CyberTracker XlsForm Reference Manual

Overview

CyberTracker is a universal data collection application which runs on mobile devices. One of the supported formats is XlsForm.

XlsForm has an extension mechanism which allows users to activate CyberTracker behavior without affecting the semantics of the form.

This reference manual descibes the CyberTracker extensions. Note that XlsForms are simply an Excel files and the extensions are columns in the worksheets.

There are three kinds of customization:

- Views, e.g. grid styles for single and multi-select lists
- Behaviors, e.g. GPS track logs and save targets
- Developer code, e.g. a new widget

Backend

CyberTracker supports ODK Central, KoBoToolbox and Survey123.

The CyberTracker extensions do not affect the semantics of the form and are transparent to backends. It is possible to use the same form to collect data across platforms (web, ODKCollect, etc) with a single form. In this scenario, CyberTracker would be chosen as a way to meet the needs of specific field workers.

Limitations

While CyberTracker supports most of the commonly used XlsForm features, it is not as mature as the existing data collection tools like ODK Collect, Kobo Collect and Survey123. Users should prefer to use those tools for mission critical projects.

TABLE OF CONTENTS

Initial setup

The following columns on the `settings` sheet are needed to begin using the CyberTracker extensions.

namespace (required)

The `ct` namespace tells other XlsForm tools to ignore columns starting with `bind::ct:`.

title	version	namespaces
My Form	2022101001	ct="http://cybertracker.org/xforms"
◀ ▶ survey	choices **settings**	○

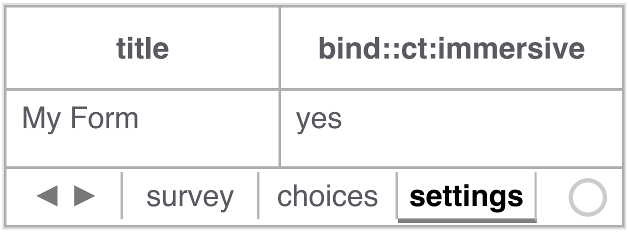
version (recommended)

The version field is used to track form versions over time. While not strictly required, it is a best practice to keep this field up to date. The XlsForm specification recommends the convention of ‘yyyymmddrr’. For example, 2022021501 is the 1st revision from Feb 15th, 2022.

Settings

immersive

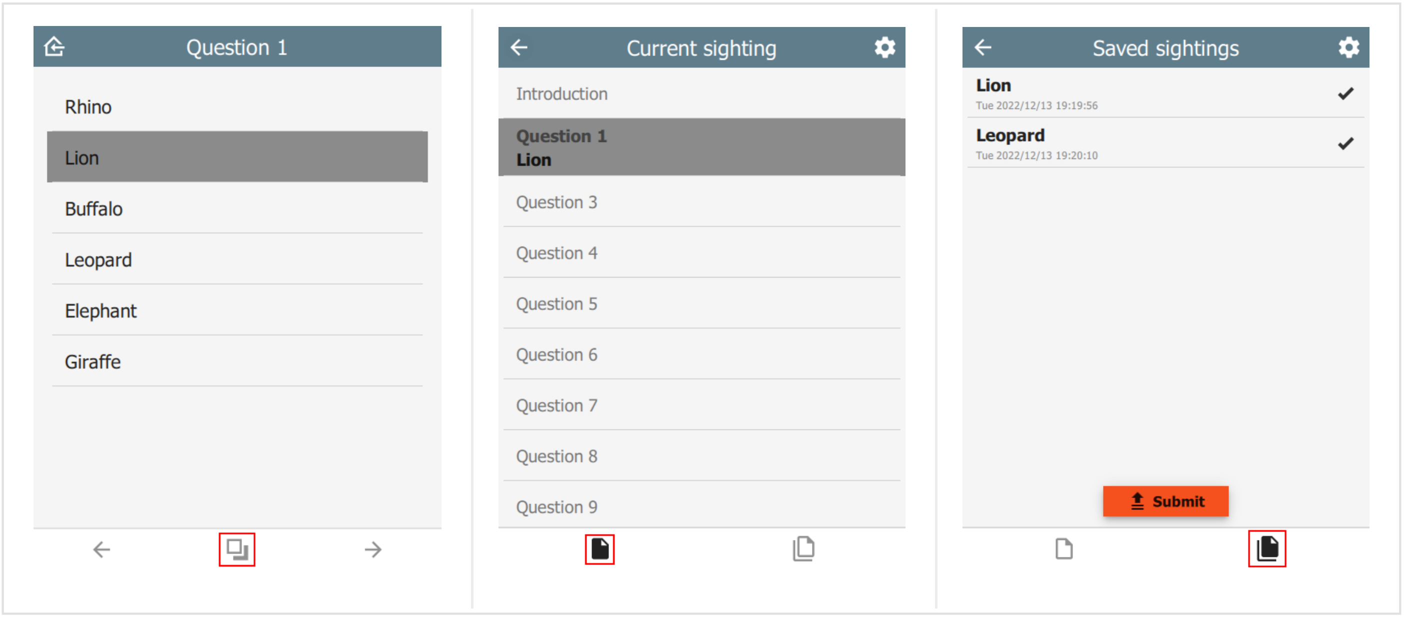
Setting this to `yes` causes the UI to use the wizard exclusively, i.e. there is no **Home** page. Default is `no`.



title	bind::ct:immersive
My Form	yes

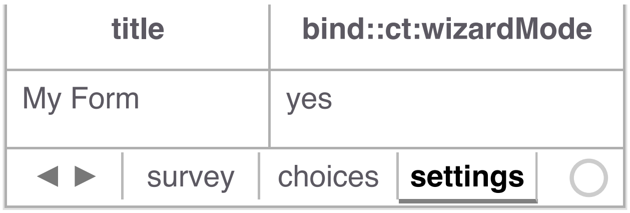
◀ ▶
survey
choices
settings

In the table below, the user context is always within a sighting and each page typically holds one question. Pressing the **options** button (highlighted in the first image) navigates to a new page which shows the current sighting on one tab and all sightings on the other. The user can edit previous sightings, but when editing is complete, the wizard will revert to the original sighting.



wizardMode

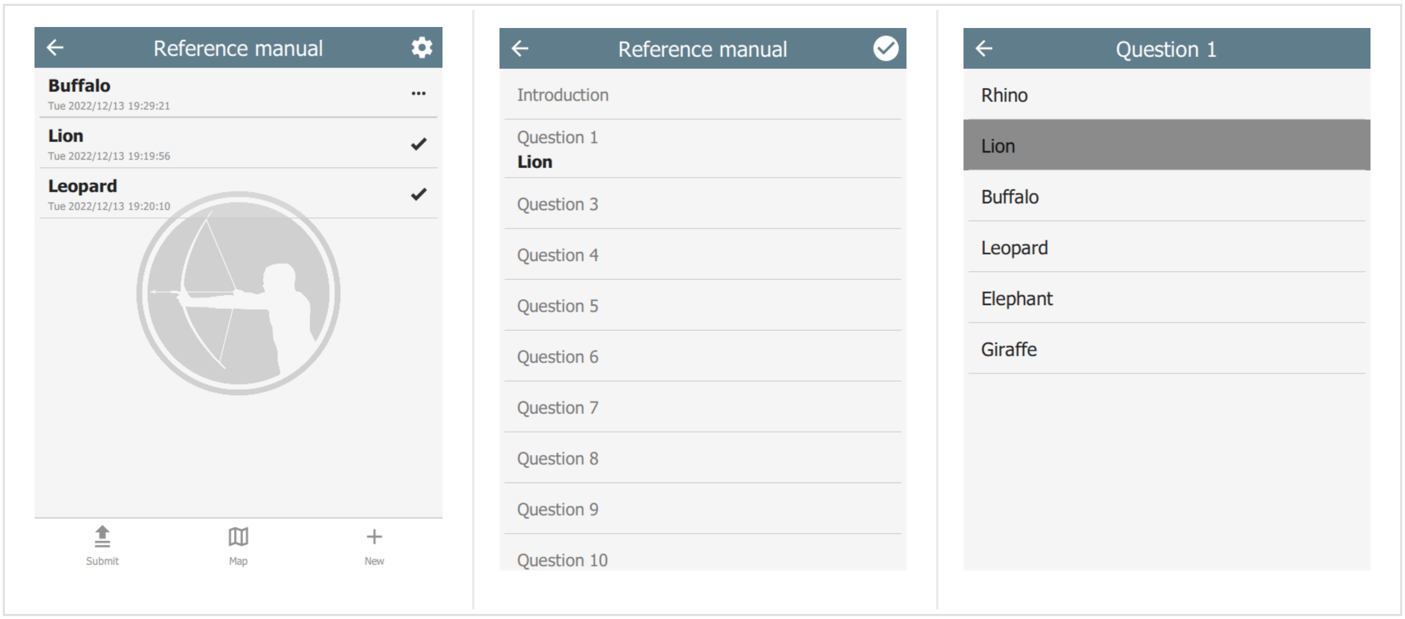
If the `immersive` column is missing or set to `no`, then the UI reverts to *non-immersive* mode. In this case, there is a **Home** page which shows all sightings. The user returns to this page after saving a sighting.



title	bind::ct:wizardMode
My Form	yes

◀ ▶
survey
choices
settings

If `wizardMode` is set to `no`, then all questions show on a single page. This mode is recommended when wanting to show all sighting data at once and is the most conventional.



Reference manual

Buffalo Tue 2022/12/13 19:29:21	...
Lion Tue 2022/12/13 19:19:56	✓
Leopard Tue 2022/12/13 19:20:10	✓

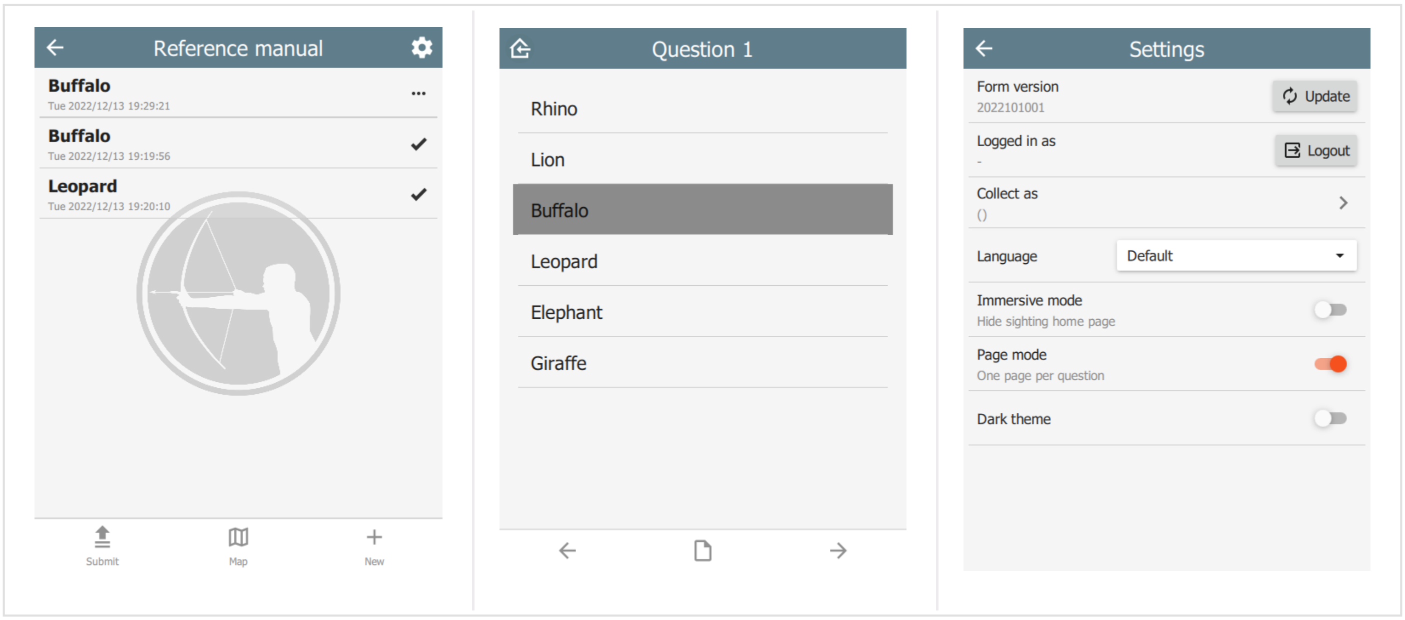
Reference manual

Introduction
Question 1
Lion
Question 3
Question 4
Question 5
Question 6
Question 7
Question 8
Question 9
Question 10

Question 1

Rhino
Lion
Buffalo
Leopard
Elephant
Giraffe

If `wizardMode` is set to `yes`, then each question will appear on its own page with **Back** and **Next** toolbar buttons to navigate between questions. The user will still return to the **Home** page between sightings. wizardMode appears on the **Settings** page as **Page mode**.



Reference manual

Buffalo Tue 2022/12/13 19:29:21	...
Buffalo Tue 2022/12/13 19:19:56	✓
Leopard Tue 2022/12/13 19:20:10	✓

Question 1

Rhino
Lion
Buffalo
Leopard
Elephant
Giraffe

Settings

Form version 2022101001	Update
Logged in as -	Logout
Collect as ()	>
Language	Default
Immersive mode Hide sighting home page	off
Page mode One page per question	on
Dark theme	off

summary

The summary attribute specifies which fields to use as the summary of a sighting on the **Home** page. For example:

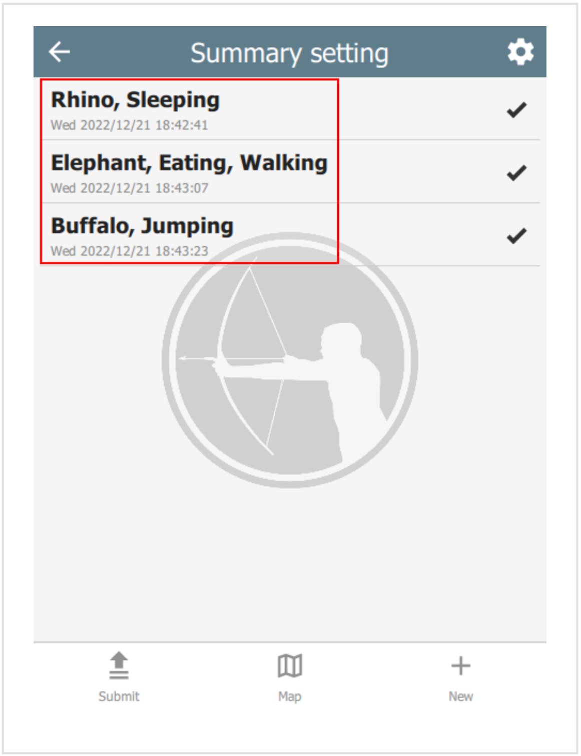
type	name	label
text	f_initial_text	Initial note
select_one animal	f_animal	Animal
select_multiple behavior	f_behavior	Behavior
text	f_final_text	Final note

◀ ▶ survey choices **settings** ○

The following setting will ensure that the summary only uses the `f_animal` and `f_behavior` questions.

title	bind::ct:summary
My Form	f_animal f_behavior

◀ ▶ survey choices **settings** ○



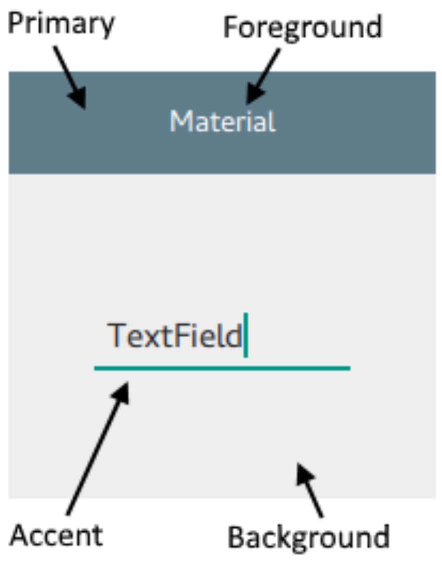
Rhino, Sleeping ✓
Wed 2022/12/21 18:42:41

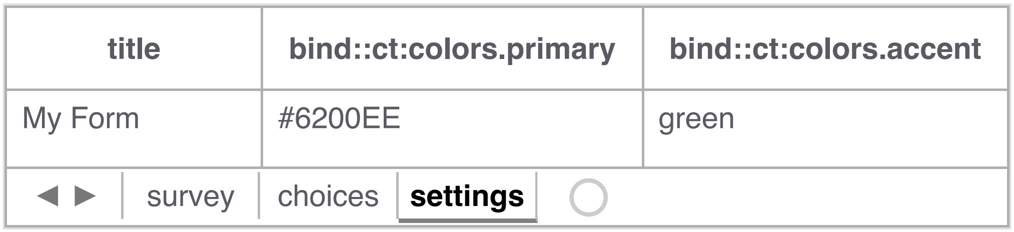
Elephant, Eating, Walking ✓
Wed 2022/12/21 18:43:07

Buffalo, Jumping ✓
Wed 2022/12/21 18:43:23

colors

The colors attribute specifies the color scheme used on the form. It follows the Material Design system:





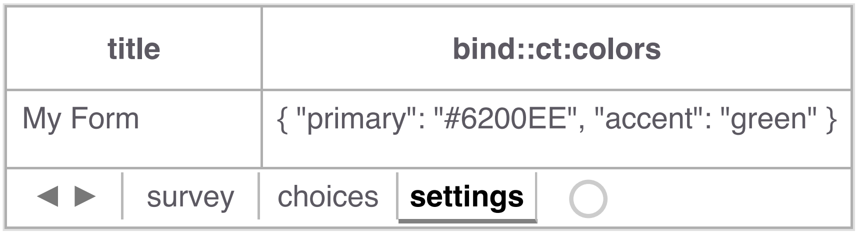
title	bind::ct:colors.primary	bind::ct:colors.accent
My Form	#6200EE	green

The following color fields are supported:

- primary & primaryDark
- accent & accentDark
- foreground & foregroundDark
- background & backgroundDark

Colors suffixed with ‘dark’ will be used when dark mode is activated. If dark colors are not provided, then ‘primary’ and ‘accent’ colors will be used, but ‘foreground’ and ‘background’ colors will be ignored.

Colors can also be provided as a JSON object:

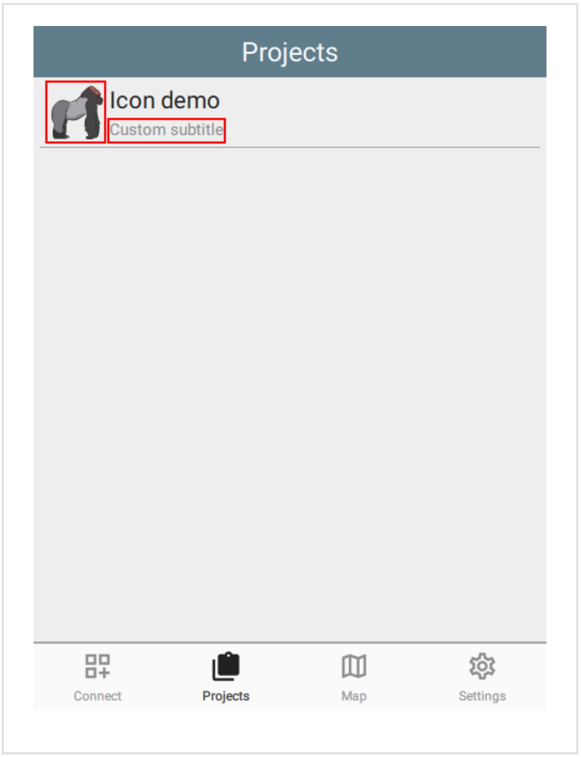


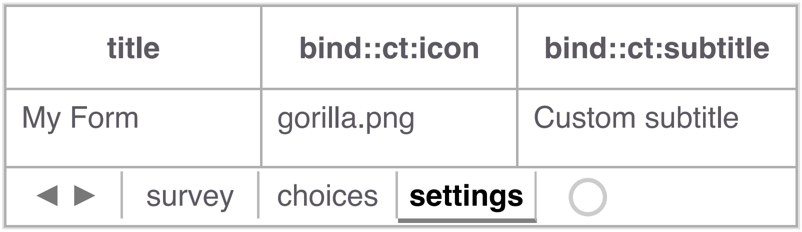
title	bind::ct:colors
My Form	{ "primary": "#6200EE", "accent": "green" }

icon, iconDark and subtitle

The `icon` attribute specifies the icon used to display the project. `iconDark` is optional and will used when dark mode is enabled.

The `subtitle` attribute specifies the text just below the form name.

