Advanced XLSForm Techniques
Brett Stokes, James Tedrick, Philip Wilson
Agenda

- Creating Editing Workflows with Survey123
- Using Custom JavaScript Functions
- Polygon and Line Creation
- Creating Selectable Images using image-map
- Working with Photos in Survey123
Survey123 Connect

Define your form in Excel using XLSForms syntax

Preview your XLSForm in Connect when happy… publish
Creating Editing Workflows with Survey123
Our Workflows

- Asset Inspection
- Approving Incoming Data

Key Concepts
- Structure data to facilitate the workflow
- Update existing features
- Use multiple forms with one feature service
Create Surveys from Existing Feature Services

- Sign into a portal (ArcGIS Online/ArcGIS Enterprise)
- Select a feature service
  - Survey123 Connect scans and creates an XLSForm
- Edit the form to provide additional logic, create a form for just a repeat layer, remove questions

- Publish

- Your XLSFile will be automatically set with:
  - Settings: submission_url
  - Settings: Form_ID
Edit Existing Survey Data

Survey123 Connect

- Add the **Inbox** option to the survey
  - Optional: can apply an expression to filter data
- This capability must be enabled on the survey BEFORE it is published
- **Editing Existing Data with Survey123 video**
Edit Existing Survey Data – the Inbox

Survey123 Connect

- You can query on fields and profile information, ex:
  - status='pending'
  - assigned=${username}
Field App Inbox folder

- Submitted surveys retrieved from a list or map view
  - Surveys returned limited by max record count setting of feature service
- Select a survey to edit responses
Asset Inspection

- Prefer creating 2 tables
  - Asset table – stores overall & important lifecycle information
  - Inspection table – stores each inspection event
- Tables can be related in the geodatabase (1:M relationship)
  - Survey123 uses GUID-based relationships by default
  - Non-GUID can be accommodated via multiple-application workflows
Asset Inspection – 2 types of forms

• Asset + Inspection
  - Survey123 Connect creates by default
  - Use the Inbox to load the assets in
  - Can carry over inspection history

• Inspection Only
  - Doesn't require Inbox
  - Doesn't require GUID-based relationship
  - Can be launched from other applications
DEMO
Approving Incoming Data

- Easiest via multiple forms
  - Submittal
  - Review/Editing
  - Approval

- Fields can be hidden, removed or made read-only at each stage
- Add 'status' or other metadata fields to chart the progress
Multi-stage forms

- Start with a table with all the questions used across the stage – makes sure you have all the fields you need
- Create a form for each stage using the 'all' feature service
Create Stages

- First form – include the 'status' fields as hidden to populate on submission
- Use the Inbox to only show those that need to be viewed at a stage
- Expose the status fields to allow the form to go to the next stage
- Use stages in another product (Dashboard, Web AppBuilder): create feature layer views using the status fields to restrict
## Feature Service Form Settings

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>form_title</td>
<td>form_id</td>
<td>instance_name</td>
<td>submission_url</td>
</tr>
<tr>
<td>Water_Stations_editing</td>
<td>Water_Stations</td>
<td></td>
<td><a href="https://www.arcgis.com/sharing/rest/content/items/717f6b4ff024272ab6027776bfdf6ec0">https://www.arcgis.com/sharing/rest/content/items/717f6b4ff024272ab6027776bfdf6ec0</a></td>
</tr>
</tbody>
</table>

- **Submission_url** – the service item’s ARCGIS REST API entry

- **Form_id** – The name of the table the top level of the form will submit data to

- These are found automatically when creating a survey from an existing Feature Service
Identifying Records – Instance name

- Instance name setting enables field users to uniquely identify survey form Drafts, Outbox, Sent and Inbox folders in the Survey123 app.

### survey

<table>
<thead>
<tr>
<th>type</th>
<th>name</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>reportdate</td>
<td>When was this assessment created?</td>
</tr>
<tr>
<td>text</td>
<td>locationname</td>
<td>What is the name of the location?</td>
</tr>
<tr>
<td>text</td>
<td>assignedto</td>
<td>Assigned To</td>
</tr>
</tbody>
</table>

### settings

<table>
<thead>
<tr>
<th>form title</th>
<th>form id</th>
<th>instance name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Damage Assessment</td>
<td>concat($assignedto, &quot;, &quot;#$locationname, &quot;, &quot;#reportdate,#%m-%d-%Y, #%H:%M')</td>
<td></td>
</tr>
</tbody>
</table>
Pulldata() JavaScript Functions
**Pulldata(@javascript) function** (Beta feature)

- The `pulldata()` function has been extended to work with your own JavaScript (JS) functions.

