¡¡Hola a todos, bienvenidos al **Desafío 2** de la Comunidad Online de Club de Apps!!

Este tutorial está pensado para que junto a sus *Clubes de Apps* puedan crear una aplicación que se convierta en una forma directa de información ante un problema que nos afecta a todos nosotros como país: los movimientos sísmicos. Como es sabido, Chile es un país sísmico que registra una enorme cantidad de temblores y terremotos en su historia contando incluso con el terremoto más fuerte de la humanidad que se haya registrado de manera instrumental, como lo fue en Valdivia el año 1960. Por lo mismo es que nunca está demás contar con instrumentos que permitan conocer información oportuna respecto a los distintos movimientos que se van registrando. El Servicio Geológico de Estados Unidos o USGS por sus siglas en inglés (United States Geological Survey) es una agencia científica gubernamental de Estados Unidos que se encarga de investigar los terrenos, recursos naturales y los posibles peligros naturales que los amenazan. Es por eso que disponen de una gran cantidad de programas que buscan entregar información oportuna a las personas que les permita tomar decisiones correctas a la hora de enfrentarse a algún tipo de desastre natural. Una de las actividades que realiza la USGS es el monitoreo de movimientos sísmicos en el mundo contando con instrumentos que permiten disponer información en tiempo real de los distintos sucesos que van ocurriendo ante la manifestación del desastre natural. En esta oportunidad nos enfocaremos en un recurso de mucha utilidad que entrega la USGS para obtener información a tiempo de los temblores y terremotos que están ocurriendo en el mundo. Para ello, la USGS dispone de un recurso online que permite consultar cuáles han sido los distintos movimientos sísmicos que han ocurrido durante un periodo de tiempo, incluso permitiéndonos clasificarlos según la magnitud que queramos. En este tutorial utilizaremos este recurso para crear una aplicación móvil que nos permita ver cuáles han sido los sismos mayores a 4.5° richter que han ocurrido durante el último mes en Chile y el mundo.

## Qué Construirás:

Para esta versión de la aplicación vamos a implementar las siguientes funcionalidades:

1. Consultar la información entregada por la USGS para los sismos 4.5+ ocurridos en el mundo durante el último mes.
2. Desplegar mediante selectores de lista el listado de todos los sismos 4.5+ ocurridos en el mundo durante el último mes.
3. Mostrar en la aplicación información relevante para cada sismo (ubicación, latitud, longitud, grado, entre otros).
4. Poder filtrar la información y mostrar solamente los sismos ocurridos en Chile.

5. Mostrar mediante Google Maps la ubicación aproximada del epicentro.

## Qué aprenderás:

Este tutorial cubre los siguientes componentes y conceptos:

1. Uso del componente **Selector de Lista** para mostrar el listado de los sismos en Chile y el mundo.
2. Uso del componente **Botón** para lanzar la ejecución de Google Maps con la información del sismo.
3. Uso del componente **Etiqueta** para mostrar los distintos datos de interés asociados al sismo.
4. Uso del componente **Web** para poder consultar información generada por un sitio web externo
5. Uso del componente **Activity Starter** para poder abrir aplicaciones externas.
6. Uso de ***procedimientos*** para implementar comportamientos repetitivos, como procesar el texto obtenido tras la consulta al sitio de la USGS.
7. Uso de ***listas*** para poder manipular gran cantidad de información
8. Conocer, entender y trabajar con datos trabajados en **JSON**

## Para Empezar

Tal como lo hemos hecho en otras ocasiones, lo primero que haremos será definir el aspecto que tendrá la interfaz gráfica de la aplicación. Siguiendo la metodología con la que hemos trabajado en las aplicaciones anteriores, tendremos el desarrollo dividido en dos partes. Una primera, que estará enfocada principalmente en el diseño que tendrá la aplicación y establecer cómo ésta va a interactuar con el usuario y una segunda etapa, donde programaremos el comportamiento que va a tener nuestra aplicación y en cómo procesaremos la información que vamos a obtener del sitio de la USGS.

### Diseñando los componentes:

Dado que ustedes ya han realizado más de una aplicación antes, lo que les ha permitido conocer en detalle los distintos componentes que tiene App Inventor para la creación de interfaces, para este desafío incentivamos la creación libre del aspecto visual de nuestra app. La idea es que habiendo entendido los objetivos del desafío puedan libremente elegir los componentes que crean ustedes entregarán los mejores resultados. El diseño que se presentará en este tutorial es netamente demostrativo y tiene la finalidad de explicarles el comportamiento que va a tener la aplicación que vamos a realizar, pero si lo desean pueden establecer su propio criterio para crear las interfaces. ¡Te incentivamos a buscar junto a tu Club de Apps los mejores

componentes que permitan sacarle el máximo provecho a la aplicación!

Para el diseño con el cuál vamos a trabajar en este tutorial vamos a necesitar que existan dos componentes que nos permitan mostrar el listado de los sismos de grado 4.5+ que han ocurrido en Chile y el mundo. Dado que esta información la vamos a consultar en tiempo real desde un sitio web en internet, no podemos saber al momento de crear la app cuántos datos vamos a tener que mostrar en pantalla. ¿Qué quiere decir esto? Que vamos a necesitar de un componente que nos permita mostrar tantos datos como resultados tenga nuestra consulta. Para ello, y para sacarle el máximo provecho a las herramientas que **App Inventor** nos entrega, es que vamos a utilizar desde el *Diseñador* los siguientes componentes:

- Un **Selector de Lista** para mostrar la lista de todos los sismos en el mundo.
- Un **Selector de Lista** para mostrar la lista de todos los sismos ocurridos en Chile
- Un componente **Web** que nos permitirá consultar por información a un sitio web externo.
- Un **ActivityStarter** que nos permitirá abrir Google Maps para mostrar el punto aproximado del epicentro del sismo.
- **Botones**, **etiquetas** y **disposiciones** para mostrar y administrar la forma en la que entregamos la información al usuario.

La imagen a continuación muestra un diseño tentativo para la aplicación. Recuerden que puedes personalizar el diseño como estimes conveniente.

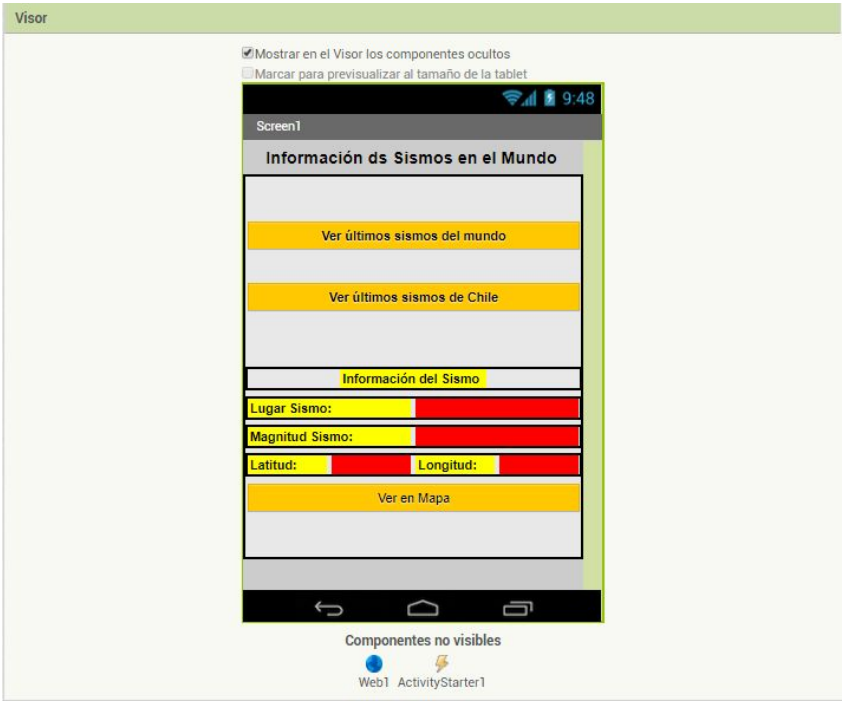


Figura 1: Interfaz principal de nuestra aplicación

**Creando la pantalla principal de la aplicación:**

Para lograr que la aplicación se vea como la figura anterior van a necesitar trabajar con disposiciones que permitan ordenar distintos elementos que agregaremos al *visor* del **Diseñador.** Para lograr que los elementos se encuentren centrados en la pantalla desde un comienzo, deberás establecer la propiedad ***DispHorizontal*** del **Screen1** como "*Centro".* De esa forma forzaremos a que cada elementos que se agregue al visor automáticamente se encuentre centrado, en relación al eje horizontal. El resto de los elementos presentes los agregaremos de la siguiente manera:

