

IBM Watson Question API  
Version 1

*IBM Watson Platform Software  
Development Kit  
Developer's Guide*



IBM Confidential



IBM Watson Question API  
Version 1

*IBM Watson Platform Software  
Development Kit  
Developer's Guide*



IBM Confidential

**Notes**

Before using this information and the product it supports, be sure to read the general information under Notices at the end of this information.

**Prerelease information disclaimer:** This document contains proprietary information. All information contained herein shall be kept in confidence. None of this information shall be divulged to persons other than (a) IBM employees authorized by the nature of their duties to receive such information, or (b) individuals with a need to know in organizations authorized by IBM to receive this document in accordance with the terms (including confidentiality) of an agreement under which it is provided.

Last updated: March 13, 2014

This edition applies to Version 1 of IBM Watson Question API. This information applies to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces 26 November 2013.

© Copyright IBM Corporation 2013, 2014.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

## Chapter 1. IBM Watson platform

### software development kit . . . . . 1

Overview of the IBM Watson Platform SDK . . . . . 1

Contents of the Watson Platform SDK . . . . . 1

## Chapter 2. Installing and configuring

### IBM Watson Platform Software

### Development Kit for Eclipse . . . . . 3

Prerequisite software for IBM Watson Platform  
software development kit . . . . . 3

Installing the SDK project . . . . . 3

Configuring the URL of the Question and Answer  
REST service . . . . . 4

Configuring the proxy server . . . . . 4

Associating Watson Platform SDK projects with the  
server in Eclipse . . . . . 5

Configuring the SDK for Java 1.6 . . . . . 5

## Chapter 3. Sample Question and Answer application . . . . . 7

## Chapter 4. Asking questions and receiving answers . . . . . 9

Overview of the Question and Answer REST service . . . . . 9

Changes roadmap on Question API. . . . . 9

Question and Answer samples and examples . . . . . 10

Example requests and responses . . . . . 10

Sample code to call the Question and Answer  
service . . . . . 20

Troubleshooting for the Question and Answer  
service . . . . . 21

Pipeline messages and HTTP codes . . . . . 21

Messages returned with an answer response . . . . . 25

Operations on Question . . . . . 25

POST /deepqa/v1/question. . . . . 26

POST /deepqa/v1/question: Asynchronous  
(deprecated) . . . . . 38

GET /deepqa/v1/question/{QID} (deprecated) . . . . . 41

POST /deepqa/v1/question/{QID}/feedback  
(deprecated) . . . . . 47

## Chapter 5. The JavaScript toolkit . . . . . 53

JavaScript samples . . . . . 53

Design patterns for an IBM Watson enabled  
application . . . . . 54

## Chapter 6. SDK PHP library overview . . . . . 69

## Notices . . . . . 71

Trademarks . . . . . 72

Privacy policy considerations . . . . . 72



---

## Chapter 1. IBM Watson platform software development kit

IBM Watson™ platform software development kit (SDK) supports developers who create applications that interact with Watson. The kit includes capabilities and samples that enable your application to submit questions to Watson and receive answers, evidence, and other information.

---

### Overview of the IBM Watson Platform SDK

By using the comprehensive set of applications and libraries in the IBM Watson Platform software development kit (SDK), you can develop applications for Watson in Java™ and JavaScript. The SDK also includes a preview PHP example.

The SDK includes the following components

- An end-to-end application that uses the JavaScript widgets and connects to Watson through the Question and Answer REST service.
- A set of samples that demonstrate how you connect to the REST service.
- A toolkit of JavaScript files and widgets that can be used in creating your own applications.
- A small PHP library to add Watson to your PHP application

**Related information:**



Overview video

A video that demonstrates some of the capabilities of the SDK samples.

---

### Contents of the Watson Platform SDK

The Watson Platform SDK is organized into a set of projects and a PHP library.

The SDK includes the following projects:

**com.ibm.watson.sdk.dojo**

A version of Dojo 1.8, which is required by the JavaScript toolkit. The Dojo toolkit is available to the toolkit and to other applications that are based on JavaScript.

**com.ibm.watson.sdk.qaapi.samples**

A set of samples that demonstrate how you connect to the Question and Answer REST service.

**com.ibm.watson.sdk.wea.proxy**

A proxy that allows cross-site access to the Watson RESTful service endpoint. You configure entries in the WebContent/WEB-INF/config.properties file in the com.ibm.watson.sdk.wea.proxy project.

**com.ibm.watson.sdk.wea.toolkit**

JavaScript widgets that you can embed and use this project with your applications.

**com.ibm.watson.sdk.wea.toolkit.app.samples**

An end-to-end sample that can be used to post a question to Watson and receive the results. The sample works with the com.ibm.watson.sdk.wea.proxy.

**com.ibm.watson.sdk.wea.toolkit.samples**

Sample applications that show examples of the toolkit widgets. You can use these samples as starting points for your own web applications.

**com.ibm.watson.sdk**

Common libraries and a small PHP library to help connect your own PHP application to Watson.

**com.ibm.watson.sdk.doc**

Information about the SDK.



---

## Chapter 2. Installing and configuring IBM Watson Platform Software Development Kit for Eclipse

Use these topics when you install IBM Watson Platform Software Development Kit on Eclipse.

---

### Prerequisite software for IBM Watson Platform software development kit

IBM Watson Platform SDK requires a development environment that supports Java development. Verify that your environment meets the requirements.

- Verify that you have IBM® SDK Java Technology Edition Version 6 or later.
- Verify the version of your Eclipse workbench:
  - Eclipse IDE for Java EE Developers version 3.7.2 or 4.3.0. The SDK might work with other versions of Eclipse, but is not tested.
  - Rational® Application Developer version 8.0 or later.
- Verify that you installed a web application server and that it is available to Eclipse. The Watson Platform SDK is tested with Tomcat version 7.0.

The Watson Platform SDK includes several JavaScript projects. You can run the applications from within Eclipse by adding a server to Eclipse and associating the server with the projects.

---

### Installing the SDK project

You install the Watson Platform software development kit (SDK) in Eclipse by importing the SDK project workspace archive file.

#### Before you begin

Verify that your environment meets the requirements of the SDK. For more information about prerequisites, see, “Prerequisite software for IBM Watson Platform software development kit.”

#### Procedure

1. Download the compressed file of the latest build of Watson Platform SDK from the build location. The file name has the form `WatsonPlatformSDK-date-time.zip`. For example, the SDK for August 14, 2013 is named `WatsonPlatformSDK-20130814-0716.zip`
2. Start Eclipse.
3. Select **File > Import**.
4. Select **General > Existing Project into Workspace**.
5. Select the archive file option and browse to the location where you saved the SDK. If you use Rational Application Developer version 8.0 or later, you might be prompted to migrate the workspace after you import the projects.

#### What to do next

Configure the URL of the Question and Answer API.

**Related information:**

 Installation video

A video that shows how to import the SDK projects to Eclipse.

---

## Configuring the URL of the Question and Answer REST service

To make requests to the IBM Watson Question and Answer REST service, the connection to the REST server URL must be configured properly. If you have your own installation of Watson, change the URL to match your server.

