**Familysearch Javascript SDK**

**Introducció**

Abans de procedir a explicar com funciona l'aplicació web desenvolupada, volíem presentar per sobre en que consisteix el SDK oficial de FamilySearch implementat en el llenguatge Javascript.

No explicarem la informació accessible a través d'aquest, doncs en realitat, es tracte de la mateixa que la presentada en la secció sis d'aquesta memòria. El que si que es desitja, no obstant, és presentar les principals funcionalitats que aquest SDK presenta i intentar exposar els motius principals pels que creiem que val la pena estudiar-ne el seu funcionament.

L'objectiu principal del SDK és el de facilitar el consum dels recursos accessibles a través de l'API de FamilySearch.

Per aconseguir-ho, el SDK envolta cada funció que realitza crides a la API amb una funció pròpia i afegeix funcions de conveniència a la resposta per tal de navegar pels resultats de forma còmode. De totes formes, el SDK no emmascara la resposta retornada per la API i els recursos JSON o XML retornats per aquesta, es troben accessibles en cas de voler accedir a una peça d'informació concreta per la que no s'ha creat una funció de conveniència.

En cas de que el SDK no estès preparat per fer front a alguna de les peticions que la API ofereix, aquest implementa un conjunt de funcions de plomeria que permeten a l'usuari realitzar amb comoditat les típiques crides GET, POST, DEL, ... , que es realitzarien de forma general contra la API, mitjançant el SDK.

Per tant, el SDK ens ofereix una cobertura casi complerta de la API oficial i en cas de que alguna consulta no es trobes suportada, sempre es podria realitzar mitjançant la via tradicional.

A continuació, s'expliquen les principals funcionalitats d'aquest SDK i en que consisteixen.

**Crides REST a la API envoltades per funcions del SDK**

Qualsevol crida que es pretengui realitzar contra la API de FamilySearch es realitza mitjançant una funció Javascript asíncrona que l'emmascara. Això ens permet deixar de preocupar-nos per les URI dels recursos als que volem accedir o haver d'afegir les capçaleres correctes a cada petició, ja que és el mateix SDK el que s'encarrega de fer-ho i de mantenir el conjunt d'URIs actualitzat.

Per exemple, si volem accedir a la persona amb identificador: 'KW7S-VQJ', ho podríem fer mitjançant la següent funció.

client.getPerson('KW7S-VQJ', {persons:true}).then(function(response) {  
 ...   
});

**Implementació basada en Promeses**

Les promeses són els objectes retornats pels blocs de codi Javascript asíncrons, com per exemple, la funció de l'apartat anterior. Una promesa es troba sempre en un dels següents tres estats:

* Pendent: Estat inicial de la promesa.
* Satisfeta: L'estat de la promesa representa una operació finalitzada amb èxit.
* Rebutjada: L'estat de la promesa representa una operació fallida.

Tan bon punt una promesa rep l'estat de satisfeta o rebutjada, ja no pot tornar a canviar d'estat.

La gracia de les promeses es que tan bon punt són resoltes, permeten executar una part del codi definida amb anterioritat (el que seria el cos de la funció de l'apartat anterior) i mentre aquestes no han de ser executades, la resta del codi pot seguir executant-se. Amb altres paraules, el codi no roman bloquejat mentre espera la resposta de la funció contra la API.

**Implementació orientada a la programació d'objectes**

Les funcions de crida del SDK, retornen objectes, o promeses, com bé s'ha explicat en l'apartat anterior. Aquests objectes disposen de funcions de conveniència que permeten accedir als resultats de la resposta sense haver de navegar per objectes de format XML o JSON.

Per exemple, en la crida de fa dos apartats podríem obtenir el nom de la Persona cercada mitjançant la navegació al recurs de la persona i seguidament, demanant-ne el nom.

client.getPerson('KW7S-VQJ', {persons:true}).then(function(response) {  
 console.log(response.getPrimaryPerson().getDisplayName());  
});

**Model de dades pràcticament idèntic al de FamilySearch**

Una altre característica destacable del SDK és que implementa un model de dades equivalent al de la API de FamilySearch, però en aquest cas, pensat per ser navegat mitjançant els estàndards dels llenguatges orientats a la programació d'objectes, en comptes dels enllaços Hypermedia.

A la figura [] podem observar el model de dades proposat pel SDK i com cada un dels objectes es troba relacionat amb els altres.