- Arrastra una *etiqueta* desde la *Paleta* y suéltala en el visor. Esta etiqueta nos permitirá mostrar un título el que se verá en todo momento que se ejecute nuestra aplicación.
    - Cambia el mensaje por defecto de la *etiqueta* desde su propiedad *texto* y haz que muestre el mensaje: "*Información de Sismos en el Mundo***".** Recuerda que puedes escoger el título que quieras.
    - Por último, y para no confundirnos, vamos a renombrar desde la sección componentes el nombre de esta etiqueta. En esta oportunidad dejaremos el nombre como *"etiquetaTítulo"*
- Arrastra una *Disposición Vertical* y suéltala inmediatamente después de la *etiquetaTítulo* que acabamos de agregar. Esta disposición nos permitirá tener control sobre los elementos que iremos agregar en la pantalla.
- Arrastra desde la *Paleta* un ***Selector de Lista.*** El *Selector de Lista* es un tipo especial de botón que una vez presionado nos permite mostrar una lista de valores.
    - Cambia el mensaje por defecto del *Selector de Lista* desde su propiedad *texto* y haz que tenga el valor: "*Ver últimos sismos del mundo".* Al presionar en este botón deberán listarse todos los sismos ocurridos en el mundo cuya magnitud haya sido de 4.5+.
    - Desde la sección componentes, renombra tu *Selector* para evitar confusiones a la hora de programar. Establece el nombre para este componente como el siguiente: *SelectorVerSismosMundo*
- Arrastra una *etiqueta* y suéltala inmediatamente debajo del *Selector de Lista* agregado anteriormente.
    - Cambia su propiedad *texto* reemplazando el valor por defecto por un texto vacío. Esta etiqueta permitirá hacer un espacio entre dos botones. Utilizaremos frecuentemente este método cuando queramos espaciar varios elementos en nuestra interfaz.
- Arrastra desde la *Paleta* un segundo *Selector de Lista.*
    - Cambia la propiedad *texto* del *Selector* y haz que tenga el valor: *"Ver últimos sismos en Chile".* Al presionar este botón deberán listarse todos los sismos ocurridos en Chile cuya magnitud haya sido de 4.5+.

- Tal como lo hicimos anteriormente, desde la sección componentes, renombra tu *Selector* para evitar confusiones a la hora programar. Establece el nombre para este componente como *SelectorVerSismosChile.*

- Agrega una nueva separación para tus componentes. Para ello arrastra dos etiquetas desde la *Paleta* cambiando su propiedad *texto* por un mensaje vacío. Verás que acabas de crear un espacio para el siguiente componente que vas a agregar.
- Agrega una *Disposición Horizontal* justo debajo de la última *etiqueta.* Desde la *Paleta* arrastra una nueva *etiqueta* y suéltala en el interior de la *Disposición Horizontal* que acabas de agregar.
  - Cambia la propiedad *texto* de la *etiqueta* para que muestre el mensaje *"Información del Sismo"*
  - Desde la sección componentes, renombra la etiqueta por *"EtiquetaSubTítulo".*
- Agrega una nueva *Disposición Horizontal* justo debajo de la disposición horizontal que agregaste anteriormente. Desde la *Paleta* arrastra dos nuevas etiquetas y suéltalas en el interior de esta nueva *Disposición Horizontal.*
  - Desde las propiedades de tu *Disposición* quita el ticket del atributo visible. La idea es que ningún elemento sea visible al momento de ejecutar tu aplicación.
  - Cambia la propiedad *texto* de la primera *etiqueta* para que muestre el mensaje: ***"Lugar Sismo: "***
  - Cambia la propiedad *texto* de la segunda *etiqueta* para que muestre un texto vacío. Posteriormente vamos a definir el valor de esta propiedad mediante bloques.
  - Desde la sección componentes, renombra la segunda etiqueta por *"EtiquetaLugarSismo".* Aquí escribiremos dónde ocurrió el temblor/terremoto una vez que hayamos obtenido la información desde el sitio web.
- Agrega otra *Disposición Horizontal* justo debajo de la *Disposición* agregada anteriormente. Esta *Disposición* la usaremos para mostrar la magnitud del sismo. Desde la *Paleta* arrastra dos nuevas etiquetas y suéltalas en el interior de la *Disposición.*
  - Desde las propiedades de tu *Disposición* quita el ticket del atributo visible. La idea es que ningún elemento sea visible al momento de ejecutar tu aplicación.
  - Cambia la propiedad *texto* de la primera *etiqueta* para que muestre el mensaje: ***"Magnitud Sismo: "***
  - Cambia la propiedad *texto* de la segunda *etiqueta* para que muestre un texto vacío. Posteriormente vamos a definir el valor de esta propiedad mediante bloques.
  - Desde la sección componentes, renombra la segunda etiqueta por *"EtiquetaMagnitudSismo".* De la misma forma que ocurría con el lugar del sismo, una vez obtenidos los datos desde el sitio web escribiremos acá la magnitud del temblor/terremoto.

- Agrega la última *Disposición Horizontal* justo debajo de la Disposición que agregaste anteriormente. En esta oportunidad, arrastra desde la *Paleta* 4 etiquetas las que deberás soltar en el interior de la *Disposición* que acabas de agregar.
    - Desde las propiedades de tu *Disposición* quita el ticket del atributo visible. La idea es que ningún elemento sea visible al momento de ejecutar tu aplicación.
    - Cambia la propiedad *texto* de la primera *etiqueta* para que muestre el mensaje: ***"Latitud: "***
    - Cambia la propiedad *texto* de la segunda *etiqueta* para que muestre un texto vacío. Posteriormente vamos a definir el valor de esta propiedad mediante bloques.
    - Desde la sección componentes, renombra la segunda *etiqueta* por *"EtiquetaLatitudSismo".*
    - Cambia la propiedad *texto* de la tercera *etiqueta* para que muestre el mensaje: ***"Longitud: "***
    - Cambia la propiedad *texto* de la cuarta *etiqueta* para que muestre un texto vacío. Posteriormente vamos a definir el valor de esta propiedad mediante bloques.
    - Desde la sección componentes, renombra la cuarta *etiqueta* por *"EtiquetaLongitudSismo".*
- Arrastra desde la *Paleta* un botón el que deberás soltar justo debajo de la última *disposición* que agregaste.
    - Cambia el valor de la propiedad *texto* para que muestre el mensaje: *"Ver en Mapa"*
    - *Quita el ticket desde la propiedad visible. La idea es que el botón no sea visible al momento de ejecutar tu aplicación.*
- Arrastra un componente ***Web*** desde la subsección ***Conectividad***, en la *Paleta*. Recuerda que no lo verás en el visor ya que es un *Componente No Visible.*
- Arrastra un ***ActivityStarter*** desde la subsección *Conectividad* en la Paleta. Recuerda que no lo verás en el visor ya que es un *Componente No Visible.* La tabla a continuación muestra los valores que debe tener tu *ActivityStarter* para que abra Google Maps de manera correcta.

| Propiedad | Valor |
|---|---|
| Acción | *android.intent.action.VIEW* |
| Clase | *com.google.android.maps.MapsActivity* |
| Paquete | *com.google.android.apps.maps* |

Figura 2: Propiedades y valores del ActivityStarter para abrir Google Maps

Si has seguido todas las instrucciones, tu aplicación debería lucir tal como se muestra en la figura 1. Dado que el diseño planteado es solo una propuesta, no hemos descrito el paso a paso para que las etiquetas mantengan el mismo aspectos que lo que se vio en la imagen. ¡Solo debes jugar con las propiedades para lograr un aspecto similar! Como el visor no es capaz de ocultar los elementos que no están visibles, es imposible hacerse una idea de cómo se verá tu aplicación una vez instalada en los equipos. Por eso la imagen a continuación muestra cómo se debería ver la app en los teléfonos (se ha escogido una imagen de fondo para que la app se vea más completa)


Figura 3: Interfaz final de nuestra aplicación y Selector de Lista con la información de los sismos.

Por último, a partir de la información entregada por el *Selector de Lista* y una vez que se haya seleccionado alguno de los sismos entregados por esta selección, la interfaz de nuestra app deberá mostrar el detalle del sismo en cuestión. Esto es, magnitud, ubicación y coordenadas del sismo. La imagen a continuación muestra el aspecto que debería tener la aplicación:


Figura 5: Interfaz básica para mostrar la información del sismo.

## Agregar comportamiento a los componentes

Como lo hemos hecho en otras ocasiones, desde el ***Editor de Bloques*** vamos a implementar el comportamiento que tendrá cada uno de los componentes que agregamos previamente a nuestra aplicación. En esta oportunidad son varios los procesos que realiza nuestra aplicación para poder mostrar la información de los sismos correctamente así que iremos agregando los bloques paso a paso.