### Procedure

To change the URL of the Question and Answer REST server that you connect to, follow these steps:

1. In Eclipse, browse to the `com.ibm.watson.sdk.qaapi.samples` project.
2. Open the `qaapi.properties` file with a text editor.
3. Enter the URL for the REST server that you are using as the value for **server** and save the file.

---

## Configuring the proxy server

To run the Question and Answer sample application, you configure the included proxy server.

### About this task

The `com.ibm.watson.sdk.wea.toolkit.proxy` project contains a small proxy server that handles cross-site scripting of the application. The Watson Question and Answer REST service are hosted on a different domain from the page with the original content. So requests are blocked by the browser if accessed directly. To handle the requests, the sample application connects to the proxy, and the proxy makes the request to the REST service. The REST services respond with the URLs of the REST service. The proxy then replaces those URLs with the Watson URLs.

### Procedure

1. Browse to the `com.ibm.watson.sdk.wea.proxy` project.
2. Open the `WebContent/WEB-INF/config.properties` file with a text editor.
3. Configure the proxy by specifying the following values
  - a. Enter the URL for the REST server you are using as the value for **qapiServerUrl**. This value is likely to be the same value that you used when you configured the URL of the Question and Answer service.
  - b. Enter the URL of the proxy as the value for **proxyServerUrl**. For example, `proxyServerUrl=http://localhost:8080/proxy1/proxy/weaQuestion/`.
4. Save the file.

### What to do next

Deploy the projects to your web application server as described in “Associating Watson Platform SDK projects with the server in Eclipse” on page 5.

## Associating Watson Platform SDK projects with the server in Eclipse

You create a relationship between the SDK toolkit projects and the server you created in Eclipse so that you can view the samples in a browser. You create the relationship by adding the projects to the server.

### Before you begin

- Install the SDK project
- Verify that you installed a web application server and that it is available to Eclipse.
- Configure the proxy server.

### Procedure

1. In the Servers view in Eclipse, right-click the server and select **Add and Remove Projects**.
2. In the list of available projects, select the projects that you want to test and click **Add**, and then click **Finish**.

To run this	Associate these projects
Sample Question and Answer application	<ul style="list-style-type: none"> <li>• com.ibm.watson.sdk.dojo</li> <li>• com.ibm.watson.sdk.wea.toolkit</li> <li>• com.ibm.watson.sdk.wea.toolkit.app.samples</li> <li>• com.ibm.watson.sdk.wea.proxy</li> </ul>
JavaScript Toolkit widgets	<ul style="list-style-type: none"> <li>• com.ibm.watson.sdk.dojo</li> <li>• com.ibm.watson.sdk.wea.toolkit</li> <li>• com.ibm.watson.sdk.wea.toolkit.samples</li> </ul>

3. Publish the applications by right-clicking the server and selecting **Publish**.
4. If necessary, start the server.

### Related information:

 Overview video

A video that demonstrates some of the capabilities of the SDK samples.

## Configuring the SDK for Java 1.6

The Eclipse projects for the IBM Watson Platform SDK are configured to use Java 1.7. To use Java 1.6, you reconfigure the projects.

### Procedure

Follow these steps to change the Eclipse workbench to use Java 1.6:

1. Change the JDK compliance:
  - a. Select **Windows > Preferences**.
  - b. In the Preferences window, expand the **Java** settings and select **Installed JREs**.
  - c. Confirm that you have a Java 1.6 installed.
  - d. Select the **Compiler** settings.
  - e. Change the **Compiler compliance level** to 1.6.
2. Update the Project Facets to use Java 1.6 for the following projects:
  - com.ibm.watson.sdk.wea.proxy

- com.ibm.watson.sdk.wea.toolkit.app.samples
- com.ibm.watson.sdk.wea.toolkit.samples
- a. In the **Project explorer** view, right-click each project and select **Properties**.
- b. In the Properties window, select **Project Facets**.
- c. Select 1.6 for the version of the Java project facet.

## Chapter 3. Sample Question and Answer application

The Question and Answer application is an end-to-end sample that posts a question to Watson and returns the answer.

### Before you begin

- Configure the proxy server to connect to the Question and Answer Rest service.
- Associate the SDK projects that are needed by the application with your web server. The Question and Answer application uses the `com.ibm.watson.sdk.dojo`, `com.ibm.watson.sdk.wea.toolkit`, `com.ibm.watson.sdk.wea.toolkit.samples.app`, and `com.ibm.watson.sdk.wea.proxy` projects.
- If necessary, start your web server.

### Running the application

Run the Question and Answer application by opening `http://<server>:<port>/watson/qa/demo/index.html` file in your browser. For example, `http://localhost:8080/watson/qa/demo/index.html`.

### About the sample application

The Question and Answer application uses the `QuestionInput`, `HistoryDialog`, and `ProgressDialog` widgets to create an end-to-end sample. To learn more about the code, browse to `com.ibm.watson.sdk.wea.toolkit.app.samples/WebContent` and open the `index.html` file. The `QuestionInput` accepts as a parameter the Watson endpoint to connect to. In this case, the included proxy handles cross-site access to the Watson REST services.

```
<div id=qiOne data-dojo-type="com.ibm.watson.ui.QuestionInput"
data-dojo-props="url:'/proxy/proxy/weaQuestion',numberAnswers:'4',numberEvidence:'2'">
</div>
```



---

## Chapter 4. Asking questions and receiving answers

You use the IBM Watson Question and Answer REST service to submit questions to Watson and receive answers.

---

### Overview of the Question and Answer REST service

Use the IBM Watson Question and Answer REST service to submit questions and retrieve answers.

**Important:** With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. If you implemented the Question and Answer REST service using the asynchronous mode, update your code to use the synchronous mode instead.

The Question and Answer REST service enables applications and tools to submit questions to a Watson pipeline and obtain a set of answers, supporting evidence and an evidence profile. You submit a question and the connection remains open until answers are returned. The response either contains the answer or indicates that the question failed or timed out.

#### REST resources

The Question and Answer service exposes a set of resources through a REST service interface. These resources represent questions that are being processed and answers to questions.

These URIs start at the context root of the IBM Watson frontend services web application. The following examples show the resource URL to submit a question for two web servers:

**Jetty**

`http://myhost.mydomain.com:8080/deepqa/v1/question`

**WebSphere® Application Server Liberty**

`http://myhost.mydomain.com:9080/WatsonApp/deepqa/v1/question`

---

### Changes roadmap on Question API

Learn about changes to the Question and Answer REST service in the roadmap.

#### Asynchronous mode deprecated

The asynchronous mode of v1 of the Question and Answer REST service is deprecated as of March 21, 2014. You do not need to make changes if you include the X-SyncTimeout header with your POST operation to the /deepqa/v1/question endpoint. The X-SyncTimeout header indicates that you are using the synchronous mode. If you implemented the Question and Answer service using the asynchronous mode, update your code to use the synchronous mode instead.

The following operations are affected by this change:

- POST /deepqa/v1/question without the X-SyncTimeout header
- GET /deepqa/v1/question/{QID}

- POST /deepqa/v1/question/{QID}/feedback

## Question and Answer samples and examples

Samples and examples to help you understand how to call the IBM Watson Question and Answer REST service.

### Example requests and responses

Follow these examples to learn how to submit questions and handle responses with the Watson Question and Answer REST service.