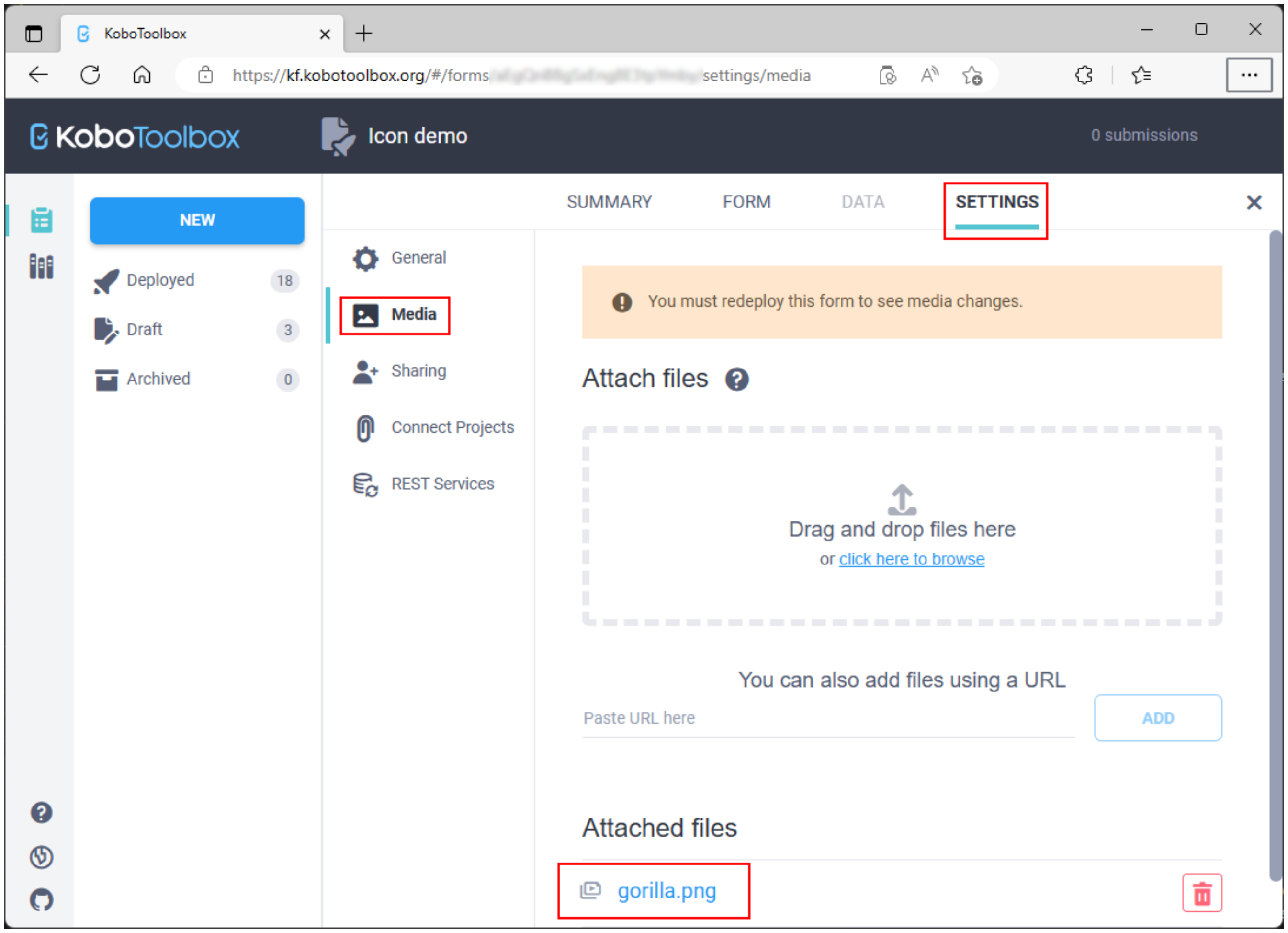




title	bind::ct:icon	bind::ct:subtitle
My Form	gorilla.png	Custom subtitle

survey
choices
settings

The icon image should be added to the form as an attached media file. In KoBoToolbox, this is done under the form settings option:



You must redeploy this form to see media changes.

You can also add files using a URL

offlineMapUrl

The `offlineMapUrl` attribute specifies a url to a downloadable zip file containing map layers. Offline maps can be added manually using the mobile app, but this provides a way to specify them with the form. The map will be downloaded, installed and updated as part of the form.

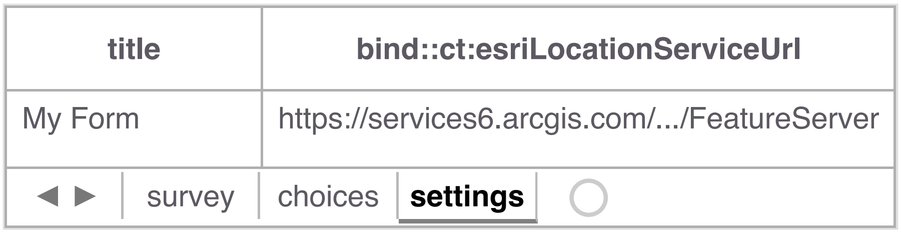
title	bind::ct:offlineMapUrl
My Form	https://cybertrackerwiki.org/assets/xlsform/offlinemap.zip

◀ ▶ survey choices **settings**

See the section on Offline maps for more information.

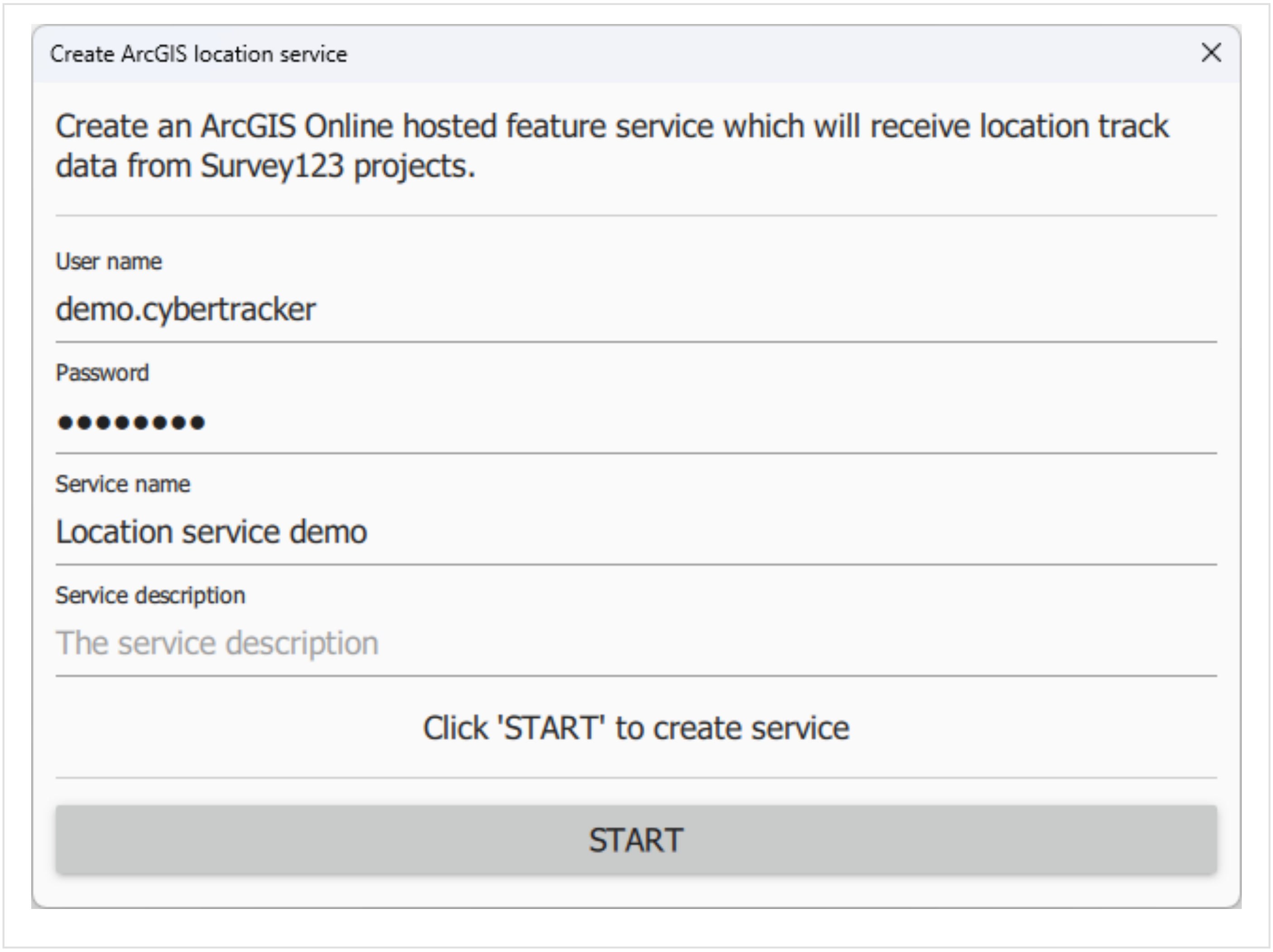
esriLocationServiceUrl

When using Survey123, CyberTracker supports uploading locations and tracks to a hosted feature service. In this case, the feature service is specified in the `bind::ct:esriLocationServiceUrl` column:



title	bind::ct:esriLocationServiceUrl
My Form	https://services6.arcgis.com/.../FeatureServer

The feature service should be created using the CyberTracker Desktop Simulator (see Download page). There is an option off the **Tools** menu called **Create ArcGIS location service**. This tool will automatically create and configure a hosted service which is compatible with CyberTracker:



Create ArcGIS location service

Create an ArcGIS Online hosted feature service which will receive location track data from Survey123 projects.

User name
demo.cybertracker

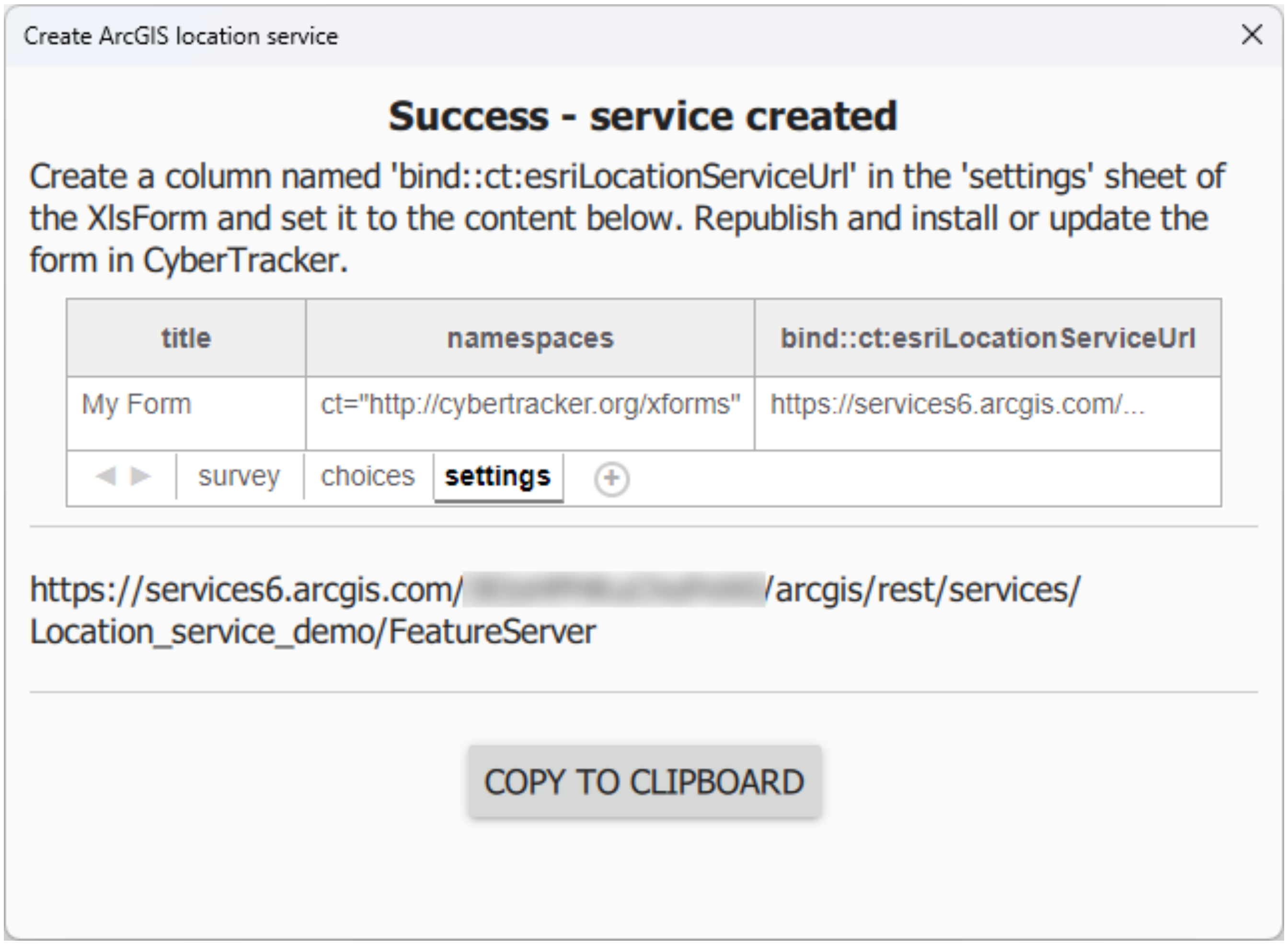
Password
●●●●●●●●

Service name
Location service demo

Service description
The service description

Click 'START' to create service

After clicking **Start**, the tool will display the following:


Create ArcGIS location service

Success - service created

Create a column named 'bind::ct:esriLocationServiceUrl' in the 'settings' sheet of the XlsForm and set it to the content below. Republish and install or update the form in CyberTracker.

title	namespaces	bind::ct:esriLocationServiceUrl
My Form	ct="http://cybertracker.org/xforms"	https://services6.arcgis.com/...

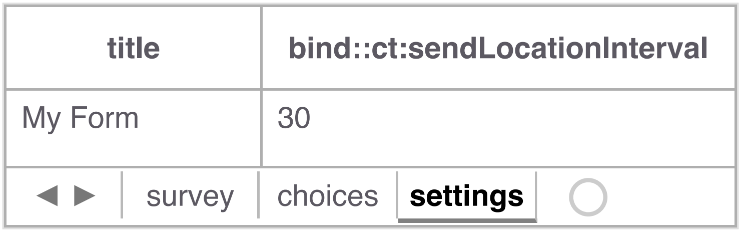
https://services6.arcgis.com/[redacted]/arcgis/rest/services/Location_service_demo/FeatureServer

The feature service contains three layers: **Tracks** (point layer), **Last Known Locations** (point layer) and **Track Lines** (Polyline layer).

If this service is not specified, then tracks are placed in a `file` type question of the sighting. See Tracks.

sendLocationInterval

If using Survey123, CyberTracker can send the current location at regular intervals - separately from tracks. The value is in seconds and is user configurable via the form Settings menu on the device. This specifies the default value. `bind::ct:esriLocationServiceUrl` must be configured.



title	bind::ct:sendLocationInterval
My Form	30

survey
choices
settings

Header

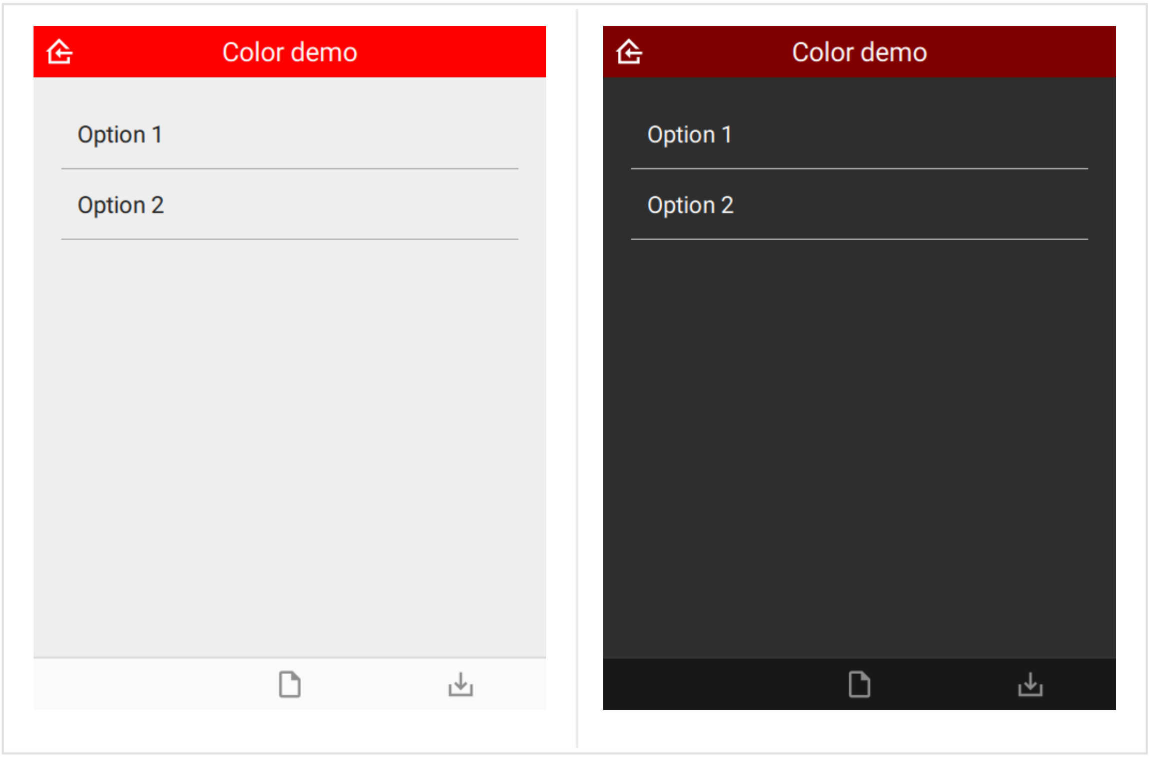
The header object supports custom header attributes. If no header object is specified, then the default header is used. By default, the header title is taken from the question label.

color and colorDark

By default, the header background color is taken from the `settings` sheet. However, it is possible to override it on an individual page.

type	bind::ct:header.color	bind::ct:header.colorDark
select_one...	#ff0000	#800000

◀ ▶ **survey** | choices | settings

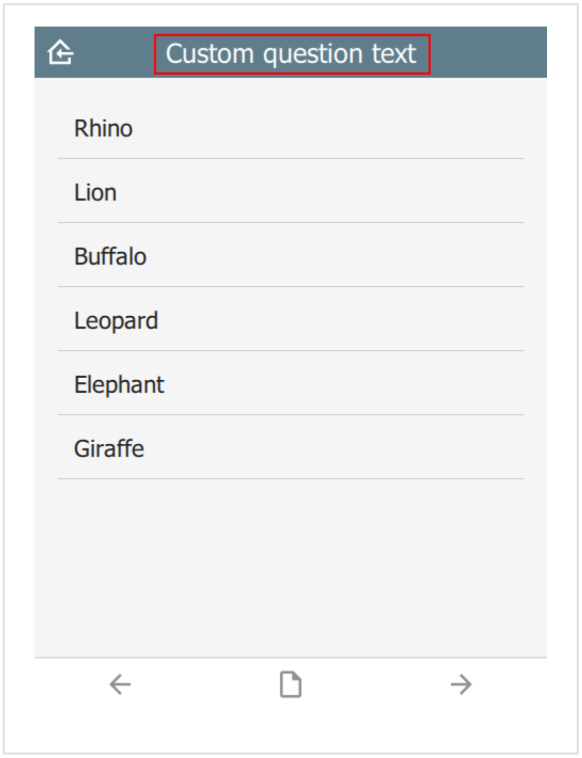


text

Custom header text.

type	name	bind::ct:header.text
select_one animal	Animal	Custom question text

◀ ▶ **survey** choices settings

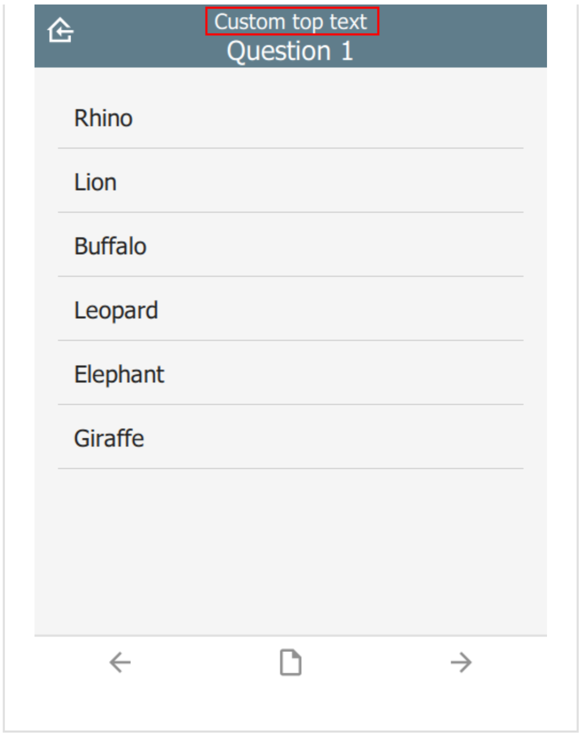


topText

Custom smaller text above main title.

type	name	bind::ct:header.topText
select_one animal	animal	Custom top text

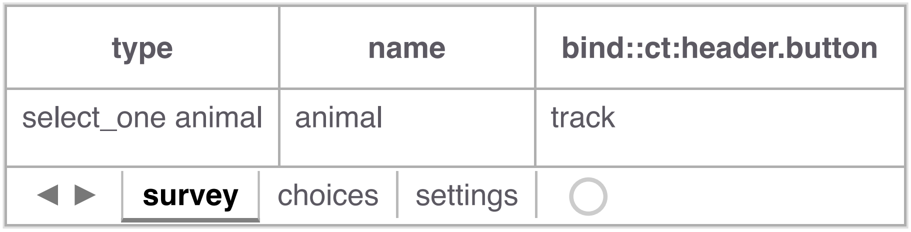
◀ ▶ **survey** choices settings



button

Type of the button in the top-right corner. Valid values are:

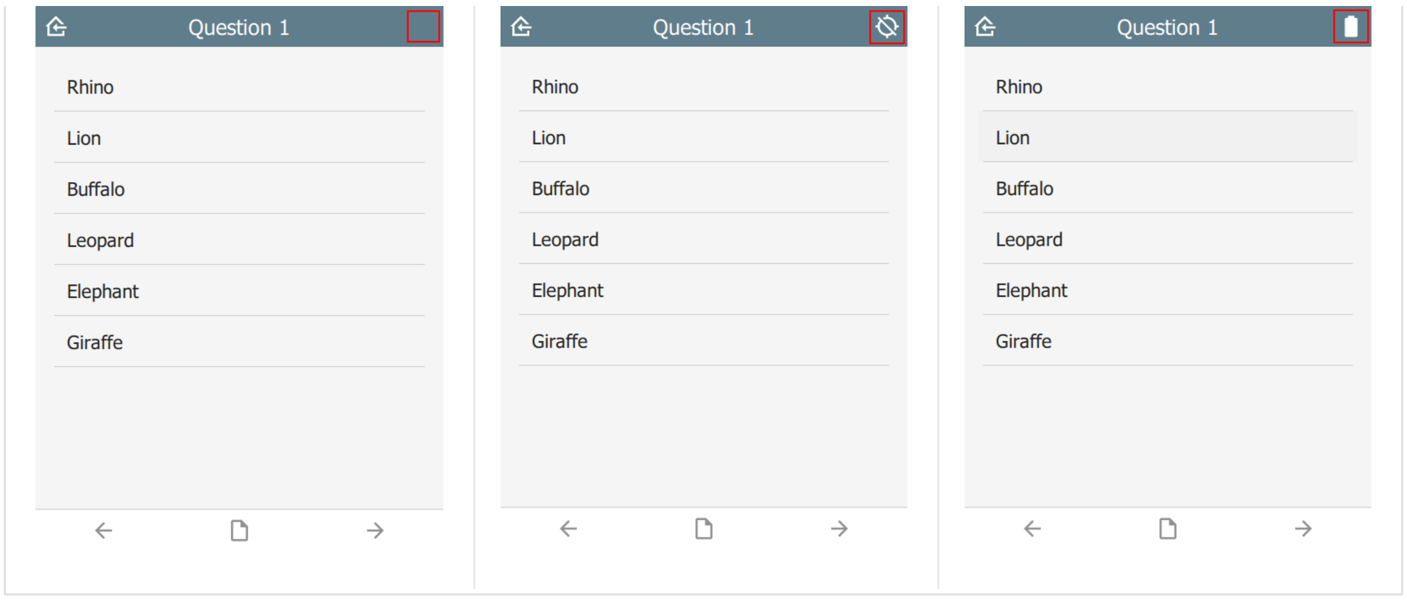
- empty - by default no button is shown
- `track` - the current state of the GPS track system
- `battery` - the current state and level of the battery



type	name	bind::ct:header.button
select_one animal	animal	track

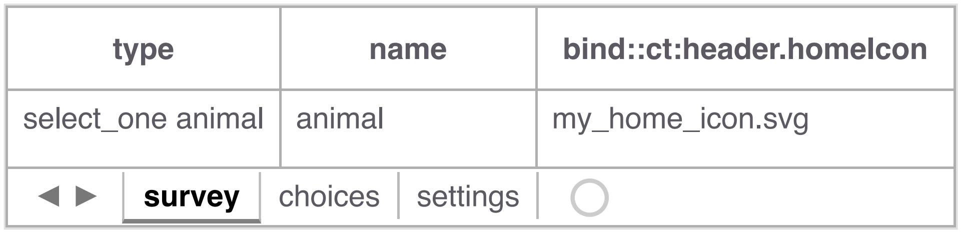
survey
choices
settings

Tapping on the button will provide more information, e.g. the track frequency or the battery level.