Lo primero que vamos a programar será el componente **Web.** La importancia y utilidad que tiene este componente es que nos permite gestionar solicitudes de tipo *HTTP* con sitios externos. ¿Qué quiere decir esto?, supongamos que queremos enviar un formulario contenido en nuestra app a un servidor externo para que registre los datos de un usuario, para enviar el contenido debemos hacer una solicitud al servidor avisándole que le enviaremos datos; esto es lo que a menudo se conoce como un ***request.*** Dependiendo el tipo de request que vamos a hacer (pedir datos, enviar datos, borrar datos, actualizar datos) el servidor nos responde avisándonos del éxito de la petición. En este tutorial vamos a realizar una solicitud para consultar la información acerca de los sismos 4.5+ ocurridos en el mundo y procesar esa respuesta para mostrar la información en nuestra aplicación.

## Consultado la información al servidor de la USGS

La idea es que nuestra aplicación muestre la información de los sismos de la manera más precisa posible, por lo que vamos a necesitar tener los datos más actuales que encontremos en la

web. Por fortuna, la USGS ha liberado de manera gratuita un recurso en línea que permite obtener el listado de todos los sismos ocurridos en el mundo según filtros que nosotros elijamos. A este recurso se le conoce como ***API Web.*** Mediante un API es posible entregar a los usuarios distinto tipo de información según el tipo de consultas que el API web nos permita realizar. Es cada vez más común ver que muchos servicios en internet permitan que los usuarios obtengan información de sus plataformas a partir de APIs. Por ejemplo Facebook, Instagram, Twitter, entre otros, tienen APIs que permiten a los desarrolladores conocer los procesos que usan en sus aplicaciones para poder darle a los programadores herramientas que permitan integrar estos servicios con sus propias aplicaciones. Por ejemplo si alguien quiere crear una aplicación que me permita mostrar todos los tweets en base a un hashtag en específico, Twitter mediante su API nos permite conectarnos con esta información. Otro ejemplo podría ser el querer desarrollar una aplicación que permita acceder a los intereses y gustos de todos mis amigos de Facebook, para lo cual Facebook ha liberado un *API* de desarrollo. El concepto de *API* es mucho más extenso que el ejemplo descrito anteriormente por lo que para la creación de esta aplicación nos vamos a centrar un un tipo específico, que es aquella *API* que nos permite obtener información almacenada en bases de datos mediante una dirección url en particular.

Si revisamos el sitio web de la USGS, que es el https://earthquake.usgs.gov/, encontraremos que dispone de un servicio web que permite consultar la información de los sismos que están ocurriendo en el mundo. Esta información es almacenada y mantenida por la USGS por lo que solo nos queda preguntar a la página cada cierto tiempo para que ésta nos entregue la información de los sismos solicitada (todo esto a través del *API*). En la siguiente dirección web, https://earthquake.usgs.gov/earthquakes/feed/v1.0/geojson.php, se detallan los distintos tipos de consultas que podemos hacer al servicio web y se explican también los distintos tipos de respuesta que podemos obtener de él. A continuación se vamos a listar alguna de los posibles consultas que podríamos realizar con este servicio web.

- Para obtener información actualizada cada 5 minutos, de terremotos de importancia ocurridos durante la última hora debemos realizar una petición a la siguiente url:

  https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/significant_hour.geojson,

- Para obtener información actualizada cada 5 minutos, de terremotos de importancia ocurridos durante la última semana debemos realizar una petición a la siguiente url:

  https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/significant_week.geojson

- Para obtener información actualizada cada 5 minutos, de sismos de grado 4.5+ ocurridos durante los últimos 30 días debemos realizar una petición a la siguiente url:

  https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/4.5_month.geojson

Existen muchas otras opciones posibles de consultas por lo que dependerá de cada uno elegir alguna y procesar la información que nos devuelve. Como se mencionó anteriormente, para esta aplicación trabajaremos con la información de los sismos cuya magnitud es igual o superior a 4.5°, así que será esa la información que procesaremos en nuestra aplicación.

Si desde el navegador web que utilizan a diario visitan alguna de las direcciones web señaladas anteriormente, estarán realizando una consulta tal cuál lo haría la app que van a crear en este tutorial. Por ejemplo, la imagen a continuación muestra el resultado obtenido si desde el navegador web visitan la url correspondiente al listado de los temblores grado 4.5+:

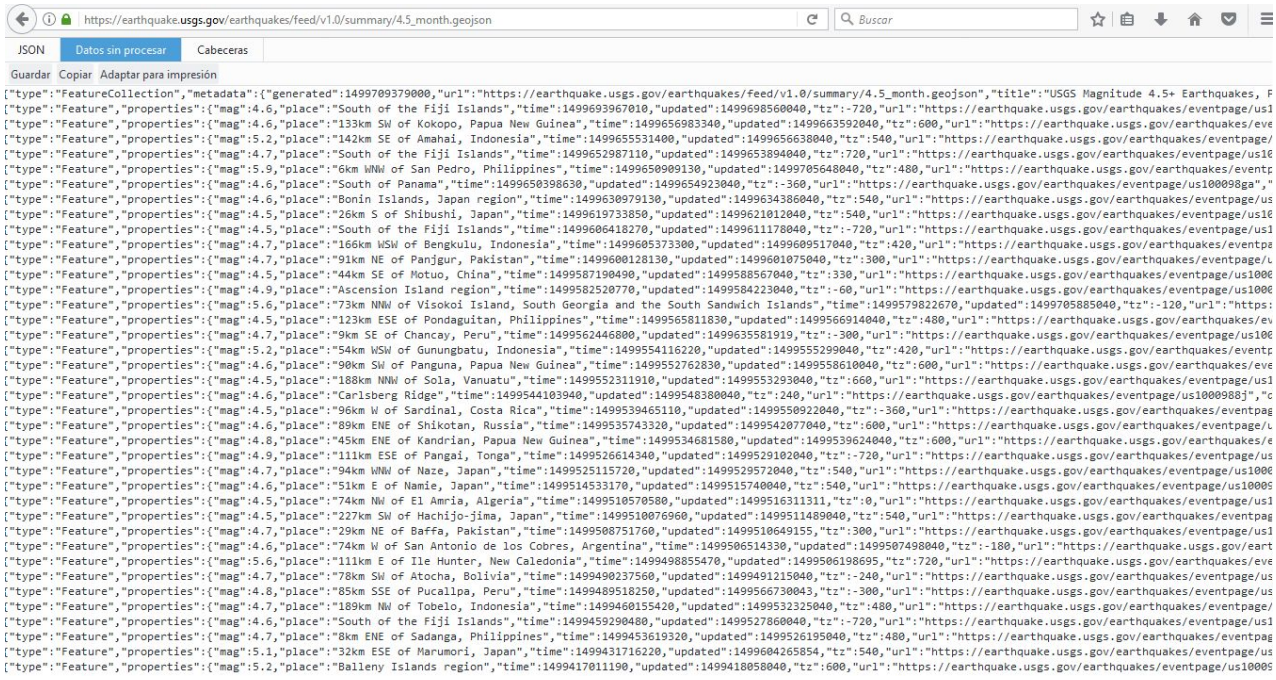


Figura 5: Resultado de una consulta al servicio web USGS

Lo que la imagen anterior nos muestra es el listado de todos los sismos de grado 4.5+ ocurridos en el mundo durante el último mes. Cualquier persona que ingrese a esta url podrá acceder a esta información y podrá utilizarla como estime conveniente. El problema está en que la información tal cual se muestra es poco práctica de utilizar ya que se encuentra organizada de una manera muy específica. Este formato se conoce con el nombre de **JSON** que no es más que un formato de texto muy liviano utilizado por muchos servicios para el intercambio de datos. El nombre proviene de las siglas **J**ava**S**cript **O**bject **N**otation. No vamos a profundizar respecto a qué es este tipo de archivo ni porque se trabaja con esta notación ya que en esta oportunidad sólo nos enfocaremos en entender cuáles son los datos que vamos a necesitar para que nuestra aplicación funcione de la manera correcta.

## Formateando la respuesta del servidor de la USGS

Como se mencionó anteriormente, cuando visitamos la url del Servicio Web de la USGS obtenemos como respuesta un archivo de texto en formato **JSON** que contiene el listado de todos los sismos grado 4.5+ ocurridos durante el último mes. Nuestra tarea ahora es entender cómo está formateada esta respuesta para saber qué es lo que vamos a utilizar en nuestra aplicación. El primer paso es conocer el formato en que viene esta respuesta por lo que analizaremos parte del listado devuelto por el servidor. Para ello utilizaremos un navegador (como firefox) que interprete el **JSON** con la respuesta para poder visualizar los datos de mejor forma (similar a lo que debería hacer nuestra aplicación móvil).


