**Introducció a la API de FamilySearch**

**Introducció**

Tota la informació disponible i necessària per tal de poder començar a treballar sobre l’API de FamilySearch és pot trobar en el portal de desenvolupadors de la pàgina web.

Aquesta secció de la pàgina web esta formada per diferent seccions, però l’estructura no acaba de resultar del tot clara per una persona que vulgui començar de cero. Si ens enfoquem més en la documentació disponible que no pas en la estructura proposada per l’organització, podem veure que la informació es distribueix en certa forma en els següents grups:

* **Requisits tècnics:** Conjunt d’informació necessària per comprendre l’estructura de l’API, els formats de dades que maneja i els passos necessaris per tal de començar a interactuar amb aquesta.
* **Recursos disponibles i rutes d’accés:** Informació detallada sobre cada recurs accessible a través de l’API. En concret disposa dels detalls de com accedir aquest recurs, les operacions que es poden realitzar sobre ell, la informació que conté i quines són les connexions amb altres recursos.
* **Capacitats extres de l’API:** Aquest recull d’articles conceptualitza característiques de l’API com poden ser els recursos de *catching, localització o throttling.*
* **Evolució i canvis produïts a l’API:** Informació semi ordenada de com l’API s’ha vist evolucionada al llarg del temps i un recull dels canvis produïts a l’API, procés de certificació, material de documentació i eines de desenvolupament.
* **Eines de desenvolupament:** Recull d’entorns de desenvolupament i eines extres que poden facilitar la feina del programador.

**Arquitectura de l’API**

**Que és una API**

Abans d'entrar en detall en com funciona una API, sembla que cal doncs definir amb una mica més de precisió en que consisteix exactament.

Una API, de l'anglès 'Application Programming Interface' o Interfície de programació d'aplicacions en català, representa el conjunt de subrutines, funcions i procediments que ofereix una biblioteca per tal de ser utilitzada pel software de tercers com una capa d'abstracció.

Aquest conjunt de subrutines, funcions i procediments, acostumen a oferir accés a certs serveis o conjunts de dades d'un particular a tercers de forma controlada.

**Arquitectura de l’API de FamilySearch**

L’arquitectura que l’API de FamilySearch utilitza és una arquitectura REST. Les sigles REST venen de l’anglès i representen l’expressió: *Representational State Transfer”.*

Les arquitectures REST es caracteritzen per estar orientades els recursos més que en les accions que es poden realitzar sobre ells i com a peculiaritat implementen sis restriccions; a saber, *uniform interface, stateless, client-server, cacheable, layered system, code on demand.* Aquestes sis restriccions son les que defineixen les bases de les arquitectures REST.

És diu que una arquitectura REST és orientada als recursos perquè estan construïdes al voltant d’objectes en comptes d’accions. Per exemple, a l’API de FamilySearch es parla de Persones i Esdeveniments en comptes de llegir persones o escriure esdeveniments.

Les representacions expliquen com els recursos són tractats per l’API i de quina forma s’han de realitzar les comunicacions entre el servidor i el client. Els formats més freqüents són JSON i XML. FamilySearch dona suport a ambdós formats.

Per posar un exemple reduït que il·lustri el que estem explicant, podem descriure de la següent forma el que podria ser una operació contra la API de FamilySearch especificant quin element seria considerat el Recurs, quin el Servei i quin la Representació.

* **Recurs:** Persona
* **Servei:** Informació de la persona (GET)
* **Representació:** Nom, cognoms, esdeveniments relacionats amb la seva vida, etcètera, en format XML o JSON.

Com també havíem comentat, l’arquitectura REST es caracteritza per sis restriccions que el sistema imposa.

**Interfície uniforme (uniform interface)**

Defineix la interfície entre el client i el servidor. Que ve a significar que utilitzem els protocol de comunicació HTTP o HTTPS i les URI (Identificador de Recurs Uniforme, també conegudes com “URL” en camps diferents), per aconseguir accés als recursos i operacions de l’API.

