"": ""media": {
      "items": {
        "$ref"    "content": {
          "type": "s": ""transcript": {
      "type" "Content of the transcript file"
```

```
      "Content of the transcript file""url"  "url",
    "url""type": "string",
        "description"    "title",
      "project",
      "type",
 "title""content"      "location",
 "location"ia",
      "transcript"
    ]
"transcript""$defs": {
  "person": {
    "title": "JSON schema for a": ""$defs""$defs""$defs""required": [
      "url",
      "content"
    ],
    "required"ng",
        "description": "Name of the person"
      },
      "email": {
        "type": "string",
        "description": "Email of the per": ""$defs": {
  "person": {
    "title"        "name"
    ]
  },
  "m"name""type"   "type": "object": ""properties": "JSON schema for a m
    "desc"JSON schema for a media file and references""description""prop
      "kind": {
      "properties""email": {
        "type"ion": "Kind of media u": ""description"view.\n\nThe \"n\\a
        "default": "audio",
        "enum": [
 "n\\a\" value is used for interviews that were not recorded and where a
      "kind" },
      "url": {"url"  "type": "string",
 ": ""description"ion": "URL of the media file (where applicable)"
      },
      "duration": {
```

```
          "type": "number",
          "description": "Duration of the media file"
        },
        "duration_": ""default"      "type": "string": ""enum"   "desscrip
        }
      },
      "required": [
        "kind": ""format": {
          "type"l",
        "duration""duration""description": "Format of the media recorded
        },
        "url": {
          "type": "string",
          "description": "URL of the media file (where applicable)"
        },
        "duration": {
          "type": "number",
          "description": "Duration of the media file"
        },
        "duration_unit": {
          "type": "string",
          "desscription": "the unit for duration"
        }
      },
      "required": [
        "kind",
        "format",
        "url",
        "duration",
        "duration_unit"
      ]
    }
  }
}
```

Links and Resources

- Information Repository

        - JSON Schema
- Specifications
    - JSON Schema Core
    - JSON Schema Validation
    - Relative JSON Pointers
- Tools
    - Online JSON Schema Validator