**Introducció a la API de FamilySearch**

**El portal de desenvolupadors**

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 la 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 la 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.
* **Evolució i canvis produïts a la API:** Informació semi ordenada de com la API s’ha vist evolucionada al llarg del temps i un recull dels canvis produïts a la API, procés de certificació, material de documentació i eines de desenvolupament.
* **Serveis extres oferts per la API:** Aquest recull d’articles conceptualitza característiques de la API com poden ser els recursos de *catching, localització o throttling.*
* **Eines de desenvolupament:** Recull d’entorns de desenvolupament i eines extres que poden facilitar la feina del programador.
* **Certificació:** Recull la informació necessària als diferents processos de certificació i regulacions a les que s'ha de fer front en cas de voler certificar l'aplicació i obtenir accés a les dades de producció. També descriu els beneficis d'una relació de col·laboració amb la entitat.

**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 (o atòmic)**

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 temporal de la API**

**Evolucions i punts rellevants pel projecte**

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.

Hi ha hagut molts canvis interessants al llarg del temps. L’accés a la llista complerta s’adjunta a la bibliografia del projecte, però no te gaire sentit mencionar-los tots en aquesta secció.

L’últim canvi introduït a la API i probablement el més important de tots els que s’han arribat a produir, fa referència a la nova estructura del back-end anomenada ‘Tree Foundation’. A falta de documentació oficial es va consultar un dels Webinair passats oferts per FamilySearch, per tal de comprendre millor la magnitud dels canvis i a quins recursos afectava i s’exposen de forma més detallada a la següent secció per tal d’oferir visibilitat a possibles futurs estudiants.

Les primeres fases del canvi que van arribar a producció a mitjans del mes de juliol, moment en el que la implementació dels exemples pel projecte a través del SDK oficial de JavaScript estava complerta. Aquests canvis van repercutir en gran mesura sobre el SDK oficial utilitzat, trencant-ne moltes de les funcionalitats.

Al final, els principals problemes que va causar pel projecte van ser interrupcions permanents contra el Sandbox o entorn de probes i la inhabilitat d’accedir als recursos Source i Discussion relacionats a una persona degut a la no adaptació del SDK oficial.

**Tree Foundation**

Com s’ha comentat en la secció anterior, Tree Foundation és la nova versió del back-end que FamilySearch està implementant. A continuació es detalla en més profunditat en que consisteix aquest canvi exactament, ja que pot suposar un canvi en com les dades principals emmagatzemades per la API han de ser accedides respecte a la forma en que aquest projecte exposa.

El projecte de Tree Foundation consisteix en canviar per complert el motor que fa funcionar el back-end de FamilySearch. El motor actual, anomenat de forma interna, Conclusion Tree i que era l’estructura amb la que la API es comunicava per tal d’extreure’n dades, passaria a ser depracat en benefici de la nova estructura implementada en el projecte Tree Foundation.

El canvi d’arquitectura té com a raó de ser poder augmentar l’escalabilitat del sistema i fer front a la demanda actual sobre la que la API es veu d’accés a les dades. En la imatge \_\_\_ es pot veure com es passa d’un sistema unificat, a un sistema distribuït.

Els beneficis esperats són un increment en la escalabilitat del sistema, una millora eventual del rendiment, tot i que aquest no és l’objectiu principal durant les primeres fases d’implementació i finalment una millor correlació entre el funcionament del sistema i com la API accedeix a les dades.

Addicionalment, el projecte canviarà part de la informació inclosa en els recursos accessibles a través de la API.

Un dels canvis que l’evolució de Tree Foundation pretén integrar i que a dia d’avui, encara no es troba a producció, és integrar dins del recurs d’una persona tota la informació que s’acostuma a consultar. Per exemple, les relacions familiars o les fonts de dades que verifiquen que la informació introduïda és correcta.

Aquest canvi pretén facilitar la navegació per les dades, evitant haver d’accedir a múltiples recursos per tal d’aconseguir tota la informació desitjada. Aquests canvis també permetran alliberar part la càrrega que suporta el sistema i millorar-ne eventualment la eficiència. La figura \_\_\_\_\_ representa de forma visual el que aquest canvi implica.

Quan Tree Foundation va arribar a producció, només els recursos Source i Discussion es trobarien incorporats dins del recurs Person i els antics enllaços a aquesta informació quedarien deprecats. Més endavant, apuntant a mitjans d’Agost, es realitzaria el mateix per les relacions familiars. Es pot veure al que ens estem referint aquí en la figura \_\_\_\_\_.

Tot i que el canvi és molt prometedor, cal tenir en e compte que te el potencial de trencar part dels SDKs oficials i moltes de les aplicacions que tractin amb versions antigues de la API, com ja ha fet les primeres versions a producció de Tree Foundation.

Per tant, caldrà estar al compte de quan aquests canvis arriben a producció i de que els SDKs oficials estan preparats per fer front a aquest canvi en cas de que vulguin ser utilitzats.

