**EBook Gratis**

# APRENDIZAJE ibm-watson-cognitive

Free unaffiliated eBook created from **Stack Overflow contributors.**

#ibm-watson-cognitive

# Tabla de contenido

# Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: ibm-watson-cognitive

It is an unofficial and free ibm-watson-cognitive ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official ibm-watson-cognitive.



Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

# Capítulo 1: Empezando con ibm-watson-cognitive

## Observaciones

Este tema proporciona instrucciones básicas para obtener credenciales para los servicios de Watson y proporciona enlaces relevantes para cada servicio y los SDK de Watson Developer Cloud.

Watson servicios homepages:

- Alquimia Idioma
- AlchemyData News
- Conversacion
- Descubrimiento
- Conversión de documentos
- Traducción de idiomas
- Clasificador de lenguaje natural
- Entendimiento del lenguaje natural
- Perspectivas de la personalidad
- Recuperar y Clasificar
- Dictado a texto
- Texto a voz
- Analizador de tonos
- Transacciones analíticas
- Reconocimiento visual

## Versiones

| Versión | Fecha de lanzamiento |
| --- | --- |
| 1.0.0 | 2016-05-05 |

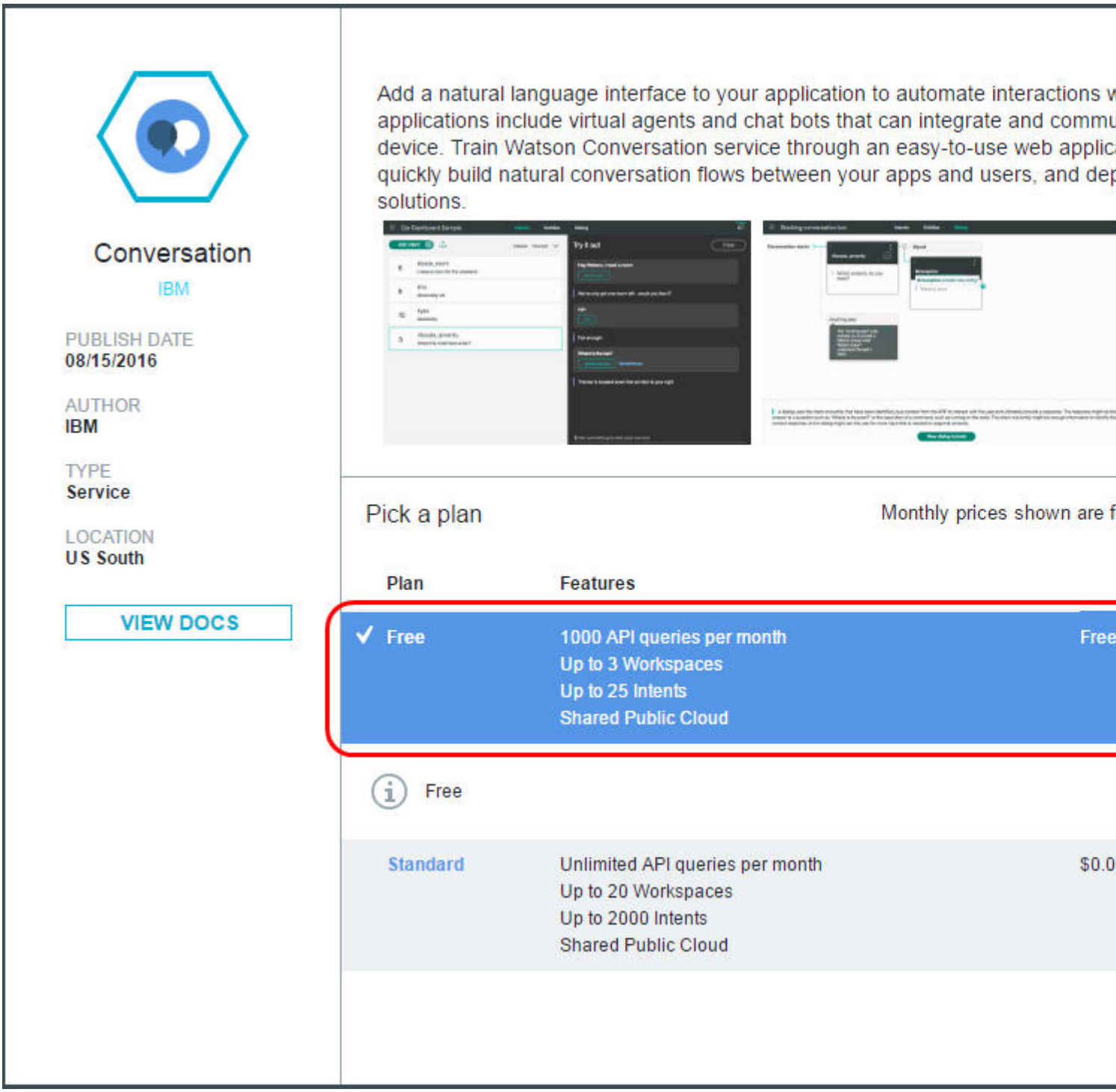
## Examples

### Obtención de credenciales API

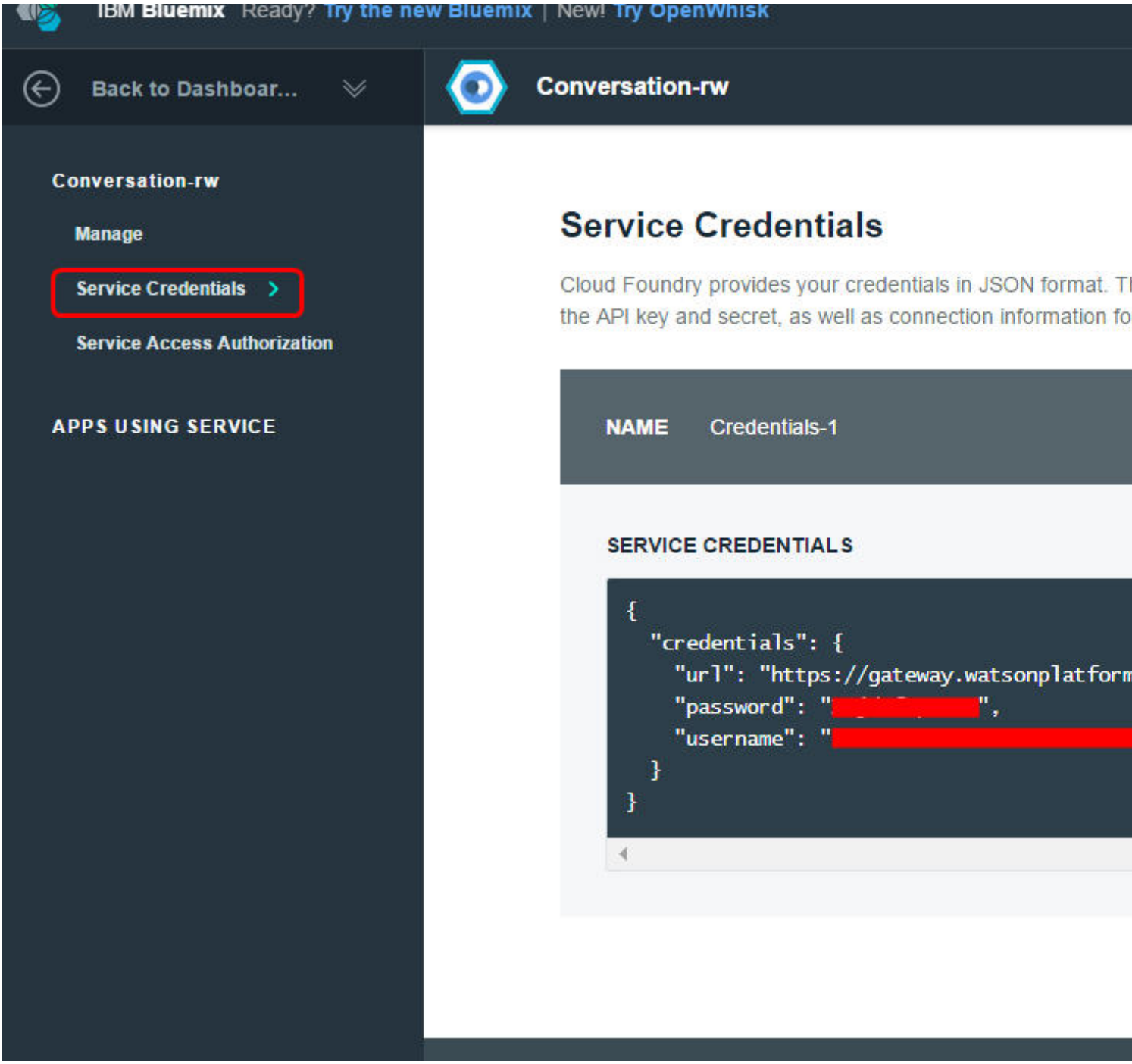
Para autenticarse en los servicios de Watson, necesita credenciales para cada servicio que planea usar. Dependiendo del servicio, deberá pasar un nombre de usuario y una contraseña con Autenticación básica, o deberá pasar una clave API en un parámetro para cada solicitud que realice.