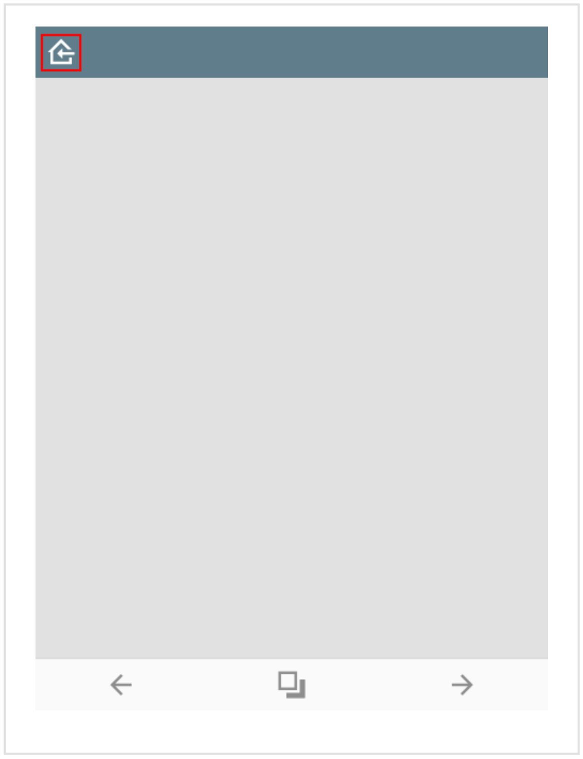
homeIcon

Override the home icon with a custom icon.



type	name	bind::ct:header.homeIcon
select_one animal	animal	my_home_icon.svg

◀ ▶ **survey** choices settings

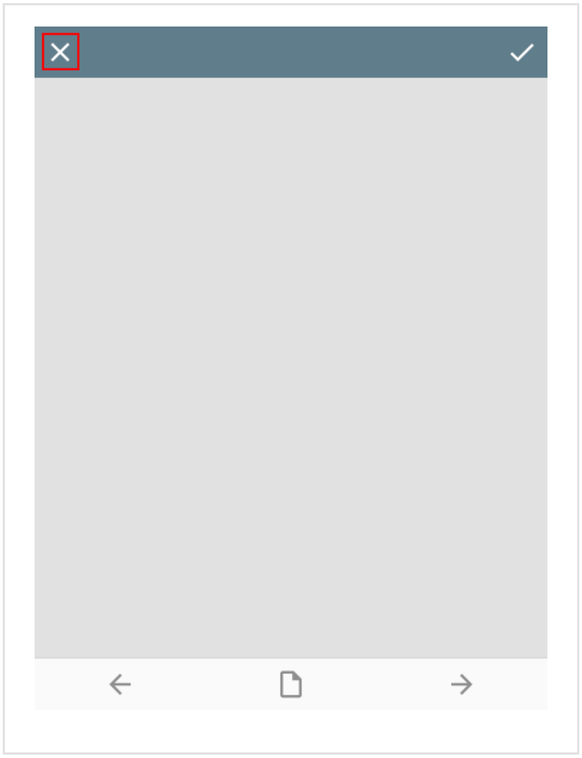


cancelIcon

When editing a sighting in `immersive` mode, the system puts a **Cancel** button in the top left corner. Clicking this button will discard any edits. This property overrides the default icon used.

type	name	bind::ct:header.cancelIcon
select_one animal	animal	my_edit_cancel_icon.svg

◀ ▶ **survey** choices settings ○

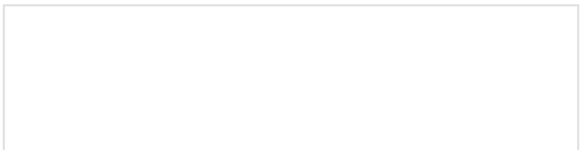


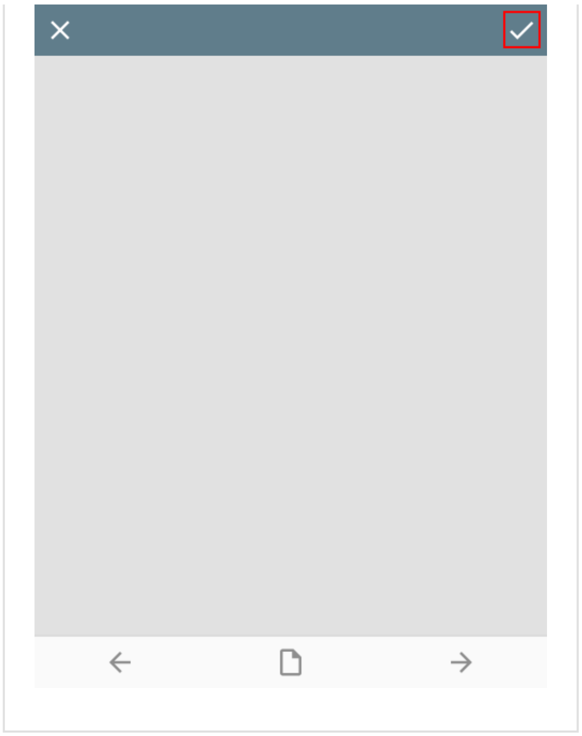
confirmIcon

When editing a sighting in `immersive` mode, the system puts a **Confirm** button in the top right corner. Clicking this button will accept edits made to the sighting. This property overrides the default icon used.

type	name	bind::ct:header.confirmIcon
select_one animal	animal	my_edit_confirm_icon.svg

◀ ▶ **survey** choices settings ○





hideHome

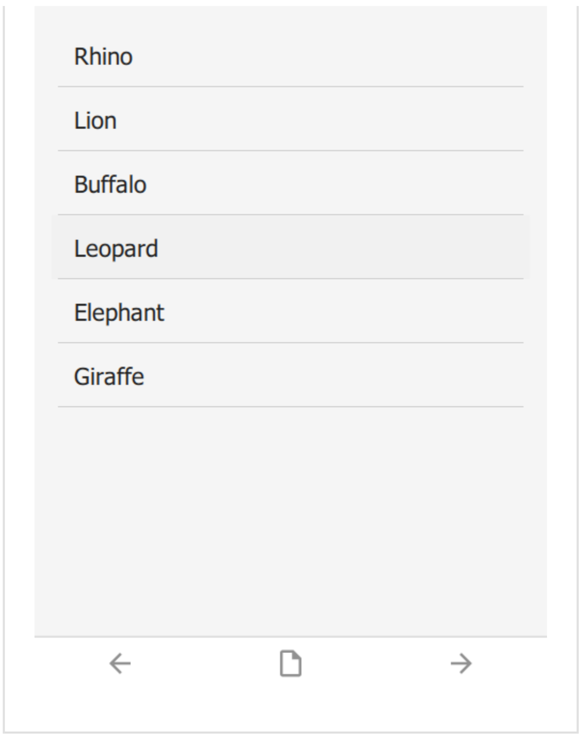
If `yes` then the home button is hidden. Default is `no`.

type	name	bind::ct:header.hideHome
select_one animal	animal	yes
◀ ▶ **survey** choices settings		○

hidden

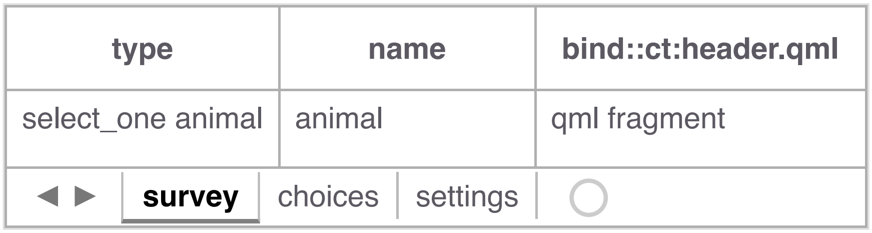
If `yes` then the header is hidden. Default is `no`.

type	name	bind::ct:header.hidden
select_one animal	animal	no
◀ ▶ **survey** choices settings		○


Rhino
Lion
Buffalo
Leopard
Elephant
Giraffe

qml

A QML fragment to use instead of the built-in header. See Developer section. For example:



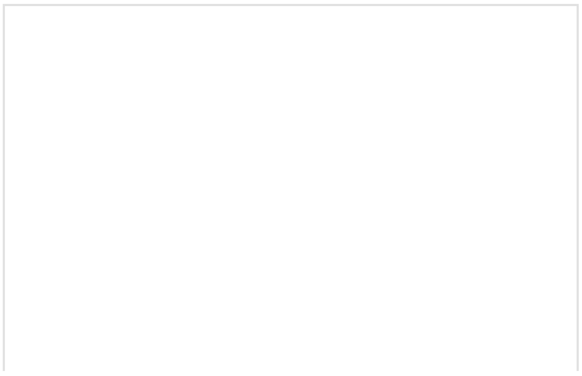
type	name	bind::ct:header.qml
select_one animal	animal	qml fragment

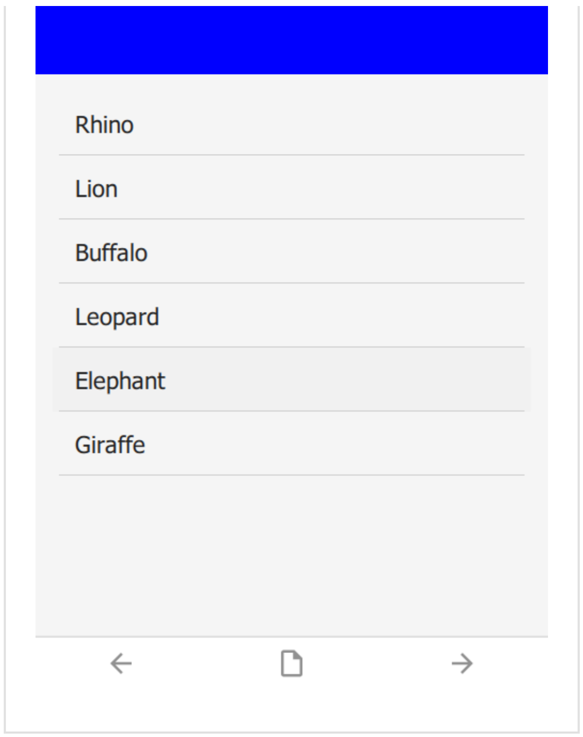
survey
choices
settings

To set the header to a blue rectangle, replace `qml fragment` above with the following:

```
import QtQuick 2.15

Rectangle {
    color: "blue"
    height: 64
}
```





qmlBase64

Base64 encoded QML (see **qml** above).

qmlFile

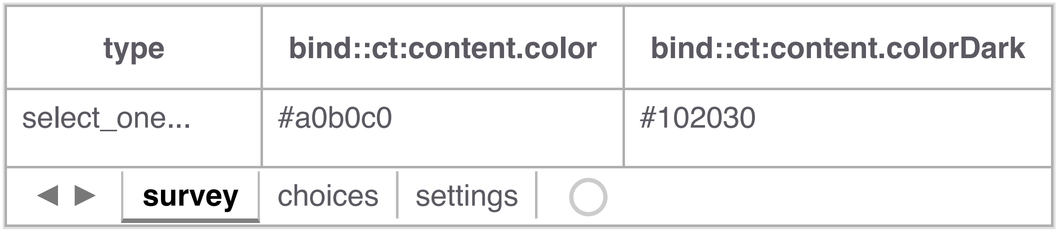
Name of a QML file which exists alongside other project files. This is not supported on ODK or KoBoToolbox, but can be used in Survey123.

Content

The content section is the middle part of the screen between the header and footer. By default it automatically selects a control for the question type, e.g. a date selector for a date question. By specifying a custom content object, more styles are available. This is especially useful for customizing lists.

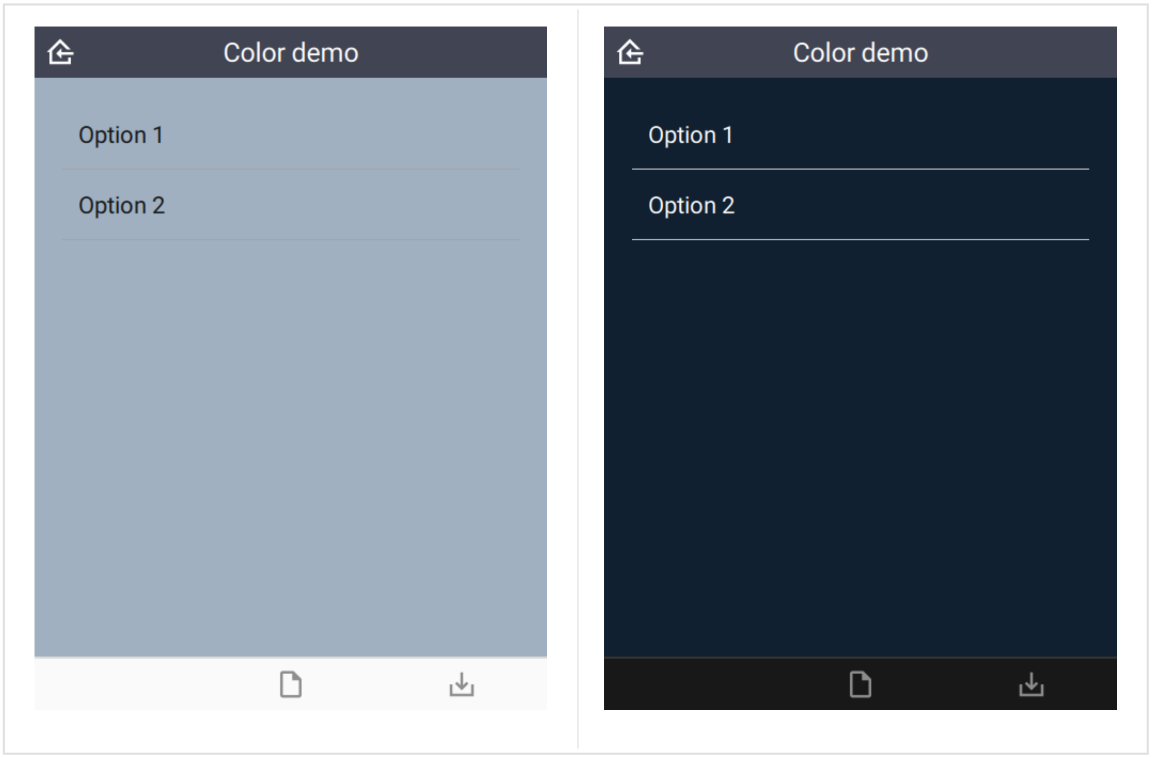
color and colorDark

By default, the content background color is taken from the `settings` sheet. However, it is possible to override it on an individual page.



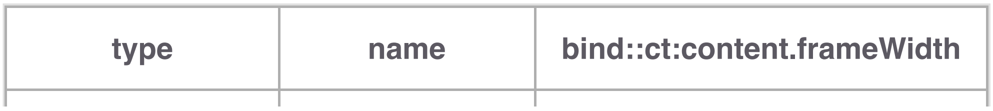
type	bind::ct:content.color	bind::ct:content.colorDark
select_one...	#a0b0c0	#102030

◀ ▶ **survey** choices settings



frameWidth

Frame width around the content area of the page. Default is 16.

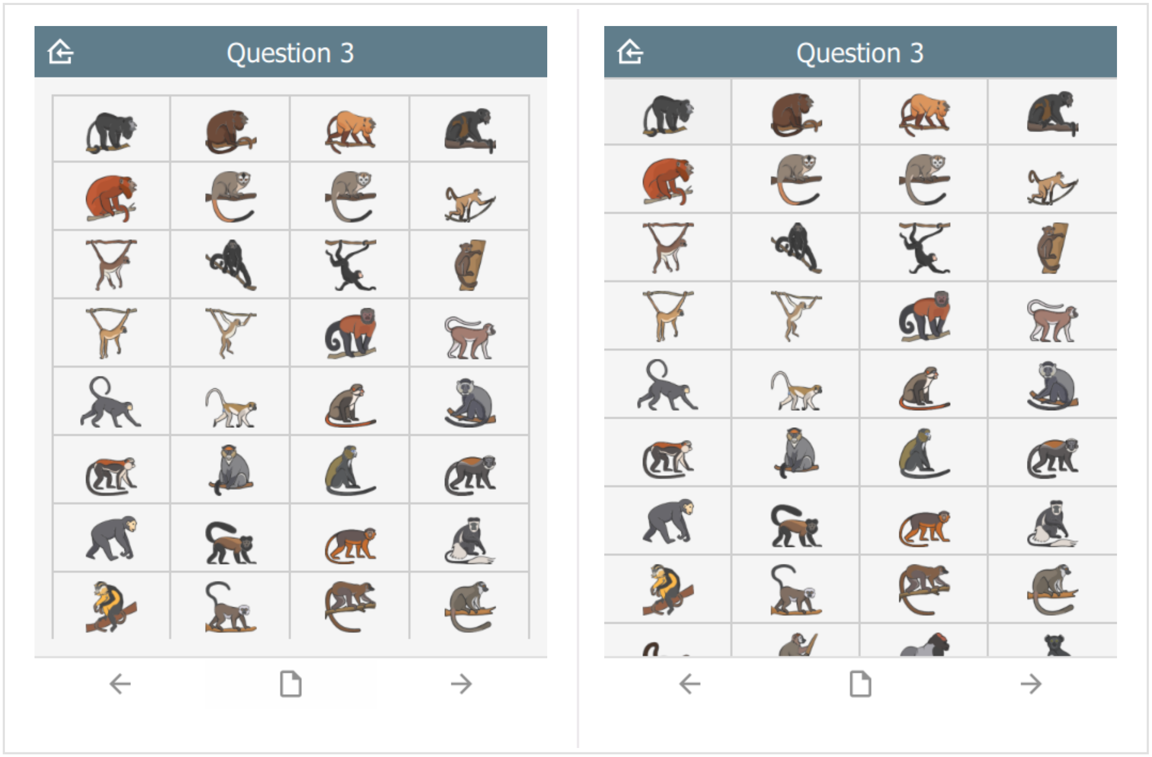


type	name	bind::ct:content.frameWidth

select_one animal	animal	0

◀ ▶ **survey** choices settings ◯

In this case, `frameWidth` was set to `0` in the second image.



style

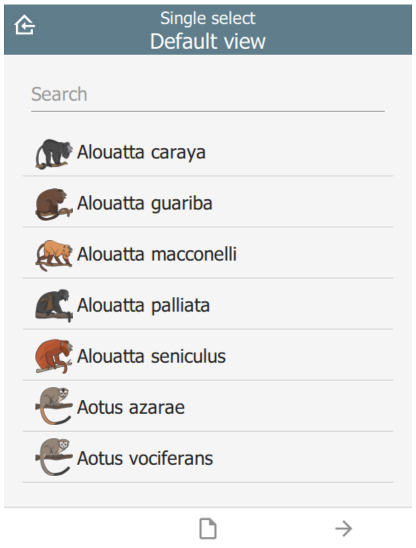
The visual appearance of the question.

