**Introducció a l'API de FamilySearch**

**El portal de desenvolupadors**

Tota la informació disponible per tal de poder començar a familiaritzar-se amb l’API de FamilySearch, pot ser trobada en el portal de desenvolupadors.

Aquest apartat de la web està format per diferents seccions, malauradament, l’estructura no acaba de resultar del tot clara per una persona que vulgui iniciar-se per primer cop en l'ús d'aquesta API.

Si ens enfoquem més en la documentació disponible, que no pas en l'estructura proposada per l’organització, podem veure que la informació es podria distribuir, 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 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 al 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 l'API:** Informació semi ordenada de com l'API s’ha vist evolucionada al llarg del temps i un recull dels canvis produïts sobre els recursos, procés de certificació, material de documentació i eines de desenvolupament.

**Serveis extres oferts per l'API:** Aquest recull d’articles conceptualitza característiques de l'API com poden ser els recursos de *caching, localització o throttling.*

**Eines de desenvolupament:** Recull d’entorns de desenvolupament i eines extres que poden facilitar la feina del desenvolupador.

**Certificació:** Recull la informació necessària per gestionar els diferents processos de certificació i informació sobre les regulacions a les quals s'ha de fer front en cas de voler certificar l'aplicació.

**Arquitectura de l’API**

**Què és una API?**

Abans d'entrar en detall en com funciona una API, estaria bé definir, amb una mica més de precisió, en què 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 en el 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 sobre la qual està creada l’API de FamilySearch és una arquitectura REST. Les sigles REST provenen de l’anglès i representen el concepte: \gls{REST}. *Representational State Transfer”.*

Les arquitectures REST es caracteritzen per estar orientades els recursos més que a les accions que es poden realitzar sobre ells i com a peculiaritat, es caracteritzen per sis regles o restriccions.

Aquestes són: *interfície uniforme, sense estat, client-servidor, emmagatzemables, sistema per capes, i codi sota petició.* Aquests sis conceptes són doncs els que defineixen les bases de es arquitectures REST.

És diu que una arquitectura REST és orientada als recursos perquè estan construïdes al voltant d’objectes i les relacions entre aquests, en comptes d’accions. Per exemple, a l’API de FamilySearch, es parla de persones i esdeveniments*,* en comptes de llegir persones o crear esdeveniments i en canvi, aquestes operacions, passen a formar part dels objectes *Persona* i *Esdeveniment*.

L'intercanvi de dades es produeix mitjançant l'ús de diferents representacions. Aquestes, expliquen com els recursos són tractats per l’API i de quina forma han de ser realitzades les comunicacions entre el servidor i el client. Els formats més freqüents són JSON i XML. FamilySearch ofereix suport per 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 l'API de FamilySearch, especificant quin element seria considerat el Recurs, quin el Servei i quin la Representació.

* **Recurs:** Persona (informació relacionada amb una persona en concret)
* **Servei:** Obtenir informació de la persona (GET)
* **Representació:** Nom, cognoms, esdeveniments relacionats amb la vida de la persona, etcètera, en format XML o JSON.

Com també s'ha comentat, l’arquitectura REST es caracteritza per la implementació de sis restriccions imposades sobre el sistema. A continuació s'exposa amb més detall, cada una d'elles.

**Interfície uniforme (uniform interface)**

Aquesta restricció s'encarrega de definir la interfície de comunicació entre el client i el servidor.

En una arquitectura REST, s'utilitzen els protocols de comunicació HTTP i HTTPS de forma conjunta amb els \gls{URI), (Identificadors de Recurs Uniforme) per aconseguir accés als diferents recursos i operacions proporcionades per l'API.

Els verbs permesos pels protocols de comunicació web són els coneguts: 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 o HTTPS mitjançant el verb i URI especificats a continuació:

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

**Sense Estat (stateless)**

Aquesta restricció implica que el servidor no emmagatzema la informació del client. Això implica que cada petició d'aquest cap a l’API, ha de contenir tota la informació necessària perquè el servidor l'identifiqui i es defineixin les regles de comunicació adequades per processar la petició.

Hi ha exemples d'operacions a l'API de FamilySearch, com per exemple el procés d'identificació Oauth V2, que no són realment RESTful, doncs aquestes sí que guarden informació del client durant les diferents parts de la comunicació.

**Client–Servidor (client-server)**

Per comprendre aquesta restricció, cal comprendre primer en què consisteix un sistema desconnectat.

En el cas de les arquitectures REST, un sistema desconnectat implica que el client mai tindrà accés directe a les bases de dades que emmagatzemen la informació, i que per tant, sempre haurà d'accedir a les dades mitjançant l’intermediari. En aquest cas, l'API.

Els protocols de comunicació descrits prèviament (HTTP i HTTPS), i la interfície de comunicació, són els encarregats de gestionar les comunicacions entre client i servidor.