**Serveis extres oferts per la API**

La API de FamilySearch no es caracteritza només per ser el centre d'informació obert amb més registres genealògics, sinó que a més a més la API que han implementat ha estat dissenyada pensada en que aquesta sigui utilitzada en gran mesura a diferents llocs del món del a forma més còmode possible i que a la vegada pugui acomodar al màxim nombre d'usuaris possibles.

Tot i que evidentment cap sistema és infalible, i de fet, com ja hem comentat, FamilySearch està realitzat una actualització del seu sistema del back-end per tal de garantir una molt millor escalabilitat del sistema en el futur, el sistema actual ja presenta una sèrie de quatre serveis dignes de menció.

**Caching**

El fet de voler oferir un servei de *caching* decent ha influenciat una bona part del disseny de la API.

Davant la pregunta evident de, perquè és important el *caching per FamilySearch?* Són tres els punts a destacar:

* El fet de que el client pugui aplicar Catching evita la repetició de certes peticions al sistema.
* L'existència de *reverse proxies* permet a aquests respondre a peticions dels clients sense necessitat d'accedir als servidors i bases de dades d'origen si aquesta ja es troba disponible. Per tant, evitant càrrega al sistema.
* Els servidors poden indicar als *proxies* que les seves dade encara són vàlides i que per tant no fa falta que facin la petició de noves dades.

**Throttling**

La funcionalitat de *throttling* s'encarrega a posar un límit al nombre de peticions en un interval específic de temps. Aquests límits són creats a nivell d'usuari, per tant, utilitzar més d'una sessió diferent no permet saltar-se aquesta limitació.

Les polítiques de *throttling* permeses són calculades mitjançant el temps de processat respecte al temps real que ha transcorregut. Diferents recursos, disposen de diferents polítiques. En cas d'excedir aquests límits la capçalera de la resposta ens indica quan de temps cal esperar entre peticions per tal de poder tornar a llençar peticions contra la API.

Aquest servei, que evidentment, és una limitació, evita que certs usuaris de forma intencionada o involuntaria, bloquegin tots els recursos per part dels servidors de FamilySearch i el servei quedi inutilitzable per altres persones.

**Sincronització**

Un dels reptes més comuns per aquelles aplicacions que emmagatzemen dades d'un tercer és el de saber si aquestes han estat modificades des de l'últim cop que van ser consultades o extretes.

El paràmetre ETag de la capçalera de les peticions contra la API s'ha modificat per tal d'obtenir la versió d'un recurs. Per tant, si realitzem una petició de només aquestes capçaleres, a canvi d'un esforç petit de la API podrem comparar si els recursos desitjats han estat modificats o no sense causar un gran impacte en els recursos utilitzats.

Un cop sabem que un recurs ha estat modificat es poden descobrir els canvis realitzats mitjançant la comparació dels dos objectes o llegint-ne l'historial de canvis. Evidentment, sempre es podria realitzar una sobre escriptura complerta del recurs.

**Internacionalització**

Un dels serveis que més m'ha sorprès trobar-me a la API de FamilySearch és el de suport a la internacionalització.

La internacionalització consisteix en adaptar una peça de software o contingut a diferents llenguatges sense necessitat de realitzar canvis en el codi. La API de FamilySearch permet realitzar les peticions de forma que certs conjunts de dades retornats, com poden ser per exemple el nom de països o persones, estigui localitzat en el idioma especificat.

La internacionalització permet que un usuari que es troba a Espanya per exemple vegi el terme 'living' (viu) en el seu idioma natiu. Un altre recurs que moltes pàgines web localitzen son les dates. És ben sabut que l'estructura Americana d'una data, per exemple, difereix de la d'una Espanyola.

El conjunt de dades que FamilySearch permet internacionalitzar és:

* Les propietats de visualització del recurs Persona.
* La informació relativa a les localitzacions també es troba localitzada.
* Certs aspectes del vocabulari controlat, o comú, de l'API.
* Dates.

**Eines de desenvolupament**

FamilySearch posa a disposició dels desenvolupadors un seguit d'eines destinades a facilitar les seves implementacions contra la API.

Són dos els recursos principals que permeten apropar la utilització de la API a qualsevol desenvolupador. Els SDKs i les aplicacions d'exemple.

**Els SDK de FamilySearch**

Els 'kits de desenvolupament software' són un conjunt d'eines pel desenvolupament de software que faciliten la creació d'aplicacions per un software, hardware, framework, sistema de computació, videojoc, sistema operatiu o plataforma de desenvolupament concreta.

En el cas de FamilySearch es tracte d'eines de desenvolupament destinades a un llenguatge de programació específic.

Però, perquè utilitzar un SDK? En el cas de FamilySearch un SDK ens ofereix la possibilitat de comunicar-nos amb la API de FamilySearch sense haver de preocupar-nos de forma directa en com aquestes recursos han de ser demanats contra la API o en parcejar els JSON/XML de les respostes. Tasca que pot esdevenir tediosa i repetitiva. Per tant, permeten centrar més l'esforç en el desenvolupament de funcionalitats que no pas en el de comunicar-se amb la API.