Cómo obtener credenciales para un servicio de Watson:

1. Regístrese para Bluemix e inicie sesión.
2. Vaya a la página de servicio para el servicio Watson deseado:
    - AlchemyLanguage and AlchemyData News
    - Conversacion
    - Diálogo
    - Conversión de documentos
    - Traducción de idiomas
    - Clasificador de lenguaje natural
    - Perspectivas de la personalidad
    - Recuperar y Clasificar
    - Dictado a texto
    - Texto a voz
    - Analizador de tonos
    - Transacciones analíticas
    - Reconocimiento visual
3. Seleccione el plan deseado y haga clic en CREAR:




4. Haga clic en el botón "Credenciales de servicio" de la página del panel de servicio para ver sus credenciales. Si no es llevado automáticamente al panel de servicios, vaya a su panel de Bluemix y haga clic en la instancia de servicio deseada.




## Llamando a las API de Watson con curl

Dependiendo del servicio, deberá utilizar la autenticación básica con un `username` y `password` o pasar una `apikey` como parámetro en cada solicitud.

Algunos servicios también admiten la autenticación de token .

OBTENER utilizando el analizador de tono:

```
curl -X GET \
-u "username":"password" \
-d "version=2016-05-19" \
-d "text=Hey! Welcome to Watson Tone Analyzer!" \
"https://gateway.watsonplatform.net/tone-analyzer/api/v3/tone
```

POST utilizando AlchemyLanguage:

```
curl -X POST \
-d "apikey=YOUR_API_KEY" \
-d "url=www.ibm.com" \
"https://gateway-a.watsonplatform.net/calls/url/URLGetRankedKeywords"
```

## Usando Watson Developer Cloud SDKs

La forma más rápida de comenzar con los servicios de Watson es usar los SDK de la nube para desarrolladores de Watson. Los siguientes repositorios de GitHub contienen instrucciones de instalación y ejemplos de uso básico:

- Androide
- iOS
- Java
- Node.js
- Pitón
- Unidad

Por ejemplo, aquí se explica cómo realizar una llamada a la API de AlchemyLanguage con el SDK de Node.js:

Instale el SDK:

```
$ npm install watson-developer-cloud
```

Guarde el siguiente código en un archivo (lo llamaremos *app.js* ). Asegúrate de reemplazar `API_KEY` con tu clave API.

```
// Instantiate the service
var AlchemyLanguageV1= require('watson-developer-cloud/alchemy-language/v1');
var alchemy_language = AlchemyLanguageV1({
  api_key: 'API_KEY'
})

var parameters = {
  extract: [
    'entities',
    'keywords'
  ]
  url: 'https://www.ibm.com/us-en/'
};

alchemy_language.combined(parameters, function (err, response) {
  if (err)
    console.log('error:', err);
  else
    console.log(JSON.stringify(response, null, 2));
});
```

Ejecutar la aplicación:

```
$ node app.js
```

Lea Empezando con ibm-watson-cognitive en línea: https://riptutorial.com/es/ibm-watson-cognitive/topic/607/empezando-con-ibm-watson-cognitive

# Capítulo 2: Alquimia Idioma

## Observaciones

AlchemyLanguage es una colección de métodos de análisis de texto que brindan una visión más profunda de su texto o contenido HTML. Consulte el tema de Primeros pasos para obtener información sobre cómo comenzar a utilizar AlchemyLanguage y otros servicios de Watson. Para obtener más detalles y ejemplos de AlchemyLanguage, consulte la referencia y la documentación de la API .

# Límites de tamaño

- Contenido HTML antes de la limpieza del texto: **600 KB**
- Texto fuente, después de la limpieza del texto: **50 KB**
- Llamadas que utilizan modelos personalizados: **5 KB**

# Ayuda de idioma

Para ver qué idiomas son compatibles con cada función, consulte la entrada de cada función en la referencia de la API .

# Detección de lenguaje

De forma predeterminada, AlchemyLanguage detecta automáticamente el idioma del texto de origen. Puede especificar manualmente el idioma de su contenido con el parámetro de consulta de `language` . (eg `language=spanish` )

# Limpieza de texto

Cuando utiliza una función HTML o URL de la API, AlchemyLanguage limpia el contenido para preparar el texto de origen para el análisis. El parámetro `sourceText` permite personalizar el proceso de limpieza con las siguientes opciones:

- `cleaned_or_raw` (predeterminado): elimina elementos del sitio web como enlaces, anuncios, etc. Si la limpieza falla, se usa el texto sin formato de la página web
- `cleaned` : elimina elementos del sitio web, como enlaces, anuncios, etc.
- `raw` formato: utiliza el texto sin formato de la página web sin limpieza.
- `cquery` : utiliza la consulta de restricciones visuales que especifica en el parámetro `cquery` . Consulte la documentación para obtener detalles sobre las consultas de restricciones visuales.
- `xpath`

: utiliza la consulta XPath que especifique en el parámetro `xpath`

- `xpath_or_raw` : utiliza los resultados de una consulta XPath, y vuelve a ser texto sin formato si la consulta XPath no devuelve nada
- `cleaned_and_xpath` : utiliza los resultados de una consulta XPath en el texto de la página web limpiado

# Examples

## Llamada combinada: use múltiples funciones en una sola llamada de API (Node.js)

El método de llamada combinada te permite usar múltiples funciones de AlchemyLanguage en una solicitud. Este ejemplo utiliza una llamada combinada para obtener entidades y palabras clave del sitio web de IBM y devuelve información de sentimiento para cada resultado.

> Este ejemplo requiere las credenciales del servicio AlchemyLanguage y Node.js.

1. Use una interfaz de línea de comandos para instalar el SDK de Watson Developer Cloud Node.js :

```
$ npm install watson-developer-cloud
```

2. Guarde el siguiente código en un archivo *app.js* en el mismo directorio. Asegúrese de reemplazar `API_KEY` con su clave AlchemyAPI:

```
var AlchemyLanguageV1 = require('watson-developer-cloud/alchemy-language/v1');
var alchemy_language = AlchemyLanguageV1({
  api_key: 'API_KEY'
})

var parameters = {
  extract: 'entities,keywords',
  sentiment: 1,
  url: 'https://www.ibm.com/us-en/'
};

alchemy_language.combined(parameters, function (err, response) {
  if (err)
    console.log('error:', err);
  else
    console.log(JSON.stringify(response, null, 2));
});
```

3. Ejecutar la aplicación:

```
$ node app.js
```

## Análisis de sentimiento: obtenga información de sentimiento para frases específicas en texto (Node.js)

La función Targeted Sentiment de AlchemyLanguage puede buscar en su contenido frases específicas y devolver información de sentimiento para cada resultado.

