ClipFile Data Model

Entity

All objects created, manipulated, and persisted by ClipFile are, at base, of the *class,* or *type*, Entity. As such, the Entity is the so-called *supertype* of all ClipFile objects. This allows ClipFile, and its users, to operate upon any ClipFile object instance using common data values and operations. Beyond that, depending upon subtype, an object instance can be differentiated and operated upon with entity type dependent operations and data. The power of ClipFile’s capabilities, especially those supporting the claims contained in this document, are directly enabled by this design; this design framework is also, and not coincidently, a recognized and well-tested common and current engineering best practice used the world over to build complex, but reliable, software systems.

All instances of every ClipFile object, no matter what its special capabilities and functionality may be, are at minimum constituted of the following values:

* EntityID – a unique system-generated value used to identify an entity. This value is guaranteed to be unique, across space and time, within the ClipFile data space Every EntityID is otherwise opaque or invisible to a user or ClipFile application, and has no other semantic value.
* EntityType – the system type, or subtype, of an entity, which self-identifies it as one of the subtypes described below, e.g. Tag, Artifact, etc.
* ClippedBy – the Account, described below, of the user which created this entity
* ClippedByDateTime – a standard encoded description of the date and time of creation of the entity with millisecond precision and time zone.
* ModifiedBy – the Account, described below, of the user who last updated this entity
* ModifiedByDateTime – a standard-encoded description of the date and time when last updated
* DeactivatedBy: the Account, described below, of the user who deactivated this entity
* DeactivatedByDateTime – a standard encoded timestamp of the data and time of deactivation of this entity
* SecurityLevel – a system-supplied, sometimes user-specified value describing a default or desired amount of security applied to operations to be performed on this entity. Example values include private, super-private, private-visible, public, etc., as described elsewhere in this document.
* ListPosition – an integer value used for ordering entities in a List, described below.
* Weighting – another integer value used for ordering entities in a List, described below.
* Children -- a List, described below, of entities explicitly and strongly associated with an entity. Entities such as Highlights, Tags, and Artifacts are commonly, but not exclusively, used as “children” of an entity. For example, a Clip of a web page can include artifacts describing and containing the HTML and raw text data on the page, user-generated Highlights excerpting the page, and user-entered Tags describing user-identified attributes of the web page, highlights, etc. This Clip can alternatively be thought to *contain* these child entities. Note that all entities, including Tags, described below, can contain child entities; indeed, Tags that are children of other Tags are an important enabling technology supporting the claims contained in this document.
* FromEntity– the entity containing the List of which this entity is a member, i.e. its parent entity.
* TopEntity – the entity, generally but not exclusively a Clip entity, described below, from which this entity, and its parent, are ‘descended’ from. More specifically, the TopEntity contains, as one or all of its children, either this entity, or more likely, an entity contained recursively as a child in another entity which contains this entity in the list constituting its children.

List

A List is possibly canonically ordered collection of entities. Multiple signifiers of possible canonical orderings for list elements are an entity’s list position and weighting, described above, as well as ratings, flags, and other attributes encoded in Tags, described below.

Account

Each user of the ClipFile system is associated with one or more Accounts. The Account entity is used to identify, authenticate, and authorize a user, or application, to access and operate upon ClipFile entities. The type-specific data that constitute an Account are comprised of elements used in current and common engineering best practices as applied to application and system security, and as such can be effectively enhanced over ClipFile’s lifetime to embrace future state-of-the-art security practices.

* The EntityID, the unique identifier found in every entity, described above, is in this case used as a user’s unique identifier within the system, which, as with all EntityIDs, is otherwise opaque or invisible to the user and to others.
* Name. A simple string which may be used to label a user.
* Email addresses, which are the “login” identities of a particular user, among other uses.
* Credentials. Associated with each user account are a set of distinct credentials of multiple types/technologies, ranging from simple password to Public Key certificate, retina scan, fingerprint image, or any known or as-yet-undeveloped so-called *shared secret* used by ClipFile’s security systems for authentication and authorization. Multiple credentials can be used for multi-factor authentication challenges, e.g. a demand for a second password to allow access to a user’s super-private data.
* Roles: lists of various user “actors” (publisher, editor, curator, author, agent, representative, broker, administrator, system administrator, general user, other) that a user may operate as in ClipFile, each of which may correspond to a set of Permissions, as well as to a set of Credentials.
* Permissions: collections of business rules governing a user’s access to ClipFile features and data. A set of ClipFile permissions will generally correspond to a set of Credentials, Roles, etc.
* Lists. Each user’s account can have multiple lists of other entities attached to it. Uses for lists can include accounts of users comprising the user’s various networks, as well as a user’s sets of Personas, Preferences, Contexts, and so forth, as described elsewhere in this document.

