

Data import to InkVisitor

- [Parsing instructions for the import of tabular data](#)

Parsing instructions for the import of tabular data

This doc describes the parsing/import instructions used in the google sheet tables meant to be imported to inkVisitor DDB1, which conforms to DDM (=DISSINET DATA MODEL).

The instructions were mostly improvised in the conversations between DZ and TH, they changed a lot during the development of the inkVisitor data model and parsing. E.g. the system and terminology probably are far from ideal. Sorry for that. :)

Prerequisites

It is helpful to understand the issues around [properties, metaproperties and 2nd order metaproperties](#).

Basics

The first **four rows** of the google sheet table (designated for parsing+import to inkVisitor DDB1) form the **parsing header**.

The **fifth** row contains a header row with column **labels**.

Each other following row represents one **main entity**, e.g. "Person" or "Location".

The foundational part of import instructions parsing instruction **keywords** (e.g. *discard*, *inside*, ...), sitting in the 4th row.

The four rows are:

1. **comment**- anything can be here, it is for humans, mostly used for the definition of the "special" instruction
2. **target object** - for parameters of relational instructions, (e.g. "id"), signals target to which this column is supposed to hooked
3. **type** - for parameters of relational instructions (e.g. "R0002"), for the first part of the property type-value pairs, i.e. dog has "color" (**type**) "brown" (value)
4. **parsing keyword** (e.g. "reference_part") - primary information on what to do with the column data
5. the label /name of the column

	A	B	C	D	E	F	G	H	I	J	K
1					discard - unification						
2									id	id	
3									R0002	Classificati	
4	inside	inside	inside	inside	discard	discard	discard	discard	reference_p	relation	discard
5	id	label	label_lang uage	entity_logical_ty pe	resource_ id	document no	page_range start	page_rang e_end	page_range concatena	class_id	class_label
6	E0137	[Rubrica]	Latin	definite	R0002		52	52	52	NA	NA
7	E0138	[Preambula]	Latin	definite	R0002		52	52	52	NA	NA

Generally, there are three types of instructions

- simple, they do not need other parameters (e.g. *inside*, *discard*)
- relational, they specify relational data and **need** other parameters (e.g. *propvalue*, *relation*)
- special, they are specified in natural language in the comment row

Here are all possible keywords:

simple	inside, discard, hooked-inside
relational	relation, reference_part, propvalue, proptype, proptype_2nd, propvalue_2nd, hooked-relation, hooked-propvalue
special	special

All columns in the table meant for parsing and import should have an instruction header!

Meaning of the parsing instruction keywords

Simple

inside	the content in the column is meant to be directly inside the entity objects, e.g. for columns like "label" or "note", that target entity need to have such attribute specified by the data model
discard	the content in the column is just ignored
hooked-inside	the content in the column is meant to be inside of some " embedded object ", which is always defined by the preceding <i>special</i> column

Relational

	description	param1 type	param2 target object	the cell can contain
relation	for making DDM "Relation"	name of the relational type (e.g. "Classification", "Identification")	can be <i>empty</i> but usually contains " id " as a signal that this entity is connected to the main entity	a concept legacyId, e.g. C3166 (defendant deposition)
reference_part	for making DDM references to resources	resource ID (e.g. R0002)	can be <i>empty</i> but usually contains " id " as a signal that this entity is connected to the main entity	string of the pages, e.g. "v216", which is transformed into Value object
propvalue	for making metaproperty , where the type is A. fixed for whole column OR B. defined by main entity field with <i>proptype</i> instruction	a concept, which defines the type, e.g. C0316 (occupation)	empty OR name of the column with <i>proptype</i>	entity legacyId, or string which is transformed into Value object
proptype	for making metaproperty , where the type can differ		empty or can contain "id"	the C entity
proptype_2nd	for making 2nd metaproperty , where the type can differ		name of the <i>propvalue</i> column, to which this 2nd order property is hooked	
propvalue_2nd	for making 2nd metaproperty , where the type is A. fixed for whole column OR B. defined by main entity field with <i>proptype</i> instruction	a concept, which defines type, e.g. C0316 (occupation)	name of the <i>propvalue</i> column, to which this 2nd order property is hooked	entity legacyId, or string which is transformed to Value object
hooked-relation	as <i>relation</i> , but it is controlled from special instruction	as relation	can be empty, but usually contains the name of the column, which has <i>special</i> instructions	
hooked-propvalue	as <i>propvalue</i> , but it is controlled from special instruction	as <i>propvalue</i>	can be empty, but usually contains the name of the column, which has <i>special</i> instructions	

Special

special	Conforms to the particular parsing function, which is fully custom and based on the instructions.
---------	---