- An advanced technique that can help you introduce logic beyond what is possible through standard XLSForm syntax.

- Invoke your own JS functions to return values and you decide what occurs within your JS function.

- The syntax for using your own JS functions with pulldata is as follows:
  - `pulldata("@javascript","yourJSFile.js","yourFunction","parameter1","parameter2")`

- Custom JS files must be stored in a folder called 'extensions' within the survey directory.

- Add as many function parameters as needed. In the example above, the JS function is expecting 2 parameters, but you could have more or less as needed by your own function.
**Pulldata(@javascript) with repeats**  
(Beta feature)

- A useful application is the ability to pass all the values from across all repeat records into your JS functions.

- The pulldata("@javascript", ...) function supports two different styles of implementation for retrieving the values from a repeat.

  1. Repeat name - if a repeat name is passed in, the JS function will receive the parameter as an array of rows of all the values in the repeat.
     - pulldata("@javascript", "yourJSFile.js", "yourFunction", ${repeat1}, "question1")

  2. Question from within a repeat - if a question from within a repeat is passed in, the JS function will receive an array of values from the specified question.
     - pulldata("@javascript", "yourJSFile.js", "yourFunction", ${question1})
Other *pulldata(@javascript)* properties (Beta feature)

- When using JS functions to access web services and secure ArcGIS services, some properties are not available through standard XLSForms. For this reason, we have also extended the collection of *properties* you can obtain through the *property()* function as follows:

  - *property('propertynname')*
    - portalurl - URL of ArcGIS Portal
    - Token - ArcGIS token
    - online - Boolean value indicating if device has network connectivity.
    - language - Language in use by current survey.
    - locale - Locale object in use for current survey. Only useful for JavaScript functions.
    - localeinfo - AppStudio LocaleInfo object for current survey. Language code reported in various notations.
    - utcoffset - Offset in minutes from UTC for the local timezone.
      - Note: This can also be done in JavaScript via var utcOffset = - new Date().getTimezoneOffset
    - timezone - Timezone code.


Known limitations (Beta feature)

• You cannot use DOM.

• You cannot use frameworks such as JQuery, Ember, Angular, etc.

• You cannot access local files.

• You cannot make asynchronous calls.

• You must be signed in.

• The survey must originate from the same ArcGIS organization as the signed in user.
Fun with shapes
Building Line / Polygon geometry from repeated geopoints
Geotrace (Line) & Geoshape (Polygon / Area) question types

- Starting at the 3.6 release, your surveys can include line and/or polygon geometry.
- Simply add a ‘geotrace’ or ‘geoshape’ question to your XLSForm.
- Can use these surveys in field app or web app (but only publish from Connect).
  - support for adding line / area questions in web designer is coming in 3.9 release.
- Only one ‘parent’ geometry type for your survey (as per all esri feature layers).
- Can include other geometry types if within a repeat (stored as related features).
Collecting Lines / Polygons

- **Capture Methods**: sketch or vertex-by-vertex
  - Specified in `body::esri:style` column of XLSForm (method=vertex or method=sketch).
  - Default is sketch, unless survey based on existing feature layer, then default is vertex.
- **Digitize on screen.**
- **Collect vertices at GPS location in field app only (location streaming in future release).**
Capture Methods

**Sketch**

- Fast, single take ie one ‘pen stroke’.
- Option to re-sketch or undo only (no edits).
- Smart shapes for regular geometries:
  - Ellipse, rectangle, triangle.

**Vertex**

- More control over geometry capture, vertices can be placed and edited with:
  - Screen clicks / taps, cross-hair, location sensor, geosearch bar.
- Entire geometry can be re-drawn
Use a `calculation` expression that returns a string of co-ordinate pairs.