Figura 6: Información de un solo objeto del JSON obtenido del servidor.

La figura anterior muestra el resultado obtenido tras consultar al Servicio Web la información de los últimos sismos. Dado que se utilizó Mozilla Firefox, fue posible interpretar el **JSON** de respuesta para que la información se mostrase ordenada. De utilizar otro navegador web o configurando Firefox para que muestre la información en bruto, el resultado debería ser el que se vio anteriormente en la Figura 5. Se cual sea el navegador que se utilice para ver la consulta, el resultado no afecta en nada a la aplicación que vamos a crear. La única diferencia está en la comodidad que existe para ver los datos. La respuesta que nos entrega el *API* está organizada como una colección de *"Features"*. ¿Qué quiere decir esto? para la USGS un sismo o un terremoto corresponde a una *"Feature"* por lo que al consultar a la url el servidor nos devuelve una lista con todas las *"Features"* ocurridas, es decir, todos los sismos o terremoto que están registrados en el sistema. A la fecha que fue realizado este tutorial es posible ver como ha ocurrido un total de 321 *"Features",* o sea, 321 sismos de grado 4.5+. La imagen a continuación muestra de manera compacta el **JSON** con al respuesta del servidor.

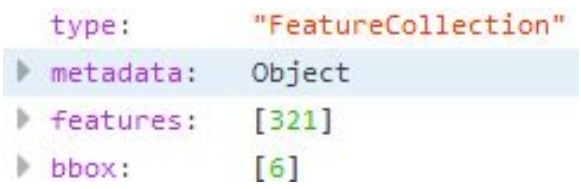


Figura 7: Vista compacta del JSON obtenido desde el servidor

Como se mencionó anteriormente, de la figura 7 vemos cómo para la USGS la información se encuentra organizada como una lista de "Features". Se puede apreciar como a la fecha han existido 321 Sismos de 4.5+. De manera muy sencilla, la principal característica del formato JSON es que nos permite organizar los datos de manera jerárquica mediante el formato clave, valor. ¿Qué quiere decir esto? Si queremos organizar datos de la sala de clase y queremos crear una suerte de libro de clases virtual, podemos crear un JSON que tenga la lista de "Alumnos". Cada uno de estos Alumnos representa un Objeto de esta lista tal cual una "Feature" representa un elemento dentro de esta lista de "Features" que nos entrega la USGS. Volviendo al ejemplo de la sala de clases, de manera particular, cada uno de estos objetos poseen propiedades propias por lo que podríamos describir un alumno como un objeto que tiene nombre, apellidos, edad, entre otros. La imagen a continuación muestra cómo se vería un posible JSON correspondiente al listado de los alumnos para una sala de clases en particular:

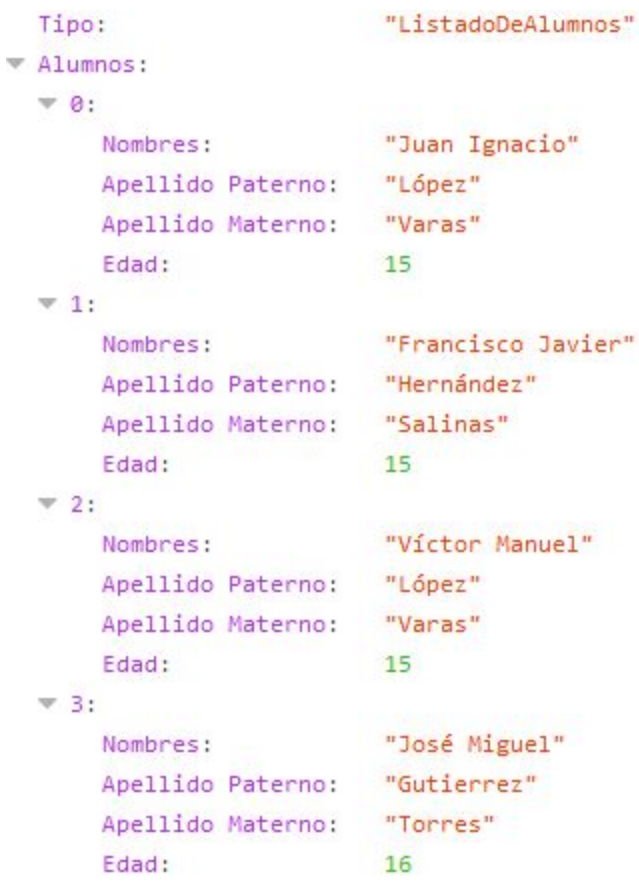


Figura 8: JSON con el listado de los alumnos de un curso

En el ejemplo anterior podemos ver cómo cada objeto de tipo alumno tiene las etiquetas Nombres, Apellido Paterno, Apellido Materna y edad, las que a su vez tienen valores distintos para cada alumno. Lo mismo que ocurre con este ejemplo es lo que pasa con el JSON obtenido del servicio web de la USGS con la única diferencia de que tiene una mayor cantidad de atributos que son necesarios para poder describir en detalle el sismo. De la figura 6 podemos darnos cuenta de que existen un sin número de propiedades que nos devuelve este JSON, algunas propiedades propias del sismo (como la magnitud, fecha, lugar, etc) así como otros elementos que no describen al sismo propiamente tal pero que permiten utilizar este JSON con otras aplicaciones. Lo que vamos a hacer nosotros con nuestra aplicación móvil es recibir este JSON del servicio web y quedarnos solo con las etiquetas que nos aporten características propias el sismo como lo son el lugar, la fecha, la magnitud y la ubicación. El resto de los elementos no los vamos a considerar por lo que no será necesario ahondar en más detalles.

## Programando los bloques

Habiendo entendido de manera general cómo está construido este archivo JSON el siguiente paso será definir qué es lo que vamos a hacer con él y el cómo nuestra aplicación se va

a conectar con el servicio web de la USGS. A continuación describiremos en detalle los pasos necesarios para lograr hacer la consulta al sitio de la USGS y qué es lo que vamos a necesitar para poder desplegar la información de manera ordenada en nuestra aplicación.

### Creando nuestras variables

Tal como lo hemos visto en otras oportunidades, el uso de variables es algo que hacemos frecuentemente cuando creamos nuestras aplicaciones. Esto se debe principalmente a que la gran mayoría de las aplicaciones que hemos creado necesitan guardar información, ya sea porque el usuario la ingresó o bien porque la app genera datos que son necesario guardar. Para poder almacenar esta información y que de esa manera funcione correctamente la app, el uso de variables es imprescindible. Dado que son muchos los datos con los cuales estaremos trabajando es necesario ordenar y organizar desde el comienzo esta información.

- Necesitaremos crear una variable que nos permita guardar la url del servicio web de la USGS. En caso que necesitemos cambiar la fuente de los datos, solamente debemos actualizar el valor que tiene esta variable. El link del servicio web fue listado anteriormente.
- En esta versión preliminar de nuestra aplicación, solo nos vamos a preocupar de guardar algunos de los datos que nos entrega la USGS. En este caso como nuestra aplicación va a mostrar la información del título del temblor, la ubicación en dónde ocurrió y la magnitud que tuvo, vamos a necesitar crear solamente estas variables. Es importante recalcar que como la información que nos entrega la USGS no corresponde a un solo dato, nuestras variables deberían poder almacenar más de un valor por cada uno de los atributos que se deseen guardar. Por fortuna App Inventor tiene un componente llamado *lista* que nos permite hacer muy sencillo el procedimiento de guardar más de un dato del mismo tipo. Vamos a necesitar una lista para la magnitud del sismo, otra para las coordenadas y una para guardar el título el sismo. También vamos a necesitar una lista que nos permitirá procesar la información de las otras listas, pero esta vez de manera temporal. Por último necesitaremos dos variables más, una para almacenar el nombre del sismo actual que se está revisando y otra para guardar la información del JSON que estamos procesando. La imagen a continuación detalla cada una de las variables que necesitaremos para poder desarrollar el desafío.


Figura 9: Variables a utilizar en nuestra aplicación

- ***datosSismos***: Nos permitirá guardar el nombre del sismo que se está procesando
- ***indice***: Nos permitirá movernos entre los elementos de la lista
- ***url:*** Dirección del lugar donde se encuentra el Servicio Web
- **coordenadasSismos**: Lista que contendrá la información de las coordenadas de todos los sismos obtenidos del JSON
- **magnitudSismos:** Lista que contendrá la información de las magnitudes de todos los sismos obtenidos el JSON
- **sismos**: Lista que contendrá la información de los nombres de todos los sismos obtenidos del JSON

El resto de las variables se irán explicando en la medida que avancemos con el desarrollo de nuestra app. Por ahora sólo vamos a crearlas entendiendo su uso posteriormente.

