Draft of taxonomy API standard				
DINA TC workshop 2017-05-18				
Editable version:	https://docs.google.com/spreadsheets/d/1uaJh1qt6mvY0ZCEB6uBKdnUnyGdpQnZcnFqTMHTS5pA/edit#gid=0			
Note1: Most of the API calls listed below can be extended with {id} to get an individual object. PUT and DELETE will require one or more {id} to identify the record(s) that will be modified or deleted. POST does not allow an {id}, since the i				
Note 2: We have favored a small number of configurable API calls, instead of providing specialized instances for searching, matching, cloning of subtrees into new classifications etc. This means that the /api/taxonomy/taxon call will requi For instance, cloning of a subtree into a new classification will require three calls (GET subtree, POST classification, POST taxons). Note that this will not carry the vernacular names with the cloned subtree.				
Note 3: The DINA API standard enforces pagination. It is important that pagination is supported, for instance to communicate order of siblings. This allows calls for next sibling and previous sibling to be composed using the call "siblings" a siblings from page 8 to page 8.				
Note 4: A classification needs to store and allow manipulation of the order of siblings (essential for some use cases). On the implementation side, the easiest solution might be to introduce an integer field giving the sort order of the sibling over the entire tree.				
Note 5: These API calls do not support specific operations for hybrid taxa. For the subtree operation, a clear understanding of how hybrid taxa will be treated is important for understanding the API functionality. A common solution is to dis				
Note 6: For use cases, SH = Specimen Handling; TH = Taxonomy Handling (see separate document describing use cases)				
Note 7: Unlike PlutoF and GBIF we have no calls referring to the root node or root nodes of each classification. Those should probably be added.				
Note 8: Terminology: What we call 'classification' here is called 'tree' in PlutoF and 'checklist' in GBIF. What is called 'taxon' here and in PlutoF is called 'name usage' in GBIF.				

DINA-Web call (suggested)	**Parameters**	**PlutoF equivalent(s) (up-to-date swagger documentation)**	**GBIF Species API (gbif.org/eveloper/species)**	**FinBIF API equivalent**
/api/taxonomy/	-	/api/taxonomy/		https://api.laji.fi/explorer/#/Taxa
/api/taxonomy/classification/	-	/api/taxonomy/tree/		N/A
/api/taxonomy/classification/	-	/api/taxonomy/tree/		
/api/taxonomy/vernacular_language/	classification	/api/taxonomy/language/		N/A
/api/taxonomy/vernacular_language/		/api/taxonomy/language/		
/api/taxonomy/mandatory_language/	classification	NA		N/A
/api/taxonomy/mandatory_language/	classification	NA		
/api/taxonomy/taxon/	classification, rank, scientific_name, author, year, vernacular_name, language, ancestor[= {taxon_id} # root taxon for search], search_type, sort_by, include, return_related[= {ancestors | parent | immediate_children | siblings| basionym | accepted}]	/api/taxonomy/taxon /api/taxonomt/taxon/{pk}/higher_taxa /api/taxonomy/taxon/{pk}/direct_children /api/taxonomy/taxon/search	/species /species/{int}/parents /species/{int}/children /species/{int}/synonyms /species/{int}/combinations # all combinations having this basionym /species/{int}/related /species/match /species/search /species/suggest	https://api.laji.fi/taxa/search *
/api/taxonomy/taxon/		/api/taxonomy/taxon	NA	
/api/taxonomy/subtree/	taxon	/api/taxonomy/taxon/{pk}/subtree		https://api.laji.fi/taxa/{id}/children **
/api/taxonomy/rank/	classification	/api/taxonomy/rank/		
/api/taxonomy/rank/	classification	/api/taxonomy/rank/		
/api/taxonomy/vernacular_name/	classification, taxon, language	/api/taxonomy/vernacular_name /api/taxonomy/vernacular_name/search		included in https://api.laji.fi/taxa/{id}
/api/taxonomy/vernacular_name/		/api/taxonomy/vernacular_name		
/api/taxonomy/act/	classification, taxon, date_from, date_to, user	/api/taxonomy/act/		N/A
/api/taxonomy/batch/	classification, file_type	NA		N/A

d is assigned upon the completion of the POST.

e a fair amount of configuration for the most common API calls, which may not be ideal. Other operations will require several calls.

and based on the pagination mechanism. For instance, to get the next sibling, asuming I have the sibling on page 7, I ask for the

s. PlutoF stores one traversal order by linking to the next and the previous taxon in that order. It appears that this traversal order is

tinguish one hybrid parent as the primary parent in operations assuming a tree structure.