**Captura d'errors**

Els errors de connexió amb la API són fàcilment tractables gràcies a la naturalesa de les crides asíncrones. Només cal afegir la clàusula '.catch' al codi de la petició com es mostra en el següent bloc de codi.

client.getPerson('KW7S-VQJ', {persons:true}).then(function(response) {  
 ...  
})

.catch(function(error) {  
 // Codi per gestionar l'error  
});

**Intentar de nou peticions GET fallides**

En cas de que una petició GET falli per motius transitoris, aquesta es reintentada automàticament per part del SDK.

**Gestió automàtica del 'Throttling'**

Com s'exposava en la secció quarta de la memòria, quan es parlava d'algunes de les funcionalitats extres de la API de FamilySearch, existia la funcionalitat de Throttling que evitava que un usuari realitzes 'masses' peticions en un període de temps limitat.

En cas de que l'usuari del SDK sigui bloquejat durant un període de temps per aquesta causa, el mateix SDK s'encarregarà de gestionar-ne la duració, evitar seguir llençant peticions contra la API i reintentar les crides tant bon punt sigui possible.

**Autentificació mitjançant un Pop-up**

Mentre es realitzi des d'un navegador, l'autentificació pot ser gestionada mitjançant un pop-up a la pàgina oficial de FamilySearch.

D'aquesta forma, no ens hem de preocupar per crear una pàgina de redirecció específica pel protocal OAuth. L'únic requisit és registrar una URI a la que es cridarà un cop es finalitzi el procés d'identificació en el mateix domini i port que el codi de la nostre aplicació.

**Autentificació automàtica**

Existeix la opció d'activar el protocol d'identificació de forma automàtica si l'usuari intenta realitzar una petició contra la API sense estar identificat. Pot esdevenir útil ja que les connexions expiren al cap d'un temps d'inactivitat.

**Emmagatzematge del Token en una cookie**

Existeix l'opció d'emmagatzemar el token d'identificació (retornat per la operació d'identificació amb FamilySearch) en una cookie.

Resulta útil de cara a utilitzar el SDK des de la capa del controlador, ja que no cal anar identificant l'usuari a cada pàgina diferent o implementar una aplicació web basada en una sola pàgina. Malauradament, viola una de les condicions per la certificació de les aplicacions, així que no s'acaba d'entendre perquè apareix en un SDK oficial mes enllà d'ajuda durant el desenvolupament.

Per tal d'intentar superar aquesta restricció, en el nostre cas hem emmagatzemat el valor del Token en el espai local del navegador.

**Disparador automàtic de la funció d'expiració**

Per tal de mantenir la coherència entre l'aplicació web i l'estat de la identificació, recordem que pot expirar per inactivitat, existeix la possibilitat d'implementar una funció que s'executi quan el token expira.

D'aquesta forma, podem garantir que en tot moment existeix una concordança d'estat entre el client i el servidor i que per tant, no correm el risc de que l'usuari pugui realitzar operacions no permeses al expirar el token.

**Utilitzable des de diferents plataformes**

El SDK és utilitzable tant des de la part del client com el servidor. En altres paraules, pot correr tant mitjançant la tecnologia Javascript en la capa del controlador com mitjançant Node.js a la capa del back end.

**Suport en el desenvolupament**

Una de les característiques principals del SDK és que es troba relativament ben documentat. Inclòs m'atreviria a dir, que de cara a obtenir una idea general de com funciona la API de FamilySearch i quina informació conté, realitza una millor feina que la pròpia documentació oficial de la API. Encara que sigui de forma involuntària, es digne de destacar.

Al mateix temps, existeixen diferents grups d'ajuda per aquells desenvolupadors que desitgin utilitzar l'API de FamilySearch i que són visitats de forma regular pels desenvolupadors dels SDK oficials i la API de FamilySearch.

Des d'aquesta memòria, es recomana passar pel grup de Google 'FamilySearch Developer Network', ja que resulta l'emplaçament ideal per qualsevol dubte que puguem tenir i un centre d'informació ideal sobre el funcionament de la API i l'estat en el que es troba a cada moment.

**Conclusió**

Com s'ha pogut observar en les seccions anteriors són molts els beneficis d'utilitzar el SDK oficial en comptes de realitzar una implementació directa contra la API. En contrapartida, el preu a pagar es pràcticament nul, més enllà d'haver d'estudiar el funcionament d'aquest apart del de la API, però resulta un esforç que val la pena.