### Consultado la información al servidor

Habiendo creado nuestras variables, el siguiente paso es consultar al Servicio Web de la USGS para obtener el JSON con la información de los sismos. Definiremos que el mejor momento para hacer eso es al momento de inicializar la pantalla de nuestra aplicación. Con esto nos aseguramos de tener el JSON con los datos de los sismos a penas encendamos nuestra aplicación:

- Desde la sección Screen1, arrastra el bloque ***CuandoScreen1.inicializar***
- Cambian la propiedad Url del componente web mediante el bloque ***PonerWeb1.Url como.*** Encaja el valor de la variable url definida previamente
- Desde el componente ***Web1*** selecciona el procedimiento ***llamarWeb1.obtener*** y encájalo

en el bloque *CuandoScreen1.inicializar*

La figura a continuación muestra el resultado de programar esta primera parte:

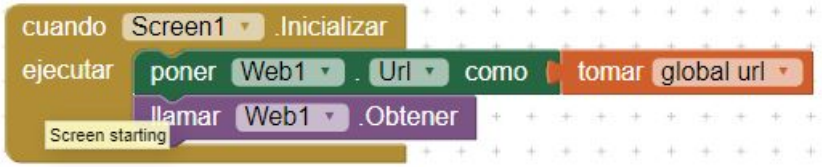


Figura 10: Bloques necesarios al momento del ejecutar la aplicación

El bloque ***llamarWeb1.obtener*** no hace más que decirle a App Inventor que queremos visitar el sitio web definido por la url previamente ingresada. Esto es lo mismo que visitar el sitio desde el navegador, por lo tanto el resultado de ***llamarWeb1.obtener*** es el archivo JSON del que hablamos anteriormente. App Inventor dispone de un bloque de eventos que permite conocer el resultado de una petición Web por lo que el siguiente paso que haremos será programar qué es lo que haremos con el resultado de la petición Web. Para ello realizaremos los siguientes pasos:

- Desde tu componente web arrastra el bloque de evento **cuandoWeb1.ObtuvoTexto.** Dentro de las variables que nos devuelve este bloque utilizaremos el **códigoDeRespuesta** y el **contenidoDeRespuesta.** Si el resultado de consultar la información del servidor fue correcto, el código de respuesta debería ser 200, en caso contrario quiere decir que ocurrió algún tipo de error. el contenido de respuesta corresponde a la información devuelta por el servidor, en este caso, el JSON con la información de los sismos.
- Como debemos verificar si la petición fue exitosa o no, desde la sección control arrastra un bloque *si entonces* y agrégalo a tu evento *cuandoWeb1.ObtuvoTexto.* Desde el engranaje azul que nos permite componer el bloque, agrega un *sino* a nuestro bloque si entonces.
- El siguiente paso será crear la condición para lo cual arrastrarás un bloque de comparación desde matemáticas. La idea es verificar si el *códigoDeRespuesta* devuelto por el evento tiene o no el valor 200. En caso contrario podríamos mostrar algún mensaje de error a los usuarios. Para eso, crea la condición que nos permite preguntar si el códigoDeRespuesta es igual o no a 200. Deberás sacar desde matemáticas un bloque para comparar y un bloque numérico con el valor 200. En caso que el el resultado sea distinto a 200, puedes agregar desde el diseñador un notificar y agregar una alerta del error en esta sección. La siguiente imagen muestra cómo debería verse tu evento una vez agregado los bloques descritos anteriormente.


Figura 11: Verificando que la petición web sea exitosa

Si la petición fue exitosa nos devolverá un valor 200 por lo que al compararla como se ve en la imagen anterior, estamos verificando que podamos trabajar con el *contenidoDeRespuesta*. El siguiente paso, y quizá el más complicado de esta aplicación, es el crear un procedimiento que permita recuperar la información que nos interesa del JSON que nos devuelve el servidor. Tal como se mencionó anteriormente, lo que nos interesa en esta oportunidad es obtener las coordenadas del sismo, la magnitud y el lugar. Es por eso que anteriormente creamos listas para guardar dichos valores. El procedimiento que crearemos a continuación permite recuperar el valor a partir de las etiquetas que contienen el dato que nos interesa. Como vimos en el ejemplo del listado de los alumnos del curso, la etiqueta *"Nombre"* contenía los datos del nombre, la etiqueta *"Apellido Paterno"* lo hacía con la información del apellido paterno y lo mismo pasaba con el resto de las etiquetas. El JSON devuelto por el servidor web trabaja de la misma manera, guardando en etiquetas específicas la información que nos interesa. Dado que la USGS dispone de esa información para ser utilizadas de distintas maneras, el JSON contiene mucha información aparte de la que vamos a utilizar nosotros. Dado que le diremos a App Inventor que busque por medio de etiquetas específicas, no nos importa que el JSON tenga más información de la que necesitamos.

El siguiente paso será crear el procedimiento al cual llamaremos **InfoDeTodosLosSismos.** Como lo hemos realizado en otras oportunidades, para crear el procedimiento debemos hacerlo desde la sección dispuesta para eso. Arrastraremos el bloque morado ***ComoProcedimiento*** lo renombraremos a *InfoDeTodosLosSismos* y mediante el engranaje azul que tiene este bloque modificaremos nuestro procedimiento para que acepte cuatro parámetros de entrada. Estos parámetros corresponden al lugar desde donde obtendremos la información a procesar, el nombre de la etiqueta que contiene el dato, el o los caracteres que representan el final del dato que vamos a procesar y por último la lista donde vamos a guardar el resultado del procesamiento una vez finalizado. La imagen a continuación muestra cómo quedará nuestro procedimiento una

vez creado:

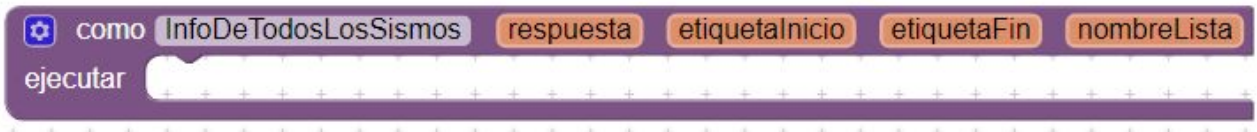


Figura 12: Procedimiento InfoDeTodosLosSismos utilizado para procesar el JSON obtenido desde el servidor

El rol que desempeña este procedimiento es el de procesar el JSON con toda la información de los sismos para extraer solo los datos que necesitaremos, guardándolos en una lista. La información que vamos a rescatar esta dada por la *etiquetaInicio* y la *etiquetaFin* por lo que si queremos personalizar nuestra aplicación para que extraiga otro tipo de información, solamente debemos especificar el valor de las etiquetas y el nombre de la lista donde guardaremos los datos. Sigue los pasos a continuación para poder extraer la información correcta desde el JSON entregado como respuesta:

- Desde variables arrastra un bloque poner y configurarlo para guardar datos en la variable *datosSismos.* Desde texto, extrae el bloque "*recorta*" y encaja en su parámetro *texto* la variable respuesta (desde donde vamos a recortar información, en este caso, desde el JSON). En el parámetro *"en"* encaja el texto que vamos a recortar, en este caso, la etiqueta inicio. El bloque recorta funciona de la siguiente manera: Divide un texto en partes utilizando el parámetro *"en"* como punto de recorte. Por ejemplo, si tenemos el texto *"Hola-cómo-te-llamas-tú"* y lo recortamos usando como punto de recorte el guión, obtendremos una lista con cada uno de los valores que fueron recortados, en este caso, algo como [Hola, cómo, te, llamas, tú]. Al ser una lista podemos utilizar cualquier bloque que hayamos visto antes con otros ejercicios. Una vez que hayas hecho todos los pasos tu bloque debería verse algo parecido a la figura 13.