type	name	bind::ct:content.style
select_one animal	animal	IconOnly

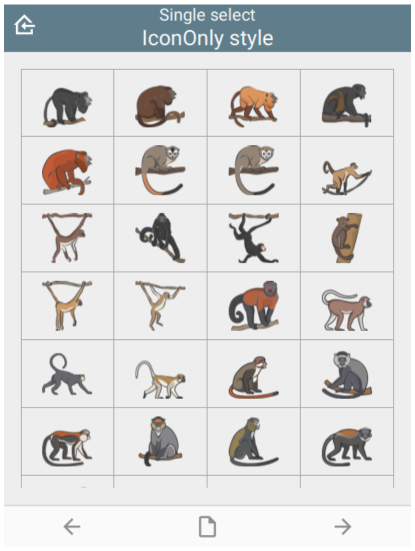
◀ ▶ **survey** choices settings ◯

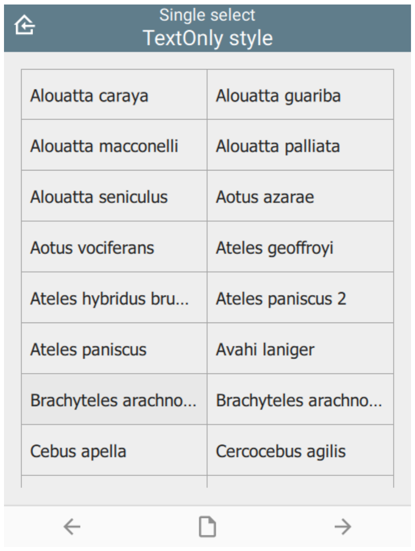
For **select_one** questions

- (not specified)
- IconOnly
- TextOnly
- TextBesideIcon
- TextUnderIcon

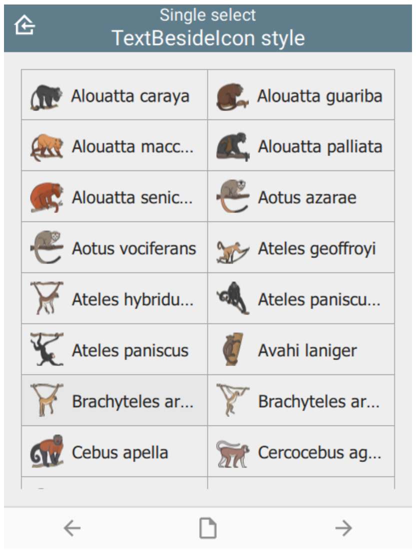


- Alouatta caraya
- Alouatta guariba
- Alouatta macconelli
- Alouatta palliata
- Alouatta seniculus
- Aotus azarae
- Aotus vociferans

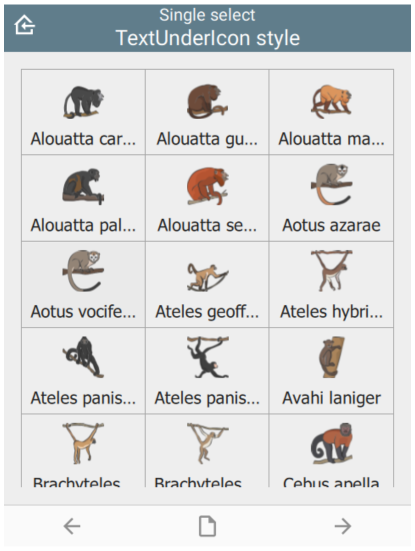




Alouatta caraya	Alouatta guariba
Alouatta macconelli	Alouatta palliata
Alouatta seniculus	Aotus azarae
Aotus vociferans	Ateles geoffroyi
Ateles hybridus bru...	Ateles paniscus 2
Ateles paniscus	Avahi laniger
Brachyteles arachno...	Brachyteles arachno...
Cebus apella	Cercocebus agilis



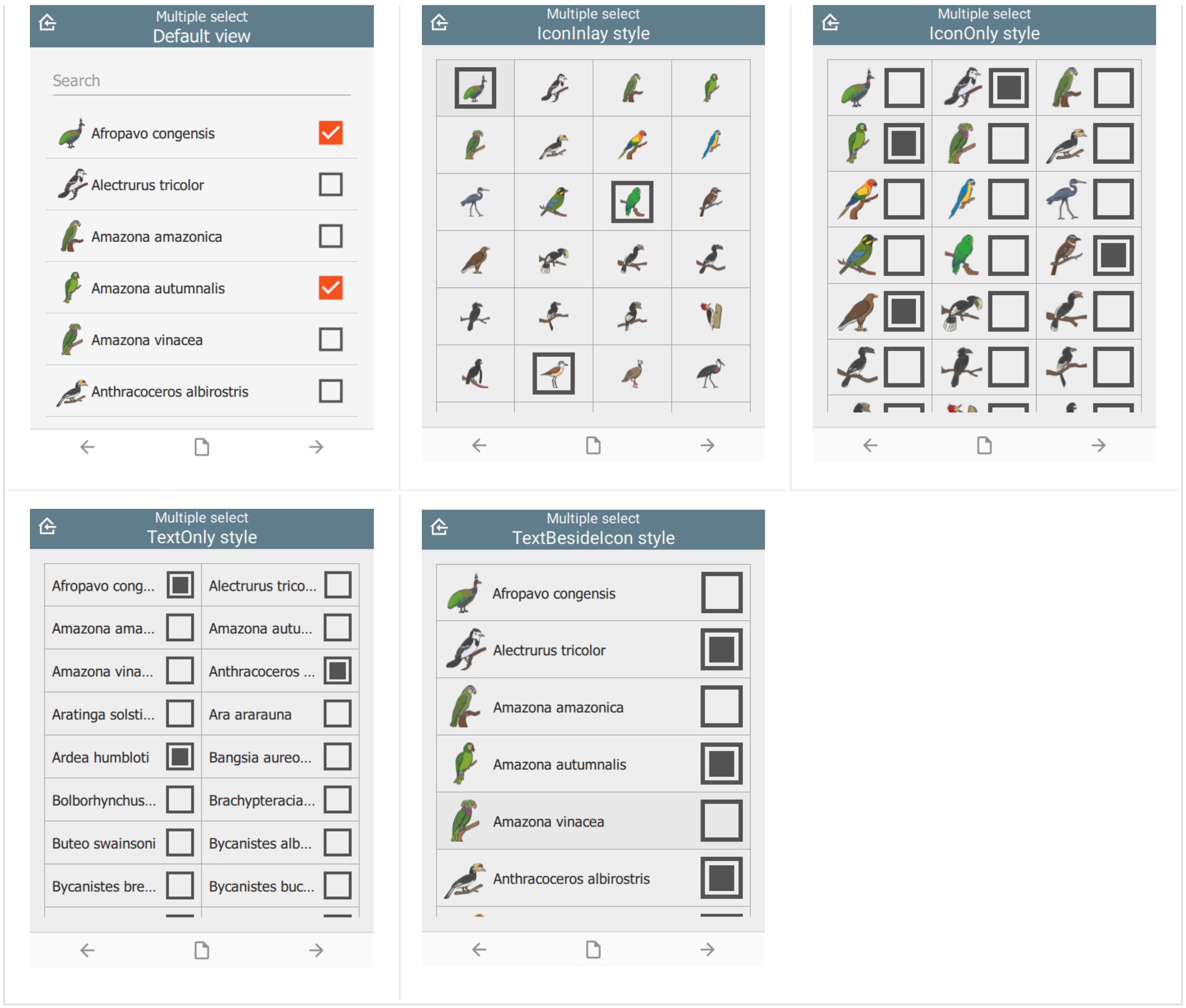
Alouatta caraya	Alouatta guariba
Alouatta macc...	Alouatta palliata
Alouatta senic...	Aotus azarae
Aotus vociferans	Ateles geoffroyi
Ateles hybridu...	Ateles paniscu...
Ateles paniscus	Avahi laniger
Brachyteles ar...	Brachyteles ar...
Cebus apella	Cercocebus ag...



Alouatta car...	Alouatta gu...	Alouatta ma...
Alouatta pal...	Alouatta se...	Aotus azarae
Aotus vocife...	Ateles geoff...	Ateles hybri...
Ateles panis...	Ateles panis...	Avahi laniger
Brachyteles	Brachyteles	Cebus apella

For **select_multiple** questions:

- (not specified)
- IconInlay
- IconOnly
- TextOnly
- TextBesideIcon

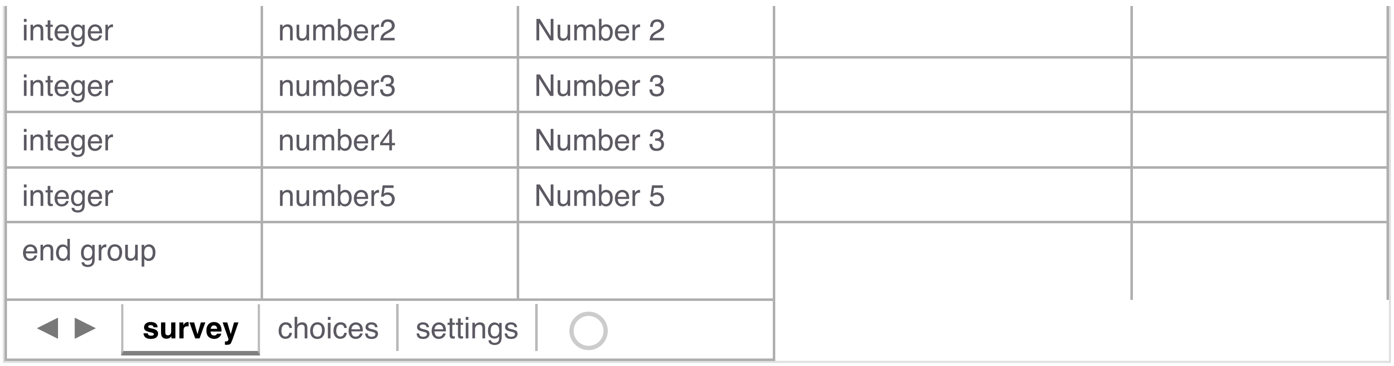


For **number list** groups:

- (not specified)
- IconOnly
- TextOnly
- TextBesideIcon

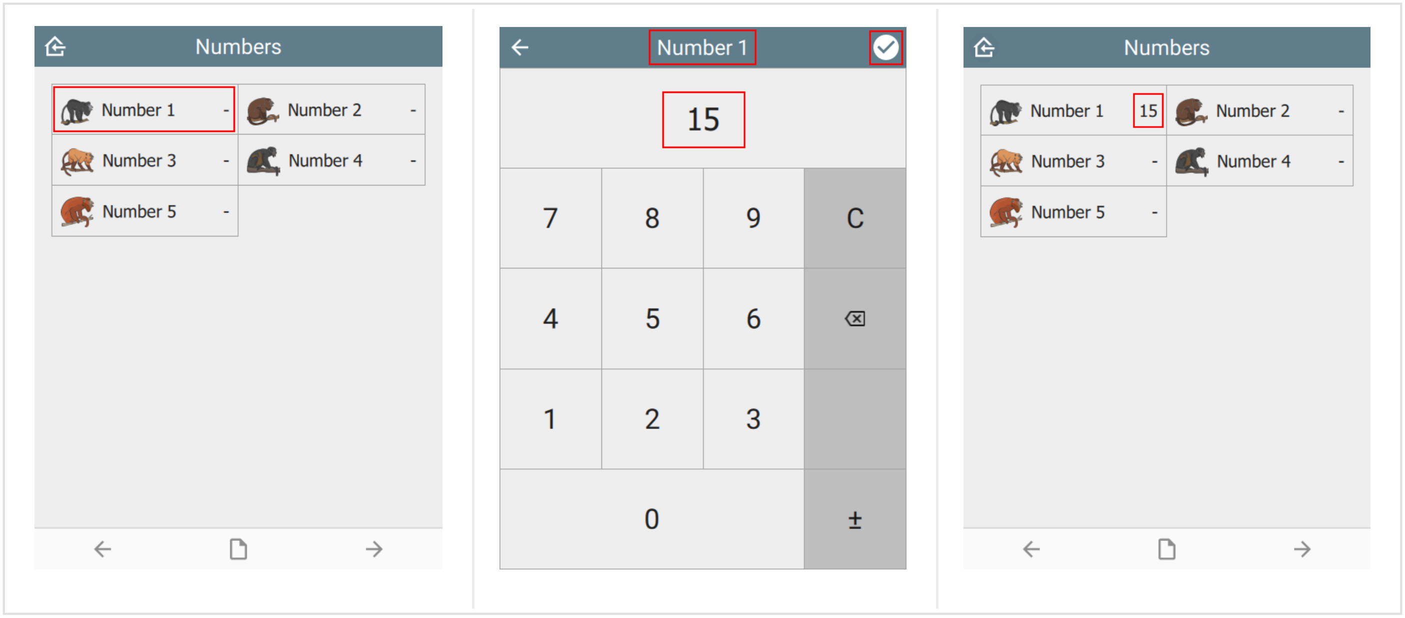
Number lists are a set of questions inside a `group`. The `appearance` column must be set to `field-list` to force all group questions to appear on the same page:

type	name	label	bind::ct:content.style	appearance
begin group	numberlist	Number list	IconOnly	field-list
integer	number1	Number 1		

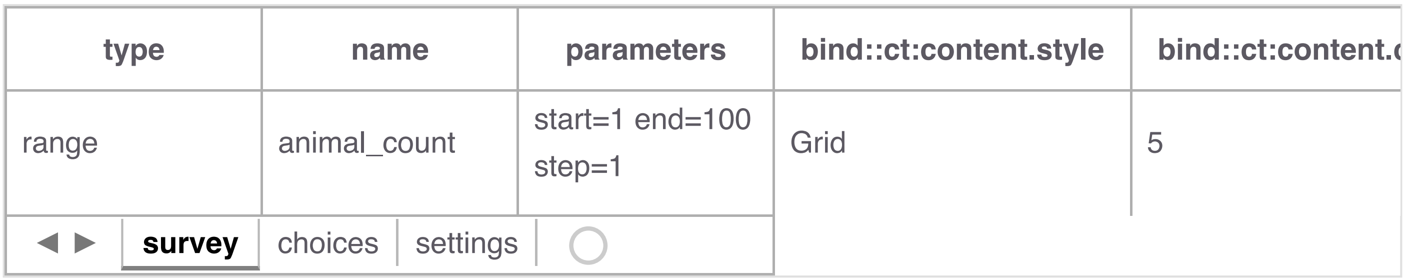


integer	number2	Number 2		
integer	number3	Number 3		
integer	number4	Number 3		
integer	number5	Number 5		
end group				

survey
choices
settings

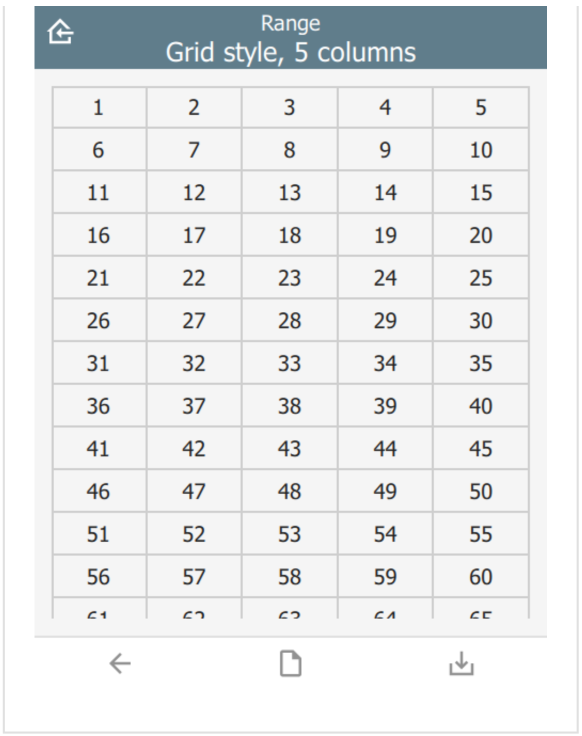


For **range** questions:



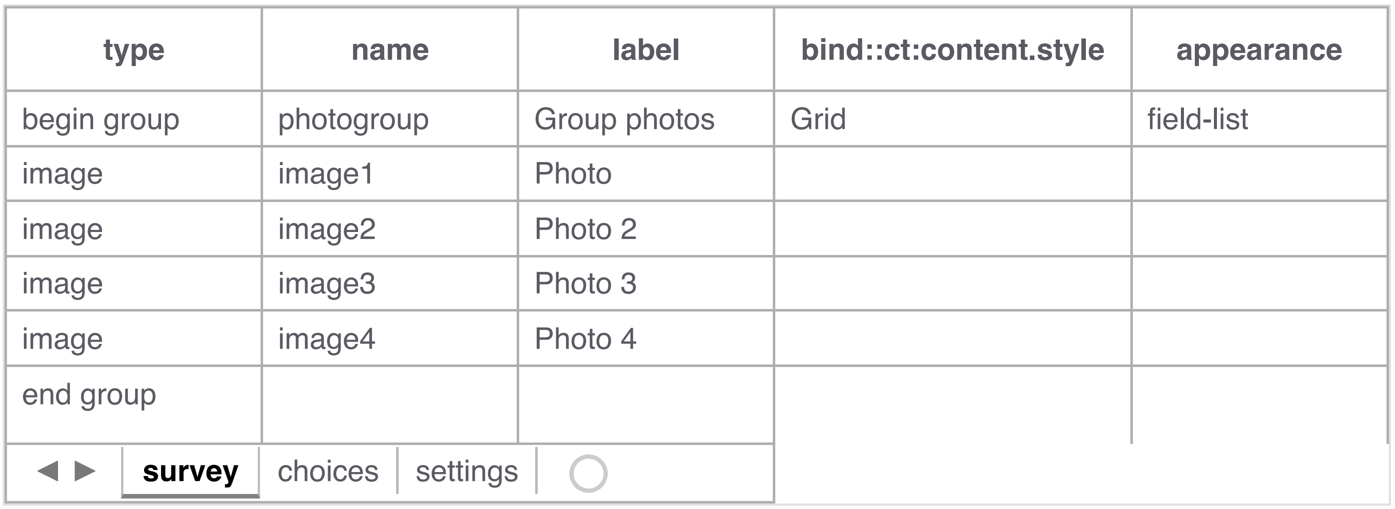
type	name	parameters	bind::ct:content.style	bind::ct:content.c
range	animal_count	start=1 end=100 step=1	Grid	5

survey
choices
settings


Range
Grid style, 5 columns

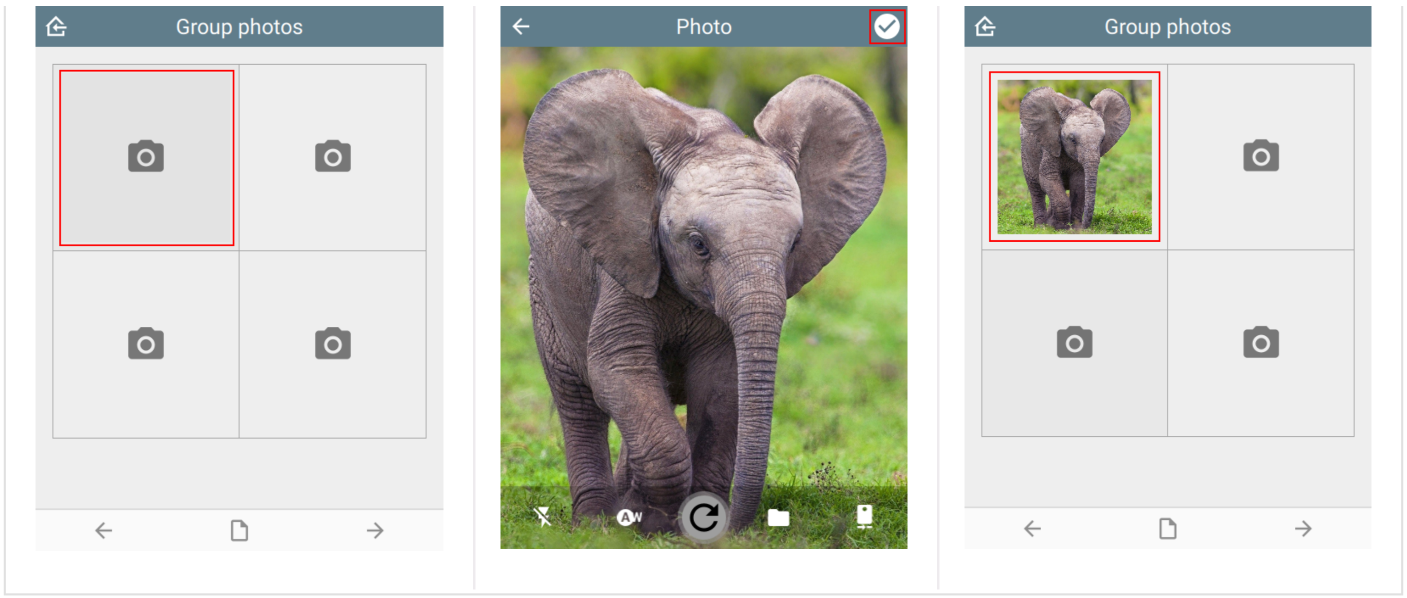
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30
31	32	33	34	35
36	37	38	39	40
41	42	43	44	45
46	47	48	49	50
51	52	53	54	55
56	57	58	59	60
61	62	63	64	65

