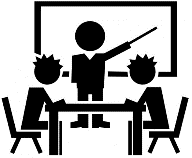
**Guía 1.2 -** Consumo de JSON



**Material presencial**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nombre Curso** | **Integración de módulos de software** | | **Unidad de Aprendizaje** | | **Realiza integración del software de acuerdo a procedimientos establecidos para su ejecución.** | |
| **Horas totales Unidad Aprendizaje** | | 20 horas | | **Fecha Actualización** | | Noviembre 2017 |
| **Escuela** | **Escuela de Informática y Telecomunicaciones** | | | | | |

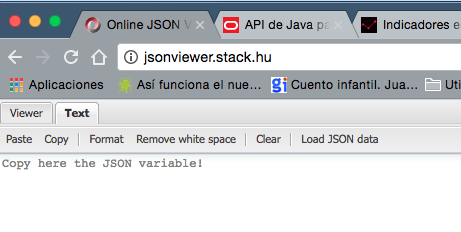
|  |  |
| --- | --- |
| **Aprendizaje Esperado** | **Realiza integración del software de acuerdo a procedimientos establecidos para su ejecución.** |
| **Indicadores de Logro** | Reconoce los tipos de integración para unir los componentes de software con el objeto de ser aplicados a un determinado tipo de aplicación. |

**OBJETIVO**

Consumir un servicio web que entrega los principales indicadores económicos para Chile en formato JSON (http://mindicador.cl/)

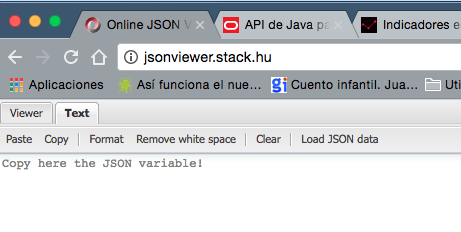
**REVISIÓN DE FORMATO DE ARCHIVO JSON**

Vamos a utilizar la herramienta online JSON viewer para visualizar el contenido de archivo JSON provisto por el servicio web: (http://jsonviewer.stack.hu/)

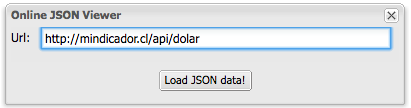


Desde el sitio asociado a la API podemos obtener la forma en que se debe consultar los indicadores y usaremos JSON viewer para conocer ese formato.

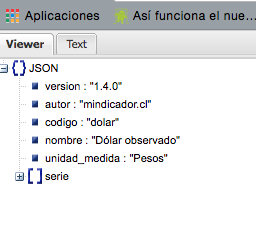
Si hacemos clic en Load JSON data nos pedirá la URL desde donde se obtienen los datos:



En este caso vamos a consultar la información relacionada con el dólar:



Dando clic en el botón será posible ver los datos asociados a ese indicador. Los datos tienen 2 formatos de visualización: “Text” y “Viewer”; esto ayuda para saber cómo se deben obtener los datos, si vamos a la pestaña “Viewer” se obtiene:

donde es posible ver que entrega información general (versión, autor, código, nombre, unidad\_medida) y datos se la variación del úlitmo mes (serie) .

Ahora, vamos a construir una aplicación que permita consumir estos datos y presentarlos.

**APLICACIÓN PARA CONSUMIR LOS DATOS.**

1. Crear un proyecto Java Web – Web Application (ver Figura 1 - Creación proyecto)
2. Dar un nombre al proyecto (ver Figura 2 - Nombre proyecto)
3. Seleccionar el servidor web (ver Figura 3 - Selección servidor web)
4. Clic en Finish.