#### Related concepts:

“Overview of the Question and Answer REST service” on page 9

Use the IBM Watson Question and Answer REST service to submit questions and retrieve answers.

### Example question post with required elements (synchronous mode)

Follow this simple example to learn the process of submitting a question and receiving answers in synchronous mode. After you understand this example, explore other more complex examples.

Although the Watson Question and Answer REST service accepts many elements, only one element is required to post a question. In this example, you post a question that includes only the required element and receive a successful response with answers.

You follow three steps to post a question and receive answers:

1. Post the question.
2. The client request blocks until a response is received or times out.
3. Parse the answers or resolve errors.

### Post the question

When you post a question, include an HTTP header. The header must include the content-type. Because this example uses synchronous mode, a value of 30 seconds exists for the **X-SyncTimeout** header element. That value represents how long that the server waits after the question is submitted until it times out. The value does not represent how long the client waits for a response from the server.

```
Accept: application/json
Content-Type: application/json
X-SyncTimeout : 30
```

Post the question. In this JSON example, only the required `questionText` element is included.

```
{
  "question": {
    "questionText": "His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar."
  }
}
```

Here is the same question that is submitted as XML:

```
<question>
  <questionText>"His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar."</questionText>
</question>
```



## Receive the response

After you submit a question, the response includes an HTTP status code. For synchronous mode, the successful HTTP status code is 200 Created. The successful response also includes a Complete question status and the answers.

The answers section contains answers that are ranked by confidence. The value for confidence is a decimal percentage that represents the confidence that Watson has in this answer. Higher values represent higher confidences. The text section contains the text of each answer.

The information that is returned in the question section includes what was submitted with the question and extra metadata that is returned by Watson.

```
{
  "question": {
    "qclasslist": [
      {
        "value": "FACTOID"
      }
    ],
    "focuslist": [
      {
        "value": "His"
      }
    ],
    "latlist": [
      {
        "value": "he"
      }
    ],
    "pipelineid": "433259107",
    "category": "",
    "items": 5,
    "status": "Complete",
    "id": "AE71A54A2BED4AE7BAAEEA98764ECB47",
    "questionText": "His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar.",
    "evidenceRequest": {
      "items": -1,
      "profile": "NO"
    }
  },
  "answers": [
    {
      "id": 0,
      "text": "Michael Jackson",
      "confidence": 0.42276
    },
    {
      "id": 1,
      "text": "Rick Derringer",
      "confidence": 0.22121
    },
    {
      "id": 2,
      "text": "Thriller",
      "confidence": 0.12293
    },
    {
      "id": 3,
      "text": "Jeff Porcaro",
      "confidence": 0.09981
    },
    {
      "id": 4,
      "text": "Quincy Jones",
      "confidence": 0.05383
    }
  ],
  "errorNotifications": [],
  "passthru": ""
}
```

## Common errors in synchronous mode

If you receive a successful HTTP return status code of 200 but a question status of Timeout, submit the question with a higher value for **X-SyncTimeout**. Alternatively, set the value to -1 to direct the server to wait until the answer is returned or the question fails. If the question status is Failed, examine the request for errors. If the request looks good, the error might be at the Watson server.

If the HTTP return status is 400, examine the request to determine the cause. For example, you included an invalid value for **X-SyncTimeout**.

This example checks the state before it retrieves the answers:

```
If Response status is "Complete"
--> Retrieve answer

elseif Response status is "Timeout"
--> Submit question with a longer X-SyncTimeout time value

elseif Response status is "Failed"
--> Examine the payload to determine the cause.

else Unexpected Response status

elseif HTTP status is 400
--> Examine the payload to determine the cause
-->(for example,an invalid X-SyncTimeout value)

else Unexpected HTTP status
```

## Example question post with required elements (asynchronous mode)

In this example, you submit a basic request to the Question and Answer REST service in asynchronous mode. The question includes only the required elements.

**Important:** With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. If you implemented the Question and Answer REST service using the asynchronous mode, update your code to use the synchronous mode instead.

In asynchronous mode, Watson returns a unique identifier URL in the response to submitting a question. With asynchronous modes, the steps to retrieve the answer are different than with synchronous mode. You poll the server with a GET request that includes the identifier to determine when Watson finishes processing the question. When processing is done, the successful response contains the answers and other information. In asynchronous mode, you follow these steps in a successful flow:

1. Post the question
2. Receive the initial response with an HTTP response code of 201. The payload data contains a response status of Accepted, and a unique URL to retrieve the answer.
3. Poll the server with a GET request to the URL while the response status is Queued.
4. When the response status is Complete, the answers are returned with the response.
5. Parse the answers or resolve errors

## Post the question

When you post a question, include an HTTP header. The header must include the content-type.

```
Accept: application/json
Content-Type: application/json
```

Post the question. In this JSON example, only the required `questionText` element is posted.

```
{
  "question": {
    "questionText": "His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar."
  }
}
```

Here is the same question that is submitted as XML:

```
<question>
  <questionText>"His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar."</questionText>
</question>
```

## Receive the initial response

After you post a question, the initial asynchronous response includes an HTTP response status code. For asynchronous mode, the successful HTTP status code is 201 Created. The response also includes a question status. If Watson is processing the question, the status is Accepted. The JSON response is similar to the following example:

```
{
  "question": {
    "links": {
      "self": "http://myhost.example.com:8080/deepqa/v1/question/
              /AE71A54A2BED4AE7BAAEEA98764ECB47",
      "feedback": "http://myhost.example.com:8080/deepqa/v1/question/
                  /AE71A54A2BED4AE7BAAEEA98764ECB47/feedback"
    },
    "id": "AE71A54A2BED4AE7BAAEEA98764ECB47",
    "questionText": "His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar.",
    "status": "Accepted",
    "items": 5,
    "passthru": "",
    "evidenceRequest": {
      "items": -1,
      "profile": "NO"
    }
  }
}
```

The `links` element is unique to the asynchronous mode. Use the `self` value to poll for updated response status and to retrieve the answer when Watson finishes processing the question. The `feedback` value is provided so that you can submit feedback on the answers.

## Poll the server and retrieve the answer

After Watson accepts the question, issue a GET request to retrieve the answer:

```
HTTP Header: Accept : application/json
GET http://myhost.example.com:8080/deepqa/v1/question/AE71A54A2BED4AE7BAAEEA98764ECB47
```

You can also issue the request with an `alt` parameter, which specifies whether XML or JSON is accepted for the response. By using the `alt` parameter, you do avoid having to set the HTTP header.

```
GET http://myhost.example.com:8080/deepqa/v1/question/AE71A54A2BED4AE7BAAEEA98764ECB47?alt=json
```

Parse the response to determine the response status. While Watson is processing the question, the response status is Queued. If something goes wrong, the response status changes to Failed.

When Watson finishes processing the question, the response status changes to Complete and the response includes the answers. The answers section contains answers that are ranked by confidence. The value for confidence is a decimal percentage that represents the confidence that Watson has in this answer. Higher values represent higher confidences. The text section contains the text of each answer.