- Each pair of co-ordinates describes a vertex of your line / polygon.
- Lat ‘space’ Long (in decimal degrees), each pair separated by a semicolon.
  - Eg. `string(concat(${vertex_1},";",${vertex_2},";",${vertex_3},";",${vertex_4},";",${vertex_5}));`

- You can use the `sum()` function to sum geopoints within a repeat.
  - Each geopoint is added to the line / polygon as a vertex in the order they are collected.
  - GNSS metadata can be collected for each vertex.
  - Currently limited to the Field App only.
DEMO:

- Edit existing Power Pole Condition Assessment survey.
- Add repeat around geopoint question.
- Add geotrace question with sum() function in calculation column.
- Publish survey and open in field app.
- Begin capturing repeated geopoints and watch as the line / polygon is constructed.
  - Note that GNSS metadata won’t be populated if geopoints are added by digitizing on screen.
More Information

- **Documentation**

- **Blog Post**
‘Image map’ appearance

Customise UI without any code
Image-map works with select* questions

- Customise the UI of your survey without any code.
- Enables fast and intuitive data collection with the user selections stored in the feature layer.
- Displays a Scalable Vector Graphic (SVG) image to interact with.
- SVGs can be edited / created to work perfectly with your survey.
- SVGs can have raster images (eg JPG & PNG) embedded.
- Can add a ‘big button’ experience to your survey very easily.
- Currently not supported in web designer / web form.

<table>
<thead>
<tr>
<th>Likelihood</th>
<th>Insignificant</th>
<th>Minor</th>
<th>Moderate</th>
<th>Major</th>
<th>Severe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Almost Certain</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
<td>Extreme</td>
<td>Extreme</td>
</tr>
<tr>
<td>Likely</td>
<td>Medium</td>
<td>Medium</td>
<td>High</td>
<td>Extreme</td>
<td>Extreme</td>
</tr>
<tr>
<td>Possible</td>
<td>Medium</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
<td>Extreme</td>
</tr>
<tr>
<td>Unlikely</td>
<td>Low</td>
<td>Medium</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Rare</td>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
</tr>
</tbody>
</table>
Getting Started with image-map

1. Download / Create / Edit your SVG (more details to follow).
2. Add a `select_one` or `select_multiple` question to your XLSForm.
3. Create the choice list as per normal.
4. Add the ‘image-map’ appearance to your select* question.
5. Add the SVG filename to the ‘media::image’ column.
6. Copy the SVG file to your survey’s media folder.
SVG File format

- An SVG file is an XML-based vector image format for two-dimensional graphics.
- Can be easily edited in any text editor, but using a vector graphics package will assist.
- They have support for interactivity and animation.
- SVGs can have raster image formats such as .png and .jpg embedded within them.
- SVG `<path>` elements with an ‘id’ value are required to work with Survey123.
- Each `<path>` element with ‘id’ value becomes a clickable part of your survey.
- `<path id>` values match the ‘name’ value in your `select_one / select_multiple` choice list.