FamilySearch disposa en l'actualitat de sis SDKs diferents per tal d'integrar aplicacions amb la seva API, encara que no tots es troben en el mateix estat ni són per la mateixa plataforma. De la mateixa forma, alguns d'aquests SDK són oficials, significant que són desenvolupats i mantinguts per la pròpia organització, mentre que d'altres han estat creats per la comunitat.

* **Java SDK:** El Java SDK és un SDK oficial que serveix per crear aplicacions d'escriptori.
* **PHP SDK:** El SDK de PHP és un SDK oficial per crear aplicacions web. No obstant, no es troba del tot actualitzat i la última versió d'aquest no integra masses de les funcionalitat que es poden realitzar contra la API.
* **C# SDK:** El SDK de C# és un altre dels SDK oficials per crear aplicacions .net.
* **Javascript SDK:** El SDK de Javascript és l'últim SDK oficial i està orientat sobretot a la creació d'aplicacions web. És el SDK que s'ha utilitzat a l'hora de crear el projecte ja que cobreix gran part de les potencialitats de la API i en facilita l'ús en gran mesura.
* **Python SDK:** El SDK de Python resultaria molt atractiu de cara a tota mena d'aplicacions però per desgracia encara es troba en fase de desenvolupament, sense dates concretes de finalització, ja que no es tracte d'un SDK oficial.
* **Ruby SDK:** La perla de Ruby és l'últim SDK disponible per FamilySearch i també es tracte d'un SDK no oficial.

**Les aplicacions d'exemple**

Les aplicacions d'exemple consisteixen en aplicacions de codi obert que permeten entendre de forma més pràctica com funcionen els SDK i de com s'han de realitzar les comunicacions per tal de poder utilitzar de forma pràctica el SDK.

Solen cobrir un conjunt d'operacions bàsiques com poden ser les de llegir l'usuari connectat, una cerca bàsica o la lectura d'una persona concreta de l'arbre. En definitiva, representen un bon punt de partida per aquells que vulguin familiaritzar-se amb un SDK concret i comprendre'n les bases que permetin elaborar funcionalitats més complicades en un futur.

De la mateixa forma que els SDK, FamilySearch disposa de sis aplicacions d'exemple, aquest cop, sent només oficials les aplicacions que empren els SDK de Javascript i Java per Android.

**Altres eines interessants**

Com a últim recurs per desenvoupadors FamilySearch llista una sèrie d'eines que poden esdevenir útils de cara a realitzar proves amb la API.

Entre aquests, destaquen differents proxy que serveixen per simular peticions HTTP i HTTPS contra la API i una utilitat no oficial que permet copiar dades de producció als entorns de desenvolupament també coneguts com a 'sandbox'.

**Procés de certificació**

Per tal de poder connectar les aplicacions desenvolupades per tercers al conjunt de dades oficial de FamilySearch les aplicacions han de passar un procés de certificació.

Les aplicacions poden ser certificades per l'ús comercial o per ús limitat. Les Apps que volen ser certificades només per ús limitat requereixen un anàlisis tècnic de com funcionen però prescindeixen dels anàlisis de negoci i marketing a les que es veuen sotmeses les aplicacions d'us comercial. Com a contrapartida, les aplicacions d'ús limitat no poden aparèixer a la galeria d'aplicacions.

Hi han dos tipus de certificacions principals:

* **Certificació de lectura:** L'aplicació ha de ser certificada per la lectura de tots aquells recursos als que accedeix. També cal que passi certs estàndards en el procés d'identificació.
* **Certificació d'escriptura:** En cas de que l'aplicació realitzi operacions d'escriptura, aquestes també han de ser testejades per part de FamilySearch.

Veurem més detalls en concret del procés de certificació en la part més pràctica del a memòria del projecte.

Per acabar aquesta secció, comentar que mantenir una relació de 'companys de negoci' proporciona certs beneficis com aparèixer en les seves galeries d'aplicacions, poder utilitzar el logotip d'aplicació certificada entre altres petits avantatges.

**Diferents entorns de desenvolupament i producció**

Es disposa molt poca informació sobre els diferents entorns als que es pot accedir quan es crea una connexió amb FamilySearch.

A base d'investigar diferents codis i possibilitats, s'ha arribat a la conclusió de que existeixen els següents entorns:

* **Sandbox:** Entorn de treball únic per cada usuari. Utilitzable durant el desenvolupament de noves aplicacions i que replica les funcionalitats de la API sense tenir accés a les dades de producció. Es pot entendre com un entorn de proves.
* **Staging:** Entorn de treball que s'utilitza per validar que una aplicació que s'ha desenvolupat en un sandbox, està realment preparada per accedir a les dades de producció.
* **Beta:** Entorn de treball sobre el que la api es va actualitzant a noves versions. Quan es vol realitzar un canvi en les funcionalitats o estructura de la API s'utilitza aquest entorn perquè els desenvolupadors puguin testejar el funcionament de les seves aplicacions en aquesta abans de convertir-la en la nova API de producció.
* **Producció:** Entorn de treball en el que es pot accedir a les dades oficials de FamilySearch. Representa l'objectiu final d'una aplicació a l'hora de comunicar-se amb la API.