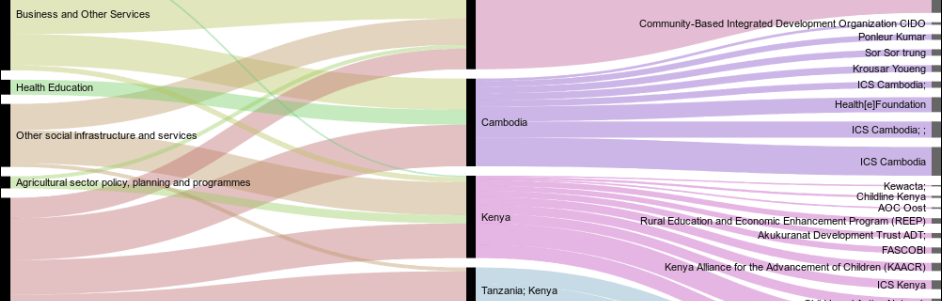
Recipe: Lasagna ICS

Recipe: Lasagna ICS

Showing ICS budget flows using IATI data

# Begin with the end in mind

Think of what you’d like to show, what kind of data do you need to make that possible and check if you have the right data.

In this case we want to show the sectors and countries in which ICS is active, and how the budgets flow between sector and country.

# Get your ingredients (the data)

We need to check if ICS includes budgets and sectors in its IATI data set.

1. Go to the IATI Registry and download the ICS IATI file over 2013.

*It’s always a good idea to open the file in a text editor (for instance Notepad++), to see what’s in it.* [*http://notepad-plus-plus.org/*](http://notepad-plus-plus.org/)

We look at the data to see if there are elements like below – which is the case.

<budget>...</budget>

For a further analysis of the data, we’ll transform it to a table format (CSV) first.

1. Download EditiX free edition (<http://www.freexmleditorsite.com/download.html>) and install it.
2. Download XSLT transformations here: <https://github.com/IATI/IATI-XSLT> and unzip the library to a place where you can easily find it again.
3. Transform the file to CSV using XSLT:
   * Open EditiX
   * Open the ICS .xml file
4. Choose XSLT  Transform using XSLT
   * In the box ‘XSLT document’ browse to the library you’ve just unzipped, and on to templates  csv. Now choose iati-activities-xml-to-csv.xsl
   * Choose a location for the resulting file. Make sure to set the File type to ‘All files’ and end your filename with “.csv”
   * Click “OK”

And then we’ll open the data set in OpenRefine (*a power tool for working with messy data*)

1. Download and OpenRefine from <http://openrefine.org/download.html> and install it.
2. When you run Open Refine, it will open in your browser, and let you create a project by importing data from your computer. Open the .csv file you’ve just created in step 4 and click next.
   * Make sure you set the field "Character encoding" to UTF-8
   * Choose “Columns are separated by Commas (CSV)”
   * You will see a preview of the data. In the top-right corner, click “create project” to finish importing the data.
3. Now let’s analyse the data ICS has included in their IATI file. You can easily do this using facets and filters.
   * At the top of each column you’ll find an arrow pointing downward that offers you a range of powerful options to analyse and edit the data in that column.
   * For instance: go to the column with recipient-countries
   * Choose Facet  Text facet
   * On the left hand side, you’ll now see a box showing all the different values in this column

Notice there are 6 different values. There are blank values as well.

1. Looking through the data set we see a couple of things:
   * Some recipient-country entries contain multiple values (Tanzania; Kenia)
   * All activities have a budget
   * Budgets are all given in EUR
   * all activities contain sector information
2. In this visualization, we want to show flow links between sectors and countries. Given our analysis we’ll be able to show this. But we’ll make sure to include more fields so that we keep some more options open.
   * Flows should have a weight (which determines their size in the eventual visualization), based on project budget.

# Shaping & cleaning your data

Now we're going to shape the data, using our “power tool” OpenRefine.

1. First we’re going to reorder the columns in the data set. Go to the first column (All), click on the arrow and choose ‘Edit columns’  ‘Reorder / remove columns’
   * Reorder or remove a few columns and click “OK”
   * Notice how in the top-left corner the Undo / Redo tab now shows (1)
   * When you click the tab, you’ll see all the ‘transformations’ you’ve made so far. Here you can easily undo and redo your steps.
   * Also notice the buttons “Extract” and “Apply”. Here you have access to the code that OpenRefine produced to execute your transformation. You can copy this code and use it in future projects.
2. Go back to how the original data was by clicking on “0. create project”. The columns you removed are back again. Now click the “Apply...” button, and copy and paste the code below to reorder the columns in the data set, keeping the columns we want to use & save as extra information.