For a fixed number of **image** questions:



type	name	label	bind::ct:content.style	appearance
begin group	photogroup	Group photos	Grid	field-list
image	image1	Photo		
image	image2	Photo 2		
image	image3	Photo 3		
image	image4	Photo 4		
end group				

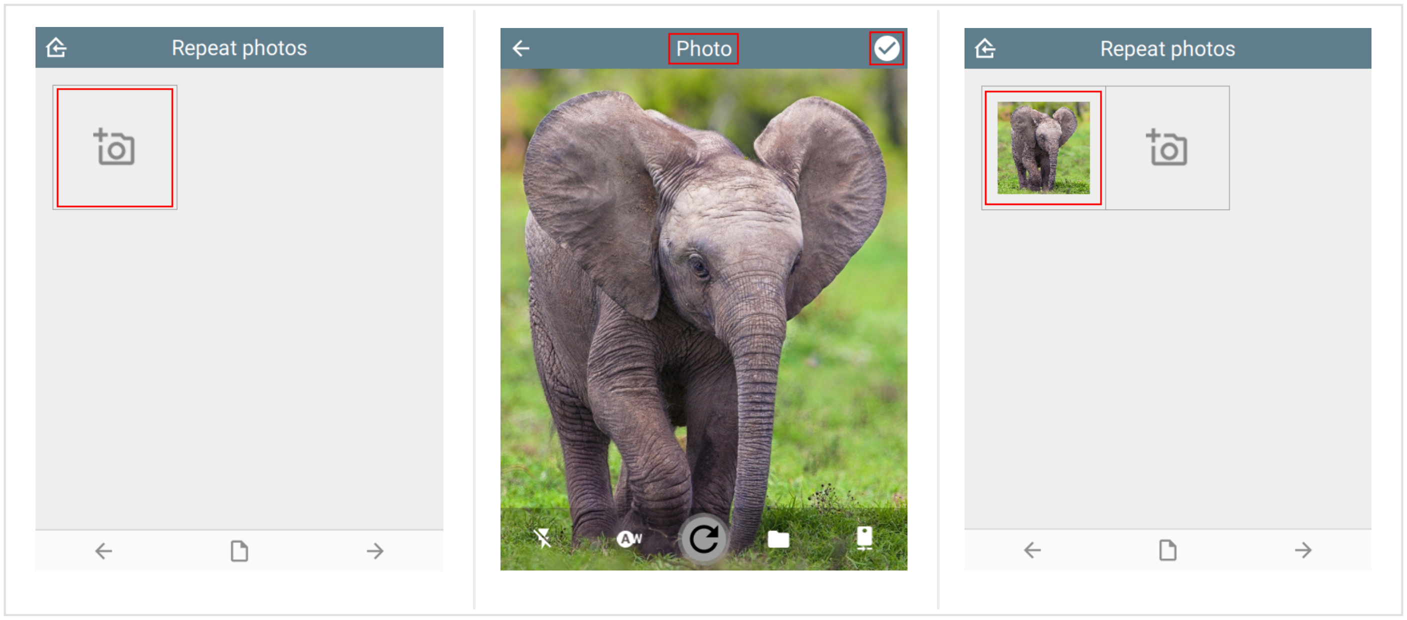
◀ ▶
survey
choices
settings



For a dynamic number of **image** questions:

type	name	label	bind::ct:content.style
begin repeat	photogroup	Repeat photos	Grid
image	image	Photo	
end repeat			

◀ ▶ **survey** choices settings



Ignored for other question types.

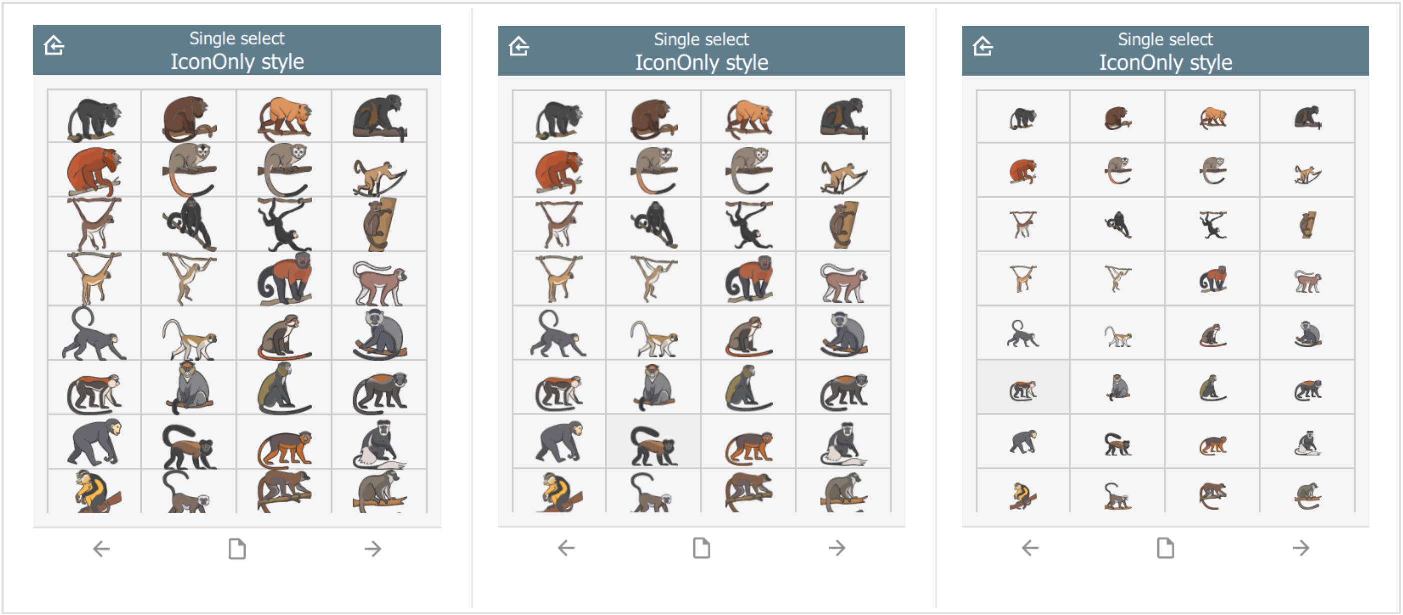
padding

The padding space between grid items. Requires `style` to be set.

type	name	bind::ct:content.padding
select_one animal	animal	8

◀ ▶ **survey** choices settings

Padding values are `0`, `4` and `8`.



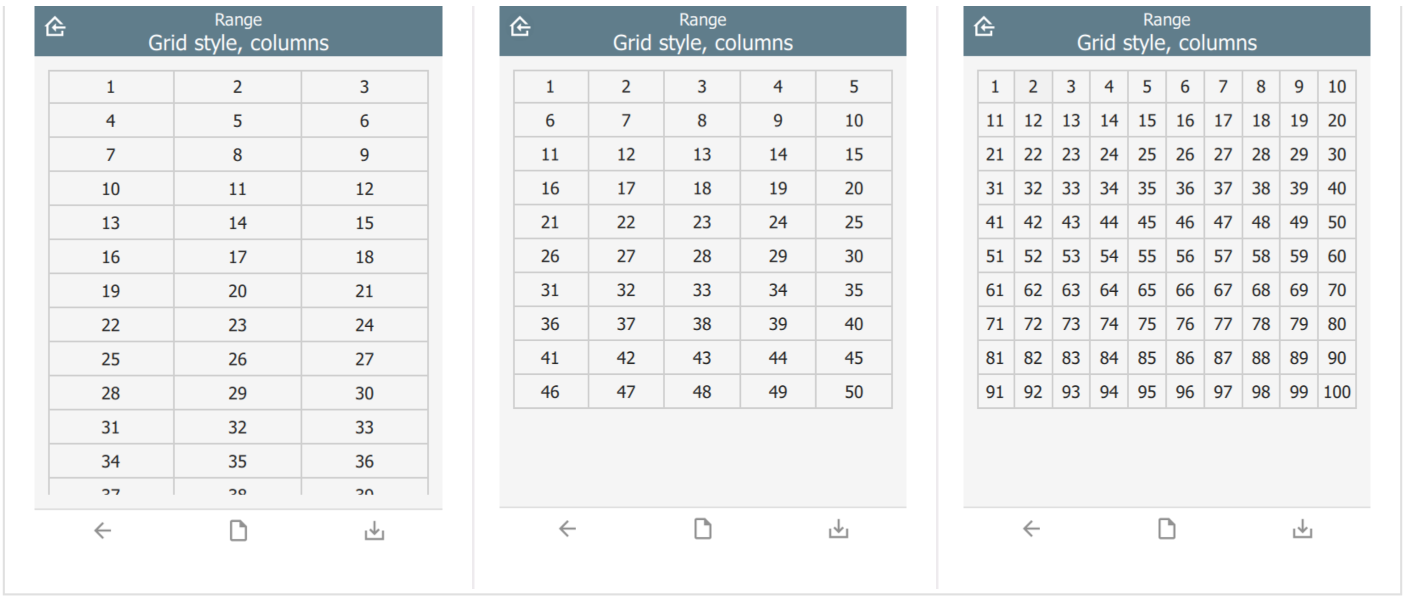
columns

Number of columns for grids. Requires `style` to be set. Defaults to 2.

type	name	bind::ct:content.style	bind::ct:content.columns
select_one animal	animal	Grid	4

◀ ▶ **survey** choices settings

For example, column values below are `3`, `5` and `10`.



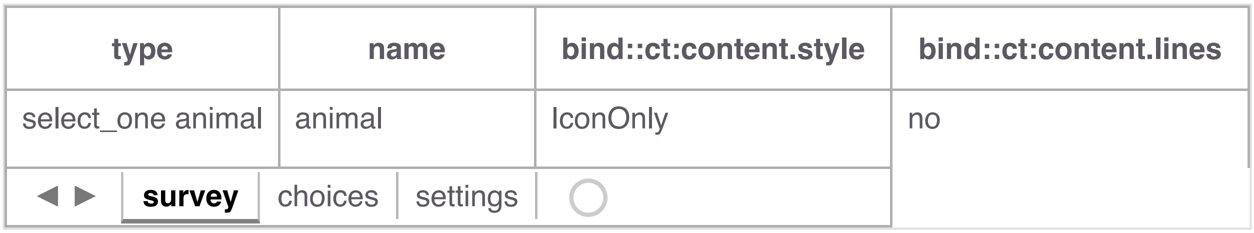
1	2	3
4	5	6
7	8	9
10	11	12
13	14	15
16	17	18
19	20	21
22	23	24
25	26	27
28	29	30
31	32	33
34	35	36

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30
31	32	33	34	35
36	37	38	39	40
41	42	43	44	45
46	47	48	49	50

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

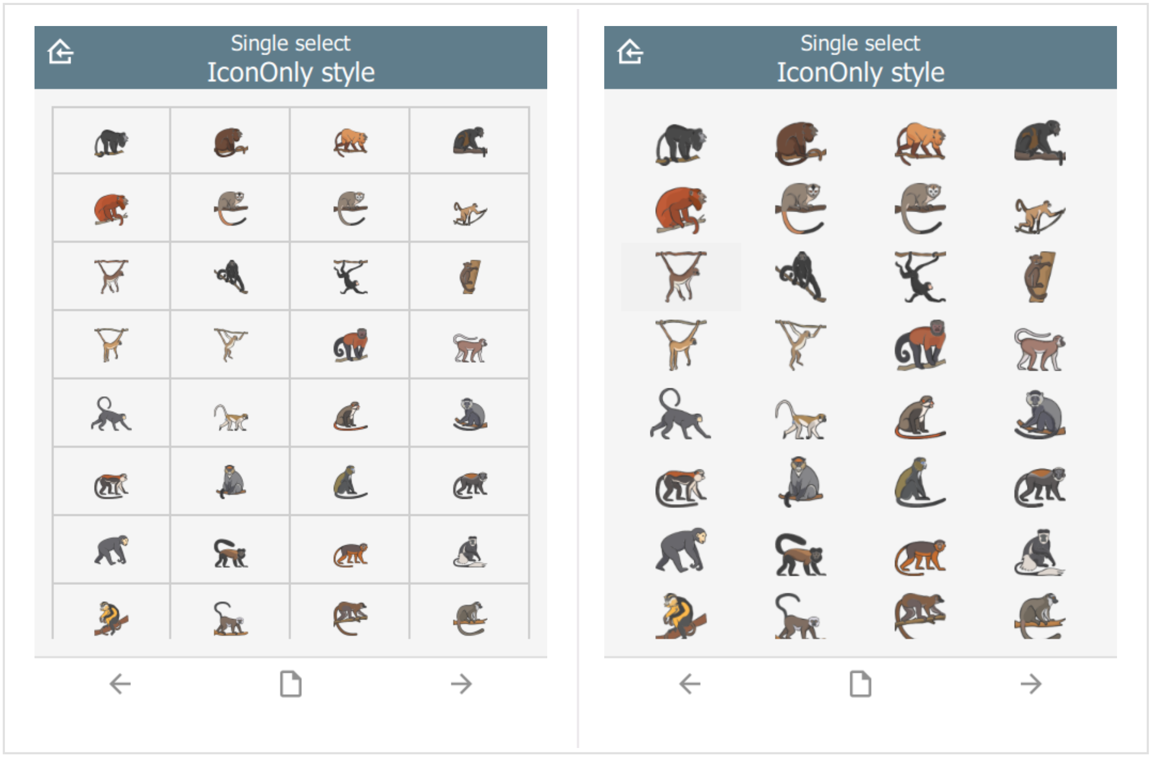
lines

Show lines between cells for grids. Requires `style` to be set. Defaults to true.



type	name	bind::ct:content.style	bind::ct:content.lines
select_one animal	animal	IconOnly	no

`lines` value below is `yes` and `no`.



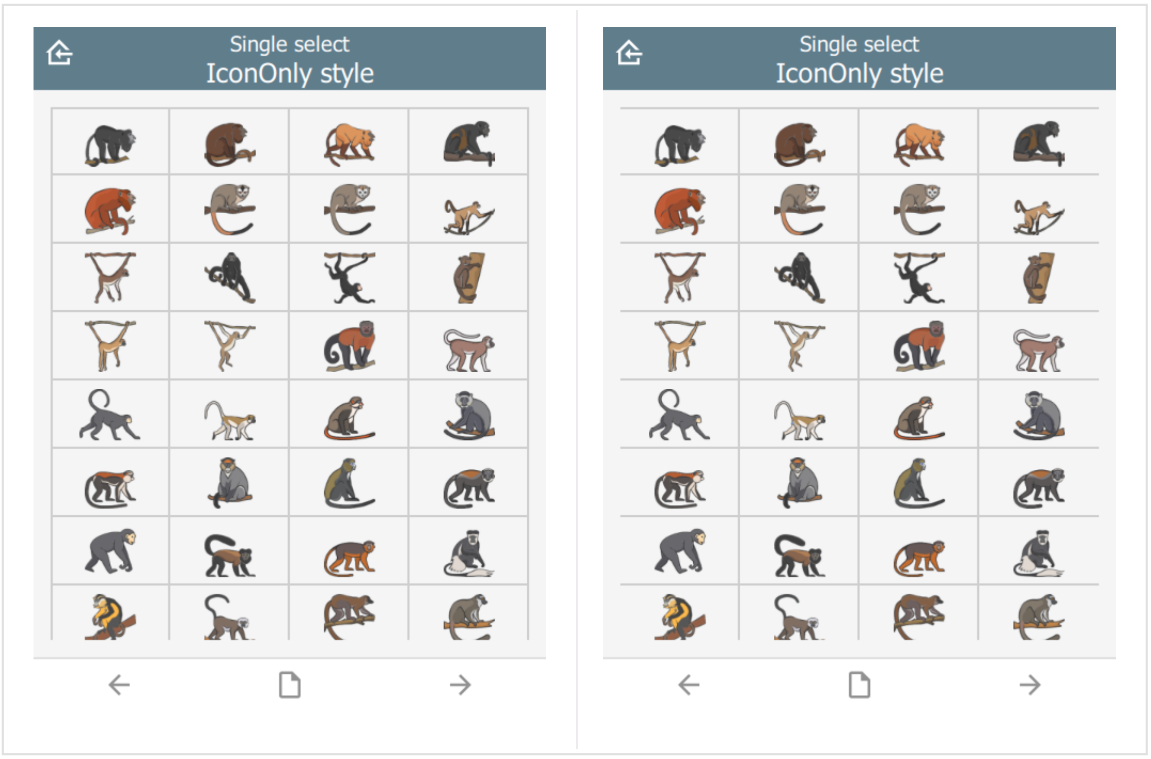
border

Show border around the outside of a grid. Requires `style` to be set. Defaults to `no` if frameWidth is 0, `yes` otherwise.

type	name	bind::ct:content.style	bind::ct:content.border
select_one animal	animal	IconOnly	yes

◀ ▶ **survey** choices settings

Border value is `yes` and `no`.



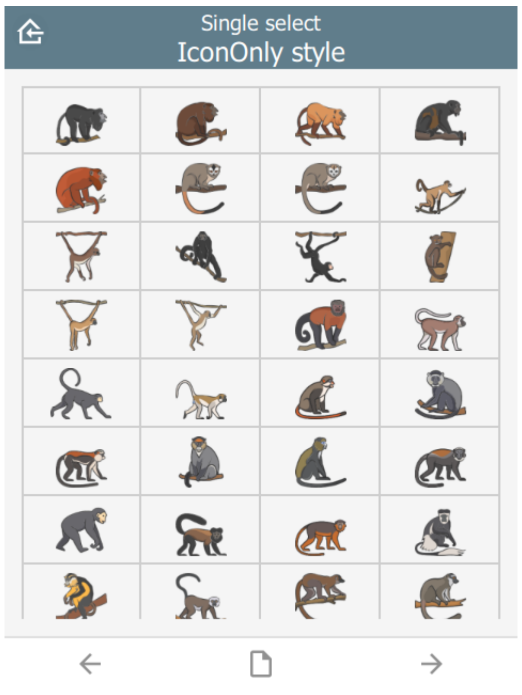
borderWidth

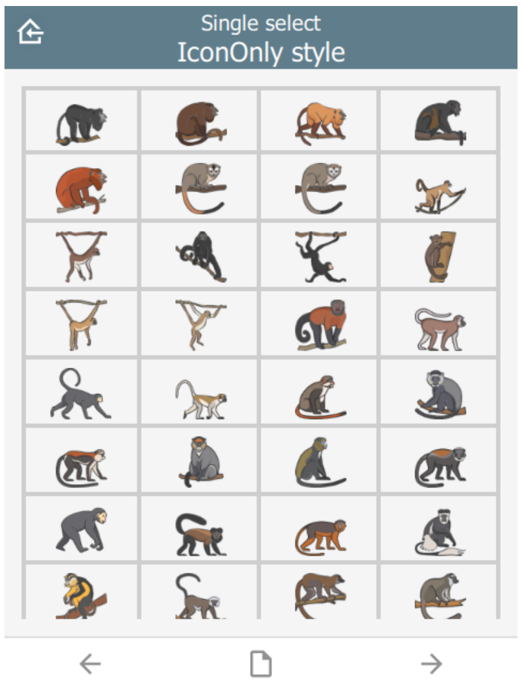
Border width for grid lines. Requires `style` to be set. Defaults to 2.

type	name	bind::ct:content.style	bind::ct:content.borderWidth
select_one animal	animal	IconOnly	2

◀ ▶ **survey** choices settings

borderWidth value is `2` and `4`.





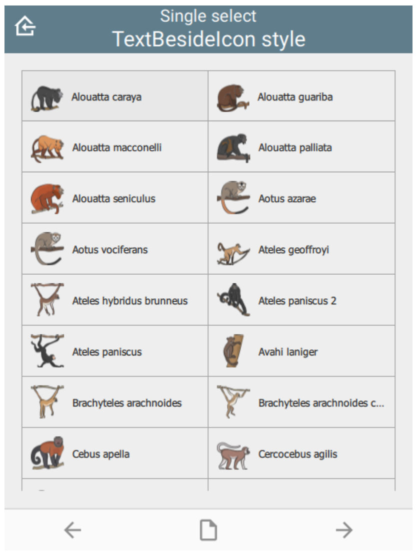
fontSize

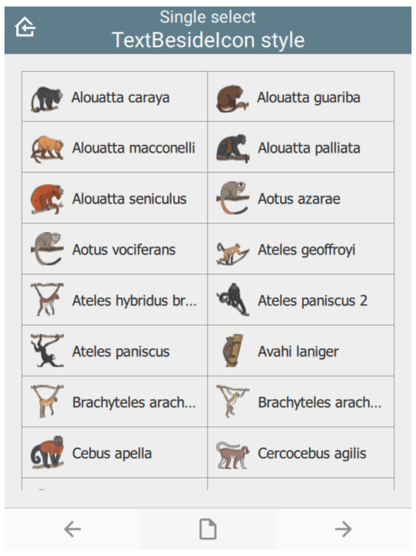
Size text font size. Requires `style` to be set. Defaults to 16. Note that the font is subject to scaling according to the **Font size** in the main Settings page.