Els verbs permesos pels protocols de comunicació web HTTP i HTTPS són: GET, PUT, POST, DELETE, OPTIONS and HEAD.

Per exemple, per fer la petició de lectura sobre el recurs d’una Persona a l’API de FamilySearch executaríem la següent crida HTTP/HTTPS:

GET /platform/genealogies/persons/2:2:PPPJ-MYZ7

**Sense Estat (stateless)**

El servidor no emmagatzema informació del client. Això implica que cada petició a l’API ha de contenir tota la informació necessària per poder processar la petició.

En altres paraules, l’estat és emmagatzemat en el client.

Hi ha exemples com el Oauth V2, que no són realment RESTful perque guarden informació del client.

**Client–Servidor (client-server)**

Hem de comprendre que el sistema client-servidor és un sistema desconnectat. Per tant hem de tenir sempre en compte que el client mai tindrà accés directe a les bases de dades que emmagatzemen la informació, sinó que ha de passar sempre per l’intermediari que és l’API.

El protocol de comunicació, o la interfície de comunicació que ja hem descrit fa un moment fa les funcions d’enllaç del sistema.

**Cacheable**

Fa referència a que les respostes del servidor poden ser guardades o emmagatzemades. Existeixen tres nivells de ‘cache’:

* **Implícit:** Si és el client el que decideix quan de temps guardarà els paràmetres o informació és tracte de implícit caching.
* **Explícit:** Si és el servidor el que mana i posa les regles, parlem de explícit caching.
* **Negociat:** Quan el client i el servidor negocien i arriben a un acord, es tracte de caching negociat.

**Sistema per capes (Layered System)**

Aquest principi es basa en que el client no pot assumir que tindrà direcció directa amb el servidor. És a dir, existeixen intermediaris en forma de software i hardware entre el client i el servidor.

Això facilita l’escalabilitat del sistema, perquè el client no s’ha de provocar de comunicar-se amb elements o tecnologies concretes que es poden veure subjectes a canvis degut a canvis de necessitat per part del servidor.

**Codi sota petició (code on demand)**

De forma excepcional el servidor pot donar accés al client a la lògica del funcionament del servidor. Alguns exemples poden ser els ‘Java Applets’ o ‘blocs de codi JavasScript.

**Conclusió**

En conclusió, seguir les regles marcades per una arquitectura REST ens permet assegurar els següents beneficis: escalabilitat, simplicitat, modificabilitat, visibilitat, portabilitat i fiabilitat.

**Formats de dades utilitzats en la comunicació client – servidor**

FamilySearch utilitza tres formats de dades diferents. Durant les comunicacions amb la API, mitjançant els camps ‘Header’ i ‘Content-Type’ de la petició és pot indicar quins sons els formats de dades que es vol utilitzar durant el transcurs de la conversació.

Els conjunts de dades i els formats sota els que estan codificats s’exposen a continuació:

* Les dades genealògiques es troben representades amb el format GEDCOM X.
* Els objectes específics del model de FamilySearch es troben definits sobre una extensió del format GEDCOM X per dades multimèdia però al resultar específics per FamilySearch, no s’han especificat de forma més general i formal.
* El format de dades Atom s’utilitza per tal de proporcionar un format pel contingut web i les meta-dades. També és implementat com una extensió del format GEDCOM X, però aquests formats no són considerats l’estàndard.

**El format de dades GEDCOM i GEDCOM X**

GEDCOM és un acrònim del anglès Genealogical Data Communications, o en català, Communicació de dades Genealògiques.

GEDCOM és una especificació que s’ha convertit en l’estàndard de l’industria degut a la seva acceptació universal. El format GEDCOM va ser desenvolupat per l’església de Jesucrist dels Sants dels Últims dies i va ser presenat per primer cop al 1984. És a dir, pel mateix col·lectiu que va crear el portal web de FamilySearch.

