GeoTag-X

Project Creation Guide

https://github.com/geotagx/geotagx-project-template

I. Introduction

As you may already know, creating a project requires writing a *task presenter* and or a *tutorial*, in HTML, CSS and JavaScript. Depending on the size and scope of the project, this can be very time consuming or even nigh impossible for someone with limited or no knowledge of these languages.

The purpose of the builder tool is to speed up the creation process by ingesting well-defined configurations of your project and generating the necessary HTML, CSS, and JavaScript code that runs it. Ultimately, this will let project creators focus more on the content of the project without having to worry too much about the technical details.

This guide will walk you through creating the configurations required by the builder tool, namely

- the project configuration which will generate a *task presenter* for your project.
- an optional, albeit recommended, tutorial configuration which will generate a *tutorial* to complement your project.

Both of the aforementioned configurations can be written in either JSON or YAML formats but to keep things consistent, the examples in this guide adhere to the JSON format. As a matter of fact, the examples used in this guide are taken from the sample project's JSON configurations.

II. The project configuration

`filename: project.json/project.yaml`

The project configuration -- based on the classic PyBossa configuration -- has been extended to include a few requirements to help create a project's task presenter. In addition to the required **name**, **short_name** and **description** fields, you will need to add the following to your project configuration:

- **why**: a short description reminding volunteers about the importance of their contribution to the project.
- **questionnaire**: a list of one or more questions asked to volunteers.

II.1. Questionnaires

The project's questionnaire is a list of entries where each is comprised of

- a unique **key*** to identify the entry
- the **type*** of question to ask
- the actual **question*** to ask
- a short **hint** about the question
- a set of **parameters** that configure the entry's behavior
- a **branch** field to determine the questionnaire's control flow

Please note that **key**, **type** and **question** are mandatory fields, while the rest are optional.

II.1.a. Keys

A key is a non-empty string that identifies each questionnaire entry. It is composed of alphanumeric characters, dashes, and/or underscores; no whitespace.

A key is used in the following ways:

- The flow in a questionnaire is not necessarily linear. It is possible to jump from question to question based on specific criteria. To jump to a specific question, you will need to know its key.
- A help file can be created to elaborate upon a question. Its filename must be identical to the key for the the question you wish to explain.
- The result to each question is mapped to its key.

Certain keywords are reserved for internal use by the project builder and can not be used as keys. The builder tool will promptly notify you when a reserved keyword is used as a key.

II.1.b. Help

To help volunteers answer questions as accurately as possible, you can either provide a short hint or a more elaborate explanation for questions that may be open to interpretation.

A hint is usually a concise way to quickly dispel ambiguity. For instance, let's take a look at the first questionnaire entry in the sample project

```
{
    "key":"isWaterVisible",
    "type":"binary",
    "question":"Can you see any water in the photo?",
    "hint":"More specifically, can you see small water bodies (...) as rivers or lakes?",
    "branch":{
        "yes":"waterColor",
        "no":"end"
    }
}
```

This entry will generate the following

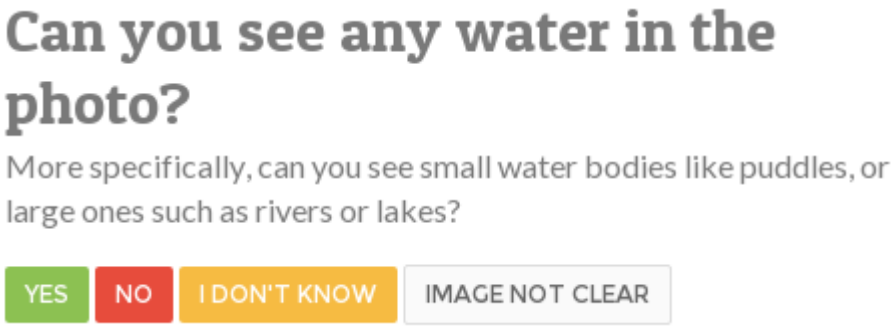

FIGURE 1 - Question hints

Some questions may require more than a hint to get the point across in which case you will need to create an HTML file in the project's *help* directory. The help file's filename must match the key to the question you would like to explain so for instance, a question with the key `waterColor` will look for the help file named `waterColor.html` in the help directory.