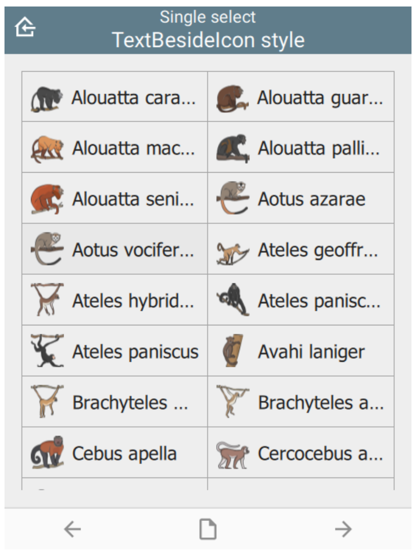
type	name	bind::ct:content.style	bind::ct:content.fontSize
select_one animal	animal	IconOnly	14

◀ ▶ **survey** choices settings

`fontSize` values are `10`, `14` and `18`.







fontBold

Set font to bold. Requires `style` to be set. Defaults to false.

type	name	bind::ct:content.style	bind::ct:content.fontBold
select_one animal	animal	IconOnly	yes

◀ ▶ **survey** choices settings

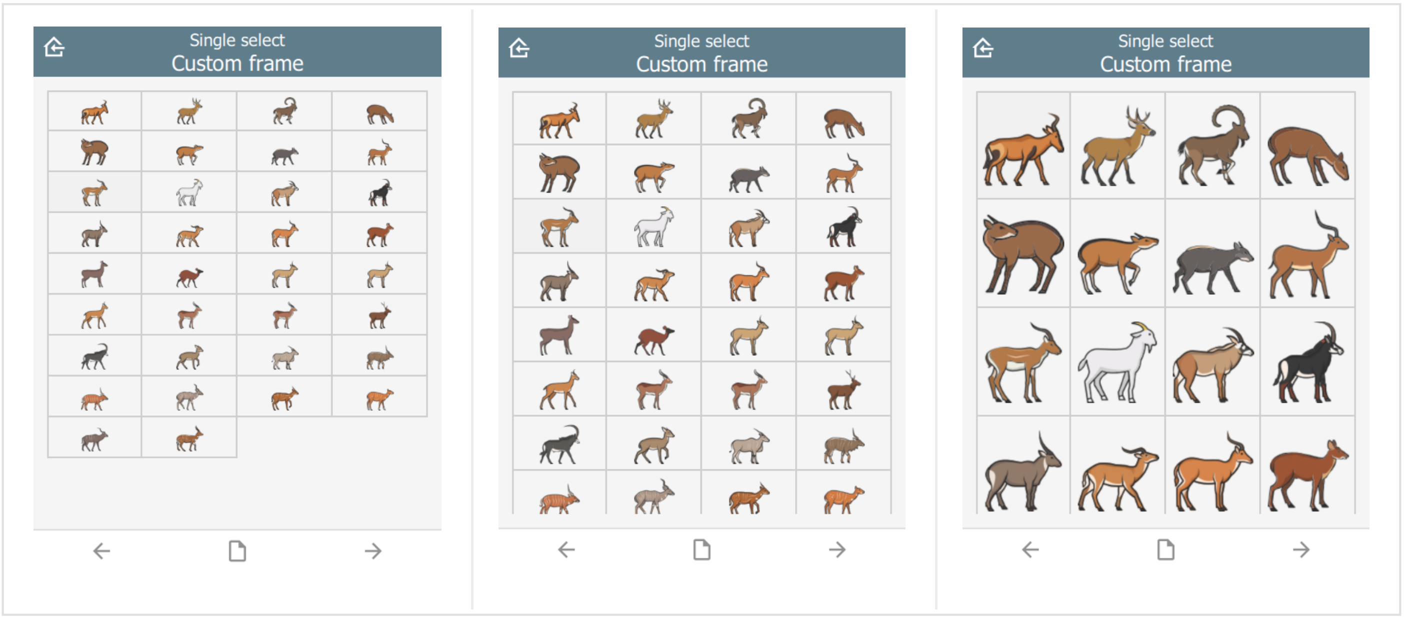
itemHeight

Set height of individual items. Requires `style` to be set. Defaults to 48.

type	name	bind::ct:content.style	bind::ct:content.itemHeight
select_one animal	animal	IconOnly	48

◀ ▶ **survey** choices settings

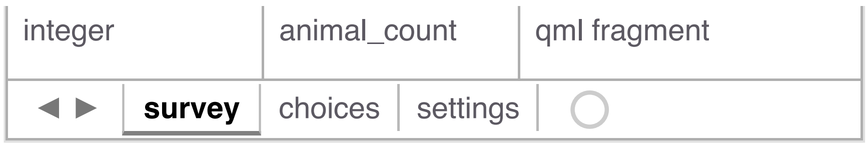
`itemHeight` values are `48`, `64` and `128`.



qml

A QML fragment to use instead of the built-in content. See Developer section. For example:

type	name	bind::ct:content.qml



integer	animal_count	qml fragment

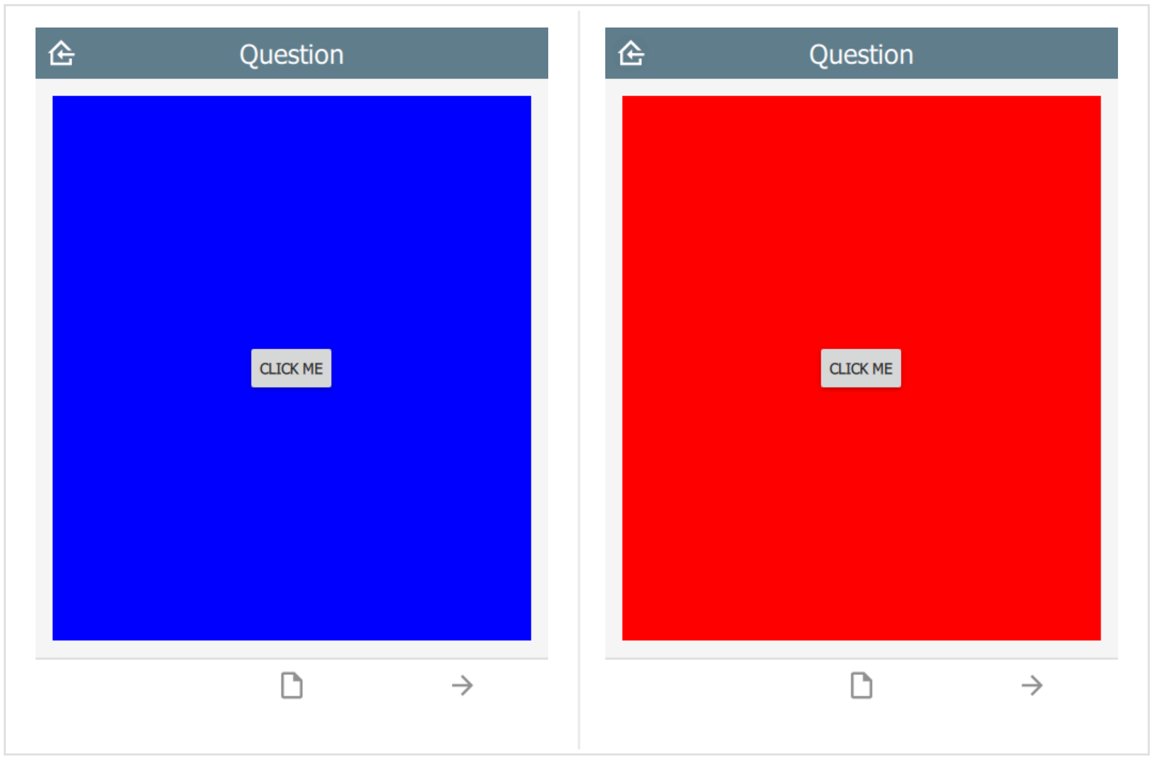
◀ ▶ **survey** | choices | settings

To set the content to a blue rectangle, replace `qml fragment` above with the following:

```
import QtQuick 2.15
import QtQuick.Controls 2.15

Rectangle {
    color: "blue"
    Button {
        anchors.centerIn: parent
        text: "Click me"
        onClicked: parent.color = "red"
    }
}
```

In the example, the content is blue, then changes to red when the button is clicked.



qmlBase64

Base64 encoded QML (see **qml** above).

qmlFile

Name of a QML file which exists alongside other project files. This is not supported on

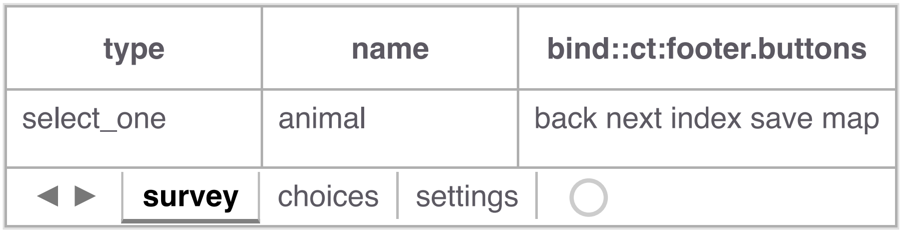
ODK or KoBoToolbox, but can be used in Survey123.

Footer

The footer object supports custom control buttons, e.g. home, back, next, save, etc. If no footer object is specified, then the default control is used.

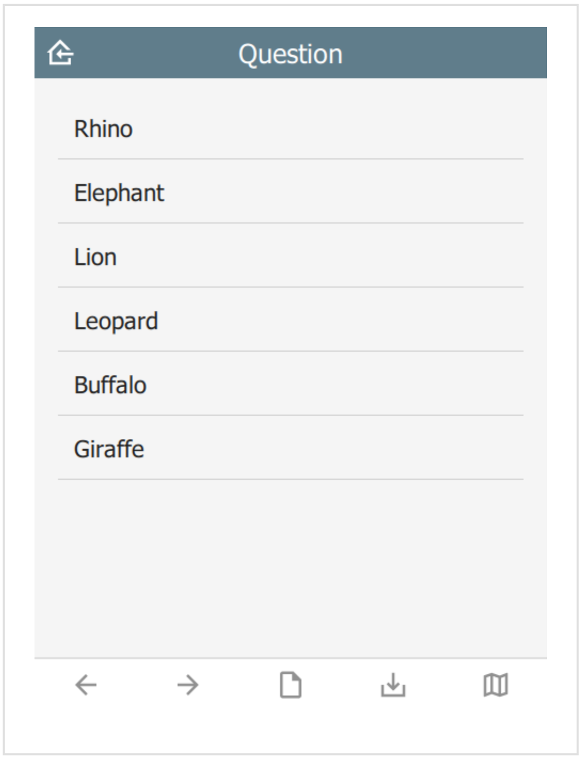
buttons

`buttons` is text which specifies which buttons should be shown on the footer toolbar.



type	name	bind::ct:footer.buttons
select_one	animal	back next index save map

◀ ▶
survey
choices
settings



home button

The `home` button returns to the **Home** page. In `immersive` mode, this returns to the Projects page, otherwise it returns to the project home page.

back button

The `back` button navigates to the prior question on the form. If the wizard is at the start of the form, the back button is hidden.

next button

The `next` button navigates to the next question on the form. If there is no next question, then the next button is hidden.

save button

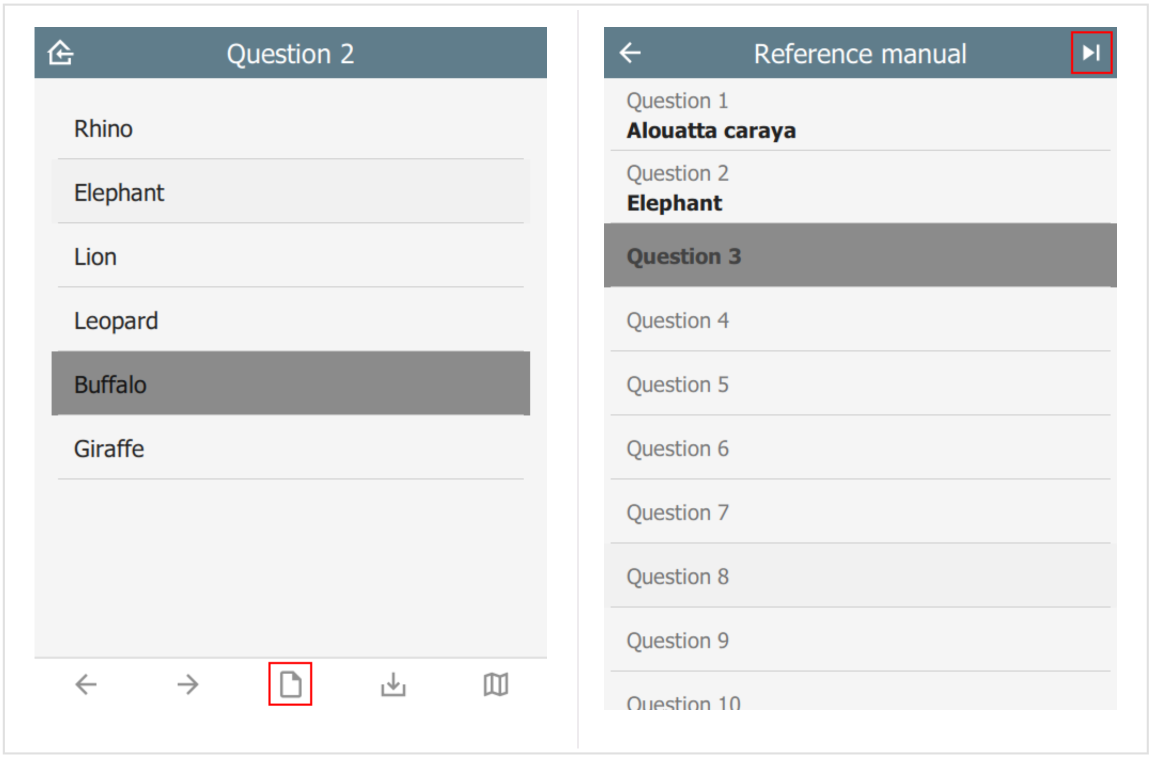
The `save` button will attempt to save the current sighting. If the sighting has invalid data, then the **Index page** will be shown with invalid fields highlighted.

nextOrSave button

The `nextOrSave` button will show as a `next` button unless there are no more questions, in which case it will become a `save` button.

index button

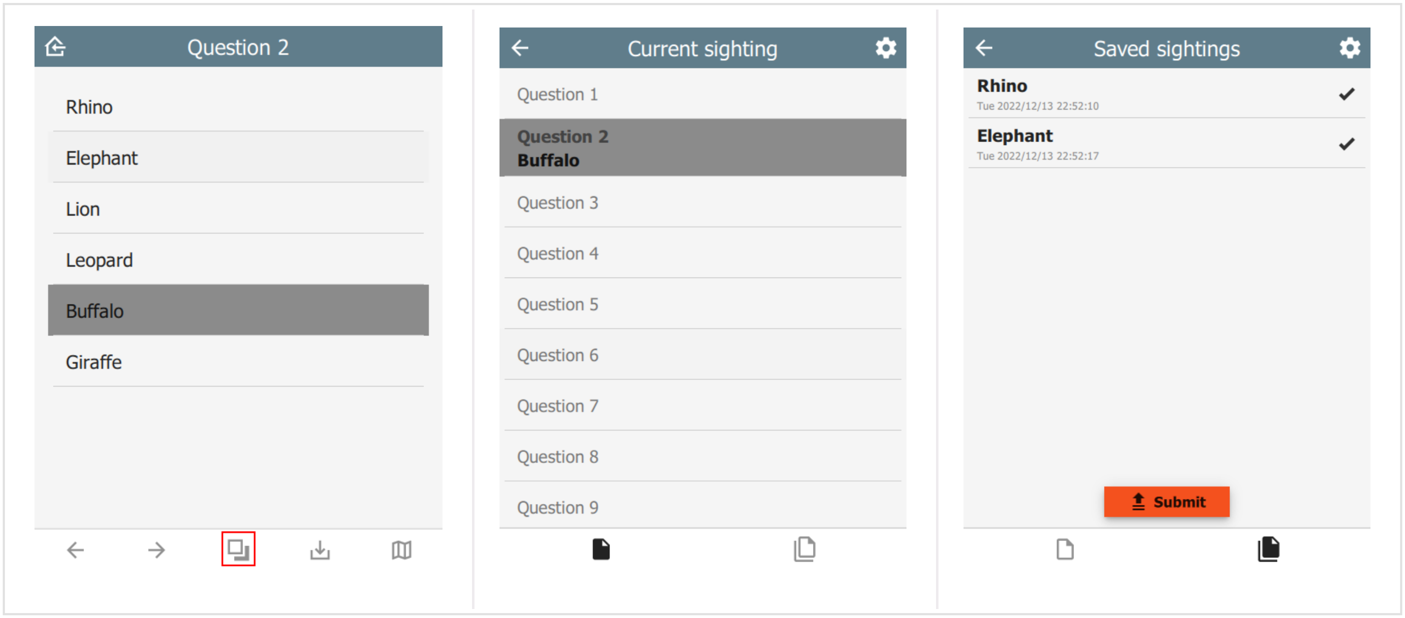
The `index` button displays a list of all the form questions. Selecting a question will navigate the wizard to it directly. A jump-to-last button on the top right of the header will jump to the next required question. If all required questions are filled in, then it jumps to the last question.



options button

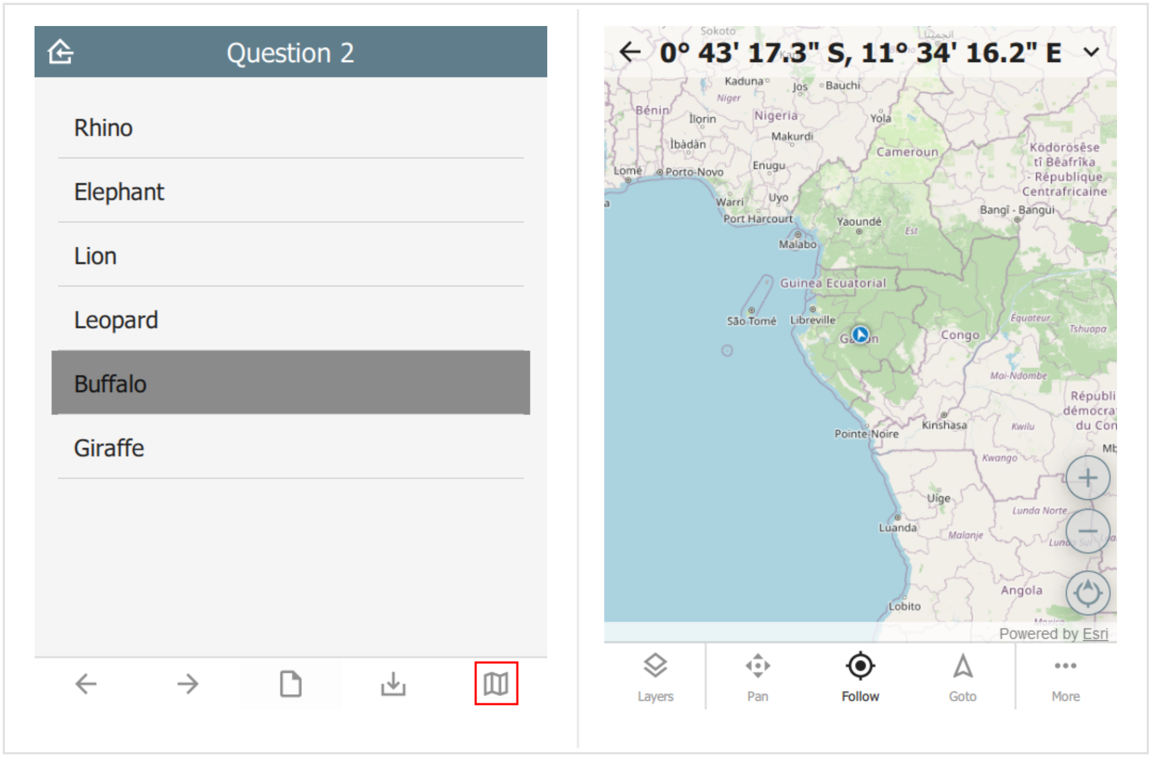
The `options` button is only available in `immersive` mode. In non-immersive mode, it becomes the `index` button (see above).

The `options` button shows an options page with two tabs: current sighting and saved sightings:



map button

The `map` button opens the map dialog.



