**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 diversos canvis i modificacions al llarg del temps, algunes, amb més repercussió i implicacions que altres. La llista de canvis és prou llarga com per no intentar reproduir-la en la memòria. De totes maneres, aquesta és adjuntada a la bibliografia del projecte.

L’últim canvi introduït a l'API i de fet, probablement, dels més importants que s'han produït fins ara, fa referència a la nova estructura del back-end anomenada Tree Foundation.

A falta de documentació oficial al respecte, es va consultar un dels Webinair passats, oberts al públic per part de FamilySearch, per tal de comprendre millor la magnitud d'aquests canvis i quines implicacions tenia. En el següent apartat, tractarem més a fons en què consisteix Tree Foundation.

Les primeres fases d'aquest darrer canvi van arribar a producció a mitjans del mes de juliol, moment en què 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, trencant-ne moltes de les funcionalitats.

Al final, els principals problemes que va causar pel projecte van ser interrupcions permanents en l'entorn de proves Sandbox i la inhabilitat d’accedir als recursos Font de Dades i Discussió relacionats amb una persona, a causa de la no adaptació del SDK oficial.

**Tree Foundation**

Com s'ha comentat en l'apartat anterior, Tree Foundation és la nova versió del back-end que FamilySearch està implementant. A continuació, es detallen en més profunditat les implicacions d'aquest canvi, ja que pot tenir un gran impacte en com les principals dades emmagatzemades per l'API hauran de ser accedides en un futur no molt llunyà.

El projecte de Tree Foundation consisteix a canviar per complet el motor que fa funcionar el back-end de FamilySearch.

El motor actual, anomenat de forma interna, Conclusion Tree i que fins ara, era l’estructura amb la que l'API es comunicava, caurà en desús 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 l'elevada demanda de dades sota la que l'API es veu sotmesa. En la imatge [ref] es pot veure com es passa d’un sistema unificat, a un sistema distribuït.

Els beneficis esperats del canvi són un increment en l'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 avui en dia, encara no es troba a producció, és integrar dins del recurs d’una persona tota la informació que s’acostuma a consultar i que ara no contenia. 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. D'aquesta forma, s'aconseguirà alliberar part la càrrega que suporta el sistema i millorar-ne, per tant, l'eficiència i disponibilitat. La figura [ref2] representa de forma visual el que aquest canvi implica en el recurs Persona.

Quan Tree Foundation va arribar a producció, només els recursos Source i Discussion es trobaven incorporats dins del recurs Persona i els antics enllaços, a aquesta informació quedaven obsolets. Més endavant, apuntant a mitjans d’Agost, s'espera que es realitzi el mateix canvi per les relacions familiars. La figura [] mostra els canvis als quals ens estem referint.

Tot i que el canvi és molt prometedor, cal tenir en compte que té el potencial de trencar part dels SDKs oficials i moltes de les aplicacions que tractin amb la versió antiga de l'API, de la mateixa forma, que ja ho han fet les primeres versions de Tree Foundation arribades a producció.

Per tant, caldrà estar al compte de quan aquests canvis arriben a producció i de quan els SDKs oficials estaran preparats, per ser utilitzats sense error, davant d'aquesta nova versió del back-end de FamilySearch.

**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 ha estat dissenyada i pensada perquè pugui ser utilitzada de forma simultània per moltes persones, des de diferents indrets del món, d'una forma còmode i personalitzada.

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

**Caching**

El fet de voler oferir un servei de *caching* decent ha influenciat bona part del disseny de l'API. Davant la pregunta evident de, perquè és important el *caching* a FamilySearch?Són tres els punts principals de la resposta:

* El fet que el client pugui aplicar tècniques d'emmagatzematge, evita la repetició de certes peticions al sistema.
* L'existència de *reverse proxies* permet, a aquests servidors, respondre a peticions dels clients sense necessitat d'accedir als servidors i bases de dades si detecten que la informació demana, ja es troba disponible en aquests. Per tant, ajuden a reduir la càrrega del sistema.
* Davant noves peticions, els servidors poden indicar als *proxies* que les seves dades encara són vàlides i que per tant, no fa falta que realitzin una nova petició i n'esperin la resposta.

**Throttling**

La funcionalitat de *throttling* s'encarrega de posar un límit al nombre de peticions, en un cert interval de temps, que un usuari pot realitzar. Aquests límits són creats en l'àmbit d'usuari i per tant, utilitzar més d'una sessió simultània, 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 resposta indica quant de temps cal esperar entre peticions, per poder tornar a interactuar amb l'API.

Aquest servei, que evidentment, suposa una limitació pels usuaris, evita que certs usuaris de forma intencionada o involuntària, bloquegin tots els recursos disponibles per part dels servidors 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 dades 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, ha estat modificat per part de FamilySearch, per obtenir la versió del recurs consultat. Per tant, si es realitza una petició de només aquestes capçaleres, a canvi d'un petit esforç de l'API, es podria comparar si els recursos desitjats han estat modificats o no sense causar un gran impacte.

Un cop es coneix que un recurs ha estat modificat, es poden descobrir els canvis realitzats mitjançant la comparació dels dos objectes o llegint l'historial de canvis. Evidentment, sempre es podria realitzar una sobre escriptura completa 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 a adaptar una peça de software o contingut, a diferents idiomes, sense necessitat de realitzar canvis en el codi. L'API de FamilySearch permet realitzar les peticions de forma que certs conjunts de dades retornades, com poden ser per exemple, el nom de països o persones, estiguin localitzades en l'idioma especificat.