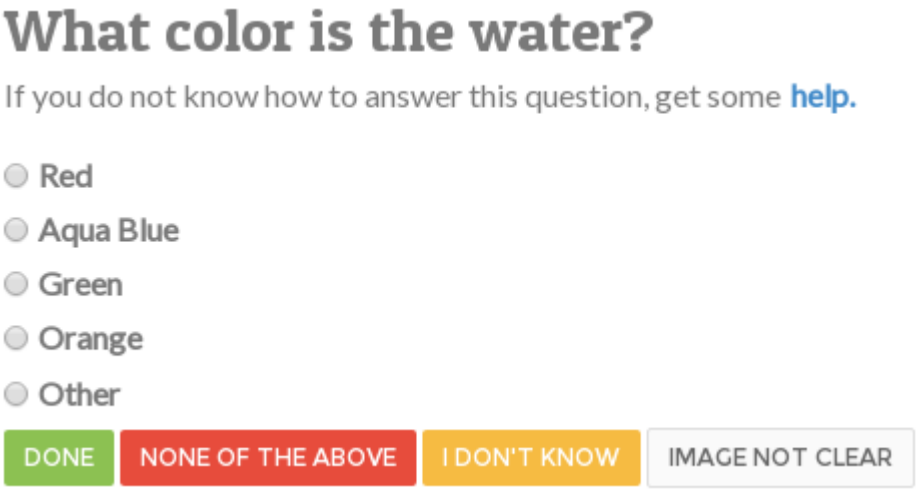

FIGURE 2 - Question help

Wait, where did the help go? Because the help can be as detailed as you would like it to be, it does not make sense to clutter the page with too much information, so it is strategically placed in a hidden modal window that is only displayed when requested.