Clip

A Clip is a collection of Entities -- in the simplest case a single Entity -- which commonly, although not exclusively, describes a “thing” or object which has been “clipped”, or saved, in ClipFile. A Clip can encapsulate many types of objects including: a web page, a video clip, an email, a “tweet,” a chapter of a book, an audio clip, etc., as described elsewhere in this document.

|  |
| --- |
| Subject |
| ClippingArtifactEntityID |
| WebPageUrl |
| WebPageEntityID |
| WebPageTitle |
| WebPageDescription |
| WebPageSource |
| WebPageAuthor |
| WebPagePublishDate |

Media

|  |
| --- |
| HtmlArtifactEntityID |
| PlainTextArtifactEntityID |

Artifact

|  |
| --- |
| DataDbName |
| DataObjectKey |
| Data |
| ByteLength |
| MIMEType |
| MD5Checksum |
| UsageCount |

Tag

Tags are entities containing data used to describe, classify, qualify, or otherwise enhance the “meaning” of a parent entity, beyond that which can be encoded within the fields of that entity object itself. A tag may be thought of as meta-data attached to an entity, encoded as a singleton or tuple. TagClass: an optional attribute of a tag, also a character string value, describing the *type* of a tag’s *value* as encoded in the TagText field. An example TagClass could be “author”, which could signify that a TagText value of “Norman Mailer” specifies the name of an author.. Similarly a TagClass of “drunk,” could signify that “Norman Mailer” is a person suffering from alcoholism.

* TagText: the tag *value,* specifying some semantic information, which is commonly, but not exclusively, free text input by a user. For example, “Norman Mailer”.
* TagClass: describes the *type* of this tag. Examples include “title”, “author”, “publish date”, “publisher”, and “web page URL”.
* ListPosition: an integer value used to define the position of the tag in a list of tags.
* TagWeighting: an integer value used to compare this tag to other tags.
* TagScale: an integer used to hold a user-defined degree of agreement with the tag’s TagText. For example, an applicatin could provide a slider to let the user choose any value between -10 and +10 to indicate how much they agree [zero to +10] or disagree [-10 to zero] with each of the statements: “I am a Republican”, “I am a Democrat”, and “I am an Independent”.

Note that ClipFile also uses Tags to add application-specific data to a parent entity. For example, a Tag with TagClass = “flag” could be used to further differentiate whether its parent entity is a “ClipFile patent document” or a “potential competitor.”

Another application-specific use for this generalized Tag functionality is Ratings. For example, given TagClass=”rating” and TagText=”Important!!!” this could indicate that its parent entity has been rated as “Important!!!”

Highlight

A Highlight entity describes a user-selected subset of the data described by its parent Clip. An example Highlight could be a paragraph of text contained within a web page article, or a snippet of audio data containing a single question and answer edited from an hour-long radio program.

* HighlightArtifact: an Artifact entity describing and containing the opaque blob of data constituting the highlight.

TagRelationship

A TagRelationship entity is used to go beyond a tag’s basic parent-child relationship using the FromEntityID field. The TagRelationship entity type is used to describe many-to-many “relationships” between tags. This entity can be used to define any type of named and directed relationship between a pair of Tags. One or more TagRelationship entities can be used to define how (and to what degree) a single Tag relates to another single Tag or to a list of Tags.

* RelateFromTagText: the text value of a pre-existing Tag entity. The “FromTag”  
  can also be thought of as the “BeginTag” or “StartTag”.

|  |
| --- |
| * RelateTagText: the text value of a pre-existing Tag entity used to describe a  directed relationship between a pair of Tags. Examples of directed relationships  include: “is a child of”, “is a parent of”, “is similar to”, “is a kind of”, “is a part  of”, “belongs to”, “is a member of”. * RelateToTagText: the text value of a pre-existing Tag entity. The “ToTag” can also be thought of as the “EndTag” or “FinishTag”. * RelateWeighting: an integer value used to describe the strength of the given relation. For example, the weighting could be used to describe the weighted  relationships among 3 Tags, “Teal”, “Blue” and “Green”:  Teal is 70% Blue  Teal is 30% Green * RelateFromTagID: the entity ID of the FromTag |
| * RelateTagID: the entity ID of the RelateTag |
| * RelateToTagID: the entity ID of the ToTag |