The information that is returned in the question section is either what was submitted with the question or information about how Watson classified and identified the question.

```
{
  "question": {
    "links": {
      "self": "http://myhost.example.com:8080/deepqa/v1/question/
/AE71A54A2BED4AE7BAAEEA98764ECB47",
      "feedback": "http://myhost.example.com:8080/deepqa/v1/question/
/AE71A54A2BED4AE7BAAEEA98764ECB47/feedback"
    },
    "qclasslist": [
      {
        "value": "FACTOID"
      }
    ],
    "focuslist": [
      {
        "value": "His"
      }
    ],
    "latlist": [
      {
        "value": "he"
      }
    ],
    "pipelineid": "433259107",
    "category": "",
    "items": 5,
    "status": "Complete",
    "id": "AE71A54A2BED4AE7BAAEEA98764ECB47",
    "questionText": "His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar.",
    "evidenceRequest": {
      "items": -1,
      "profile": "NO"
    },
    "answers": [
      {
        "id": 0,
        "text": "Michael Jackson",
        "confidence": 0.42276
      },
      {
        "id": 1,
        "text": "Rick Derringer",
        "confidence": 0.22121
      },
      {
        "id": 2,
        "text": "Thriller",
        "confidence": 0.12293
      },
      {
        "id": 3,
        "text": "Jeff Porcaro",
        "confidence": 0.09981
      },
      {
        "id": 4,
        "text": "Quincy Jones",
        "confidence": 0.05383
      }
    ]
  },
}
```

```

    "errorNotifications": [],
    "passthru": ""
  }
}

```

## Common errors in asynchronous mode

If you receive a successful HTTP return status code of 201 but a question status of Failed, examine the request for errors. If the request looks good, the error might be at the Watson server.

If the HTTP return status is 400, examine the request to determine the cause. For example, you included an invalid value for an element.

## Example post for answers with evidence

In this example, you ask for evidence to be returned with the answers from Watson. To get the API to return evidence for a particular answer, you include an evidence request with the question.

### Post the question

When you ask IBM Watson a question, the system processes the question to find the answer. Part of that processing involves identifying one or more documents within the knowledge base that contain the answer. The document snippets that show the answer in context are referred to as evidence passages. The Question and Answer REST service returns the background passages for each answer.

The basic example for posting question included only the `questionText`. This JSON example includes the `evidenceRequest` element with two attributes: `items` and `profile`. The value of 1 for the `items` attribute specifies one supporting passage for each answer. The evidence profile identifies the sources of the evidence and is returned in this example:

```

{
  "question": {
    "questionText": "His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar."
    "evidenceRequest": { "items": 1, "profile": "yes" }
  }
}

```

Here is the same question that is submitted as XML:

```

<p>
  <question>
    <questionText>"His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar."</questionText>
    <evidenceRequest items="1" profile="yes" />
  </question>
</p>

```

## Receive the response

In the response, the `evidenceProfile` element contains the sources for the evidence and is unique to the Watson pipeline for which it was configured. The evidence passage and related information is contained in the evidence section.

```

{
  "question": {
    "qclasslist": [
      {
        "value": "FACTOID"
      }
    ],
    "focuslist": [
      {
        "value": "His"
      }
    ]
  }
}

```

```

},
"latlist": [
  {
    "value": "he"
  }
],
"pipelineid": "433259107",
"category": "",
"items": 5,
"status": "Complete",
"id": "AE71A54A2BED4AE7BAAEEA98764ECB47",
"questionText": "His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar.",
"evidenceRequest": {
  "items": 1,
  "profile": "YES"
},
"answers": [
  {
    "id": 0,
    "text": "Michael Jackson",
    "confidence": 0.42276
    "evidenceProfile": [
      {
        "name": "FeatureGroup_Linguistic",
        "value": "0.9999993141075441"
      },
      {
        "name": "FeatureGroup_Type",
        "value": "2.9766025382203176E-7"
      },
      {
        "name": "FeatureGroup_Spatio-temporal",
        "value": "1.8527814689180015E-7"
      },
      {
        "name": "FeatureGroup_Relevance",
        "value": "2.0295405528818982E-7"
      }
    ]
  },
  "evidence": [
    {
      "document": "http://myhost.example.com:8080/deepqa/v1/question/document/611396/48/50",
      "id": "611396",
      "title": "Rick Derringer",
      "text": "He played guitar or mandolin with \"Weird Al\" on many albums. For example, he offered both an homage and parody to Eddie Van Halen when we played the guitar solo on the track \"Eat It.\" Van Halen played the solo in \"Beat It\" on a Michael Jackson album.",
      "copyright": "http://en.example.org/wiki/Copyrights",
      "termsOfUse": "http://en.example.org/wiki/Terms"
    }
  ]
},
{
  "id": 1,
  "text": "Rick Derringer",
  "confidence": 0.22121
  "evidenceProfile": [
    {
      "name": "FeatureGroup_Linguistic",
      "value": "0.9999992097788784"
    },
    {
      "name": "FeatureGroup_Type",
      "value": "3.343194026679231E-7"
    },
    {
      "name": "FeatureGroup_Spatio-temporal",
      "value": "1.9799165313408648E-7"
    },
    {
      "name": "FeatureGroup_Relevance",
      "value": "2.579100657814232E-7"
    }
  ]
},
"evidence": [
  {
    "document": "http://myhost.example.com:8080/deepqa/v1/question/document

```

```

        /243536/86/88",
        "id": "243536",
        "title": "Music of Louisiana",
        "text": "Danny Johnson has been the guitarist in Steppenwolf for the last 16
                years. He has performed with many of the greats, including Rod
                Stewart, Rick Derringer, (Eddie Van Halen) Private Life, Danny
                Johnson and the Bandits, and Axis.",
        "copyright": "http://en.example.org/wiki/Copyrights",
        "termsOfUse": "http://en.example.org/wiki/Terms"
    }
],
},
{
    "id": 2,
    "text": "Thriller",
    "confidence": 0.12293
    "evidenceProfile": [
        {
            "name": "FeatureGroup_Linguistic",
            "value": "0.9999993127936775"
        },
        {
            "name": "FeatureGroup_Type",
            "value": "3.0981451898308303E-7"
        },
        {
            "name": "FeatureGroup_Spatio-temporal",
            "value": "1.6811651172801184E-7"
        },
        {
            "name": "FeatureGroup_Relevance",
            "value": "2.0927529166792349E-7"
        }
    ],
    "evidence": [
        {
            "document": "http://myhost.example.com:8080/deepqa/v1/question/document/
                        1287699/82/83",
            "id": "1287699",
            "title": "1984 (Van Halen album)",
            "text": "The album 1984 made it to #2 on the Billboard charts. Thriller,
                    by Michael Jackson was number one, and featured Van Halen on solo
                    for the track \"Beat It.\".",
            "copyright": "http://en.example.org/wiki/Copyrights",
            "termsOfUse": "http://en.example.org/wiki/Terms"
        }
    ]
},
{
    "id": 3,
    "text": "Jeff Porcaro",
    "confidence": 0.09981
    "evidenceProfile": [
        {
            "name": "FeatureGroup_Linguistic",
            "value": "0.9999992183919334"
        },
        {
            "name": "FeatureGroup_Type",
            "value": "3.3287720392044685E-7"
        },
        {
            "name": "FeatureGroup_Spatio-temporal",
            "value": "1.9713754980690284E-7"
        },
        {
            "name": "FeatureGroup_Relevance",
            "value": "2.5159331287407007E-7"
        }
    ],
    "evidence": [
        {
            "document": "http://myhost.example.com:8080/deepqa/v1/question/document/
                        9752690/23/24",
            "id": "9752690",
            "title": "Lukather",
            "text": "In addition to Toto members Jeff Porcaro and David Paich, Eddie
                    Van Halen, Steve Stevens, Richard Marx, Jan Hammer also worked on
                    Lukather. ",