> Este ejemplo requiere las credenciales del servicio AlchemyLanguage y Node.js

1. Use una interfaz de línea de comandos para instalar el SDK de Watson Developer Cloud Node.js :

```
$ npm install watson-developer-cloud
```

2. Guarde el siguiente código en un archivo *app.js* en el mismo directorio. Asegúrese de reemplazar `API_KEY` con su clave AlchemyAPI:

```
var AlchemyLanguageV1 = require('watson-developer-cloud/alchemy-language/v1');
var alchemy_language = new AlchemyLanguageV1({
  api_key: 'API_KEY'
})

var parameters = {
  text: 'Grapes are the best! I hate peaches.',
  targets: [
    'grapes',
    'peaches'
  ]
};

alchemy_language.sentiment(parameters, function (err, response) {
  if (err)
    console.log('error:', err);
  else
    console.log(JSON.stringify(response, null, 2));
});
```

3. Ejecutar la aplicación:

```
$ node app.js
```

## Conceptos: identificar conceptos de una página web (Node.js)

AlchemyLanguage puede detectar conceptos generales a los que se hace referencia en su contenido. El servicio devuelve enlaces de Datos vinculados para cada concepto y una URL a un sitio web relevante cuando sea posible.

> Este ejemplo requiere las credenciales del servicio AlchemyLanguage y Node.js

1. Use una interfaz de línea de comandos para instalar el SDK de Watson Developer Cloud Node.js :

```
$ npm install watson-developer-cloud
```

2. Guarde el siguiente código en un archivo *app.js* en el mismo directorio. Asegúrate de reemplazar `API_KEY` con tu clave AlchemyAPI.

```
var AlchemyLanguageV1 = require('watson-developer-cloud/alchemy-language/v1');
var alchemy_language = new AlchemyLanguageV1({
  api_key: 'API_KEY'
})

var parameters = {
  url: 'http://www.cnn.com'
};

alchemy_language.concepts(parameters, function (err, response) {
  if (err)
    console.log('error:', err);
  else
    console.log(JSON.stringify(response, null, 2));
});
```

3. Ejecutar la aplicación:

```
$ node app.js
```

Lea Alquimia Idioma en línea: https://riptutorial.com/es/ibm-watson-cognitive/topic/6817/alquimia-idioma

# Capítulo 3: Dictado a texto

## Observaciones

IBM Watson Speech to Text ofrece una variedad de opciones para transcribir audio en varios idiomas y formatos:

- **WebSockets** : establezca una conexión persistente a través del protocolo WebSocket para la transcripción continua
- **Sin sesión** : transcriba el audio sin la sobrecarga de establecer y mantener una sesión
- **Sesiones** : cree intercambios de turnos largos con el servicio o establezca varias conversaciones paralelas con una instancia particular del servicio
- **Asíncrono** : proporciona una interfaz HTTP sin bloqueo para transcribir audio. Puede registrar una URL de devolución de llamada para recibir notificaciones del estado y los resultados del trabajo, o puede sondear el servicio para conocer el estado del trabajo y recuperar los resultados manualmente.

Consulte el tema de Introducción para aprender cómo comenzar con Speech to Text y otros servicios de Watson. Para obtener más detalles y ejemplos de Speech to Text, consulte la referencia de la API y la documentación .