**Emmagatzematge en el client (cacheable)**

Aquesta restricció fa referència a si les respostes retornades, des del servidor, al client, poden ser emmagatzemades per aquest i durant quant de temps les pot guardar. Existeixen tres nivells de configuració diferents:

* **Implícit:** Si és el client el que decideix quant de temps guardarà les dades o informació retornada, es tracte d'emmagatzematge implícit.
* **Explícit:** Si és el servidor el que mana i posa les regles, parlem d'emmagatzematge explícit.
* **Negociat:** Quan el client i el servidor negocien i arriben a un acord, es tractea d'emmagatzematge negociat.

**Sistema per capes (layered system)**

Aquest principi, o restricció, es basa en el fet que el client no pot assumir que tindrà connexió directa amb el servidor. És a dir, poden existir diferents intermediaris en forma de hardware i software entre client i servidor.

Això, facilita l’escalabilitat i persistència del sistema gràcies al fet que el client no s’ha de preocupar de comunicar-se amb elements o tecnologies específiques. D'aquesta forma, el servidor es pot veure subjectes a canvis de forma transparent pels clients.

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

Restricció que regula com, de forma excepcional, el servidor pot proporcionar accés al client sobre certes parts de la lògica del funcionament. Alguns exemples poden ser els *Java Applets* o blocs de codi *JavasScript*.

**Conclusió sobre les arquitectures REST**

En conclusió, seguir les regles marcades per una arquitectura REST permet assegurar els següents beneficis de cara a la gestió de les comunicacions entre clients i servidor: escalabilitat, simplicitat, modificabilitat, visibilitat, portabilitat i fiabilitat.

**Formats de dades utilitzats pel sistema**

FamilySearch utilitza tres formats de dades diferents per representar la informació emmagatzemada en les seves bases de dades i dos formats extres per codificar aquesta informació i enviar-la a través del núvol.

Els conjunts de dades utilitzats per representar els recursos són els que segueixen:

* Les dades genealògiques es representen mitjançant el format GEDCOM X.
* Els recursos o objectes específics del model de FamilySearch, es representen mitjançant una extensió del model de dades GEDCOM X.
* El format de dades Atom, o atòmic, s’utilitza per proporcionar un format simple per les meta-dades.

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

El terme \gls{GEDCOM}, és un acrònim de l'anglès Genealogical Data Communications. (Comunicació de Dades Genealògiques)

El format GEDCOM consisteix en un conjunt de regles d'aplicació per tal de representar informació genealògica. Aquest format de dades, creat per FamilySearch l'any 1984, s'ha convertit en l'estàndard de la indústria.

Per simplificar-ho, podríem entedre un fitxer en format GEDCOM, com un fitxer de text que emmagatzema informació genealògica d’una persona i les metadades necessàries per poder enllaçar als diferents fitxers de la mateixa persona.

Tot i que l’última versió, datada del 1996, segueix sent molt utilitzada, FamilySearch va proposar durant l'any 2012, canviar aquest estàndard per la seva nova versió anomenada GEDCOM X.

El format de dades per la \gls{GEDCOM X}, representava un nou projecte de codi obert i es diferenciava del seu antecessor en la implementació d'un sistema que facilitava la inclusió d'arbres genealògics i fonts de dades als recursos ja existents.

Al mateix temps, el nou estàndard també donava suport a l'intercanvi i enllaçament de dades a través del núvol i es creava així la primera versió de l'API de FamilySearch.

A continuació, a la taula [ref] s'ofereix un petit exemple de com un recurs és codificat part del recurs persona sota el format de dades GEDCOM X.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | Gènere (GEDCOM X) | OPCIONAL. |
| Names | Els noms de la persona | Llista de Noms. (GEDCOM X) L'ordre d’entrada és preservat. | OPCIONAL. Si més d’un nom és introduït, s’assumeix que aquests han estat introduïts en ordre de preferència, amb el més preferit, introduït primer. |
| Facts | Esdeveniments relacionats amb la vida d’una persona. | Llista d'Esdeveniments (GEDCOM X). L'ordre d'entrada és preservat. | OPCIONAL. |

Així doncs, podem veure com la instància del recurs Persona conté un camp booleà, que indica si aquesta pot ser utilitzada de forma pública o només en l'àmbit privat 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 recurs amb l’estructura que s’exposa a la taula[ref2] i els valors possibles per l’enumeració de gènere, s’indiquen en la taula [ref3]

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

|  |  |
| --- | --- |
| **Gènere URI (Gedcom X)** | **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 a través de l'API de FamilySearch, no entrarem més en detall en aquests formats.

L’objectiu d’aquest apartat era explicar quin és l'estàndard de representació de dades genealògiques utilitzat per l'API. Per qualsevol informació extra que es vulgui consultar, s’adjunta a la bibliografia del projecte, l’enllaç a la documentació del model conceptual.