```

```

        "copyright": "http://en.example.org/wiki/Copyrights",
        "termsOfUse": "http://en.example.org/wiki/Terms"
    }
]
},
{
    "id": 4,
    "text": "Quincy Jones",
    "confidence": 0.05383
    "evidenceProfile": [
        {
            "name": "FeatureGroup_Linguistic",
            "value": "0.9999992965709585"
        },
        {
            "name": "FeatureGroup_Type",
            "value": "3.0864687031138067E-7"
        },
        {
            "name": "FeatureGroup_Spatio-temporal",
            "value": "1.7915949516887396E-7"
        },
        {
            "name": "FeatureGroup_Relevance",
            "value": "2.1562267602537032E-7"
        }
    ],
    "evidence": [
        {
            "document": "http://myhost.example.com:8080/deepqa/v1/question/document
                        /26496842/272/273",
            "id": "26496842",
            "title": "Eddie Van Halen",
            "text": "Quincy Jones invited Eddie van Halen to play the guitar solo
                    on \"Beat It,\" for Michael Jackson's album Thriller. ",
            "copyright": "http://en.example.org/wiki/Copyrights",
            "termsOfUse": "http://en.example.org/wiki/Terms"
        }
    ]
}
],
"errorNotifications": [],
"passthru": ""
}
}

```

## Example post to control the number and format of answers

Watson returns five plain text answers unless you specify otherwise. In this example, you specify the number of answers you want returned and that you want the answer text in HTML format.

You control the number of answers which Watson returns by specifying a value for **items** when you post the question. If you do not specify the number of items, Watson assumes that you want five answers. The maximum number of answers Watson returns is 10. Pay attention to the confidence values of the answers. The answers are ranked by confidence values, and after five answers, the confidence value might approach 0.0.

```

POST
{
    "question":{
        "questionText":"His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar." ,
        "items" : 6
    }
}

```

If Watson is not able to return the number of answers that you request, the response includes a message, as shown in this example JSON code:

```

    "errorNotifications" : [{
        "id" : "LateSPRAnnotator",
        "text" : "AQWAL02017W Unable to obtain the requested number of answers."
    }
],

```



By default, Watson returns answer text as plain text. You can specify that Watson also return the answer text with extra HTML formatting. You specify the special formatting by including the **formattedAnswer** element with a value of true. The formatted answer text is returned in the `formattedText` section of the response.

```
POST
{
  "question":{
    "questionText":"His 1983 hit \"Beat it\" featured Eddie Van Halen on guitar." ,
    "formattedAnswer" : true
  }
}
```

The response contains an HTML-formatted version of the answer. The following example shows a fragment of `formattedText`.

```
.....
.....
"answers": [
  {
    "id": 0,
    "text": "Michael Jackson",
    "formattedText" : "<p>Michael Jackson</p>"
    "confidence": 0.42276
  },
  {
    "id": 1,
    "text": "Rick Derringer",
    "formattedText" : "<p>Rick Derringer</p>",
    "confidence": 0.22121
  },
  .....
  .....
```

## Synonym feedback example

Watson uses synonyms to improve answer recall. In release 2.2 and later, you can provide the list of synonyms that you want Watson to use when it answers a question, which might improve retrieval results.

You provide a list of synonyms by resubmitting a question and including the synonyms that Watson returned with the answer. For synonyms that you want Watson to ignore, set the **isChosen** attribute to false. Watson uses the updated list of synonyms in the query expansion phase and returns new answers.

Follow these steps to specify synonyms to use for answering a question:

1. Post a question.
2. Watson returns an answer.
3. Adjust the synonyms.
4. Resubmit the question with the adjusted synonyms.
5. Watson returns the updated answer.

## Post a question

Follow your normal process to submit a question.

```
<question xmlns="http://www.ibm.com/xmlns/deepqa/question">
  <questionText>What colour does milk have?</questionText>
</question>
```

## Watson returns the answer and synonyms

The response with the answer includes a list of synonyms that were used in the `synonymList` section:

```
<synonymList>
  <term lemma="colour" partOfSpeech="noun" value="colour">
    <synSet name="Wordnet_refer-verb-2">
      <synonym isChosen="true" weight="1.0">color</synonym>
      <synonym isChosen="true" weight="1.0">colouring material</synonym>
    </synSet>
  </term>
</synonymList>
```

When no synonyms are found, an empty `synonymList` is returned

## Adjust the synonyms

Parse the list of synonyms and change the value of `isChosen`. For example, specify that Watson not use the synonym `colouring material` by changing the value of `isChosen` from `true` to `false`.

```
<synonym isChosen="false" weight="1.0">colouring material</synonym>
```

## Resubmit the question with the adjusted synonyms

Resubmit the original question and include the updated list of synonyms.

```
<question xmlns="http://www.ibm.com/xmlns/deepqa/question">
  <questionText>What colour does milk have?</questionText>
  <synonymList>
    <term lemma="colour" partOfSpeech="noun" value="colour">
      <synSet name="Wordnet_refer-verb-2">
        <synonym isChosen="true" weight="1.0">color</synonym>
        <synonym isChosen="false" weight="1.0">colouring material</synonym>
      </synSet>
    </term>
  </synonymList>
</question>
```

## Watson returns the updated answer

Review the new answer. If necessary, change the synonyms again and resubmit the question.

## Sample code to call the Question and Answer service

The IBM Watson Platform SDK includes several samples that are written in Java that call the Question and Answer REST service. Both the code and the output of the samples might give you ideas for how to code your own application.

**Important:** With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. If you implemented the Question and Answer REST service using the asynchronous mode, update your code to use the synchronous mode instead.

The samples cover both synchronous and asynchronous mode and JSON and XML formats. The samples are stored in the `com.ibm.watson.sdk.qaapi.samples` project under the `com.ibm.watson.sdk.qaapi.web.rest.questionanswer` package:

### QAJsonSample

Asynchronous implementation that returns JSON data.

**QASyncJsonSample**

Synchronous implementation that returns JSON data.

**QASyncXmlSample**

Synchronous implementation that returns XML data.

**QAXmlSample**

Asynchronous implementation that returns XML data.

Run a sample in Eclipse by right-clicking the package and selecting **Run As > Java Application**.

**Related reference:**

“Operations on Question” on page 25

You can use POST and GET requests on the question resource.

---

## Troubleshooting for the Question and Answer service

For problems that are associated with the Question and Answer service, you can use troubleshooting information to help isolate and resolve those problems.

### Pipeline messages and HTTP codes

A request to the Question and Answer REST service might result in a warning or error before Watson processes the request. Use the error response information that is returned by the pipeline to diagnose and resolve the issue.

#### HTTP response status codes

If there is an issue with the request, the response header includes an HTTP status code of 4xx or 5xx. For example, if you submit a string value when an integer is expected, the response includes 400 Bad Request. The error response body also includes the HTTP status code.