Custom button icons

The button icons can be overridden with custom ones. To do this, create columns with the name of the button followed by `Icon`. For example:

- homeIcon
- backIcon
- nextIcon
- saveIcon
- indexIcon
- optionsIcon
- mapIcon

type	name	bind::ct:footer.mapIcon
select_one	animal	my_custom_map_icon.svg

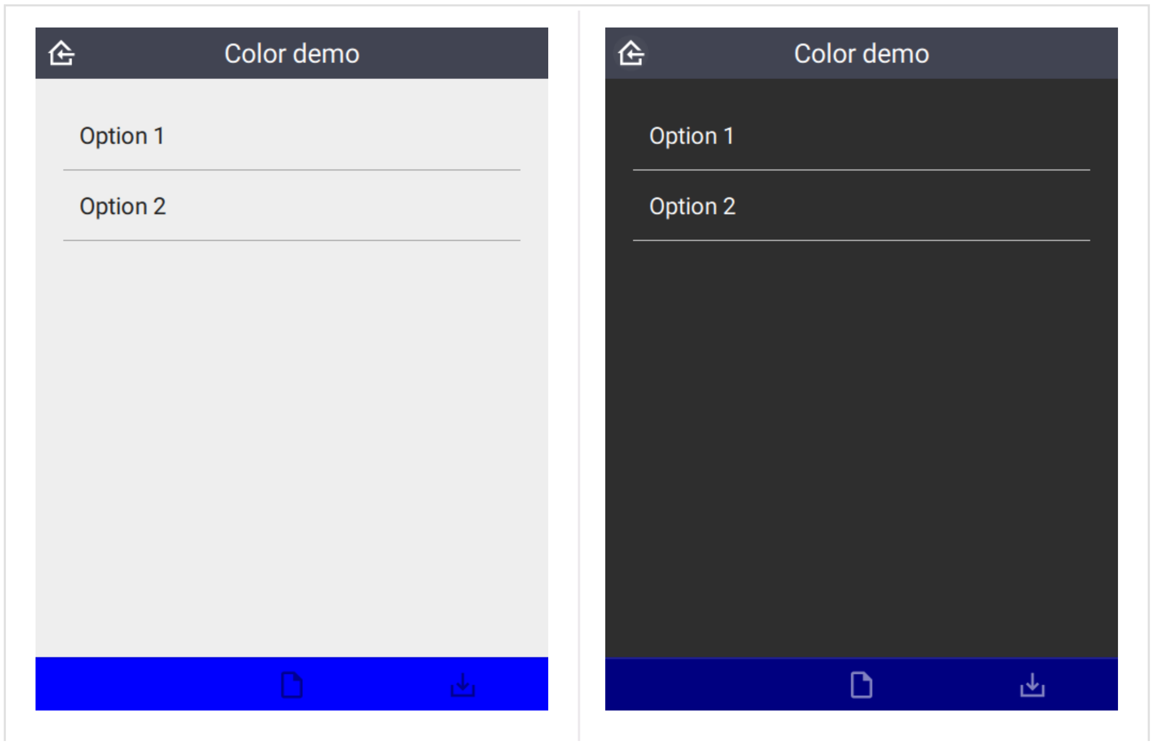
◀ ▶ **survey** choices settings

color and colorDark

By default, the footer background color is taken from the `settings` sheet. However, it is possible to override it on an individual page.

type	bind::ct:footer.color	bind::ct:footer.colorDark
select_one...	#0000ff	#000080

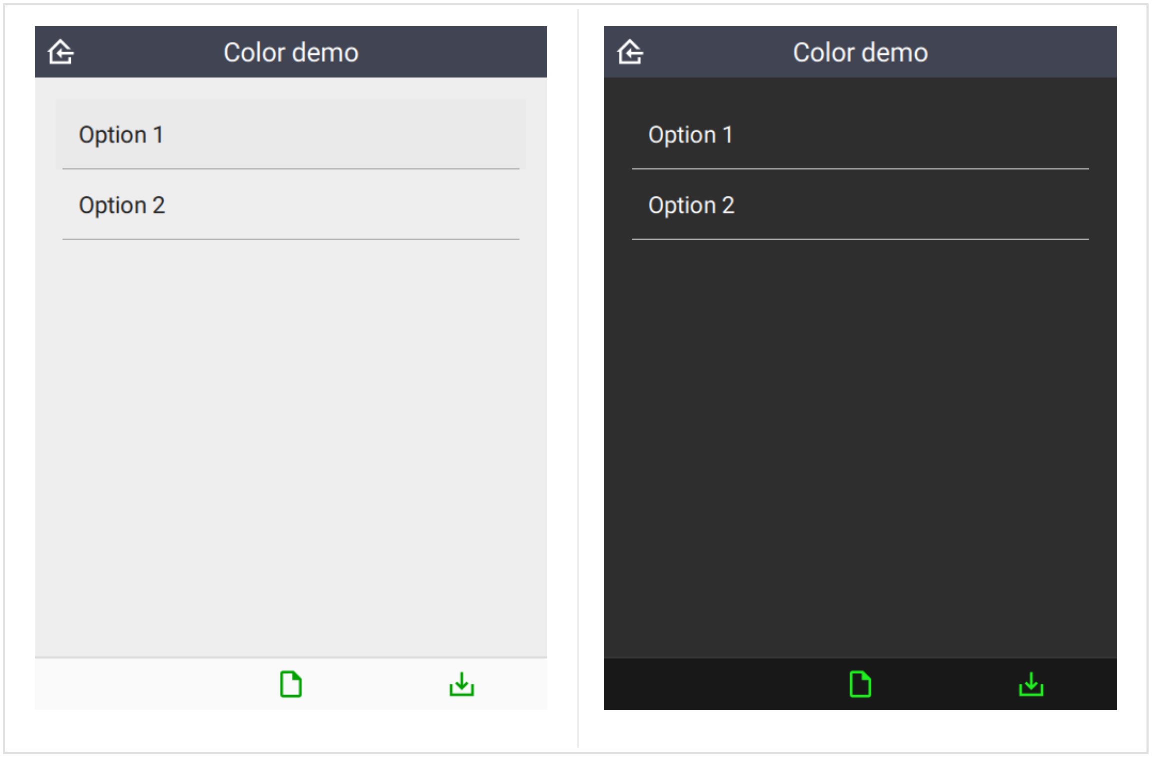
◀ ▶ **survey** choices settings



buttonColor and buttonColorDark

Override the default button color with a custom one. This applies to all buttons.

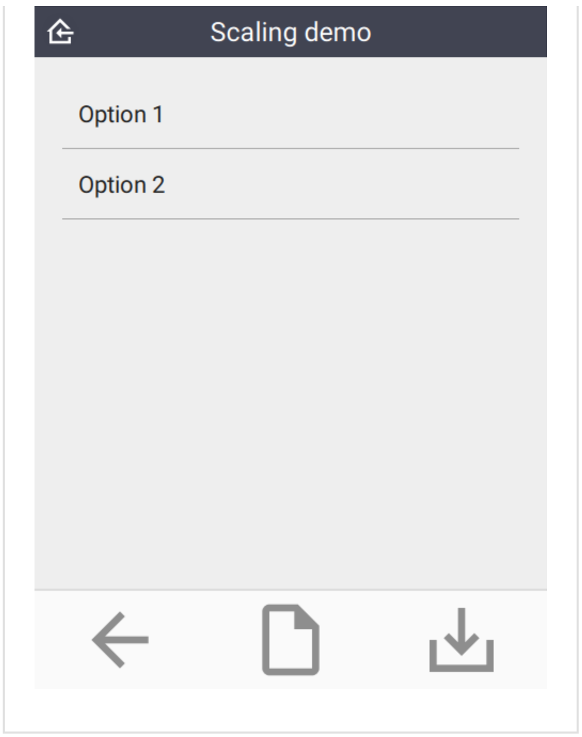
type	bind::ct:footer.buttonColor	bind::ct:footer.buttonColorDark
select_one...	#00a000	#20f020
◀ ▶ **survey** choices settings ○		



buttonScale

An additional scaling factor to apply to the button size. This is typically useful for increasing the size of footer buttons. The scaling factor will be capped to allow at least 6 buttons to fit in the footer.

type	bind::ct:footer.buttonScale
select_one...	3.5
◀ ▶ **survey** choices settings ○	



hidden

If `yes` then the footer is hidden. Default is `no`.

type	name	bind::ct:footer.hidden
select_one animal	animal	no
◀ ▶ **survey**	choices	settings ○

qml

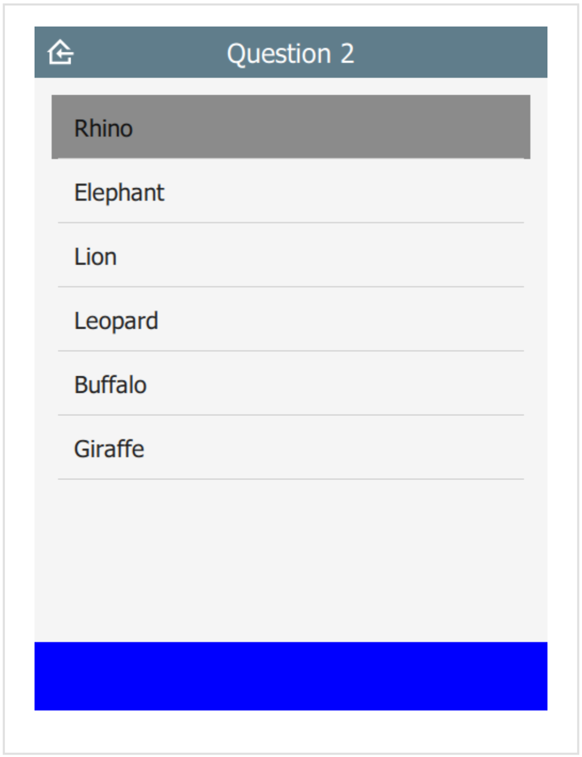
A QML fragment to use instead of the built-in footer. See Developer section. For example:

type	name	bind::ct:footer.qml
integer	animal_count	qml fragment
◀ ▶ **survey**	choices	settings ○

To set the footer to a blue rectangle, replace `qml fragment` above with the following:

```
import QtQuick 2.15

Rectangle {
    color: "blue"
    height: 64
}
```



qmlBase64

Base64 encoded QML (see **qml** above).

qmlFile

Name of a QML file which exists alongside other project files. This is not supported on ODK or KoBoToolbox, but can be used in Survey123.

Save

When the user presses the **Save** button, this triggers the save behavior.

snapLocation

Setting `snapLocation` to the name of a `geopoint` question will create a popup to acquire the GPS location. This feature is only active when `wizardMode` is enabled.

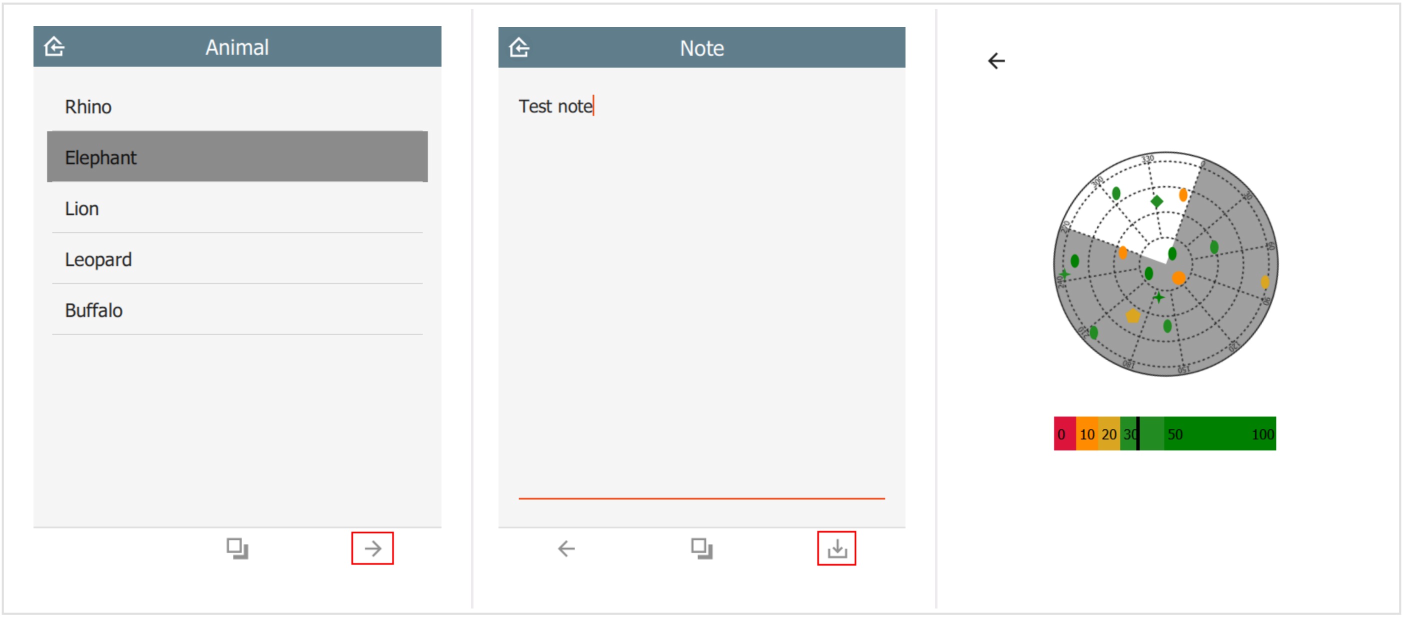
type	name	label
geopoint	f_location	Location
select_one animal	f_animal	Animal
text	f_note	Note

◀ ▶ **survey** choices settings

title	bind::ct:save.snapLocation
My form	f_location

◀ ▶ survey choices **settings**

In this example, the user flow will be:



targets

In the example below, the user will be presented with a popup containing the choices **Restart** or **Another**. After the sighting is saved, a new sighting will be created starting at the targeted question. All prior question data will be replicated into the new sighting.

This value must be a valid JSON array.

type	name	label
select_one animal	f_animal	Animal
select_multiple behavior	f_behavior	Behavior
text	f_note	Note

◀ ▶ **survey** choices settings ○

`restart` and `another` are taken from the `choices` sheet in the `saveTargets` list name.

list_name	name	label
saveTarget	restart	Restart
saveTarget	another	Another

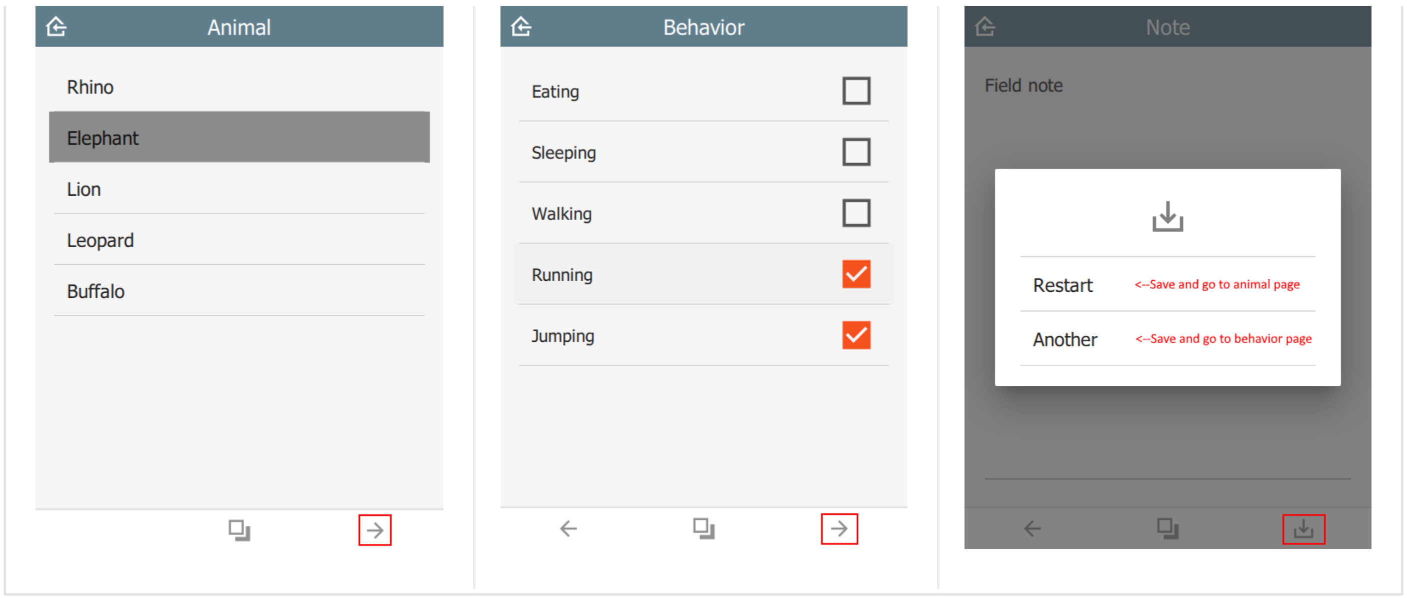
◀ ▶ survey **choices** settings ○

Note that `question` is the name of the targeted question in the `survey` table. If the question is not relevant, then this choice will be hidden.

title	bind::ct:save.targets
My form	[{ "choice": "restart", "question": "f_animal"}, { "choice": "another", "question": "f_behavior"}]

◀ ▶ survey choices **settings** ○

Note that if `immersive` is set to false, then the target list will automatically contain the **home** button. This option will save and return to the **Home** page without automatically creating a new sighting.



track

In the example below, there is a `select_one` question called `f_track` with choices `start`, `stop` and `nochange`. When the user presses **Save**, the track timer is adjusted depending on which choice was selected. The values in `updateIntervalSeconds` and `distanceFilterMeters` are the new track settings.

This value must be a valid JSON array.

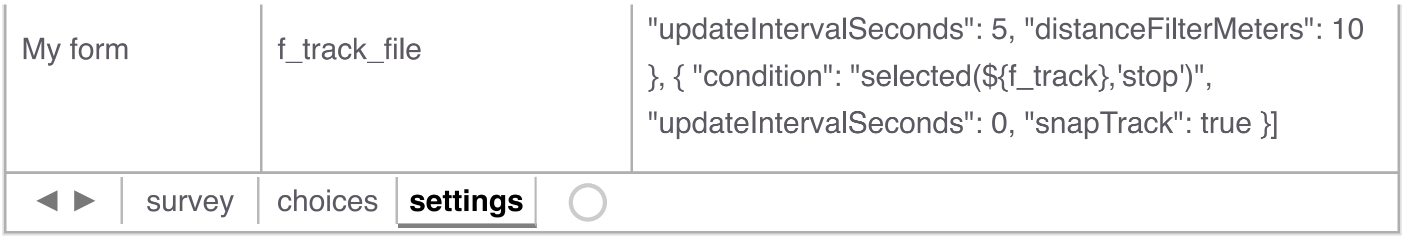
type	name	label
file	f_track_file	
select_one track_items	f_track	Configure track
text	f_note	Note

◀ ▶ **survey** choices settings

list_name	name	label
track_items	start	Start
track_items	stop	Stop
track_items	nochange	No change

◀ ▶ survey **choices** settings

title	bind::ct:save.trackFile	bind::ct:save.track
		[{ "condition": "selected(${f_track}, 'start')",



My form	f_track_file	"updateIntervalSeconds": 5, "distanceFilterMeters": 10 }, { "condition": "selected(${f_track},'stop')", "updateIntervalSeconds": 0, "snapTrack": true }]

◀ ▶
survey
choices
settings

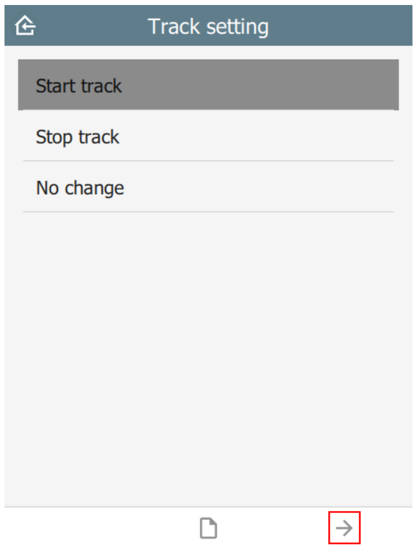
`condition` is an XlsForm expression which activates this option if matched, e.g. ${start_stop}='start'. Check out the ODK Form Logic documentation.

`updateIntervalSeconds` is the number of seconds between GPS readings. Set to 0 to disable the track timer.

`distanceFilterMeters` is the minimum distance between readings in meters. This is optional and by default no distance filter is used.

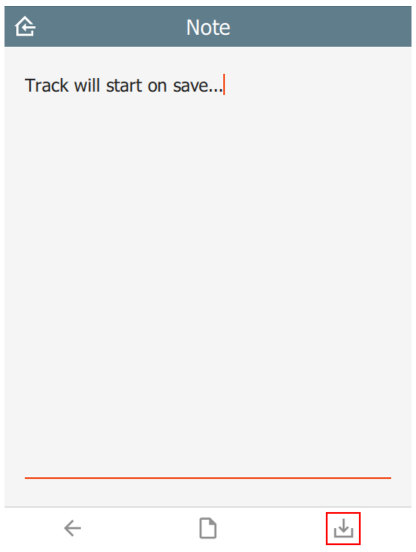
`snapTrack` causes the system to snapshot all the track points (since prior snap).

If using Survey123 and `esriLocationServiceUrl` is specified, then the track data will be sent to the feature service. Otherwise, a track file will be created and added to a `file` type question in the form. The question selected must be of type `file` and should have a `trackFileFormat` column specified.



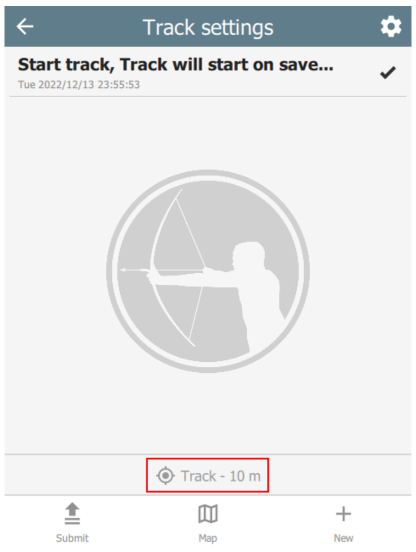
Track setting

Start track
Stop track
No change



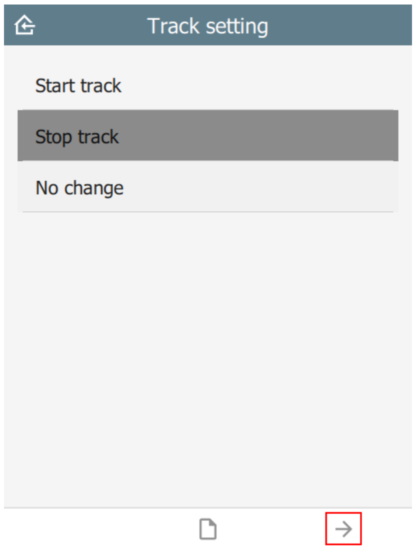
Note

Track will start on save...