## Examples

### Reconociendo un archivo de audio usando WebSockets en Java

Usando el `Java-SDK 3.0.1`

```
CountDownLatch lock = new CountDownLatch(1);

SpeechToText service = new SpeechToText();
service.setUsernameAndPassword("<username>", "<password>");

FileInputStream audio = new FileInputStream("filename.wav");

RecognizeOptions options = new RecognizeOptions.Builder()
    .continuous(true)
    .interimResults(true)
    .contentType(HttpMediaType.AUDIO_WAV)
    .build();

service.recognizeUsingWebSocket(audio, options, new BaseRecognizeCallback() {
  @Override
  public void onTranscription(SpeechResults speechResults) {
    System.out.println(speechResults);
    if (speechResults.isFinal())
      lock.countDown();
  }
```

```
});

lock.await(1, TimeUnit.MINUTES);
```

## Transcribiendo un archivo de audio usando WebSockets (Node.js)

Este ejemplo muestra cómo utilizar el servicio IBM Watson Speech to Text para reconocer el tipo de archivo de audio y producir una transcripción del texto hablado en ese archivo.

> Este ejemplo requiere credenciales de servicio de voz a texto y Node.js

1. Instale el módulo npm para el SDK de Watson Developer Cloud Node.js :

```
$ npm install watson-developer-cloud
```

2. Cree un archivo JavaScript (por ejemplo, *app.js* ) y copie el siguiente código en él. Asegúrese de ingresar el `username` y la `password` para su instancia de servicio de voz a texto.

```
var SpeechToTextV1 = require('watson-developer-cloud/speech-to-text/v1');
var fs = require('fs');

var speech_to_text = new SpeechToTextV1({
  username: 'INSERT YOUR USERNAME FOR THE SERVICE HERE',
  password: 'INSERT YOUR PASSWORD FOR THE SERVICE HERE',
  url: 'https://stream.watsonplatform.net/speech-to-text/api'
});

var params = {
  content_type: 'audio/flac'
};

// Create the stream,
var recognizeStream = speech_to_text.createRecognizeStream(params);

// pipe in some audio,
fs.createReadStream('0001.flac').pipe(recognizeStream);

// and pipe out the transcription.
recognizeStream.pipe(fs.createWriteStream('transcription.txt'));

// To get strings instead of Buffers from received `data` events:
recognizeStream.setEncoding('utf8');

// Listen for 'data' events for just the final text.
// Listen for 'results' events to get the raw JSON with interim results, timings, etc.
['data', 'results', 'error', 'connection-close'].forEach(function(eventName) {
  recognizeStream.on(eventName, console.log.bind(console, eventName + ' event: '));
});
```

3. Guarde el archivo de audio de muestra *0001.flac* en el mismo directorio. Este código de ejemplo está configurado para procesar archivos **FLAC** , pero puede modificar la sección de `params` del código de ejemplo para obtener transcripciones de archivos de audio en otros formatos. Los formatos admitidos incluyen **WAV** (tipo `audio/wav` ), **OGG** (tipo `audio/ogg` ) y otros. Consulte la referencia de la API Speech to Text para obtener una lista completa.

4. Ejecute la aplicación (use el nombre del archivo que contiene el código de ejemplo)

```
$ node app.js
```

Después de ejecutar la aplicación, encontrará el texto transcrito de su archivo de audio en el archivo *transcription.txt* en el directorio desde el que ejecutó la aplicación.

Lea Dictado a texto en línea: https://riptutorial.com/es/ibm-watson-cognitive/topic/675/dictado-a-texto

# Capítulo 4: Reconocimiento visual

## Examples

### Obtener una lista de clasificadores personalizados

Esto enumera todos los clasificadores personalizados que ha entrenado.

```
'use strict';

let watson = require('watson-developer-cloud');

var visualRecognition = watson.visual_recognition({
  version: 'v3',
  api_key: process.env['API_KEY'],
  version_date:'2016-05-19'
});


let url =
"https://upload.wikimedia.org/wikipedia/commons/1/1c/Chris_Evans_filming_Captain_America_in_DC_cropped.


visualRecognition.classify({url: url}, function(error, results) {
  console.log(JSON.stringify(results,null,2));
});
```