- 400 - Bad Request
- 401 - Unauthorized
- 404 - Not Found
- 405 - Method Not Allowed
- 500 - Internal Server Error

#### Error response body

The body of the response contains information about errors. The following syntax shows the structure of response elements common to all warnings and errors from the Question and Answer service.

```
<error>
  <errorCode>String</errorCode>
  <message>String</message>
  <severity>String</severity>
  <statusCode>Integer</statusCode>
</error>
```

#### Error elements

The following table explains the error code elements for the Question and Answer REST service.

Table 1. Elements of the error container in response payload. A list of the error code elements that might be returned from the pipeline.

Name	Description
error	The container for all errors in the response.
error/errorCode	A string that uniquely identifies a message. For more information about the codes, see Table 2.
error/message	The message text.
error/severity	The severity code (HIGH MED LOW).
error/statusCode	The HTTP response code, usually 400, that is also returned in the header. Including the code in the response simplifies where you must check to get complete information.

## Message ID

The following table lists the unique IDs that might be returned in the **errorCode** element by the Question and Answer REST service.

Table 2. Message IDs in the **errorCode** element. A list of message ID values in the **errorCode** element of the **error** element.

Error code	Explanation
AQWTL00139E	The question text length is longer than the allowed length of {0}. The value that is shown in the message AQWTL00139E is the value for the <b>qa.max.question.size</b> parameter in the configuration set.
AQWTL00140E	A word in the question is longer than the allowed length of {0} characters. The value that is shown in the message AQWTL00140E is the value for the <b>qa.max.question.word.size</b> parameter in the configuration set.
AQWTL00141E	The question contained all numeric characters.
AQWTL00142E	The question contained all space characters.
AQWTL00156E	Required field <b>questionText</b> element is missing.
AQWTL00157E	Field <b>items</b> is invalid.
AQWTL00158E	Field evidence items attribute is invalid.  The <b>/question/evidenceRequest/@items</b> attribute must be an integer in the range 1 - 10.
AQWTL00159E	Field evidence profile attribute is invalid.  The <b>/question/evidenceRequest/@profile</b> attribute must be yes or no.
AQWTL00160E	Value for items is not an integer between 1 and 10.  The <b>/question/items</b> element must be an integer in the range 1 - 10.
AQWTL00161E	The context could not be read from the URL.
AQWTL00163E	Field <b>passthru</b> exceeds the configured limit in size.  The maximum size of the data is defined in the configuration set in the <b>qa.question.passthru.maxsize</b> parameter.
AQWTL00180E	<b>X-SyncTimeout</b> header value is invalid.
AQWTL00181E	Poison Question filter could not find the view {0}.

*Table 2. Message IDs in the **errorCode** element (continued).* A list of message ID values in the **errorCode** element of the **error** element.

Error code	Explanation
AQWTL00184E	The question is blocked from submission to the pipeline. The question was not processed.
AQWTL00185E	The filter field name is null. The filter field name is a required field. Specify the filter name as the field name in the index.
AQWTL00186E	The filter field name {0} is not indexed. The filter field name {0} was not indexed during ingestion of the corpus. The filter field name {0} must be indexed during ingestion of the corpus.
AQWTL00187E	The filter type name "{0}" is not valid. The supported filter types are dateRangeFilter, queryFilter, or prefixFilter. Set the "{0}" filter type to one of the supported types.
AQWTL00188E	The filter type is null. The supported filter types are dateRangeFilter or queryFilter. The filter type is a required field. Set the filter type to dateRangeFilter, queryFilter, or prefixFilter.
AQWTL00189E	The values for both minDate and maxDate are null. One of those fields must have a value. At least one date field is required for the dateRangeFilter. The value of the second field can be null to allow open-ended date range filtering. Set either the minDate or maxDate fields to a date in the following format: YYYYMMDDHHMMSS.
AQWTL00190E	The date range is not valid. The {0} maxDate is earlier than the {1} minDate. The minDate filter term must be before the maxDate in the format. Specify the minDate before the maxDate in the following format: YYYYMMDDHHMMSS.
AQWTL00191E	The value for maxDate has the wrong format: {0}. The correct format is YYYYMMDDHHMMSS. The value for the date must be either a string in the format YYYYMMDDHHMMSS or a null or empty value. Specify the date in a YYYYMMDDHHMMSS format. To leave the filter open-ended, include one date with a null or empty value.

Table 2. Message IDs in the **errorCode** element (continued). A list of message ID values in the **errorCode** element of the **error** element.

Error code	Explanation
AQWTL00192E	<p>The value for minDate has the wrong format: {0}. The correct format is YYYYMMDDHHMMSS.</p> <p>The value for the date must be either a string in the format YYYYMMDDHHMMSS or a null or empty string.</p> <p>Specify the date in a YYYYMMDDHHMMSS format. To leave the filter open-ended, include one date with a null or empty value.</p>
AQWTL00193E	<p>The filter values are missing. Add values that match the index.</p> <p>The filter values list is a required field for the queryFilter or prefixFilter.</p> <p>Specify values for the filter.</p>
AQWTL00194E	<p>The filter values list is null.</p> <p>The filter values list is a required field for the queryFilter or prefixFilter.</p> <p>Specify values for the filter.</p>
AQWTL00195E	<p>The fieldSet holding the indexed fields for filter fieldNames was not initialized. The fields were likely not indexed.</p> <p>The fields were not indexed or are indexes that do not support filtering.</p> <p>The fields must be indexed during ingestion of the corpus.</p>
AQWTL00196E	<p>The dateRangeFilter includes the wrong date format.</p> <p>The format for dateRangeFilter filterType is &lt;start date&gt;:&lt;end date&gt;. The value for the start or end date must be either a string in the format YYYYMMDDHHMMSS or a null or empty string.</p> <p>Specify dates in a YYYYMMDDHHMMSS format. Specify the start date and then the end date, and separate the dates with a colon (:). To leave the filter open-ended, include one date with a null or empty value.</p>
AQWTL00197E	<p>The processing pipeline is not configured for metadata persistence.</p> <p>The metadataPersistence configuration setting must be set to true to support filters.</p> <p>Set the metadataPersistence configuration to true in the pipeline_common.xml extension file.</p>

## Error code examples

The following example error message is triggered when the request does not include the required **questionText** element.

XML example

```

<error>
  <errorCode>AQWTL00156E</errorCode>
  <message>AQWTL00156E Required field questionText element is missing.</message>
  <severity>LOW</severity>
  <statusCode>400</statusCode>
</error>

```

JSON example

```

{
  "error" : {
    "errorCode" : "AQWTL00156E"
    "message" : "AQWTL00156E Required field questionText element is missing.",
    "severity" : "LOW",
    "statusCode" : 400
  }
}

```

## Messages returned with an answer response

When the Question and Answer service returns an answer and issues exist, information about the issue is returned with the response.

An error notification is a recoverable warning message that provides more information about an answer.

### Error notification syntax

```

<errorNotifications>
  <error id="String">
    <text>String</text>
  </error>
</errorNotifications>

```

### Example error notification with not enough passages

The following sample error is returned when the number of supporting passages is less than what was requested in the `/question/evidenceRequest@items` element.

```

- <errorNotifications>
- <error id="LateSPRAnnotator">
  <text>AQWAL02014W Unable to obtain the requested number of supporting passages for
  all answers.</text>
</error>
</errorNotifications>

```

### Example error notification when no answers are found

The following sample error is returned when Watson cannot find an answer.