For example, TagRelationships can be used to describe a variety of blue   
or blue-ish colors:

|  |  |  |
| --- | --- | --- |
| **RelateFrom**  **TagText** | **Relate**  **TagText** | **RelateTo**  **TagText** |
| Aquamarine | Is a type of | Blue |
| Blueberry | Is a type of | Blue |
| Blue Bell | Is a type of | Blue |
| Blue Gray | Is a type of | Blue |
| Blue Green | Is a type of | Blue |
| Cerulean | Is a type of | Blue |
| Cobalr Blue | Is a type of | Blue |
| Cornflower Blue | Is a type of | Blue |
| Denim | Is a type of | Blue |
| Indigo | Is a type of | Blue |
| Midnight Blue | Is a type of | Blue |
| Navy Blue | Is a type of | Blue |
| Pacific Blue | Is a type of | Blue |
| Pewter Blue | Is a type of | Blue |
| Sapphire Blue | Is a type of | Blue |
| Sky Blue | Is a type of | Blue |
| Steel Blue | Is a type of | Blue |
| Teal | Is a type of | Blue |
| Turquoise Blue | Is a type of | Blue |

Conversely, if the From and To tags are switched then the “Is a type of” type of relationship would need to be inverted to something like “Is a primary component of”.

|  |  |  |
| --- | --- | --- |
| **RelateFrom**  **TagText** | **Relate**  **TagText** | **RelateTo**  **TagText** |
| Blue | Is a primary component of | Aquamarine |
| Blue | Is a primary component of | Blueberry |
| Blue | Is a primary component of | Blue Bell |
| … | … | … |
| Blue | Is a primary component of | Sky Blue |
| Blue | Is a primary component of | Steel Blue |
| Blue | Is a primary component of | Teal |
| Blue | Is a primary component of | Turquoise Blue |

ClipFile Entity Functionality

ClipFile Entities implement various operations upon themselves, and upon the data they describe, in order to enable the activities of ClipFile users and applications. These activities, described elsewhere in this document, include Discovery, Curating, and Sharing. The primary ClipFile entity operations designed to support these and other ClipFile activities can be described as follows:

Entity, List: create, find, update, disable Entity

These operations comprise the so-called CRUD operations – create, read, update, delete – which are common and current engineering terms-of-art describing the basic operations performed upon an idealized set of persistent data objects, such as those performed by ClipFile entity instances, in software and hardware systems. All ClipFile Entities and Lists, of all Entity types, implement these operations.

Furthermore, every Entity also implements two operations, get*x* and set*x*, for all *x* which are names of fields in an entity; these functions allow the application to retrieve and replace the values contained within those entity fields. The use of “getter and setter” functions in an idealized persistent object, as implemented by a ClipFile entity instance, is also a common and current engineering best practice utilized in the design and implementation of software systems.

Account: loginUser, logoutUser

Accounts implement loginUser to authenticate and certify a user, application, a specific computer system, or combination thereof, and then to “arm” ClipFile to authorize operations on ClipFile objects, based upon the data contained in the Account entity, such as Credentials, Roles, Permissions, and possibly Lists of Users, Contexts, Personas, Profiles and the like. logoutUser is performed to gracefully exit the user from ClipFile, and to help prevent access to data and operations controlled by the Account entity when the user or application has not been authenticated.

Clip: createSnapshot.

A Snapshot is a copy of a fully populated Clip made at a specific moment in time and space. Snapshots are primarily, but not exclusively, used in the Sharing functionality, described elsewhere.

Examples of common ClipFile activities implemented by one or more of these operations.:

Clip the web page www://nytimes.com/front-page/Manned-Russian-spacecraft-lands-on-Mars.html

Tag the clip with “I don’t believe this!”, “personal opinion”.

Get me all my highlights on any clip over the past month, sorted by importance

Get me all of my highlights related to Apple for the past week

Show me the users with whom I’ve shared clips about the Boston Red Sox over the past month.

Change the TagText of this tag from “who” to “whom.”

Delete this highlight from this clip.

Mark this Tag as “private”, for my eyes only.