### Obtener información sobre un clasificador personalizado específico

Esto devuelve información sobre un ID de clasificador específico que ha entrenado. Esto incluye información sobre su estado actual (es decir, si está listo o no).

```
'use strict';

let watson = require('watson-developer-cloud');

var visualRecognition = watson.visual_recognition({
  version: 'v3',
  api_key: process.env.API_KEY,
  version_date:'2016-05-19'
});


visualRecognition.getClassifier({classifier_id: 'DogBreeds_1162972348'}, function(error,
results) {
  console.log(JSON.stringify(results,null,2));
});
```

### Entrena un clasificador personalizado

La formación de un clasificador personalizado requiere un corpus de imágenes organizadas en grupos. En este ejemplo, tengo un montón de imágenes de manzanas en un archivo ZIP, un

montón de imágenes de plátanos en otro archivo ZIP y un tercer grupo de imágenes de cosas que *no* son frutos para un *conjunto negativo* . Una vez que se crea un clasificador personalizado, estará en `training` estatal y tendrá que usar la identificación del clasificador para verificar si está listo (usando el ejemplo 'Obtener información sobre un clasificador personalizado específico').

```
'use strict';

let watson = require('watson-developer-cloud');
let fs = require('fs');

var visualRecognition = watson.visual_recognition({
  version: 'v3',
  api_key: process.env.API_KEY,
  version_date:'2016-05-19'
});


let custom_classifier = {
  apple_positive_examples: fs.createReadStream('./apples.zip'),
  banana_positive_examples: fs.createReadStream('./bananas.zip'),
  negative_examples: fs.createReadStream('./non-fruits.zip'),
  name: 'The Name of My Classifier'
}

visualRecognition.createClassifier(custom_classifier, function(error, results) {
  console.log(JSON.stringify(results,null,2));
});
```

### Eliminar un clasificador personalizado

```
'use strict';

let watson = require('watson-developer-cloud');
let fs = require('fs');

var visualRecognition = watson.visual_recognition({
  version: 'v3',
  api_key: process.env.API_KEY,
  version_date:'2016-05-19'
});

let classifier_id_to_delete = 'TheNameofMyClassifier_485506080';

visualRecognition.deleteClassifier({classifier_id: classifier_id_to_delete}, function(error,
results) {
  console.log(JSON.stringify(results,null,2));
});
```

### Clasificar una imagen

# Prerrequisitos

Primero, debes instalar el SDK de `watson-developer-cloud` .

```
$ npm install watson-developer-cloud
```

# Clasificar una URL de imagen

Usaremos una imagen del Capitán América de Wikipedia.

```
'use strict';

let watson = require('watson-developer-cloud');

var visualRecognition = watson.visual_recognition({
  version: 'v3',
  api_key: "<YOUR API KEY GOES HERE>",
  version_date:'2016-05-19'
});


let url =
"https://upload.wikimedia.org/wikipedia/commons/1/1c/Chris_Evans_filming_Captain_America_in_DC_cropped.


visualRecognition.classify({url: url}, function(error, results) {
  console.log(JSON.stringify(results,null,2));
});
```

Lea Reconocimiento visual en línea: https://riptutorial.com/es/ibm-watson-

cognitive/topic/718/reconocimiento-visual

# Capítulo 5: Recuperar y Clasificar

## Observaciones

El cliente Solrj y el SDK de Java son independientes, por lo que puede actualizarlos individualmente. Siempre asegúrese de usar la última versión del SDK de Java.