Per tal de simplificar podríem dir que el format GEDCOM és representat com un fitxer de text que conté informació genealògica d’una persona i les meta dades necessàries per poder enllaçar els diferents registres de la mateixa persona.

Tot i que l’última versió, datada del 1996, segueix sent molt acceptada, FamilySearch va proposar al 2012 canviar-la per un projecte més gran que seria conegut pel nom de GEDCOM X.

GEDCOM X era un nou projecte de codi obert i oferia un sistema que facilitava la inclusió de arbres genealògics amb les fonts de dades i els registres, donava suport al intercanvi i enllaçament de dades a través del núvol i creava la API sobre la que aquest projecte ha estat craet.

A continuació, per tal de donar un petit exemple de com un recurs és codificat sota el format de dades GEDCOM X, s’ofereix a la taula \_\_\_\_ la representació del recurs ‘Persona’.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nom** | **Descripció** | **Format de dades** | **Restriccions** |
| Private | Indica si la instància de la persona ha estat designada com a privada o pública. | Booleà | OPCIONAL. Una descripció de com les aplicacions han de tractar les dades de caràcter privat. |
| Gender | El gènere de la persona | http://gedcomx.org/v1/Gender | OPCIONAL. |
| Names | Els noms de la persona | Llista de http://gedcomx.org/v1/Name. El ordre d’entrada es preserva. | OPCIONAL. Si més d’un nom és introduït, s’assumeix que aquests han estat introduïts en ordre de preferència, amb el més preferit introduït com a primer. |
| Facts | Esdeveniments relacionats amb la vida d’una persona. | Llista de <http://gedcomx.org/v1/Fact>. L’ordre es preservat. | OPCIONAL. |

Així doncs, podem veure com la instància de ‘Persona’ conté un camp booleà que   
indica si aquesta és de caràcter privat o públic i tres camps que es troben codificats sota els estàndards del format GEDCOM X.

Per exemple, el format de dades ‘http://gedcomx.org/v1/Gender’ representaria un objecte amb l’estructura que s’exposa en la taula \_\_\_\_ i els valors possibles per l’enumeració de gènere s’indiquen en la taula \_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| **Nom** | **Descripció** | **Format de dades** | **Restriccions** |
| type | Valor enumerat indicant el gènere | Enumeració | REQUERIT. El gènere ha de ser especificat i utilitzar un valor dels acceptats per l'enumeració és recomanat. |

|  |  |
| --- | --- |
| **URI** | **Descripció** |
| http://gedcomx.org/Male | Gènere Masculí |
| http://gedcomx.org/Female | Gènere Femení |
| http://gedcomx.org/Unknown | Gènere no especificat |

Com que l’objectiu del projecte no és estudiar la codificació GEDCOM o GEDCOM X, sinó comprendre quina informació es troba realment disponible sota aquesta codificació per part de FamilySearch, no entrarem més en detall.

L’objectiu d’aquest apartat era explicar quina era la convenció utilitzada per la API i per qualsevol informació extra s’adjunta en la bibliografia del projecte l’enllaç a la documentació del model conceptual.

**Format de dades FamilySearch**

El format de dades FamilySearch defineix una serialització de formats específics per la plataforma de desenvolupadors de FamilySearch. Això implica que aquestes definicions de format no es troben estandarditzades i acceptades globalment, fet que cobrà tot el sentit del món ja que aquestes fan referència només a l’univers de FamilySearch.

Els formats de dades específics de FamilySearch són una extensió dels formats media de l’especificació GEDCOM X.

Les extensions i nous objectes que FamilySearch ha creat de forma específica per oferir accés a les seves dades genealògiques i meta dades a través de la API, segueixen les pautes de la codificació GEDCOM X.

**Format de dades Atom**

Els formats de dades Atom, o dades atòmiques, són utilitzats per proporcionar llistes o enumeracions de dades. Aquestes llistes són utilitzades per exemple en les col·leccions ordenades de resultats com podrien ser les respostes a les funcions de la API cerca de persones o obtenir el historial de canvis d’una persona en concret.