Figura 13: Recortando el JSON a partir de la etiquetaInicio

- Como al recortar se crea una lista que contiene todos los elementos a partir de tu etiqueta de inicio, debemos ignorar el texto que está antes de tu etiqueta. Por lo mismo guardaremos en nuestra variable *primerElemento* el valor *cierto.* Habiendo hecho esto encajaremos este bloque justo debajo del bloque que se ve en la Figura 13. El paso a continuación es recorrer la lista que se generó tras recortar el JSON a partir de la etiqueta a buscar (después vamos a definir si son las coordenadas, magnitud, etc). Como hemos

visto anteriormente, para recorrer una lista necesitamos el bloque *"por cada".* Desde control arrastra un bloque *por cada,* indícale que recorreremos la lista *datosSismos* y encájalo en tu procedimiento una vez que hayas terminado. El resultado es el que se muestra en la imagen a continuación:

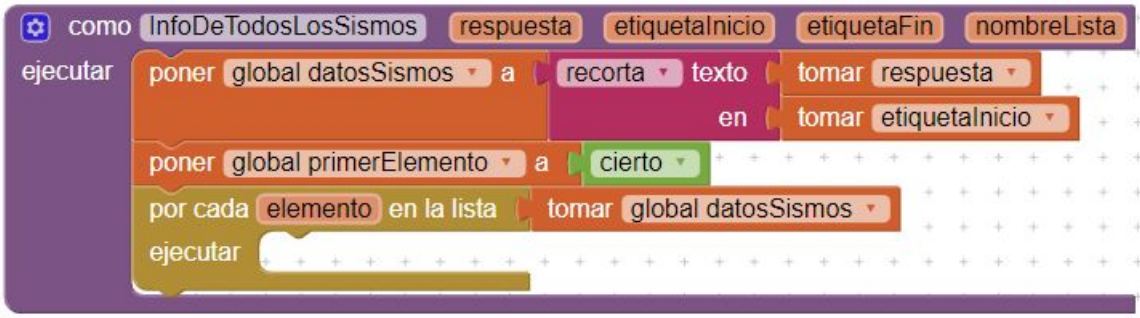


Figura 14: Creación de un bloque por cada para recorrer la lista generada tras el recorte

- El siguiente paso es ignorar el primer elemento de la lista (ya que contiene valores antes de la etiqueta a buscar) para lo cual utilizaremos un *si-entonces-sino.* La validación es muy sencilla: si el valor guardado en la variable *primerElemento* es cierto, entonces ignora el elemento de la lista, cambia el valor de *primerElemento* a falso y volvemos nuestra lista temporal a vacía (esto es porque cada vez que llamemos al procedimiento debemos limpiar la lista, de lo contrario los datos a mostrar en la aplicación estarán duplicados. Los bloques tras la validación debería quedar como en la siguiente imagen:

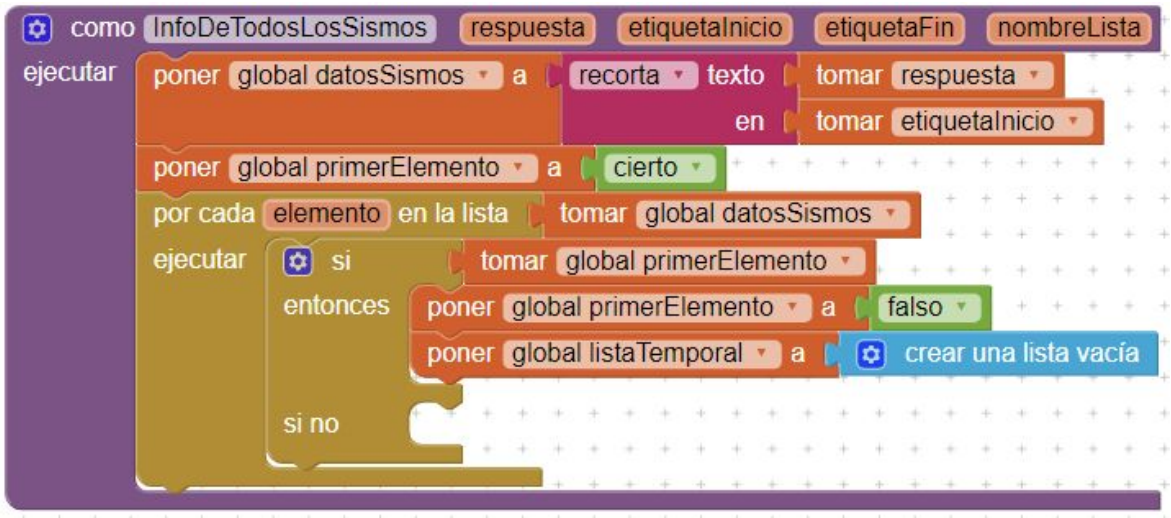


Figura 15: Validación para ignorar el primer elemento de la lista

## Procesando una etiqueta

Hasta ahora tenemos un procedimiento que recibe un JSON como dato de entrada y lo separa en una lista a partir de una etiqueta a buscar. Sin embargo la lista que crea el bloque

*recorta* está separada a partir de una etiqueta de inicio pero no por eso está correctamente formateada. Por ejemplo, si en la siguiente cadena queremos sacar el valor que está entre el carácter "-" y "+" debemos indicarle a AppInventor que la recortaremos por el principio y por el final. Con los procedimientos que hemos creado hasta ahora solamente hemos acotado el JSON por el principio. Si en la siguiente cadena de ejemplo, "-hola+soy-una+cadena" recortamos solo por el principio obtendremos lo siguiente: *["hola+soy","una+cadena"]*. Si bien logramos cortar la cadena a partir del guión, debemos hacer lo mismo para el carácter "+". Lo que haremos será crear un nuevo procedimiento pero que esta vez recorte nuestra cadena por una etiqueta de fin. Los bloques a continuación muestran cómo debería quedar este nuevo procedimiento que vamos a crear:

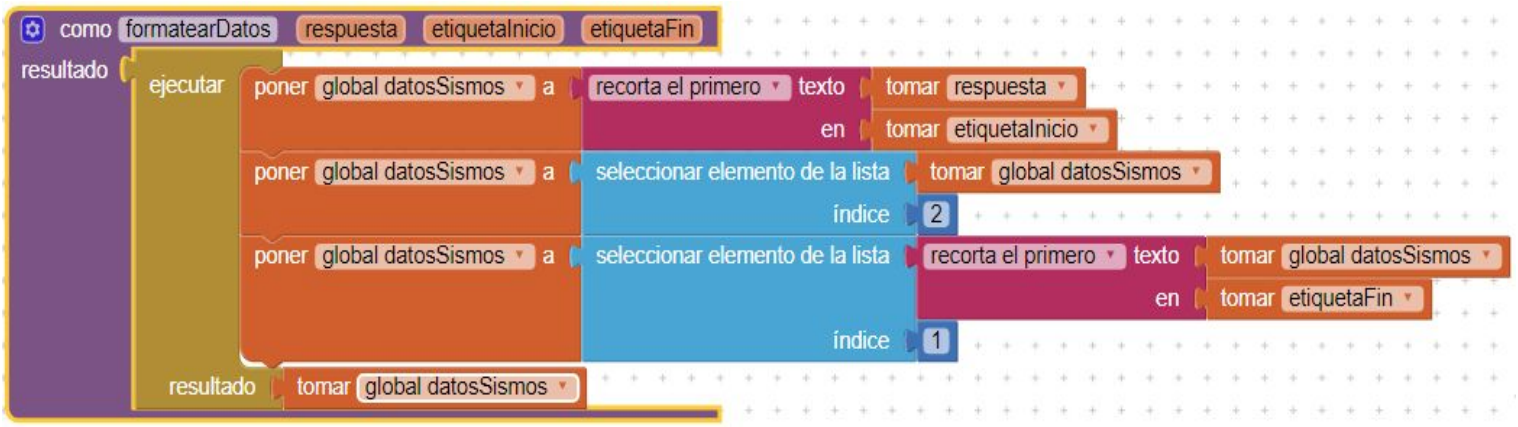


Figura 16: Bloques del procedimiento formatearDatos

Es importante señalar que a diferencia de otras veces, este procedimiento corresponde al ***ComoProcedimientoResultado*** ya que una vez en ejecución, el resultado de este procedimiento es el valor de la etiqueta sin ningún texto extra. Si llamamos a este procedimiento para recuperar el valor del ApellidoPaterno del primer niño del ejemplo del listado de curso, este procedimiento entrega como resultado "López". Para obtener el bloque amarillo "ejecutar-resultado" debes dirigirte a control y arrastrarlo sobre tu procedimiento. De manera muy simple, este procedimiento formate un solo objeto del JSON que le entreguemos por parámetro, a diferencia del procedimiento anterior que creaba una lista por cada elemento que encontrase en el JSON. Ambos son necesarios porque el primero busca todas las ocurrencias de la etiqueta (dado que son muchos temblores, necesitamos encontrarlos todos) mientras que el segundo lo vamos a llamar por cada una de las ocurrencias encontradas anteriormente para obtener el valor de la etiqueta. Una vez terminado este procedimiento lo que haremos será llamarlo desde el procedimiento creado anteriormente, tal cual se muestra en la siguiente figura:

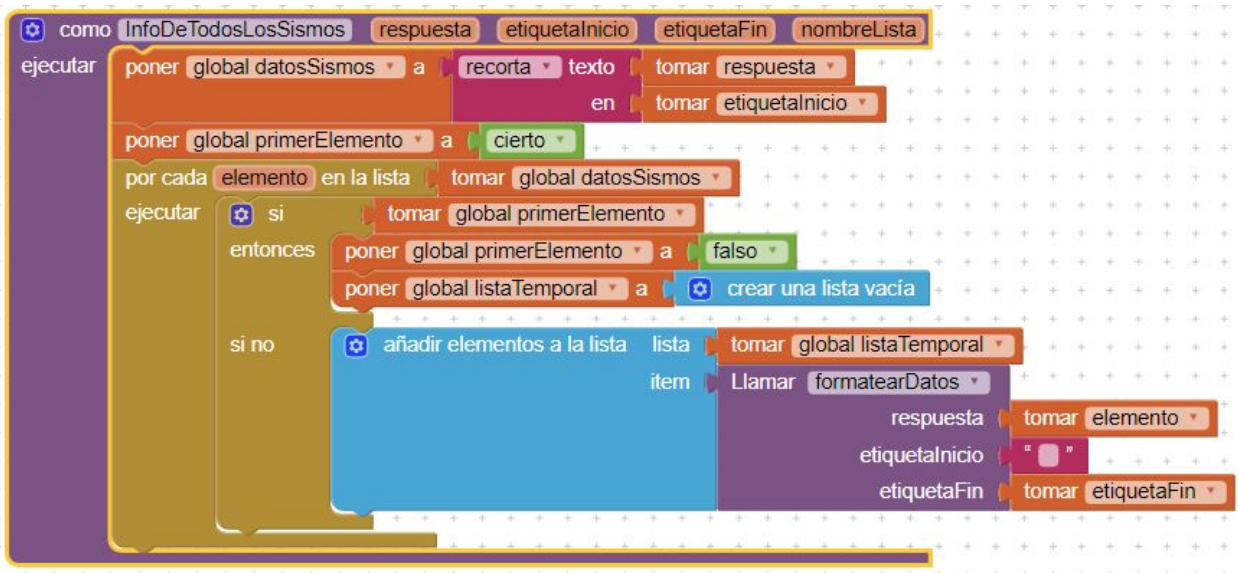


Figura 17: Bloques del procedimiento InfoDeTodosLosSismos

Lo que hemos realizado en este punto es completar los bloques del si no de la condición creada anteriormente. Vamos a utilizar una lista temporal para ir guardando cada uno de los datos que fueron extraídos del JSON.

El último paso será copiar la *ListaTemporal* a la lista correspondiente según el parámetro *nombreLista*. Realizamos este paso ya que debemos tener almacenados en listas distintas las coordenadas, nombre del sismo y la magnitud de los sismos así que copiaremos esta lista temporal en alguna de las listas que creamos en el primer paso de este desafío. Realizaremos una pregunta con *si-entonces* para saber en cual lista copiar el contenido. La figura a continuación muestra cómo deberían ser la totalidad de los bloques contenidos en el procedimiento que acabamos de crear. ¡El último paso que nos queda es llamar a este procedimiento!

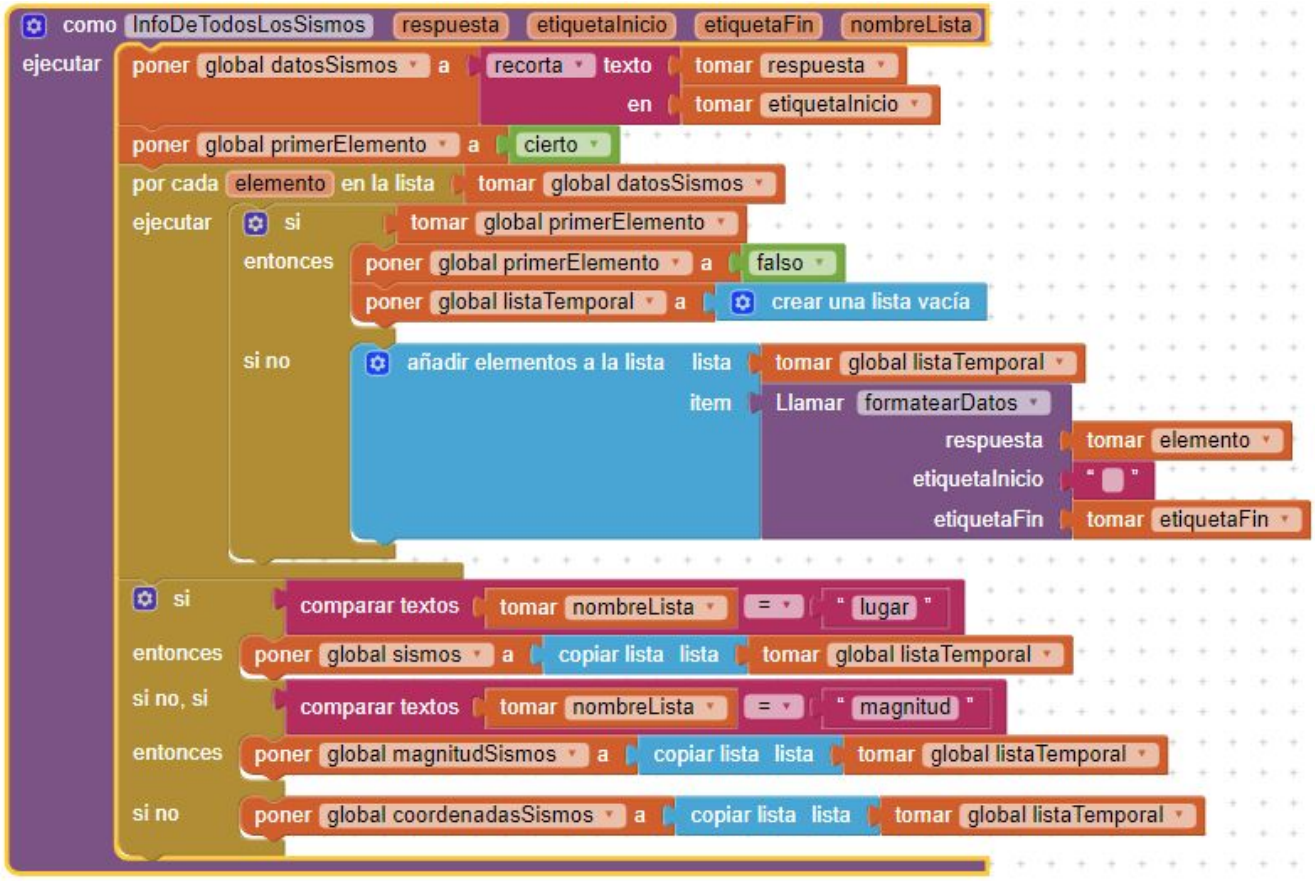


Figura 18: Bloques del procedimiento InfoDeTodosLosSismos

## Terminando el evento ObtuvoTexto

Ahora que hemos creado el procedimiento que nos entregará el valor contenido en las distintas etiquetas del JSON, el paso que nos va quedando es llamar a este procedimiento de la manera correcta. Si analizamos el formato que tiene el JSON devuelto por la USGS, veremos que las etiquetas que nos importan son las que tienen los siguientes nombres: *"mag"* (es la etiqueta que contiene la magnitud), *"place"* (es la etiqueta que describe el lugar del epicentro) y la etiqueta *"coordinates"* (que es la etiqueta que contiene las coordenadas del epicentro). Habiendo encontrado esto, el siguiente paso es hallar las etiquetas de término para cada uno de nuestros datos. En el caso de la etiqueta *"place"* los datos terminan en **",** mientras que en "*mag"* solo terminan en una coma (",") y en "coordinates" terminan con un cierre de llaves y corchetes (**"]}"**). Habiendo definido esto, completaremos los bloque que nos faltan obteniendo algo similar a lo que se muestra en la siguiente figura:

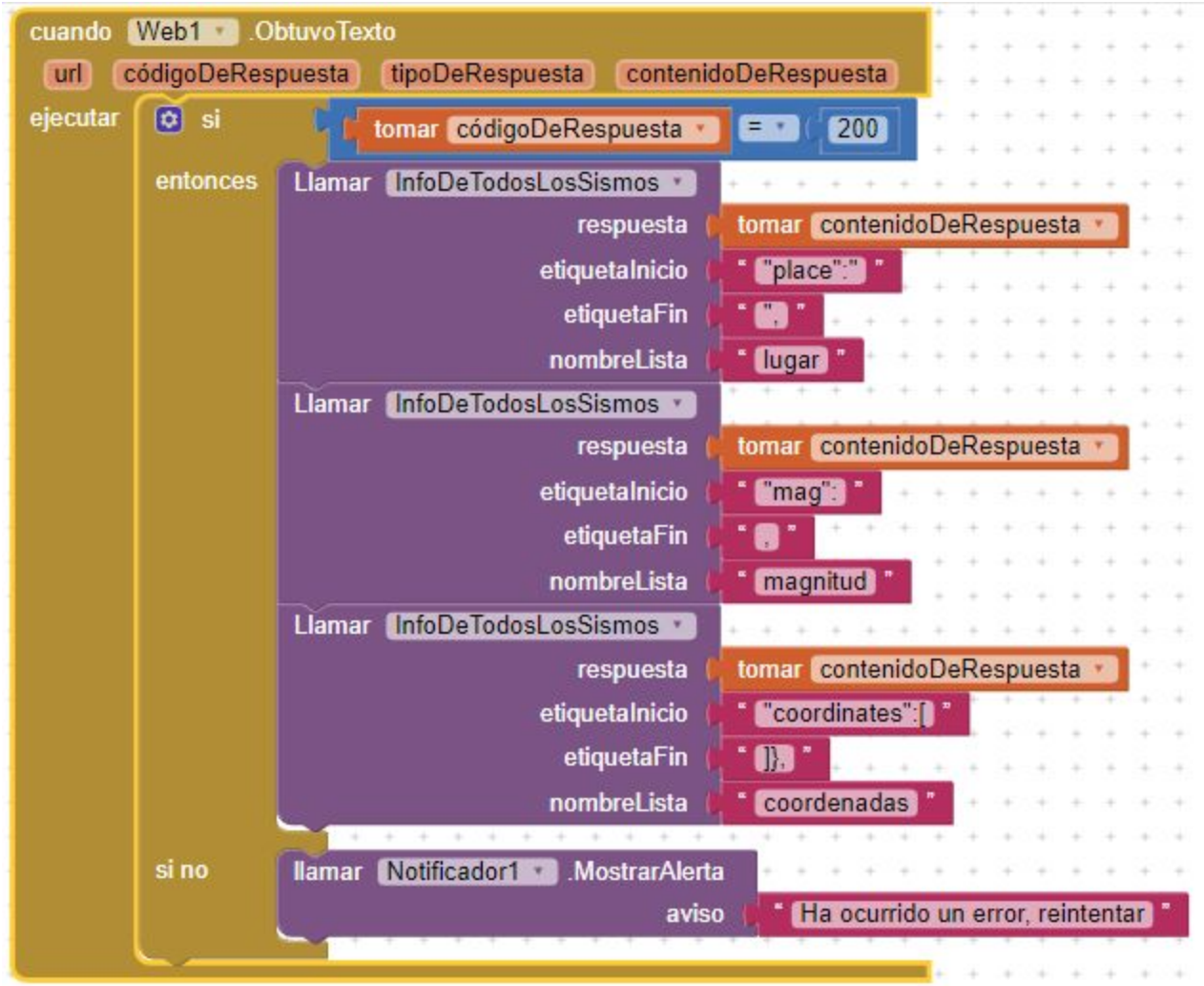


Figura 19: bloques para evento ObtuvoTexto

## Trabajando con el selector de lista

Hasta ahora tenemos toda la información que vamos a necesitar guardadas en listas, sin embargo no hemos programado de qué manera la vamos a mostrar. Para ello vamos a trabajar con el selector de lista de tal manera que al momento de presionarlo se listarán todos los sismos ocurridos durante el último mes y tras seleccionar alguno de ellos se deberá escribir en las etiquetas los valores del sismo seleccionado. El primer paso es muy sencillo de realizar ya que solamente deberás utilizar el evento ***CuandoSelectorVerSismosMundo.presionar*** al que le encajaremos la lista *sismos* tras el cambio de la propiedad elementos del selector. La figura a continuación muestra cómo debería quedar este bloque:

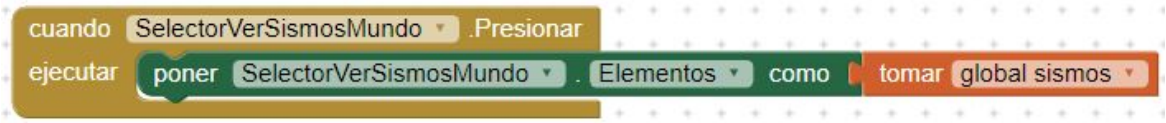


Figura 20: configurando los elementos del selector de lista

Habiendo configurado los elementos del selector de lista nos quedará definir qué es lo que se realizará tras la selección de uno de ellos. Dado que tenemos todos los elementos guardados en listas, lo único que tenemos que hacer en esta sección es seleccionar un sismo de la lista para mostrar toda la información asociada a este sismo (que no es más que escritura de etiquetas). Los bloques a continuación muestran el resultado de esperado tras seleccionar algún elemento del selector de lista:

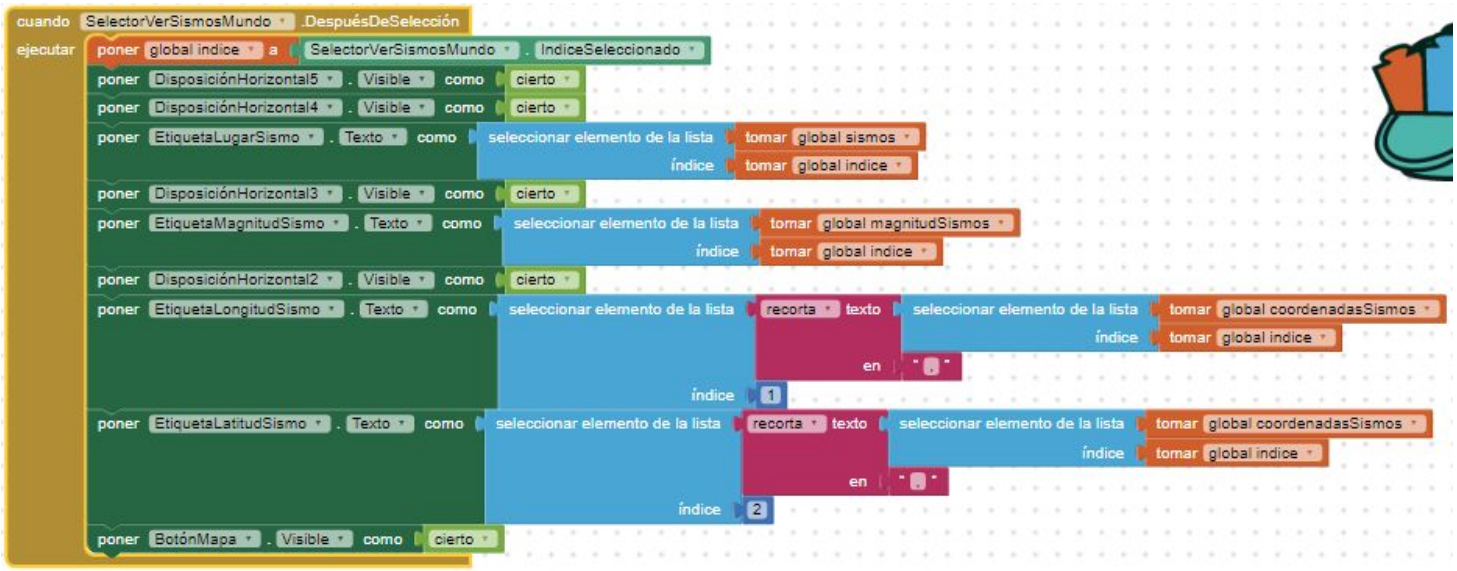


Figura 21: Bloques terminados para el evento DespuésDeSelección del selector de lista

Si lograste seguir los pasos hasta acá habrás creado el esqueleto de tu aplicación totalmente funcional. Este es el momento donde deberán personalizar la aplicación según lo estimen en sus clubes de apps. A continuación se listan un par de desafíos para que puedan trabajarlos junto a sus clubes de apps.

## Desafíos para tu club de Apps:

- La información que muestra la aplicación es acerca de todos los ismos del mundo, algo que puede ser poco práctico si lo que deseo es consultar la información para los sismos ocurridos solo en Chile. Implementen junto a su Club de Apps un nuevo selector de lista

que solo muestre los sismos ocurridos en Chile

- Puede resultar interesante no solo mostrar la información de del sismo escrita en la pantalla del teléfono, sino que también hacerlo mediante un mapa. Dado que ya tienen la información la latitud y longitud del epicentro del sismo, implementar los bloques que se necesiten para que la aplicación pueda mostrar el epicentro del sismo en Google Maps
- Otro dato interesante de rescatar corresponde a la fecha del sismo. Esa información está presente en el JSON entregado por la USGS. Implementen los bloques necesarios para permitir que la aplicación muestre la fecha del sismo.