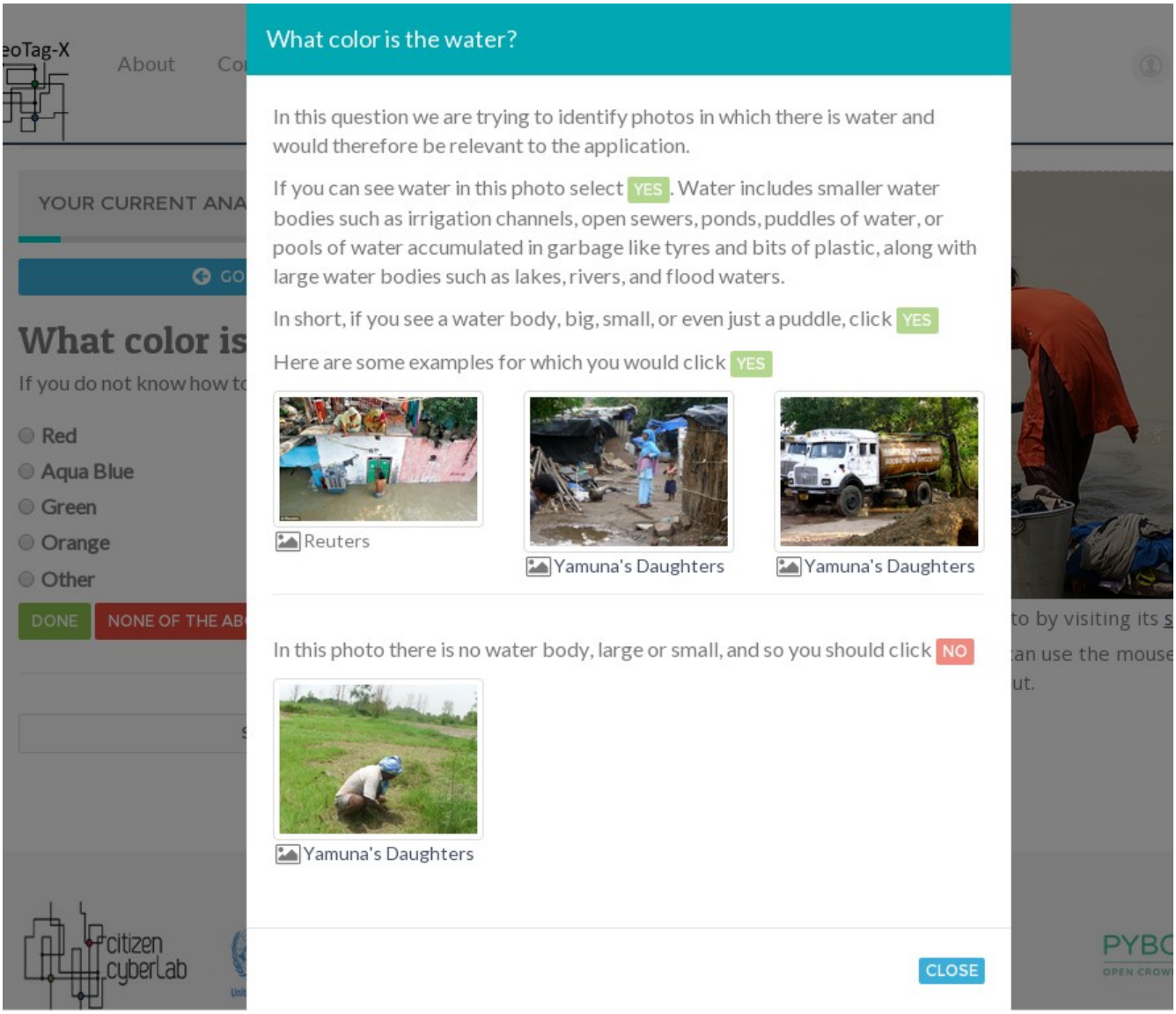

FIGURE 3 - Help modal window

II.1.c. Input types and parameters

Binary input `type:binary`

This is the input to the most basic type of question you could ask: a yes-no question with an expected answer that is either **Yes** or **No**.

Dropdown lists `type:dropdown-list`

Dropdown lists allow you to pick a single value from many available choices.

Parameter	Description
prompt (string)	This is always the first unselectable option in the list and is usually a message that invites a user to interact with the list. By default, this value is set to "**Please select an option**".
size (number)	Specifies the minimum number of options that are simultaneously presented. If omitted, this parameter is set to **1**.

options (array)	Specifies the list's available options. Each option is a ⟨*label, value*⟩ pair where the *label* is text indicating the meaning of the option, and the *value* indicates the value that is submitted when the option is selected.
	
	Example:
	
	`"options":[`
	`  {`
	`  "label":"Red",`
	`  "value":"R"`
	`  },`
	`  {`
	`  "label":"Green",`
	`  "value":"G"`
	`  },`
	`  {`
	`  "label":"Blue",`
	`  "value":"B"`
	`  }`
	`]`

Single-choice input `type:select`

Just like dropdown-lists, single-choice inputs allow you to select a single value from many. The only difference between the two is that single-choice inputs present more than one option by default.

Parameter	Description
size (number)	Specifies the maximum number of options that are simultaneously presented. If unspecified, this parameter is set to **8**.
options (array)	Specifies the input's available options. Each option is a ⟨*label, value*⟩ pair where the *label* is text indicating the meaning of the option, and the *value* indicates the value that is submitted when the option is selected.

Check lists `type:checklist`

Checklists allow you to pick more than one option from many.

Parameter	Description
size (number)	Specifies the maximum number of options that are simultaneously presented. If unspecified, this parameter is set to **8**.
options (array)	Specifies the checklist's available options. Each option is a ⟨*label, value*⟩ pair where the *label* is text indicating the meaning of the option, and the *value* indicates the value that is submitted when the option is selected.

Illustrative check lists `type:illustrative-checklist`

Illustrative checklists provide an extra visual aid to help select an option

Parameter	Description

options (array)	Specifies the checklist's available options. Each option is a ⟨*label, value, illustration, more examples*⟩ quadruplet where the *label* is text indicating the meaning of the option, and the value indicates the *value* that is submitted when the option is selected.
	
	The *illustration* is a ⟨*image, source*⟩ pair where the *image* is a direct link to the image used in the illustration, and *source* is a nested ⟨*name, URL*⟩ pair that contains the *name* or label of the source and a *URL* link to the web page.
	
	Specify *more examples* by linking to a page that provides more examples of the item you wish to illustrate. For instance, this could be a link to a Google search query like https://www.google.com/#q=tomato+plant.
	
	Example (please note that URLs have been truncated):
	
	`"options":[`
	`{`
	`"label":"Cattle",`
	`"value":"Cattle",`
	`"illustration":`
	`{`
	`"image":"http://www.interservicos.com/(...)/3n.png",`
	`"source":`
	`{`
	`"name":"Interservicos",`
	`"url":"http://www.interservicos.com/(...)/detail/306/"`
	`}`
	`},`
	`"more_examples":"https://www.google.com/#q=cattle"`
	`},`
	`{`
	`"label":"Camel",`
	`"value":"Camel",`
	`"illustration":`
	`{`
	`"image":"http://www.animalcorner.co.uk/(...)/cameldrom.jpg",`
	`"source":`
	`{`
	`"name":"Animal Corner",`
	`"url":"http://www.animalcorner(...)/camel_dromedary.html"`
	`}`
	`},`
	`"more_examples":"https://www.google.com/#q=camel"`
	`}`
	`]`

Short text input `type:text`

This is the input type to use for questions that require short (single-line) textual input.

Parameter	Description
placeholder (string)	Specifies a hint to the user of what kind of text is expected.
maxlength (number)	Specifies the maximum number of characters that can be entered into the input. By default, this parameter is set to **128** characters.

