JSON file format for EPPI Mapper

You can find an example of the minimum requirements for the JSON file used by EPPI Mapper [here](#).

(This is an edited version of a standard JSON export from EPPI Reviewer, stripped of everything that isn't needed to create a simple map. The original version of the JSON file exported from EPPI Reviewer can be found [here](#).)
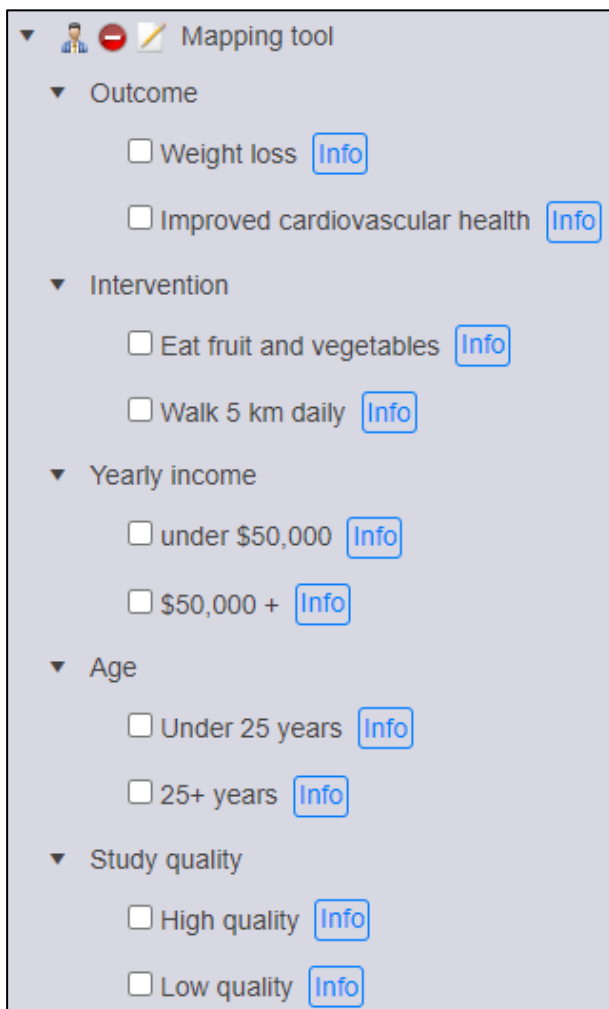
(We have endeavoured to make the files easy to read. You can easily view them in a text editor such as Notepad++, available from https://notepad-plus-plus.org/downloads/. You may also find tools such as JSON Viewer useful for looking at JSON data; see https://codebeautify.org/jsonviewer.)

Details

We have started with a basic mapping tool in EPPI-Reviewer.

For the map the columns will be the 'Outcome' section, while the rows will be 'Intervention' section.

There are two codes in both the Outcome and Intervention (i.e. there are only two levels of hierarchy in this example).

The segmenting attribute is 'Study quality' (i.e. there can be up to 2 bubbles in a cell).

The filters are 'Yearly income' and 'Age'.

We have coded 2 items using this coding tool.

Looking at the 'minimum' file, the JSON file has 2 sections.

The top is the "Codesets" section and defines the coding tool you see in the screenshot above.

The minimum information needed in this section are unique Id numbers and the code names.

```
1    {
2        "CodeSets": [
3            {
4                "SetName": "Mapping tool",
5                "Attributes": {
6                    "AttributesList": [
7                        {
8                            "AttributeId": 1,
9                            "AttributeName": "Outcome",
10                           "Attributes": {
11                               "AttributesList": [
12                                   {
13                                       "AttributeId": 10,
14                                       "AttributeName": "Weight loss"
15                                   },
16                                   {
17                                       "AttributeId": 11,
18                                       "AttributeName": "Improved cardiovascular health"
19                                   }
20                               ]
21                           }
22                       },
23                       {
24                           "AttributeId": 2,
25                           "AttributeName": "Intervention",
26                           "Attributes": {
27                               "AttributesList": [
28                                   {
29                                       "AttributeId": 20,
30                                       "AttributeName": "Eat fruit and vegetables"
31                                   },
32                                   {
33                                       "AttributeId": 21,
34                                       "AttributeName": "Walk 5 km daily"
35                                   }
36                               ]
37                           }
38                       },
39                       {
40                           "AttributeId": 3,
41                           "AttributeName": "Yearly income"
```

It should be simple to match up what you see in the file with what the coding tool looks like.