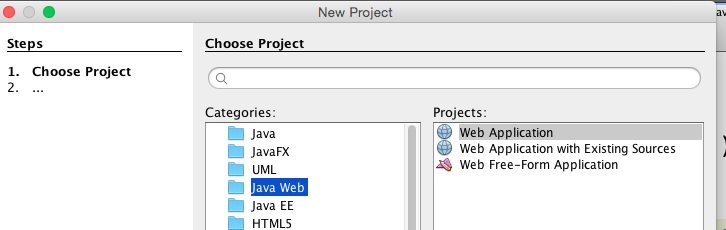


Figura - Creación proyecto

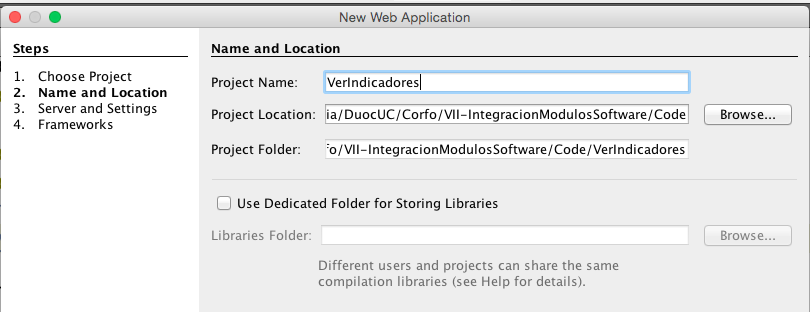


Figura - Nombre proyecto

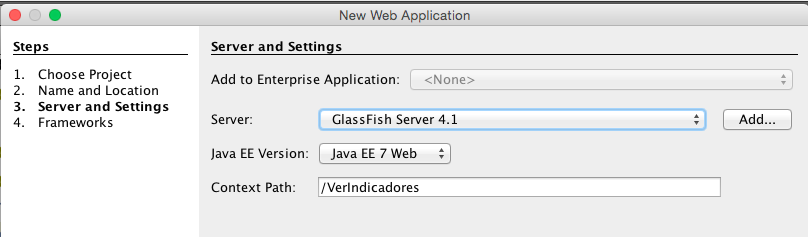
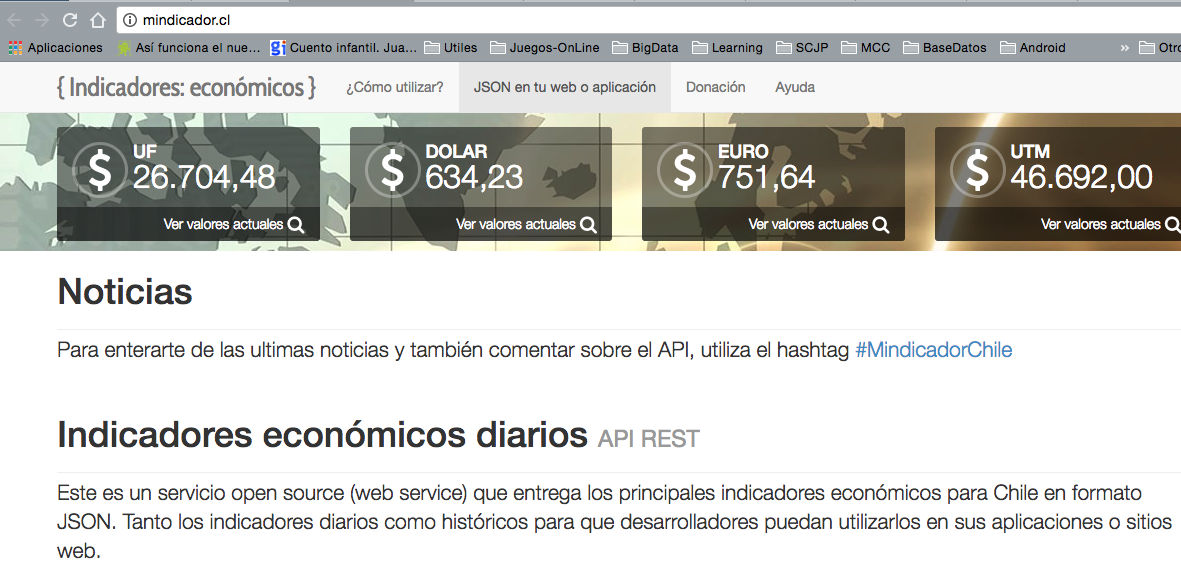