La internacionalització permet, per exemple, que un usuari que es troba a Espanya vegi el terme `living' (viu), en el seu idioma natiu. Un altre recurs que moltes pàgines web localitzen són les dates. És ben sabut que l'estructura americana d'una data, per exemple, difereix d'una data europea en la posició dels termes any, mes i dia.

El conjunt de dades que FamilySearch permet localitzar són:

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

**Transferència de registres en grans quantitats**

Existeix la possibilitat per les organitzacions que així ho desitgin, de signar un acord amb FamilySearch, que els hi permeti mantenir la seva pròpia còpia de dades oficials de FamilySearch, relativa a persones difuntes.

Per realitzar aquestes transferències, que mouen alts volums de dades, FamilySearch disposa d'una API especial que permet a les organitzacions gestionar, mantenir i actualitzar, les dades emmagatzemades en els seus sistemes a través de diferents sessions.

Aquesta opció pot permetre a altres organitzacions ordenar i emmagatzemar les dades amb una estructura diferent i habilitar, d'aquesta forma, la possibilitat de realitzar diferents tipus d'estudi. 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 aproparla utilització de l'API a qualsevol desenvolupador. Els SDKs i les aplicacions d'exemple.

**Els SDK de FamilySearch**

Els Kits de Desenvolupament de Software, són un conjunt d'eines que faciliten la creació d'aplicacions per un software, hardware, framework, sistema de computació, videojoc, sistema operatiu o plataforma de desenvolupament determinada.

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

Però, perquè esdevé útil la utilització d'un SDK? En el cas de FamilySearch un SDK ens ofereix la possibilitat de comunicar-nos amb l'API sense haver de preocupar-nos, de forma directa, en com les URIs han de ser codificades o en parcejar els JSON/XML de les respostes. Taques que poden esdevenir tedioses i repetitives.

Per tant, els SDK permeten als desenvolupadors centrar-se en el desenvolupament de funcionalitats i no en la gestió de comunicacions amb l'API.

FamilySearch disposa en l'actualitat de sis SDKs diferents. Tots ells faciliten la interacció amb l'API, encara que no tots permeten les mateixes funcionalitats. A més a més, alguns d'aquests SDK són oficials, significant que són desenvolupats i mantinguts per l'organització FamilySearch, mentre que altres, són creats i gestionats per la comunitat.

Els diferents SDK disponibles es llisten a continuació:

* **Java SDK:** El Java SDK és un SDK oficial que serveix per crear aplicacions d'escriptori amb el llenguatge Java.
* **PHP SDK:** El SDK de PHP és un SDK oficial per crear aplicacions web. Desafortunadament, no es troba del tot actualitzat i l'última versió d'aquest no integra masses de les funcionalitats que es poden realitzar contra l'API.
* **C# SDK:** El SDK de C# és un altre dels SDK oficials i serveix per crear aplicacions amb la tecnologia .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 en aquest projecte ja que cobreix gran part de les funcionalitats de l'API i en facilita l'ús.
* **Python SDK:** El SDK de Python resultaria molt atractiu de cara a tota mena d'aplicacions, però per desgràcia, encara es troba en fase de desenvolupament i sense dates concretes de finalització.
* **Ruby SDK:** La perla de Ruby és l'últim SDK disponible per FamilySearch i es tracta 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.

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

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

**Altres eines interessants**

L'últim recurs per desenvolupadors que FamilySearch ofereix, és un llistat d'eines que poden ser interessants o esdevenir útils, de cara a la realització de proves amb l'API.

Entre aquestes, destaquen diferents intermediaris que serveixen per simular peticions HTTP i HTTPS contra l'API de FamilySearch i una utilitat, no oficial, que en teoria permet copiar dades de producció als entorns de desenvolupament.

**Procés de certificació**

Per tal de poder connectar les aplicacions desenvolupades per tercers al conjunt de dades oficial de FamilySearch, aquestes aplicacions han de ser sotmeses a 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 sobre el seu funcionament, mentre que de les aplicacions d'ús comercial també són analitzades des d'un punt de vista de negoci i marketing. Com a contrapartida, les aplicacions d'ús limitat no poden aparèixer a la galeria d'aplicacions.

Hi ha tres tipus de certificacions principals:

* **Certificació de lectura:** L'aplicació ha de ser certificada per la lectura de tots aquells recursos als quals accedeix. També cal que compleixi amb certs estàndards del procés d'identificació.
* **Certificació d'escriptura:** En cas que l'aplicació realitzi operacions d'escriptura, aquestes també hauran de ser revisades per part de FamilySearch.
* **Certificació per transferència d'arxius en grans quantitats:** Per aquelles organitzacions que tinguin permís per accedir als protocols de transferència de dades en grans quantitats, cal certificar i revisar, conjuntament amb FamilySearch, les operacions realitzades contra aqueta API.

Es veuran més detalls sobre el procés de certificació en la part pràctica de la memòria del projecte.

Un cop una aplicació ha estat certificada, en cas que aquesta modifiqui les operacions de lectura o escriptura que realitza, caldrà 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 en cas de no complir-los, pot significant el retirament dels drets d'accés a producció.

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

**Diferents entorns de treball**

Es disposa de molt poca informació sobre els diferents entorns als quals es pot accedir relatius a l'API de FamilySearch. A força d'investigar diferents codis i possibilitats, s'ha arribat a la conclusió de què existeixen els següents entorns de treball:

* **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 qual l'API va desplegant 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 abans de desplegar la nova versió a producció.
* **Producció:** Entorn de treball que pot accedir a les dades oficials de FamilySearch. Totes les aplicacions aspiren a connectar-se a aquest entorn de treball.

https://familysearch.org/developers/docs/guides/evolution