

# VR File Generator

---

VR File Generator generates ...

## Understanding JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. This file generator generate JSON files that configure VR experiments, and you can also customize the different forms with JSON files.

In JSON, values must be one of the following data types:

- a string: sequence of characters in double quotes
  - e.g. `"this is a string"`
- a number: integer or decimal
  - e.g. `2`, `1.5`
- a boolean
  - `true` or `false`
- an array: a list of JSON values in square brackets, separated by commas
  - e.g. `["string value", 2.3, true, { "key": "value" }]`
- an object: a collection of key-value pairs in curly brackets, separated by commas
  - Keys must be strings. Note that an object cannot contain two identical keys.
  - Values can be any JSON value: string, number, boolean, array, object
  - e.g.

```
{
  "key_for_string": "string value",
  "key_for_number": 3,
  "key_for_array": [1, 2],
  "key_for_object": {
    "key": "string value",
    "key2": 3.5
  }
}
```

## Customize forms

This app allows users customize forms (configuratoin file form and object form) with JSON files. The file should be a JSON array containing form entry definitions. Each definition configures an entry in the form.

Note: Before using this feature, you need to make sure you can describe the desired outcome: What should the form look like? What should the output JSON file look like? You should be able to hand write the output JSON files.

### Form entry definition

Form entry definitions are JSON objects. For each definition, three key-value pairs are required:

- **"label"**: This is the label for this entry at display. The value must be a string. Note that this does not affect the output of this entry in the resulting JSON file.
- **"key"**: This is the key for this entry in the resulting JSON file. The value must be a string.
- **"type"**: This refers to the input type of this entry. This file generator supports five different types: **"text"**, **"number"**, **"switch"** (true or false), **"selection"** (a dropdown menu), **"list"** (a list of texts or numbers).
- Depending on the input type, different key-value pairs are required, which is discussed in details below. \* (Optional) means that the key-value pair is optional.

For example:

```
{
  "key": "subjNum",
  "label": "Subject Number",
  "type": "number"
}
```

This definition produces:



And if I type in **8**, the output file will look like this:

```
{
  "subjNum": 8,
  ..., // other entries
}
```

## Types

### **"text"**

- (Optional) **"defaultValue"**: Default value for this entry. This must be a string.
- (Optional) **"addonAfter"**: An add-on at the end of the input. This must be a String. The output of this entry is the concatenation of user input and the **"addonAfter"** value.

For example:

```
{
  "key": "trialFile",
  "label": "Trial file",
  "type": "text",
  "defaultValue": "config",
  "addonAfter": ".json"
}
```



In the output:

```
{
  "trialFile": "config.json",
  ...
}
```

### "number"

- (Optional) "defaultValue": Default value for this entry. This must be a number.

For example:

```
{
  "key": "subjNum",
  "label": "Subject Number",
  "type": "number",
  "defaultValue": 8
}
```



In the output:

```
{
  "subjNum": 8,
  ...
}
```

### "switch"

- (Optional) "defaultValue": Default value for this entry. This must be a boolean, so either **true** or **false**.

For example:

```
{
  "key": "collectConfidence",
```

```

    "label": "Collect Condifence Rating",
    "type": "switch",
    "defaultValue": false
  }

```



In the output:

```

{
  "collectConfidence": false,
  ..., // other entries
}

```

### "select"

- **"options"**: Defines the options in the dropdown menu. This must be an object (collection of key/value pairs). The keys are values that appear in the output JSON file, and the values are the keys' labels for display. For instance: { "male": "Male", "female": "Female" }.
- (Optional) **"defaultValue"**: Default value for this entry. This must be one of the keys in **"options"**. For instance: "male".

For example:

```

{
  "key": "subjSex",
  "label": "Subject Sex",
  "type": "select",
  "defaultValue": "male",
  "options": {
    "male": "Male",
    "female": "Female",
    "non_binary": "Non binary"
  }
}

```



In the output:

```

{
  "subjSex": "male",
  ...
}

```

## "list"

- **"itemType"**: The type of elements in list. Either **"number"** or **"text"**.
- **"listSize"**: The number of elements in list. Should be smaller than 5.
- (Optional) **"defaultValue"**: Default value for this entry. This must be a list containing **"listSize"** elements of type **"itemType"**. If **"listSize"** is 2 and **"itemType"** is **"text"**, then **"defaultValue"** could be **["value1", "value2"]**.

For example:

```
{
  "key": "coordinates",
  "label": "Coordinates (X, Y, Z)",
  "type": "list",
  "itemType": "number",
  "listSize": 3,
  "defaultValue": [
    1,
    2,
    3
  ]
}
```



In the output:

```
{
  "coordinates": [1, 2, 3],
  ...
}
```

## Import form setting

When you have a JSON file that conforms to the rules above, you can import this setting by navigating to the 'Form Settings' page by clicking on the 'Import form settings' button on the home page.




The app will parse the imported file and report debugging errors, if there are any.



If the imported file contains no error, you can click on the 'Save settings' button and save the settings.



The change will be reflected in the form:  Import