The error is also returned when you assert an answer in the `/question/answerAssertion` element and no supporting passages are returned.

```

<errorNotifications>
- <error id="FinalMerger">
  <text>AQWAL02015W No answers returned. Try a more specific question.</text>
</error>
</errorNotifications>

```

---

## Operations on Question

You can use POST and GET requests on the question resource.

## POST /deepqa/v1/question

Post a question.

### Purpose

Post a question for Watson so that you can receive an answer.

**Important:** With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. If you implemented the Question and Answer REST service using the asynchronous mode, update your code to use the synchronous mode described here instead.

### Request syntax

The input and output message document formats are described with an informal syntax. Because of the formal nature of XPath notation, the principle description is in XML format. The values indicate XML schema data types. For JSON, the XML element names correspond to JSON object names.

The special characters that follow the elements denote usage. Each element without one of these appended characters is a required element. For JSON, elements that are shown with an asterisk (\*) or plus sign (+) correspond to JSON arrays, and the names are not used. The following special characters are used in the syntax:

- ?      0 or 1 occurrence is allowed. The element can be included with a null or empty text.
- \*      0 or more occurrences are allowed. The element can be included with a null or empty text.
- +      1 or more occurrences are allowed.

```
<question xmlns="Host:Port/Context root/deepqa/v1/question"
  xmlns:qa="Host:Port/Context root/deepqa/v1/question">
  <category>String</category>?
  <context>String</context>?
  <lat>String</lat>?
  <questionText>String</questionText>
  <items>Integer</items>?
  <evidenceRequest items="Integer" profile="String"/>?
  <answerAssertion>String</answerAssertion>?
  <formattedAnswer>true|false</formattedAnswer>?
  <passthru>String</passthru>?
</question>
```

### Request header

Table 3. Request header elements

Name	Value	Required or optional
Accept	Specifies the format of the response. The values of Content-Type and Accept must be the same. application/json   application/xml	Optional
Content-Type	Specifies the format of the input message. The values of Content-Type and Accept must be the same. application/json   application/xml	Required



Table 3. Request header elements (continued)

Name	Value	Required or optional
X-SyncTimeout	<p>A question is submitted by using a POST operation and the connection remains open until an answer is returned. The question response either contains the answer or it indicates that the question failed or timed out.</p> <p><i>Time interval in seconds</i></p> <p>The time interval represents the number of seconds that the service waits after the question is submitted for the answer to be returned. A response status is returned. For more information about the response status, see Table 6 on page 32.</p> <p>Example:</p> <pre>Resource resource = restClient.resource     ("http://localhost:8080/deepqa     /v1/question"); resource.header("X-SyncTimeout", "1");</pre> <p><b>Important:</b> The timeout value that you set refers to how long the server waits for an answer from the pipeline. The value does not indicate how long the browser waits for a response from the POST operation. The default value is 30 seconds. Setting this value higher does not make a difference if the browser timeout is set at 30 seconds.</p> <p><b>-1</b> No timeout is set. Wait until the answer is returned or the question fails.</p> <p>Example:</p> <pre>Resource resource = restClient.resource     ("http://localhost:8080/deepqa     /v1/question"); resource.header("X-SyncTimeout", "-1");</pre>	Required

## Parameters

The Question and Answer service supports these parameters in the resource URL.

Table 4. Resource parameters

Name	Value	Required or optional
alt	<p>Specifies whether XML or JSON is used for the request and accepted for the response. The <b>alt</b> parameter takes precedence when both the <b>alt</b> parameter and the HTTP Accept header are specified. <code>application/json application/xml</code></p> <p><b>Tip:</b> Use the <b>alt</b> parameter during only development and testing.</p>	Optional

## Request elements

Table 5. Elements of the request payload

Element Name	Description	Required or optional
<b>/question</b>	The container for the submitted question and its associated information.	Required
<b>/question/answerAssertion</b>	<p>Specify an answer to receive the supporting evidence passages for that answer. Without this element, Watson searches for answers from the <b>questionText</b>. When you assert an answer, Watson uses that answer instead to search for supporting evidence passages.</p> <p>If you configured your pipeline to support a ranked list of evidence, the supporting evidence appears in the <b>question/evidenceList</b> section. If you include the <b>evidenceRequest</b> element with the question, the supporting evidence appears in the <b>answers/answer/evidence</b> section.</p> <p>If no supporting passages are returned for the asserted answer, the API returns a message that no answers were found.</p> <p><b>Restriction:</b> Your processing pipeline must be configured to support AnswerAssertionPrimarySearch. Otherwise, the element is ignored.</p>	Optional
<b>/question/category</b>	The category of the question in terms of a constraint on the possible answers.	Optional
<b>/question/context</b>	<p>A natural language string that is composed of words that provide extra information for Watson to consider when it determines answers. The maximum length of this element is 1024 characters.</p> <p>The <b>context</b> element can have one of the following values:</p> <p><b>plain text</b></p> <p>The context is provided as text in the <b>context</b> element:</p> <pre>&lt;context&gt;Fauna&lt;/context&gt;</pre> <p><b>url</b></p> <p>A valid URL that points to the context string. The service accepts text and HTML in the response.</p> <pre>&lt;context&gt;http://www.example.com/fauna.txt&lt;/context&gt;</pre> <p>The first 1024 characters of the contents in the file that is identified by the URL are used as the context value. If the file is not found (HTTP 404), the service returns an error and the question is not submitted to Watson.</p>	Optional

Table 5. Elements of the request payload (continued)

Element Name	Description	Required or optional
<code>/question/evidenceRequest</code>	<p>Specifies that you want Watson to return supporting evidence for each answer in the <b>answers/answer/evidence</b> section of the answer response.</p> <p><b>Important:</b> Do not confuse this element with the <code>/question/evidencelist</code> section of the response. The evidence that is returned from an evidence request was not used when Watson searched for the answer. This evidence is returned after Watson generates, scores, and ranks the answers. However, evidence from an evidence request might be useful when you want background passages for an answer.</p> <p>The element has two attributes:</p> <p><b>@items</b> An integer in the range 1 - 10 that defines the number of supporting passages to return for each answer. Fewer passages might be returned for answers that do not have good passage support.</p> <p>The default value is 3 passages.</p> <p><b>@profile</b> Whether to return evidence profiles for each possible answer.</p> <p>The evidence profile indicates where the evidence came from.</p> <p>Attribute values: yes   no (default).</p> <p>A yes value indicates that information is returned about where the evidence was found. With the no value, the information is not returned.</p>	Optional
<code>/question/filters</code>	<p>The container for a set of filters. Filters allow the use of metadata to restrict answers to come from specific documents.</p> <p><b>Restriction:</b> This element is accepted only when the IBM Watson processing pipeline is configured to support metadata. In a default Watson system, this element returns message code AQWTL00197E.</p> <p>When you include more than one filter in this container, they are combined with a logical AND to restrict the search. For example, you want to filter for answers from Wikipedia documents created in 2013. In contrast, multiple entries in the <code>/question/filters/filter/values/value</code> container are combined with a logical OR.</p>	Optional