Figura - Selección servidor web

Ahora vamos a escribir el código para consumir el servicio que ofrece (http://mindicador.cl)



Nuestra aplicación usará la API de java para poder obtener la información disponible en:

<http://www.oracle.com/technetwork/es/articles/java/api-java-para-json-2251318-esa.html> (disponible al 25 de nov. de 17)

**REVISIÓN DEL JSON QUE VAMOS A CONSUMIR**

Usando la herramienta online que hemos comentado revisamos la estructura del JSON que vamos a consumir; si le indicamos la URL de carga el resultado es el de la Figura 4 - JSON a consumir. Si extendemos algunos elementos podemos ver la Figura 5 - Revisando JSON

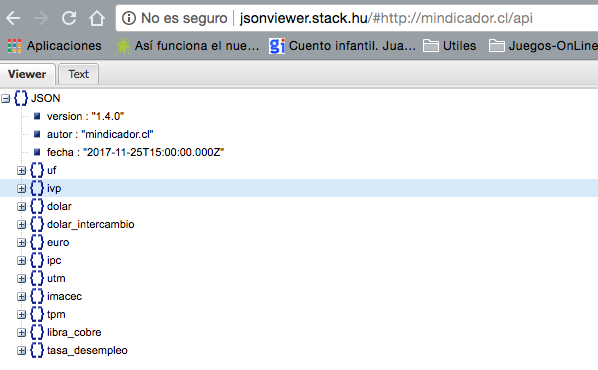


Figura - JSON a consumir

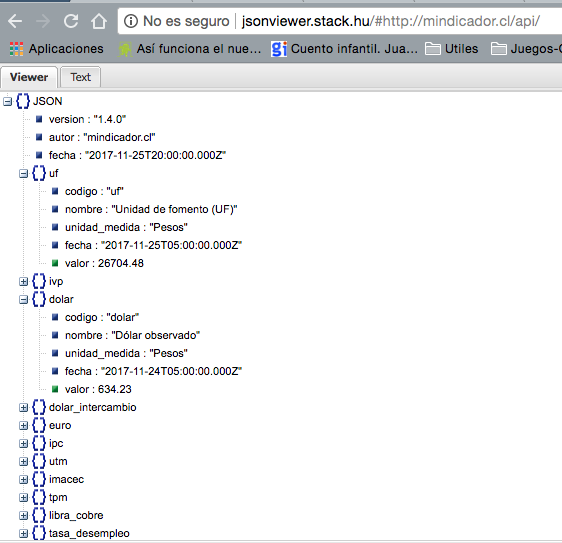


Figura - Revisando JSON

**CREACIÓN DE LA CLASE DE PRUEBA**

En nuestro proyecto vamos a crear un package llamado consola. Dentro de este package vamos a incluir las clases que permitirán consumir la API REST.

Primero, vamos a crear la clase MainIndicadores y en su método main vamos a dejar las líneas de código que permitirán leer parte de la información contenida en el JSON (ver Figura 6 - Clase a implementar)

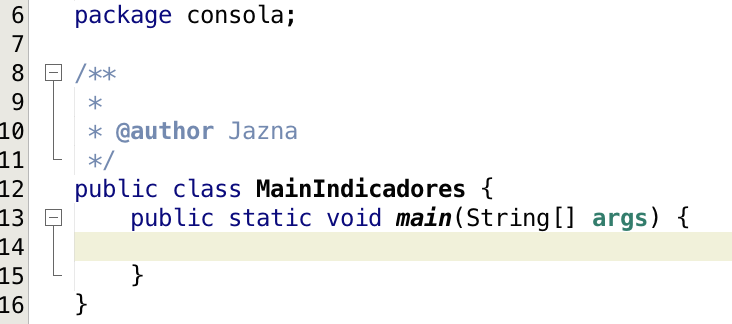


Figura - Clase a implementar

Las primeras líneas de código que escribiremos serán las de la Figura 7 - Obtiene objeto JSON a partir de la URL