Comment/explanation	**Methods**	**Use Case(s)**
Documentation of the API, returns all the endpoints	GET	
Returns the classification(s)	GET	
Creates/modifies/deletes a classification	POST/PUT/DELETE	
Returns the language(s) of vernacular names	GET	
Creates/modifies/deletes a language of vernacular names	POST/PUT/DELETE	
Returns the mandatory language(s) for vernacular names	GET	
Creates/modifies/deletes the mandatory language(s) for vernacular names	POST/PUT/DELETE	
Returns taxon record(s). This API call can be used for autocomplete searches, matching searches (returns type of match among other things), etc by setting appropriate parameters. We suggest that search_type adheres to standardized terms used in e.g. Apache Lucene. Examples: ?filter[classification]=168 # all taxa for one classification ?filter [classification]=168&filter[scientific_name] =alt&search_type=begins_with&include=scientific_name,id # appropriate for autocomplete ?filter[classification]=168&filter[scientific_name] =xxxxxxx&search_type=fuzzy&include=scientific_name,id, match_type&return_related=accepted # appropriate for name matching {id}/?return_related=immediate_children # get immediate children of a taxon	GET	SH-A, SH-C
Creates/modifies/deletes taxon record(s)	POST/PUT/DELETE	SH-B, TH-A, TH-B, TH-C
Returns the entire subtree rooted at the specified taxon. Is this the same as /api/taxonomy/taxon/?filter[ancestor]={taxon_id}? Does it return both taxa and related objects?	GET	
Returns the rank(s) that are used	GET/POST/PUT/DELETE	
Creates/modifies/deletes ranks	POST/PUT/DELETE	
Returns vernacular name(s)	GET	
Create/modify/delete vernacular name(s)	POST/PUT/DELETE	
Returns all acts (creation, updates and deletes) on taxon records	GET	
Allows you to download an entire classification using Darwin Core Archives or similar formats. Receipt will be the ID of the download operation.	GET	

/api/taxonomy/batch/	classification, file_type	NA		
/api/taxonomy/batch/{id}/		NA		N/A
Possible extensions				
DINA-Web call (suggested)	**Parameters**	**PlutoF taxonomy API equivalent**	**GBIF Species API equivalent**	
/api/taxonomy/root (?)	classification	/api/taxonomy/tree{pk}/root_children	species/root	N/A
				N/A
/api/taxonomy/common_ancestor/	taxon_list	/api/taxonomy/taxon/<id>/higher_taxa_intersection?given_ids=<included_id>,<included_id>,<included_id>...		
/api/taxonomy/filter		/api/taxonomy/filter/		N/A
/api/taxonomy/filter		/api/taxonomy/filter/		
				Notes on the FinBIF taxon API
				The taxonomy API is read-only. (Updates are done either with taxon editor G
				Many of the API endpoints are built for creating species pages, so they main
				* Doesn't allow selecting which fields are returned, instead always returns all
				** Returns only immediate children. Getting the entire subtree would require

Allows you to create or modify a classification by uploading an entire classification using Darwin Core Archives or similar formats. Receipt will be the ID of the upload operation.	POST/PUT	All use cases
Gives you progress information on the batch operation {id}. On completion of a download you will get the URL of the result file and an expiry time stamp.	GET	All use cases
Comment/explanation	**Methods**	**Use Case(s)**
Gets the root node(s) of classifications; writes could be done through the /api/taxonomy/taxon write calls. It is not clear that there is anything special about root node(s) in a classification except that it/they do not have an ancestor. If there is more than one root node in a classification, it can be tricky for a backend implementation to find them unless the root nodes are treated in a special way.	GET	
Returns the most recent common ancestor of the specified taxa (if they are in the same classification)	GET	
Returns defined filter(s)	GET	
Create/delete defined filter(s)	POST/DELETE	
UI or database batch updates.)		
ly return different kinds of non-taxonomic data for a single taxon identified by its ID.		
public fields.		
several API calls - one for each child.		