Home-at-Home API Specification

Locations

A **location** represents a physical location, usually with an address. It could be "Home" or "Office" or even "Car." It differentiates items by location.

Schema

- `_id`: The long UUID of the location.
- `shortId`: A shorter, human-readable ID
- `name`: Name of the location
- `address`: Address of the location (optional)
- `description`: A description of the location (optional)

Paths

- `locations`
  - GET `/locations`
    - Returns all locations
  - GET `/locations/{shortId}`
    - Returns the location with the matching **short ID**
  - POST `/locations`
    - Request body:
      - `name` **required**
      - `address`
      - `description`
    - Response:
      - JSON object of the added location with generated IDs
  - DELETE `/locations/{shortId}`
    - Deletes the location with the matching **short ID**
  - PUT `/locations/{shortId}`
    - Updates the location with the matching **short ID** (all fields optional; only provided fields will be updated)
    - Request body:
      - `name`
      - `address`

        - `description`
    - Response:
        - Updated object in JSON

Spaces

A **space** represents a space in a location. As such, **a space must have a location** or it will not be created/updated. Spaces can also have "sub-spaces," which is a space within another space, like "corner of the living room" or containers that don't have a lot of intrinsic properties. A bookshelf may not need descriptions, pictures, receipts, or warranty information, but it's useful to know what's inside of it, so it could be classified as a space.

Schema

- `_id`: Long UUID of the space
- `shortId`: Human readable shorter ID
- `name`: Name of the space
- `description`: A description of the space
- `location`: The location of the space; i.e. the house the space is in
- `parent`: The **UUID** of the parent space containing this space

Paths

- GET `/spaces`
    - Gets all spaces **by location**
    - Request body:
        - location: **UUID of location** to search for spaces
    - Response:
        - 200: Array of JSON spaces belonging to the location
        - 400: Location missing
        - 404: No spaces at location
        - 500: All other errors
- GET `/spaces/short`
    - Finds one space by **short ID**
    - Query parameters:
        - `id`: Short ID
    - Example query: GET `http://localhost:4000/spaces/short?id=A7WIP`
    - Responses
        - 200: Space as JSON

    - 400: Query parameter missing
    - 404: Space not found
    - 500: All other errors
- GET `/spaces/long`
  - Find one space by **UUID**
  - Query parameters:
    - `id`: UUID
  - Responses:
    - 200: Space as JSON
    - 400: Query parameter missing
    - 404: Space not found
    - 500: All other errors
- GET `/spaces/search`
  - Find space by name
  - Query parameters:
    - `name`: Space name
  - Responses:
    - 200: Space as JSON
    - 400: Query parameter missing
    - 404: Space not found
    - 500: All other errors
- POST /spaces
  - Add a new space
  - Request body:
    - `name` **required**
    - `location` **required**
    - `parent`
    - `description`
  - Responses:
    - 201: New Space as JSON
    - 500: Server error
- PUT ==`/spaces/{id}
  - Update space by **short ID**
  - Request body (all fields optional, fields not included will not be updated):
    - name
    - description
    - location

  - parent: **short ID** of parent space. Must be valid if included.
- Responses:
  - 200: Updated Space as JSON
  - 400: Invalid location
  - 400: Invalid parent space (if parent space is in request body)
  - 500: All other errors