The *AttributeId* numbers can be anything you want them to be. They just need to be unique (i.e. never repeated)

Once the coding tool is defined, you will then have a "References" section - where you define each reference and what codes are applied to it.

```json
90          ],
91          "References": [
92              {
93                  "Codes": [
94                      {
95                          "AttributeId": 10,
96                          "ItemAttributeFullTextDetails": []
97                      },
98                      {
99                          "AttributeId": 11,
100                         "ItemAttributeFullTextDetails": []
101                     },
102                     {
103                         "AttributeId": 21,
104                         "ItemAttributeFullTextDetails": []
105                     },
106                     {
107                         "AttributeId": 31,
108                         "ItemAttributeFullTextDetails": []
109                     },
110                     {
111                         "AttributeId": 41,
112                         "ItemAttributeFullTextDetails": []
113                     },
114                     {
115                         "AttributeId": 51,
116                         "ItemAttributeFullTextDetails": []
117                     }
118                 ],
119                 "Outcomes": [],
120                 "ItemId": 3292125,
121                 "Title": "Lack of resources will delay developments",
122                 "ParentTitle": "Issued opinion",
```

For each reference there is a "Codes" section where the **AttributeId** values will match up with the **AttributeId** values in the "Codesets". This is how you define what codes are applied to each reference.

Below the "Codes" section is where the reference is defined.

The **ItemID** value can be anything you want it to be, but it must be unique for each reference.

There are several different fields for each reference. These fields are selectable in EPPI-Mapper when you generate the map, so what you decide to include in your JSON file is based on what you require when generating the map.

(In the example JSON file, you will see 2 references.)

The 'minimal' JSON file can be used as a template to manually generate a JSON file compatible with EPPI-Mapper (when you are not using EPPI-Reviewer to generate a JSON export).

To manually create a JSON file with more rows and columns (and references and bubbles and filters) you just need to expand the minimal file.

(The template file would need to be altered if you wanted to show 3 levels of hierarchy in your row or column "codesets". Do contact us if you have queries on this.)

This 'minimal' version of the JSON file is for creating a map with basic functionality (which seems to cover a major proportion of EPPI-Mapper maps).

If you want more functionality in the map (such as having URL links), then some of the fields that were removed from the "minimal" example JSON file will need to be put back in. Depending on what you require, we can hopefully assist you in doing this.

The example map is generated using the 'minimal' JSON file and is based on:
 - 'Outcome' columns,
 - 'Intervention' rows,
 - 'Study quality' bubbles,
 - 'Yearly income' and 'Age' filters.
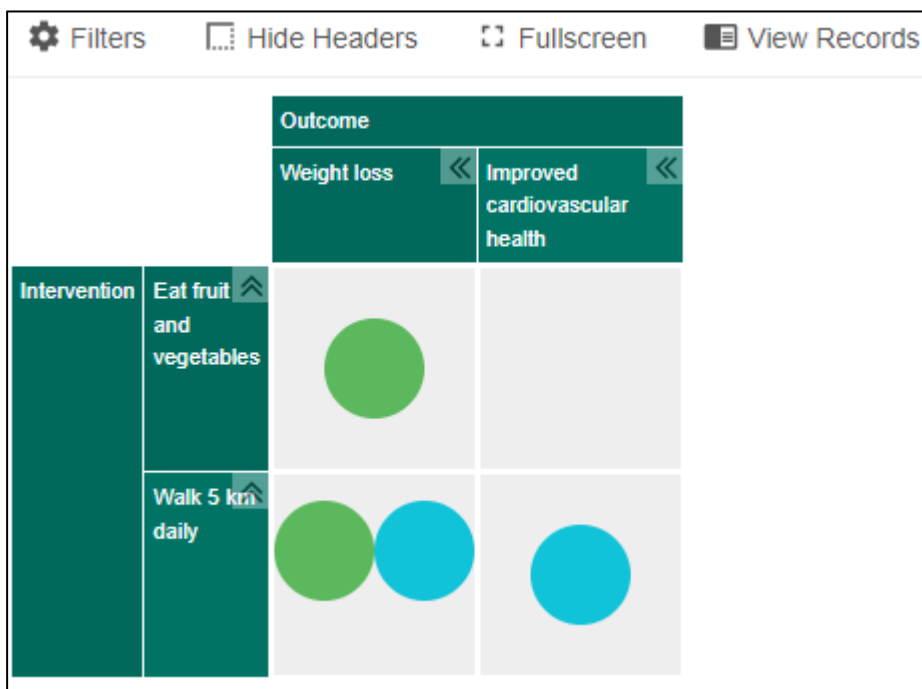
Do let us know if you have further questions on this.

eppisupport@ucl.ac.uk