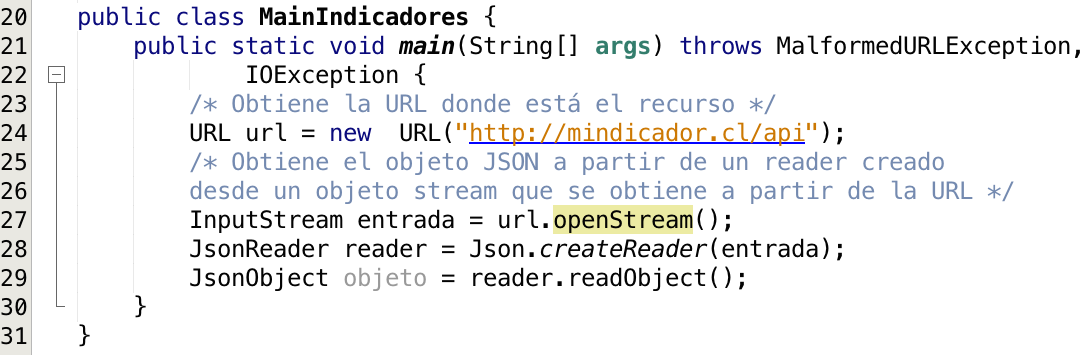


Figura - Obtiene objeto JSON a partir de la URL

Luego debemos trabajar con el objeto que hemos leído y usar los métodos definidos en la API que Java nos ofrece. Ahora vamos a escribir las líneas de código para imprimir el valor de la UF y algunas características que se tienen del dólar. Primero se debe notar que dólar y uf son 2 objetos dentro del JSON; teniendo en cuenta eso vamos a escribir nuevas líneas de código de acuerdo a lo que indica la Figura 8 - Obtener valor de la UF.

En la línea 32 de la Figura 8 - Obtener valor de la UF se obtiene el valor de la propiedad *“valor”* asociada al objeto JSON llamado *“uf”;* el método *get()* y el método *getJsonObject()* , respectivamente, utilizan como argumentos estas propiedades.