Consulte la página de lanzamiento de GitHub para ver las actualizaciones https://github.com/watson-developer-cloud/java-sdk/releases

## Examples

### Busque y clasifique utilizando la función Recuperar y clasificar en Java

Instale las dependencias requeridas:

```
'org.apache.solr:solr-solrj:5.5.1'
'org.apache.httpcomponents:httpclient:4.3.6'
'com.ibm.watson.developer_cloud:java-sdk:3.2.0'
```

El siguiente código asume que tienes una colección de Solr con documentos y que has entrenado a un clasificador; de lo contrario, sigue este tutorial.

```
public class RetrieveAndRankSolrJExample {

  private static HttpSolrClient solrClient;
  private static RetrieveAndRank service;

  private static String USERNAME = "<username>";
  private static String PASSWORD = "<password>";
  private static String SOLR_CLUSTER_ID = "<your-solr-cluster-id>";
  private static String SOLR_COLLECTION_NAME = "<your-collection-name>";
  private static String RANKER_ID = "<ranker-id>";

  public static void main(String[] args) throws SolrServerException, IOException {

    // create the retrieve and rank instance
    service = new RetrieveAndRank();
    service.setUsernameAndPassword(USERNAME, PASSWORD);

    // create the solr client
    String solrUrl = service.getSolrUrl(SOLR_CLUSTER_ID);
    solrClient = new HttpSolrClient(solrUrl, createHttpClient(solrUrl, USERNAME, PASSWORD));

    // build the query
    SolrQuery query = new SolrQuery("*:*");
    query.setRequestHandler("/fcselect");
    query.set("ranker_id", RANKER_ID);

    // execute the query
    QueryResponse response = solrClient.query(SOLR_COLLECTION_NAME, query);
    System.out.println("Found " + response.getResults().size() + " documents!");
```

```
    System.out.println(response);
  }

  private static HttpClient createHttpClient(String uri, String username, String password) {
    final URI scopeUri = URI.create(uri);

    final BasicCredentialsProvider credentialsProvider = new BasicCredentialsProvider();
    credentialsProvider.setCredentials(new AuthScope(scopeUri.getHost(), scopeUri.getPort()),
        new UsernamePasswordCredentials(username, password));

    final HttpClientBuilder builder = HttpClientBuilder.create()
        .setMaxConnTotal(128)
        .setMaxConnPerRoute(32)

.setDefaultRequestConfig(RequestConfig.copy(RequestConfig.DEFAULT).setRedirectsEnabled(true).build())

        .setDefaultCredentialsProvider(credentialsProvider)
        .addInterceptorFirst(new PreemptiveAuthInterceptor());
    return builder.build();
  }

  private static class PreemptiveAuthInterceptor implements HttpRequestInterceptor {
    public void process(final HttpRequest request, final HttpContext context) throws
HttpException {
      final AuthState authState = (AuthState)
context.getAttribute(HttpClientContext.TARGET_AUTH_STATE);

      if (authState.getAuthScheme() == null) {
        final CredentialsProvider credsProvider = (CredentialsProvider) context
            .getAttribute(HttpClientContext.CREDS_PROVIDER);
        final HttpHost targetHost = (HttpHost)
context.getAttribute(HttpCoreContext.HTTP_TARGET_HOST);
        final Credentials creds = credsProvider.getCredentials(new
AuthScope(targetHost.getHostName(),
            targetHost.getPort()));
        if (creds == null) {
          throw new HttpException("No creds provided for preemptive auth.");
        }
        authState.update(new BasicScheme(), creds);
      }
    }
  }
}
```

Lea Recuperar y Clasificar en línea: https://riptutorial.com/es/ibm-watson-cognitive/topic/6053/recuperar-y-clasificar

# Creditos

| S. No | Capítulos | Contributors |
|---|---|---|
| 1 | Empezando con ibm-watson-cognitive | Community, Garrett M, German Attanasio |
| 2 | Alquimia Idioma | Garrett M |
| 3 | Dictado a texto | Garrett M, German Attanasio, seh, WvH |
| 4 | Reconocimiento visual | German Attanasio, Joshua Smith, seh |
| 5 | Recuperar y Clasificar | German Attanasio |