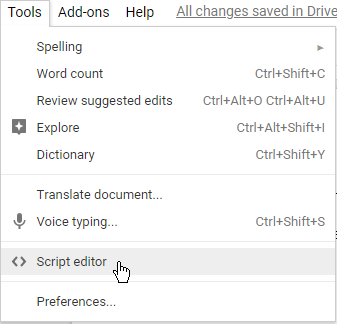
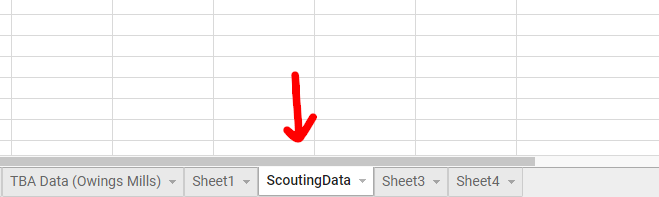
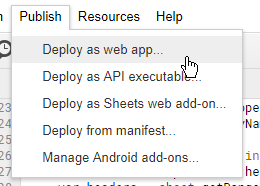
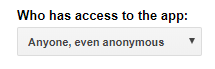
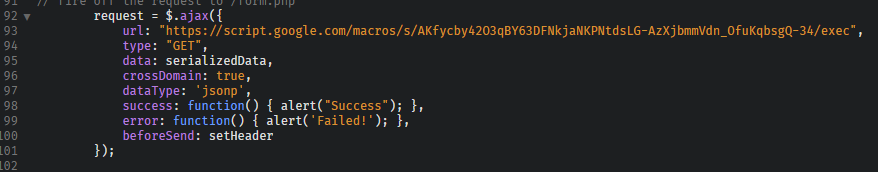
How to set up a google sheet to receive scouting data:

1. Make a google sheet.
2. Go to Script editor under the Tools menu. 
3. Copy this code into the editor:

|  |
| --- |
| function doGet(e){  return handleResponse(e);  }  // Enter sheet name where data is to be written below  var SHEET\_NAME = "[your sheet name here (no spaces)]";  var SCRIPT\_PROP = PropertiesService.getScriptProperties(); // new property service  function handleResponse(e) {  // shortly after my original solution Google announced the LockService[1]  // this prevents concurrent access overwritting data  // [1] http://googleappsdeveloper.blogspot.co.uk/2011/10/concurrency-and-google-apps-script.html  // we want a public lock, one that locks for all invocations  var lock = LockService.getPublicLock();  lock.waitLock(30000); // wait 30 seconds before conceding defeat.    try {  // next set where we write the data - you could write to multiple/alternate destinations  var doc = SpreadsheetApp.openById(SCRIPT\_PROP.getProperty("key"));  var sheet = doc.getSheetByName(SHEET\_NAME);    // we'll assume header is in row 1 but you can override with header\_row in GET/POST data  var headRow = e.parameter.header\_row || 1;  var headers = sheet.getRange(1, 1, 1, sheet.getLastColumn()).getValues()[0];  var nextRow = sheet.getLastRow()+1; // get next row  var row = [];  // loop through the header columns  for (i in headers){  if (headers[i] == "Timestamp"){ // special case if you include a 'Timestamp' column  var date=new Date();  row.push(Utilities.formatDate(date, 'America/New\_York', 'MMMM dd, yyyy HH:mm:ss Z'));  } else { // else use header name to get data  row.push(e.parameter[headers[i]]);  }  }  // more efficient to set values as [][] array than individually  sheet.getRange(nextRow, 1, 1, row.length).setValues([row]);  // return json success results  return ContentService  .createTextOutput(JSON.stringify({"result":"success", "row": nextRow}))  .setMimeType(ContentService.MimeType.JAVASCRIPT);  } catch(e){  // if error return this  return ContentService  .createTextOutput(JSON.stringify({"result":"error", "error": e}))  .setMimeType(ContentService.MimeType.JAVASCRIPT);  } finally { //release lock  lock.releaseLock();  }  }  function setup() {  var doc = SpreadsheetApp.getActiveSpreadsheet();  SCRIPT\_PROP.setProperty("key", doc.getId());  } |

1. Replace [your sheet name here (no spaces)] with the name of the sheet (no not the whole spreadsheet, the little tab at the bottom:)
2. Make the sheet match the index.html file of the site by putting each field’s name into the first row of the column you want it to be in (for instance make A1 scouterName (if that’s what you named your scouter name field) to have the A column fill up with scouter names as they come in). Keep in mind that every field has to be represented somewhere in the sheet, or the site won’t work!
3. In the Script editor, click **Publish** and then **Deploy as web app...**. 
4. Make sure to set **Who has access to the app** to **Anyone, even anonymous**. 
5. After deploying, copy the resulting URL.
6. Pull up your app.js file and replace the URL shown here (line 93) with the one you just copied.
7. The sheet should work once you push all changes to your server-side site.
8. Any questions? Email me, or comment on this.

How to set up Your Custom Form:

1. Open the index.html file.
2. To delete a field:
   1. Delete everything between the <div class = “medium 2 cell” (or whatever it is)> to the corresponding </div>.
3. Figure out which field you want to add (a number field with + and - buttons, a text field, a switch, some radio buttons, etc)
4. Insert the corresponding template. Replace columnName with your column name (duh) (no spaces) etc. Make sure you put it in the right place! It should be after the corresponding </div> for a <div class = “medium 2 cell” (or whatever)>
5. Templates:

**Switch:**

<div class="large-3 medium-4 cell">

<div class="switch">

<label class='color-dark'>Label</label>

<input class="switch-input" value="yes" id="columnName" type="checkbox" name="columnName">

<label class="switch-paddle" for="columnName"> </label>

</div>

</div>

**Number field (with + and - buttons):**

<div class="large-3 medium-4 cell">

<div class="switch">

<label class='color-dark'>Label</label>

<div class='flex'>

<button onclick="countDown('columnName')" class='button' type='button' id='columnName'/>-</button>

<input id='columnNameVal' type='number' value='initialValue' name='columnName' class='val' min='minimumNumber'>

<button onclick="countUp('columnName')" class='button' type='button' id='columnName'/>+</button>

</div>

</div>

</div>

**Text field:**

<div class='auto cell'>

<label>Label

<textarea name="columnName" placeholder="(hint text (grey text that shows up when nothing has been entered yet))" rows='amountOfLines' cols='widthInCharacters'></textarea>

</label>

</div>

**Dropdown Menu:**

<div class="medium-2 cell">

<label>Label</label>

<select name='columnName'>

<option value="" disabled selected>defaultOption</option>

<option value='firstOption'>firstOption</option>

<option value='optionTwo'>optionTwo</option>

.

.

.

</select>

</div>

**Small text field (for names etc):**

<div class="medium-4 cell">

<label>Label</label>

<input type="text" id='columnName' name='columnName' />

</div>

**Number field without + and - (for match number, score, etc):**

<div class="medium-2 cell">

<label>Match Number</label>

<input type="number" name='columnName' min='minimumNumber' step='1' />

</div>

Add anything you want to add to the floating bit above this comment and within the appropriate <div>:

Need more templates, or just having trouble? Email me at [miloszecket@gmail.com](mailto:miloszecket@gmail.com)

Databases

To incorporate your own MySQL database (if you have one), simply edit the app.php file to contain your MySQL server credentials, as well as line 117 of the app.js file to contain the URL to wherever you put app.php. The database you choose should have a table named results (or whatever, if you change it within app.php) with the same amount of columns as fields in your form, and they should all have the right names. I am currently trying to update the php to automatically create such a table, and will update this when I am finished.