Long text input `type:longtext`

This is the input type to use for questions that require long (multi-line) textual input.

Parameter	Description
placeholder (string)	Specifies a hint to the user of what kind of text is expected.
maxlength (number)	Specifies the maximum number of characters that can be entered into the input. The default value is set to **512** characters.

Number input `type:number`

This is the input type to use for questions that strictly require numeric input.

Parameter	Description
placeholder (string)	Specifies a hint to the user of what kind of numeric value is expected.
min (number)	Specifies the smallest possible value that is accepted by the input.
max (number)	Specifies the largest possible value that is accepted by the input.
maxlength (number)	Specifies the maximum number of digits that can be entered into the input. By default, this parameter is set to **256** digits.

Date and time input `type:datetime`

Datetime inputs will present a calendar and clock from which a date and time can be selected.

Parameter	Description
mindate (string)	Specifies the earliest date that is accepted by the input. The date is specified in the YYYY-MM-DD format, .e.g. 2014-09-16.
maxdate (string)	Specifies the latest date that is accepted by the input. Like the *mindate* parameter, the value is written in the YYYY-MM-DD format.
mintime (string)	Specifies the earliest time that is accepted by the input. The time is specified in the HH:MM:SS format, e.g. 18:30:00.
maxtime (string)	Specifies the latest time that is accepted by the input. Like the *mintime* parameter, the value is written in the HH:MM:SS format.

Date input `type:date`

This is a constrained version of the datetime input that only displays a calendar from which a date can be selected. It is useful in cases where obtaining a time value is not a requirement.

Parameter	Description
min (string)	Specifies the earliest date that is accepted by the input. The date is specified in the YYYY-MM-DD format, .e.g. 2014-09-16.
max (string)	Specifies the latest date that is accepted by the input. Like the *min* parameter, the value is written in the YYYY-MM-DD format.

URL input `type:url`

This is the input type to use for questions that require URL input. While visually similar to the text input, URL inputs perform extra data validation to make sure the input is either an empty string or an absolute URL.

Parameter	Description
placeholder (string)	Specifies a hint to the user of what kind of text is expected. If unspecified, this value is set to "**Please enter a URL e.g. http://www.example.com**".
maxlength (number)	Specifies the maximum number of characters that can be entered into the input. The default value is set to **2000**[1] characters.

1 This is the recommended maximum length of a URL accepted by most web browsers. For more information, please reference http://stackoverflow.com/a/417184.

Geotagging input `type:geotagging`

This will present a map that can be used for geolocalization.

Parameter	Description
location (string)	Specifies the map's initial location. If the location can not be found, then the map is centered at 0°N 0°W.

II.1.d. Control flow

By default, the questionnaire has a linear flow which means that questions are asked in the order they are defined. In some cases, it is interesting to ask a specific subset of questions based on how a volunteer progresses through your questionnaire. The simplest example I can give is the first question asked in the sample project

```
{
    "key":"isWaterVisible",
    "type":"binary",
    "question":"Can you see any water in the photo?",
    "hint":"More specifically, can you see small water bodies (...) rivers or lakes?",
    "branch":{
        "yes":"waterColor",
        "no":"end"
    }
}
```

The purpose of this yes-no question is to determine whether or not an image is relevant to the analysis. If it is, i.e. the answer to the question is `yes`, then the question with the key `waterColor` is displayed but if it is not, the questionnaire is promptly ended and a submission form is displayed.

Please note that `end` is not a question key but a reserved keyword used to explicitly end a questionnaire.

With the `branch` mechanism, you are able to dictate the questionnaire's flow. It comes in two flavors: **conditional branching** where the progress is contingent upon the answer to a question, and **strict branching** where the progress is absolute. The above JSON snippet is an example of conditional while the following is one of strict branching

```
{
    "key":"smallPoolsOfWaterNearShelters",
    "type":"binary",
    "question":"Are these small pools of water near human shelters or a settlement?",
    "branch":"sourcesOfDrinkingWaterVisible"
}
```

The branching mechanism in this example stipulates that whatever the outcome to this question, it is strictly followed by the question with the key `sourcesOfDrinkingWaterVisible`.

III. The tutorial configuration

`tutorial.json/tutorial.yaml`

Project tutorials are a great way of introducing volunteers to your project and while optional, it is highly recommended that you included at least one in your project.

The tutorial configuration is comprised of one or more tutorials where each one contains:

- an **image** to analyze during the tutorial.
- a link to the web page where the aforementioned image was found, its **source** so to speak. This page usually provides contextual information about the image that may prove to be invaluable to volunteers.
- a set of **assertions** about the image.