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
     width="360" height="360" viewBox="0 0 100 100">
  <rect x="0" y="0" width="100" height="100"></rect>
  <path id="beginner" d="M 5.4998702,5.4998718 H 129.50013 V 194.50013 H 5.4998702 Z"/>
  <path id="intermediate" d="M 135.48042,5.4804192 H 259.51959 V 194.51958 H 135.48042 Z"/>
  <path id="advanced" d="M 265.48041,5.4804192 H 389.51956 V 194.51958 H 265.48041 Z"/>
  <path id="expert" d="M 395.48041,5.4804192 H 519.51956 V 194.51958 H 395.48041 Z"/>
</svg>
```
Minimum SVG requirements

- **Must contain a <path> element that has an id value.** Non-<path> elements such as <rect>, <ellipse>, <circle>, <image> and <text> will not be clickable. Likewise, <path> elements without an id value will not be clickable.

- **No empty groups.** Having empty groups ie <g> </g> within your SVG makes all <path> elements become un-clickable.

- **Can only have 1 group per nest level.** You can have as many nested levels as you like, but if you have more than one group at a particular level (including the parent level), all of the <path> elements become un-clickable.

- **SVG height and width (Viewport) must be defined.** While your image will likely display properly without these, your clickable areas may become offset from the underlying image.

- **Viewbox (if specified) must match Viewport.** You don't need to specify a 'viewBox' parameter in your <svg> element. However, if you do specify a viewBox that differs from your height and width values, your clickable areas will be offset from the underlying image.
• Open vector graphics editor (Illustrator, Inkscape, online SVG editors like Vectr, BoxySVG and DrawSVG).
• Import the raster image and resize page to selection (ensure scale=1)
• Create a simple object (eg ellipse) over one of the circles.
• Convert object to path and assign <path id> value
• Copy / Paste to cover all selectable areas.
• Save SVG to survey media folder and open in text editor to view contents.
• Open XLSForm with select_multiple question.
• Add image-map appearance ro select_multiple question and SVG filename to media::image column.
• Refresh survey in Connect and view.
More Information

• Documentation

• Blog Post

• ‘Image-map’ Sample in Survey123 Connect
  ![Image-map Sample in Survey123 Connect](survey123-image-map-sample.png)
Watermarks
Photo Watermarking
Watermark expressions in XLSform

watermark=ASSET ID: 44087\n@[latitude longitude:ddm]\nCompass
@[compass]&size=25&color=red&outlineColor=white
watermark= - Used in the bind:esri::parameters column to determine which information will be displayed on the image.

ASSET ID: 44087 - freestanding text.

@[latitude longitude:ddm] - the @[ ] notation is used to define a placeholder.

\n – used to define line breaks

& - used to separate parameter properties

Other properties used in this example are size, color, and outline color.

Refer to the Survey123 documentation for full list.
Watermark Logos and Placement

• Add a logo or image to the watermark by placing a .png file in the media folder of the survey:
  • image=esri.png&imageSize=80
  • Image size associated with logo can be set using the ‘imageSize=<pixel size> format

• Draw multiple watermarks on the same photo. The watermark expression should include a space between the beginning of each watermark placement parameter

Example: topLeftWatermark=esri®&image=esri.png&imageSize=80 bottomRightWatermark=“ASSET ID: 4453
Lat @[latitude:dms]
Lon @[longitude:dms]&color=yellow”
Watermark considerations

• The watermark placeholder starts with @[ ]
  • unique to watermarks
  • different to the ${ }$ placeholder

• When a watermark expression includes a space, it must be wrapped in quotations:
  • watermark=@[time] does not require quotations
  • watermark="@[latitude longitude:dms]" does require quotations

• Formatting can be applied to placeholder values:
  • @[longitude:ddm] can be specified to display the longitude in degree decimal minutes.

• To reference values from previous questions, the expression should be constructed in the calculation column of a separate question (note or calculate)

• The watermark expression can then reference the calculated value from that question using the $${ }$$ format.
DEMO
Image Annotation
Image Annotation and Smart Sketching

• Using a standard image question you can draw or annotate on an image to provide additional information in your survey.

• appearance = draw - the image question will display a draw button which loads a blank canvas on which you can sketch.

• appearance = annotate - the image question will display a button to open a canvas. Annotate provides options to use an image from the camera, an existing image from the gallery or a screen shot of a map.

• Survey123 includes a smart sketching tool to generate shapes and add text annotations to the sketch.

• The canvas (saved image) will be the same size/dimension as the device’s screen resolution.
Defaults and calculations

- A default image can be used to set an image as the canvas background. This is done by referencing an image in the default column in the media folder of the survey:
  - default=turtle.jpg

- A calculation can be used to dynamically set the default image using an expression in the calculation column:
  - concat(${animal},".png")

- Defaults and calculations behave differently with image questions using draw and annotate:
  - Using draw, the default image can not be changed by the user.
  - Using annotate the user can change the image by capturing a new photo, selecting an image from gallery, or inserting a map.
DEMO
Pulldata() EXIF
Extract EXIF metadata from images

- Photos typically store certain metadata within the image file

- This information includes things like:
  - Date when the photo was taken
  - Location (and direction) where the photo was taken
  - Camera/Device used to take the photo
  - And over 200 more tags...

- All image metadata is stored following the EXIF standard

- Pulldata ("@exif") allows you to extract this information and use it for storage, constraints etc
XLSForm Syntax

pulldata("@exif", <image>, <property>)

decimal lat Latitude pulldata("@exif", ${image}, "gpslatitude")
decimal lon Longitude pulldata("@exif", ${image}, "gpslongitude")
decimal alt Altitude pulldata("@exif", ${image}, "gpsaltitude")
Extracting data from EXIF blog post
Summary

- Creating Editing Workflows with Survey123
- Using Custom JavaScript Functions
- Fun with shapes – Line and Polygon Creation
- Creating Selectable Images using image-map
- Working with Photos in Survey123