[

{

"op": "core/column-reorder",

"description": "Select columns we need",

"columnNames": [

"iati-identifier",

"titles”,

"participating-orgs\_implementing",

"recipient-countries",

"sectors",

"sector\_vocabularies",

"sector\_codes",

"budget\_values",

"budget\_value\_currencies"

]

},

1. Make sure there are no empty values

Why is this important? From the Schoolofdata.org course on Cleaning data with spreadsheets (<http://schoolofdata.org/handbook/recipes/cleaning-data-with-spreadsheets>):

*“In many spreadsheets you come across there will be empty (“blank”) cells. They may have been left blank intentionally, or in error. Either way, they are missing data, and it’s useful to be able to find, quantify, display and correct them if needed.*

*You may want to actively specify that the data is missing, rather than leaving a blank cell. - Filling blank cells isn’t always useful and it’s important to choose the right term to denote a missing value.”*

Not all visualization tools know how to handle empty values, or empty values may lead to misrepresentations, so we’ll replace all the empty values in the columns we are going to use.

* + For instance: select the column sectors
  + Open a Text Facet
  + Scroll down to the (blank) value, and click ‘edit’ (appears when you mouse over the value)
  + Change the value to ‘unspecified’.

The code below fills all the empty values in our data set

{

"op": "core/mass-edit",

"description": "Replace blanks in column participating-orgs\_implementing",

"engineConfig": {

"facets": [],

"mode": "row-based"

},

"columnName": "participating-orgs\_implementing",

"expression": "value",

"edits": [

{

"fromBlank": true,

"fromError": false,

"from": [],

"to": "unspecified"

}

]

},

{

"op": "core/mass-edit",

"description": "Replace blanks in column recipient-countries",

"engineConfig": {

"facets": [],

"mode": "row-based"

},

"columnName": "recipient-countries",

"expression": "value",

"edits": [

{

"fromBlank": true,

"fromError": false,

"from": [],

"to": "unspecified"

}

]

},

{

"op": "core/mass-edit",

"description": "Replace blanks in column sectors",

"engineConfig": {

"facets": [],

"mode": "row-based"

},

"columnName": "sectors",

"expression": "value",

"edits": [

{

"fromBlank": true,

"fromError": false,

"from": [],

"to": "unspecified"

}

]

},

{

"op": "core/mass-edit",

"description": "Replace blanks in column budget\_values",

"engineConfig": {

"facets": [],

"mode": "row-based"

},

"columnName": "budget\_values",

"expression": "value",

"edits": [

{

"fromBlank": true,

"fromError": false,

"from": [],

"to": "unspecified"

}

]

},

1. Because our CSV-transformation ‘flattened’ the dataset – and may have put multiple implementing partners and countries in the same cell (separated by ‘;’) – we now want to split those again. Again, OpenRefine comes to the rescue.

* Go to the column ‘participating-orgs\_implementing’
* Click on the arrow
* Select ‘Edit cells’  ‘Split multi-valued cells’
* Multiple values are currently separated by ‘;’.
* Click ‘OK’.
* Did you notice that the number of rows hasn’t changed? So (if you were using the Hivos IATI data set) no rows have been split.

1. The code below will split multi-valued cells in all the columns containing countries.

[

{

"op": "core/multivalued-cell-split",

"description": "Split multi-valued cells in column recipient-countries",

"columnName": "recipient-countries",

"keyColumnName": "iati-identifier",

"separator": ";",

"mode": "plain"

}

]

1. If you look closely at the data in the column “participating-orgs\_implementing”, you'll notice some rows with one or more ';' at the end. If you look at the original XML, you'll see that it contains “empty” elements like this:

<participating-org role="Funding" />

<participating-org role="Implementing" />

<participating-org role="Funding" />

<participating-org role="Implementing" />

If we use a similar “split multi-valued cells”, we end up with empty rows, so instead, we'll replace the ';' and spaces:

[

{

"op": "core/text-transform",

"description": "Replace ';'",

"engineConfig": {

"facets": [],

"mode": "record-based"

},

"columnName": "participating-orgs\_implementing",

"expression": "grel:trim(replace(value, ';', ''));",

"onError": "keep-original",

"repeat": false,

"repeatCount": 10

}

]

Our data set should be in good shape now.

1. Export your data. Choose ‘Export’  ‘Tab separated value’.

**If you are working with a different data set you may have to take some extra steps.**

**On to the visualization!**

In this recipe we’ll be using RAW to visualize the flows.

1. Open the resulting Tab-separated file in a text editor and copy all the text.
2. Go to http://app.raw.densitydesign.org/
3. In the top field, paste the text you just selected.
4. Next, choose an Alluvial Diagram from the options.
5. Move down, to “Map your dimensions”. Drag “sectors” to “steps”, then drag “recipient-countries” below it. Finally, drag “budget\_values” to size.
6. Finally, you can adapt the width and height of the diagram, and set the “Sort by” to “automatic” to clean it up a little.
7. You can also add another step, for instance to show the flow in each country to implementing partners: drag “participating\_orgs\_implementing” below the two steps and you'de done.

Now you have a flow diagram showing the size of budgets in each sector, and how it goes to each country. You can download an image or embed it in your website.

Congratulations!

You have just made your first (?) budget flow chart!