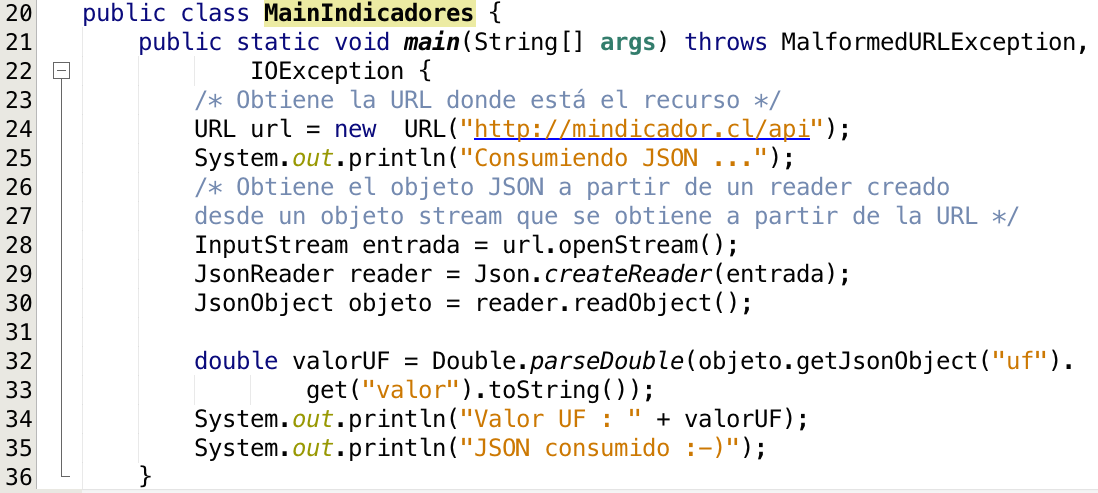


Figura - Obtener valor de la UF

**Obtener toda la información de un indicador particular**

Dentro del package consola vamos a crear una clase llamada MainIndicadores donde dejaremos el código para poder consumir los datos que obtendremos de la API REST, pero ahora será la información de un indicador en particular. En esta clase escribiremos un método main que contendrá el código que aparece en Figura 4 - Código para obtener los datos.

Algunas de estas líneas de código ya han sido usadas en las experiencias anteriores y otras son nuevas y hacen uso de la API de Java que habíamos mencionado anteriormente.

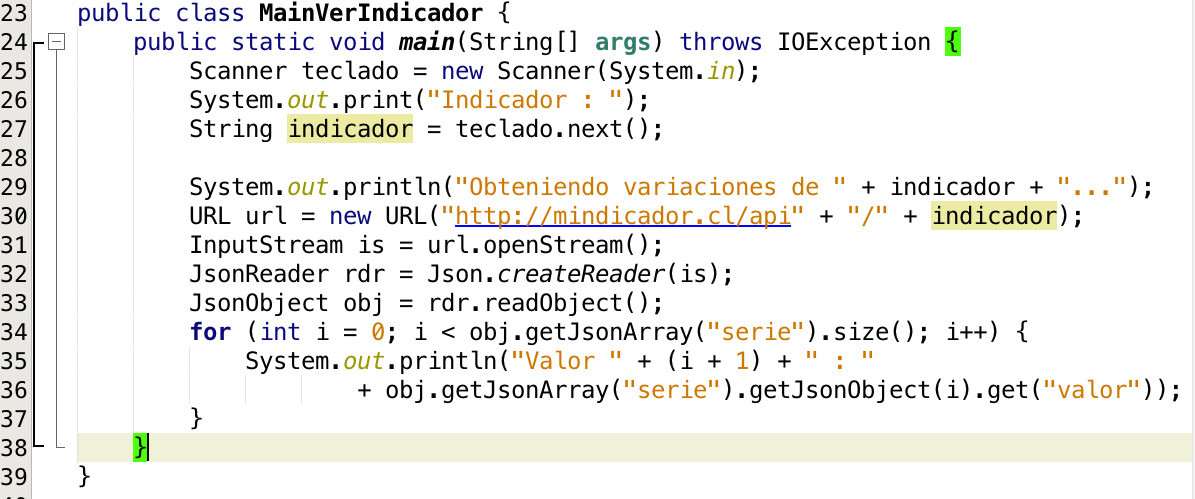


Figura - Código para obtener los datos

En la línea 25 se crea un objeto de la clase Scanner que permitirá obtener desde teclado el nombre del indicador del cual se desea la información.

En la línea 27 se obtiene lo que se ingresa desde teclado. Según lo revisado se debe manejar la URL: [http://mindicador.cl/api/{tipo\_indicador](http://mindicador.cl/api/%7btipo_indicador)} usando el nombre del indicador del cual se desea información.

En la línea 30 se construye la URL que se define por el proveedor del servicio para consumir los datos.

Las líneas 31 a la 33 se crean los objetos que permiten obtener la información disponible en el JSON.

El ciclo que aparece en la línea 34 permite recorrer el array JSON que contiene las variaciones del indicador.

**CIERRE DE ACTIVIDAD**

En este punto usted debería haber probado todos los códigos que se proponen en esta actividad. Preocupese de preguntar si algo no le ha resultado y, lo más importante, preocupese de entender todo lo que hizo.