**Format de dades FamilySearch**

El format de dades FamilySearch, defineix el format d'aquells objectes específics relacionats amb la plataforma de dades pròpia de l'organització. Això implica que aquestes estructures no formen part de cap estàndard i manquen de sentit fora del context pel qual han estat definides.

L'estructura dels objectes específics de FamilySearch ha estat creada com una extensió de l'especificació GEDCOM X. Per tant, segueixen una estructura molt similar.

**Format de dades Atom (o atòmic)**

Els formats de dades Atom, o atòmic, és utilitzat per proporcionar un format pel contingut web i les metadades.

Aquest format és utilitzat, entre altres llocs, en les col·leccions ordenades de resultats, com podrien, per exemple, les respostes a la funció de cerca de persones o l'obtenció del historial de canvis d’una persona.

**Codificacions dels formats de dades**

Els formats de dades que s’han exposat en els apartats anteriors no són més que unes convencions que marquen l'estructura a seguir per representar els diferents objectes o recursos utilitzats per l'API de FamilySearch.

Tanmateix, aquestes estructures han de ser codificades per tal de poder ser transmeses a través del núvol i en concret, FamilySearch, proporciona suport a les dues codificacions més comunes i utilitzades per aquesta finalitat. Els llenguatges XML i JSON.

**El llenguatge XML**

El llenguatge de marcatge extensible (XML), és un llenguatge de marcatge que defineix un conjunt de regles a seguir per tal de codificar, documents i informació, en un format llegible i processable, tant per éssers humans, com màquines.

El llenguatge va ser definit pel Consorci World Wide Web i tracta d’emfatitzar la simplicitat, generalitat i usabilitat del model, per l'ús a través d'Internet.

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

<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 observar, cada camp, objecte o peça d'informació, es troba envoltada per dues etiquetes que en marquen l'inici i final. El text situat a l'interior d'aquestes etiquetes, indica de quin camp es tracta. Per exemple, pel camp subjecte, tenim les etiquetes *<subject>* i *</subject>*.

**El llenguatge JSON**

El format Notació d'Objectes JavaScript (JSON), és un estàndard de format obert que, de la mateixa forma que el llenguatge XML, pretén crear codificacions llegibles tant per éssers humans com màquines 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 d'un objecte a representar està format per una clau i un valor associat a aquesta clau.

El llenguatge JSON deriva del JavaScript. Al principi, només aquesta plataforma incorporava funcions per codificar i descodificar aquest llenguatge de marcatge.

Durant els últims anys, el format JSON s'ha vist convertit en l'estàndard de la indústria per l'intercanvi de dades a través d'Internet i en conseqüència, ha provocat que molts altres llenguatges de programació hagin incorporat 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 forma següent: ·

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

**Evolució temporal de l'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 l'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 l'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 l'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 l'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 l'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 l'API accedeix a les dades.

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

L'API de FamilySearch no es caracteritza només per ser el centre d'informació obert amb més registres genealògics, sinó que a més a més l'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 l'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 l'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 l'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 l'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 l'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. L'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.

**Transferència de registres en grans quantitats**

Existeix la possibilitat per les organitzacions de signar un acord amb FamilySearch que els hi permeti mantenir la seva pròpia copia de les dades emmagatzemades a l'API de FamilySearch referents a persones difuntes.

Per realitzar aquestes transferències de grans quantitats de dades, FamilySearch disposa d'una API especial encarregada de mantenir i actualitzar les dades emmagatzemades de forma remota a través de diferents sessions.

Aquesta opció pot permetre a altres organitzacions ordenar les dades d'una forma diferent en les seves pròpies bases de dades i habilitar-les per estudis diferents. Com per exemple, estudis de mineria de dades.

**Eines de desenvolupament**

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

Són dos els recursos principals que permeten apropar la utilització de l'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 l'API de FamilySearch sense haver de preocupar-nos de forma directa en com aquestes recursos han de ser demanats contra l'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 l'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 l'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 l'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 l'API.

Entre aquests, destaquen differents proxy que serveixen per simular peticions HTTP i HTTPS contra l'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 ha tres menes 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.
* **Certificació per transferència d'arxius en grans quantitats:** FamilySearch ofereix la possibilitat a les organitzacions que signin un permís especial, la possibilitat de mantenir una copia pròpia de les dades emmagatzemades per FamilySearch. En cas d'utilitzar l'API específica dedicada a la transferència d'aquests blocs de dades, cal certificar-ne també les operacions.

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

Un cop una aplicació ha estat certificada, en cas de que aquesta modifiqui les operacions de lectura o escriptura que realitza, cal tornar a certificar-la abans de desplegar els canvis a producció. Existeix un procés encarregat de controlar que es compleixen els estàndards marcats per FamilySearch i que de no complir-se pot acabar significant el retirament dels drets d'accés a producció.

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 l'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 l'API es va actualitzant a noves versions. Quan es vol realitzar un canvi en les funcionalitats o estructura de l'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 l'API.