**Codificacions del formats de dades**

Els formats de dades que s’han exposat en els apartats anteriors no són més que unes convencions que marquen com hauria de ser l’estructura dels objectes o recursos segons el seu origen.

No obstant, aquestes estructures es poden codificar sobre diferents tecnologies o llenguatges. FamilySearch dona suport a les dues estructures de dades més comuns i utilitzades en el intercanvi de dades a traves del web, els llenguatges XML i JSON.

**El llenguatge XML**

La llengua de marcatge extensible, també coneguda per l’acrònim XML, és un llenguatge de marcatge que defineix un conjunt de regles per tal de codificar documents i informació en un format que sigui tant llegible pels humans o per màquines.

El llenguatge va ser definit pel Consorci World Wide Web i tracte d’emfatitzar la simplicitat, generalitat i usabilitat mitjançant Internet.

Una versió reduïda de la representació en XML del recurs ‘Nota’, amb camps: subjecte, text i atribució, es representaria de la següent forma:

<Note xmlns="http://gedcomx.org/v1/" id="..." xml:lang="...">  
 <subject>...</subject>  
 <text>...</text>  
 <attribution id="...">  
 <contributor resourceId="..." resource="..." />  
 <modified>...</modified>  
 <changeMessage>...</changeMessage>  
 <creator resourceId="..." resource="..." />  
 <created>...</created>  
 </attribution>  
</Note>

Com es pot veure, cada camp, objecte o peça d’informació del recurs es troba envoltant per dos etiquetes que utilitzant el nom del camp, en marquen el inici i final. Per exemple, pel camp subjecte tenim les etiquetes ‘<subject’> i ‘</subject>’.

**El llenguatge JSON**

El format JSON també anomenat Notació d’Objectes JavaScript és un estàndard de format obert que, de la mateixa forma que el XML, pretén crear codificacions llegibles per l’ésser humà d’objectes i al mateix temps, poder transmetre aquestes dades a través del núvol de forma ordenada.

Aquest format es basa en el concepte ‘clau – valor’. És a dir, cada camp te un valor associat i els camps són accessibles només a través de la clau apropiada.

El llenguatge JSON deriva del JavaScript i tot i que al principi només aquest incorporava funcions per codificar i descodificar en aquest llenguatge, el fet d’haver-se convertit en l’estàndard més gran de la comunicació en línea, ha provocat que molts altres llenguatges de programació hagin creat les seves pròpies funcions de codificació i descodificació.

Una versió reduïda de la representació en JSON del recurs ‘Nota’, amb camps: subjecte, text i atribució, es representaria de la següent forma:

{  
 "lang" : "...",  
 "subject" : "...",  
 "text" : "...",  
 "attribution" : {  
 "contributor" : { },  
 "modified" : "...",  
 "changeMessage" : "...",  
 "creator" : { },  
 "created" : "...",  
 "id" : "..."  
 },  
 "id" : "..."  
}

**Evolució en el temps de la API**

L’api de FamilySearch s’ha vist subjecte a molts canvis i modificacions en el temps, amb algunes d’aquestes modificacions afectant en una proporció considerable els recursos accessibles a través de la API.

Els canvis més interessants que s’han produït a la API durant els últims anys es citen a continuació:

L’últim canvi citat, el que fa referència a .... va succeir quan la implementació dels exemples del projecte a través del SDK oficial de JavaScript estava complerta. En concret, després de varis canvis de data, les modificacions és van fer afectives al \_\_\_\_\_\_ i aquestes van repercutir en gran mesura sobre el SDK que s’havia utilitzat per la implementació, trencant-ne moltes de les seves funcionalitats.

A falta de documentació es va consultar un dels Webinair passats que FamilySearch havia ofert, i que es troben guardats en la documentació per desenvolupadors, per tal de comprendre millor la magnitud dels canvis i a quins recursos afectaven.