Track settings

Start track, Track will start on save...
Tue 2022/12/13 23:55:53

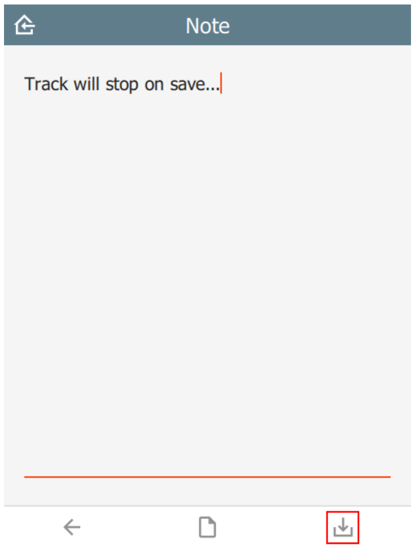


Track setting

Start track

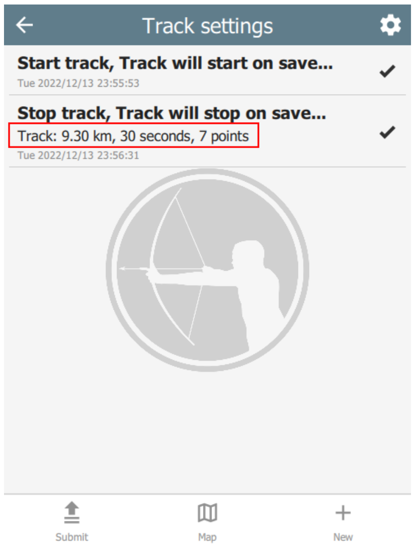
Stop track

No change



Note

Track will stop on save...



Track settings

Start track, Track will start on save... ✓

Tue 2022/12/13 23:55:53

Stop track, Track will stop on save... ✓

Track: 9.30 km, 30 seconds, 7 points

Tue 2022/12/13 23:56:31

Offline maps

What is an offline map?

Offline maps are map layers that are installed in CyberTracker. They can be used on the Map page. With the exception of WMS layers, they do not require a network connection.

Packages

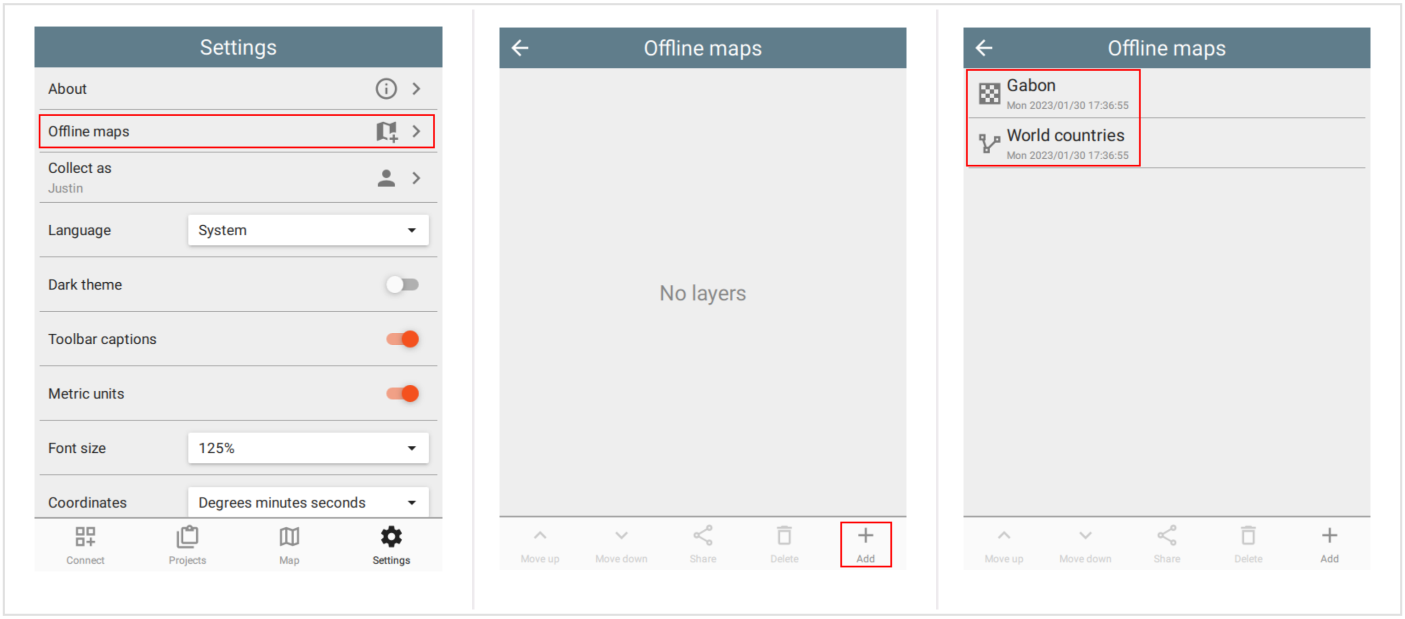
An offline map package is a zip file containing one or more layer files. See this sample file. Note that the `layers.json` file is optional - by default the system will automatically discover files with supported extensions. Many map layers require several files with the same base name, for example shape files require a .shp, .shx, .dbf and .prj file. These should all be in the base directory of the zip file.

Package installation

On **desktop**, a map package can be installed using `Install package` from the `File` menu.

On **mobile**, CyberTracker registers as a handler for zip files. When opening a zip, a prompt is displayed asking which app to open the file with. Select CyberTracker and the map will be installed. This is useful, because it is possible to send people a link via email or SMS.

Maps can also be installed directly from the `Offline maps` page. This can be reached via `Settings` or the gear icon on the Map Layers page:



Settings

About	
Offline maps	
Collect as	Justin
Language	System
Dark theme	off
Toolbar captions	on
Metric units	on
Font size	125%
Coordinates	Degrees minutes seconds

Offline maps

No layers

Offline maps

Gabon
Mon 2023/01/30 17:36:55

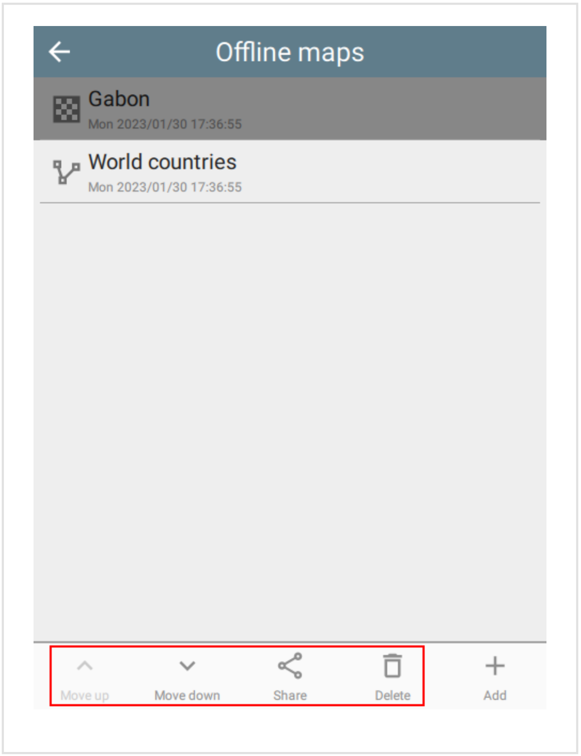
World countries
Mon 2023/01/30 17:36:55

Layer order and opacity

CyberTracker will discover and install layers in a zip file automatically. While this is often acceptable, when there are multiple layers, it is useful to specify the order and opacity of each. To do this, add a `layers.json` file to the zip and specify each of the layers:

```
[
    {
        "filename": "Gabon.mbtiles",
        "name": "Gabon",
        "active": true,
        "opacity": 1.0
    },
    {
        "filename": "Country.shp",
        "name": "World countries",
        "active": true,
        "opacity": 0.5
    }
]
```

Note that the Settings page for offline maps also supports re-ordering the layers, sharing with others and even deleting them:

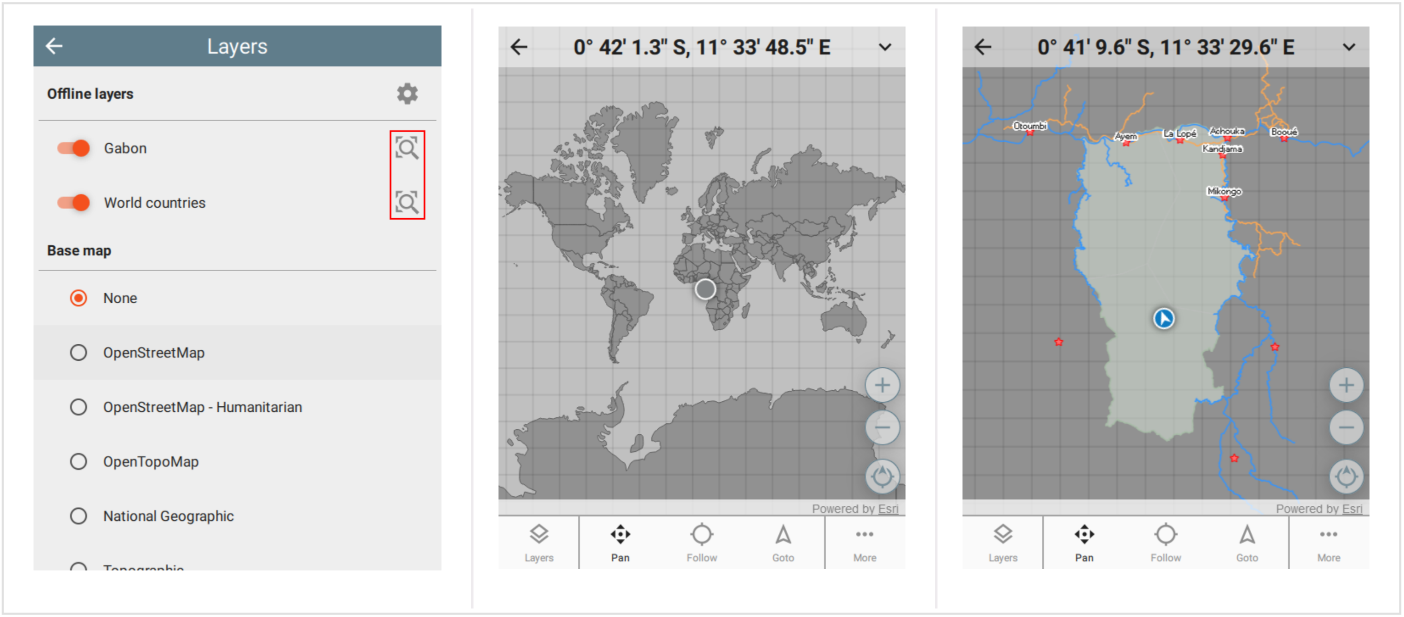


Sharing

Offline map packages can be shared to other devices. This shares the entire original package, not just the selected layer.

Zoom to layer

On the Map Layers page, selecting the `Zoom to` button will zoom to the entire extent of the layer.



Supported formats

The following layer formats are supported:

- ESRI formats: shapefile (shp), tile package (tpk), vector tiles package (vtpk)
- ASRP/USRP
- CIB1, 5, 10
- DTED0, 1, 2
- GeoTIFF
- HFA
- HRE
- IMG
- JPEG
- JPEG 2000
- NITF
- PNG
- RPF
- SRTM1, 2

- Mosaic Dataset in SQLite (read-only)
- MapBox: mbtiles
- Google: KML
- GeoJSON

WMS layers

Web Map Service is an online layer protocol. While these layers are actually online, they can be added as layers using the Offline map system. To do this, create a JSON file with the extension `.wms` and add it to the package zip file. For example:

```
{
    "layer": "0",
    "service": "https://basemap.nationalmap.gov/arcgis/services/USGSHydroCached/MapServer
}
```

Miscellaneous

fixCount

For `geopoint` question types, the number of skipped readings before a fix is taken. The default value is 4. Some GPS devices return old readings before real readings. To overcome this, setting the `fixCount` will cause the system to require several readings before the final location is taken.

type	name	parameters
geopoint	f_location	fixCount=4

◀ ▶ **survey** choices settings ○

track file format

When the user presses **Save** and creates a track file, it is stored in a `file` field specified in the `settings` sheet in the `bind::ct:save.trackFile` column.

By default the format of the track is zipped geojson, but this can be changed by using the `format` parameter of the question itself. Supported values are `geojson` and `kmz` (not supported on Survey123).

Survey123 users should prefer to use a location service - see esriLocationServiceUrl. If a location service is specified, this question should be removed.

type	name	appearance	parameters
file	f_track_file	hidden	format=kmz

◀ ▶ **survey** choices settings ○

title	bind::ct:save.trackFile
My form	f_track_file

◀ ▶ survey choices **settings** ○

Developers

Introduction

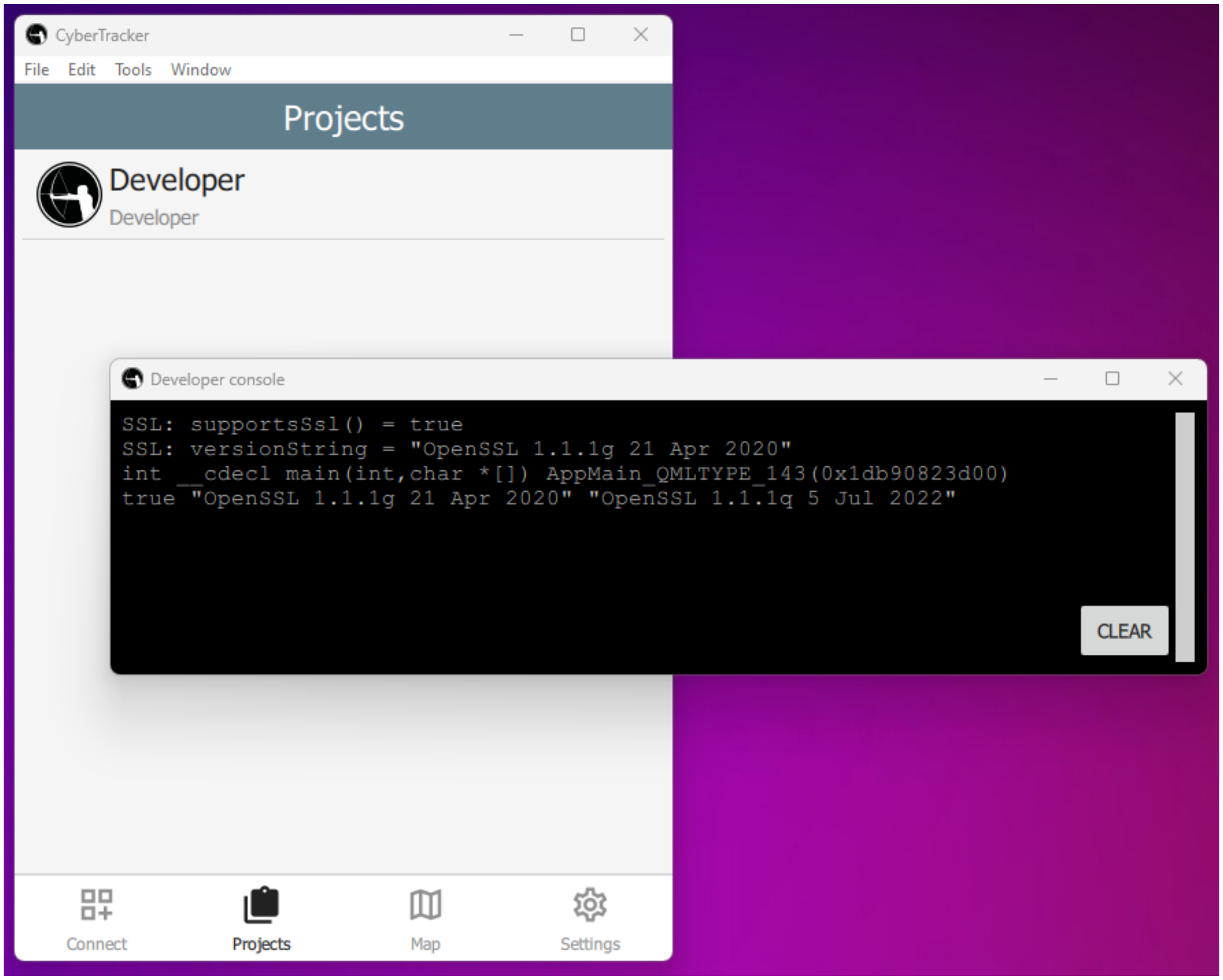
CyberTracker is built on the Qt Framework. The Qt user-interface language is called QML and it provides a concise way to describe components and layouts. The scripting language is Javascript.

QML fragments can be added to an XlsForm. This enables a high degree of customization beyond what is already available. In particular it allows custom widgets and layouts to be used in data entry.

Setup

Install CyberTracker on your desktop computer by following the instructions on the Download page.

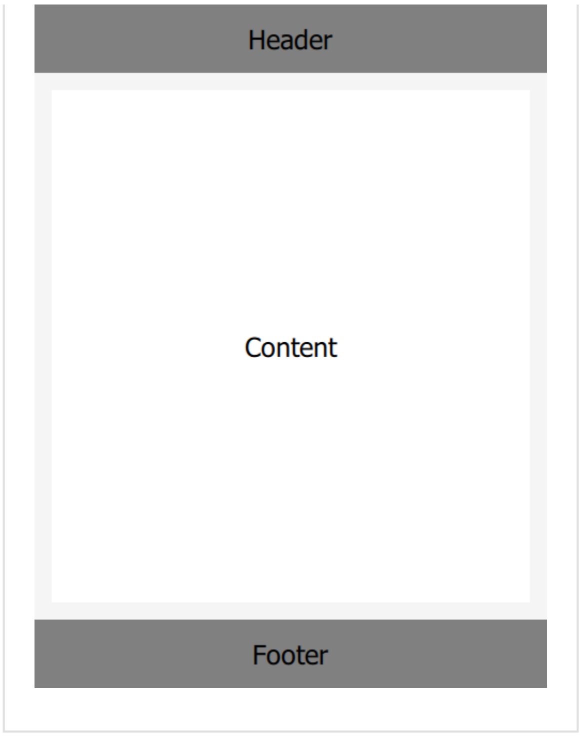
After launching, open the **Window** menu and select **Toggle developer console**.



```
SSL: supportsSsl() = true
SSL: versionString = "OpenSSL 1.1.1g 21 Apr 2020"
int __cdecl main(int,char *[]) AppMain_QMLTYPE_143(0x1db90823d00)
true "OpenSSL 1.1.1g 21 Apr 2020" "OpenSSL 1.1.1q 5 Jul 2022"
```

Page layout

Each question in an XlsForm is given one page in the UI. The page is divided into 3 segments: header, content and footer:



Each can host QML and these are specified in the `bind::ct:header.qml`, `bind::ct:content.qml` and `bind::ct:footer.qml` columns. Note that you may also use `qmlFile` (file alongside project files) or `qmlBase64` (base64 encoded QML).

recordUid and fieldUid

XlsForm question values are identified by their `recordUid` and `fieldUid`.

`recordUid` uniquely identifies the current record. For simple forms there is only record per sighting, but using repeats and groups, multiple records will be created.

`fieldUid` uniquely identifies the question within a form. It comes from the `name` column of the `survey` sheet.

Given the following form:

type	name	label	bind::ct:content.qmlFile
text	my_field_name	My field name	test.qml

◀ ▶ **survey** choices settings

And the following test.qml:

```
import QtQuick 2.15
```

```
Item {
    property string recordUid
    property string fieldUid

    Component.onCompleted: {
        console.log("recordUid = " + recordUid)
        console.log("fieldUid = " + fieldUid)
    }
}
```

The developer console will output something like:

```
recordUid = 7f1ed933401b43878fee6f0d38c7f92a
fieldUid = my_field_name
```

Setting form values

Form values can be changed using a FieldBinding component. This enables change notifications so that the Label will automatically update when the button is clicked.

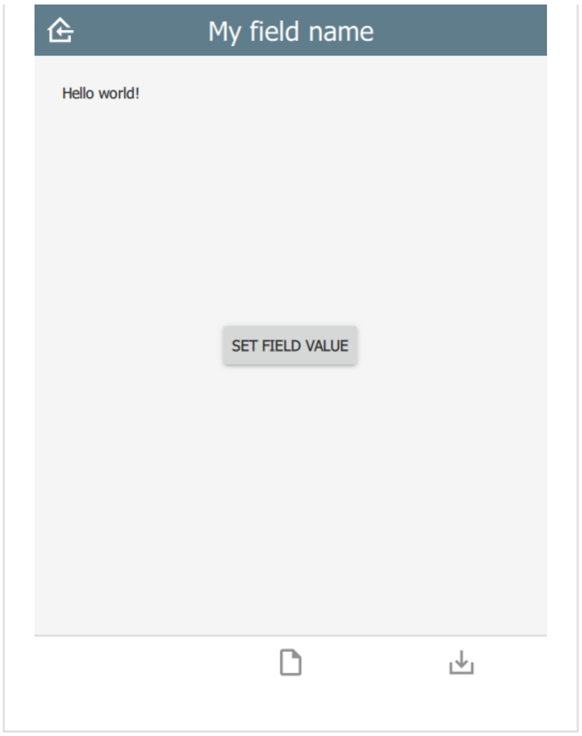
```
import QtQuick 2.15
import QtQuick.Controls 2.15
import CyberTracker 1.0 as C

Item {
    property alias recordUid: fieldBinding.recordUid
    property alias fieldUid: fieldBinding.fieldUid

    C.FieldBinding {
        id: fieldBinding
    }

    Label {
        x: 10
        y: 10
        text: fieldBinding.value
    }

    Button {
        anchors.centerIn: parent
        text: "Set field value"
        onClicked: {
            fieldBinding.setValue("Hello world!")
        }
    }
}
```



The console window will give an error of the form: `Unable to assign [undefined] to QString` when first launched, because `fieldBinding.value` is initially undefined. This is generally harmless, but can be removed by checking for undefined:

```
Label {
    x: 10
    y: 10
    text: fieldBinding.value || ""
}
```

Frequently Asked Questions

Which backends support XlsForm?

CyberTracker supports ODK Central, KoBoToolbox and Survey123.

Are CyberTracker extensions visible to other tools?

XlsForm extensions support custom columns by using the `namespaces` value in the `settings` sheet. Columns prefixed with `bind::ct:` are only used by CyberTracker and are ignored (but preserved) by other tools.