Table 5. Elements of the request payload (continued)

Element Name	Description	Required or optional
<b>/question/filters/filter</b>	<p>The container for a filter. The element has two attributes:</p> <p><b>@filterType</b> The type of filter. These attribute values are supported:</p> <p><b>dateRangeFilter</b> Filter the answer by date.</p> <p><b>metadataFilter</b> Filter the answer by an indexed metadata field name.</p> <p><b>prefixFilter</b> Filter the answer by terms that start with the same prefix.</p> <p><b>queryFilter</b> Filter the answer by an indexed metadata field name.</p> <p><b>@fieldName</b> The name of the indexed metadata field.</p> <p>When the <b>@filterType</b> is <b>dateRangeFilter</b>, set the value of <b>@fieldName</b> to <i>indexedKey.metadata</i>.</p> <p>When the <b>@filterType</b> is <b>prefixFilter</b> or <b>queryFilter</b>, set the value of <b>@fieldName</b> to <i>indexedmetadata field name</i>.</p>	Optional
<b>/question/filters/filter/values</b>	<p>The container for a set of values.</p> <p>Starting with version 2.4, use this container for any <b>filterType</b>.</p> <p>With version 2.3, use this container only when the <b>filterType</b> attribute is <b>queryFilter</b>.</p> <p>When you include more than one value in this container, they are combined with a logical OR to expand the search. For example, you want to filter for answers from multiple corpora. In contrast, multiple filters in the <b>/question/filters</b> container are combined with a logical AND.</p>	Optional

Table 5. Elements of the request payload (continued)

Element Name	Description	Required or optional
<code>/question/filters/filter/values/value</code>	<p>The metadata value of the indexed field name <code>/question/filters/filter/@fieldName</code>. The value in this element must match the value in the index.</p> <p>With version 2.4, the format for <code>@dateRangeFilter</code> filterType is <i>start date:end date</i>. The date format is a string in the format <code>YYYYMMDDHHMMSS</code>. Specify an open-ended range by substituting the string <code>null</code> for the value. However either the <i>start date</i> or the <i>end date</i> must have a date. For example <code>20140101000000:null</code> returns all dates from Jan 1, 2014 to the present date. By specifying the value <code>null:20131231235959</code>, you filter by all dates before December 1, 2013 at 23:59:59.</p> <p>With version 2.7, the value of a <code>@queryFilter</code> filterType can be multiple terms separated by one or more spaces. The value must match the full term of the filter, but is not case-sensitive. For example, the following JSON example matches index values of <code>Ibm Publications</code> and <code>ibm publications</code>, but not <code>IBM Publication</code>.</p> <pre>"filters":[   {     "filterType":"queryFilter",     "fieldName":"indexedPublisher",     "values":["IBM Publications"]   } ]</pre>	Optional
<code>/question/filters/filter/minDate</code>	<p>With version 2.3, the start date of the <code>/question/filters/filter/@dateRangeFilter</code>.</p> <p>The date format is a string in the format <code>YYYYMMDDHHMMSS</code>. For example, <code>20131021134957</code> represents October 21, 2013 at 1:49:57 p.m. Specify an open-ended range by substituting the string <code>null</code> for the value. However either <code>minDate</code> or <code>maxDate</code> must have a date.</p> <p>In version 2.4, <code>minDate</code> and <code>maxDate</code> are ignored when you specify a date filter with the <code>/question/filters/filter/values</code> element.</p>	Optional
<code>/question/filters/filter/maxDate</code>	<p>With version 2.3, the end date of the <code>/question/filters/filter/@dateRangeFilter</code>.</p> <p>The date format is a string in the format <code>YYYYMMDDHHMMSS</code>. Specify an open-ended range by substituting the string <code>null</code> for the value. However either <code>minDate</code> or <code>maxDate</code> must have a date.</p> <p>In version 2.4, <code>minDate</code> and <code>maxDate</code> are ignored when you specify a date filter with the <code>/question/filters/filter/values</code> element.</p>	Optional
<code>/question/formattedAnswer</code>	Include this element and set it to <code>true</code> to receive both formatted answer text and unformatted text in an answer response. The answer is returned in HTML format.	Optional

Table 5. Elements of the request payload (continued)

Element Name	Description	Required or optional
/question/items	An integer in the range 1 - 10 that represents the number of possible answers to be returned. If you do not specify the number of items, the request assumes five answers.	Optional
/question/lat	The lexical answer type (LAT) of the question. The LAT is a word or noun phrase that appears in the question, or is implied by it. In Watson, the LAT specifies the type of the answer that is appropriate. In most cases, Watson extracts the LAT and you do not need to pass it with the question.	Optional
/question/passthru	Specifies a string that you include with the question. The <b>passthru</b> data is not submitted with the pipeline but does pass through to the answer.  You can specify the encoding of a binary large object if it can be represented by using a string.  The maximum size of the data is defined in the configuration set as a value in <code>qa.question.passthru.maxsize</code> . If the size of the data exceeds the value in <code>qa.question.passthru.maxsize</code> , the question is not submitted to the pipeline and produces the following error:  <code>AQWTL00163E_FIELD_PASSTHRU_EXCEEDS_SIZE = "Field passthru exceeds configured limit in size"</code>  To modify or add elements to the submitted CAS, use the question extension.	Optional
/question/questionText	The text of the question to be answered.	Required
/question/synonymList	The container for the set of synonyms. You can provide feedback to Watson by resubmitting a question and including an updated list of synonyms. For more information, see "Synonym feedback example" on page 19	Optional

## Response outcomes

The response contains an HTTP code and a response status. The response status indicates the current progress of the processing for a question. When the Complete status is returned, an answer set is included in the response document. The following table shows the successful HTTP response code for synchronous mode.

HTTP Response code	Description
200	Created

Other response status values are listed in the following table.

Table 6. Response status in synchronous mode

Response status	Description
Complete	The answer is returned.

Table 6. Response status in synchronous mode (continued)

Response status	Description
Timeout	The answer did not return within the time interval that is specified in the <X-SyncTimeout> header element. The question is canceled.
Failed	An error is returned while Watson processed the question.

## Response syntax

For more information about the syntax notations, see “Request syntax” on page 26.

```
<question xmlns="Watson REST server URI">
  <category>String</category>
  <lat>String</lat>
  <questionText>String</questionText>
  <evidenceRequest items="Integer" profile="String"/>
  <status>String</Status>
</question>
```

## Response elements

In addition to the elements shown in the following table, all input elements are returned as they were submitted on the POST.

Table 7. Elements of the response payload

Element Name	Description
<b>/question</b>	The container for the answer response and its associated information. The element has one attribute:  <b>@id</b> An integer that is assigned by the service to identify this question and its answers.
<b>/question/answers</b>	The container for the set of ranked answers. Answers are included only if the value of <b>/question/status</b> is Complete.
<b>/question/answers/answer</b>	The container for information about an individual answer. The number of answers that are returned is dependent on the number of <b>/question/items</b> requested when the question was submitted. The element has one attribute:  <b>@id</b> An integer that uniquely identifies an answer in the context of the question.
<b>/question/answers/answer/text</b>	A string that contains an answer to the question in the form of text.
<b>/question/answers/answer/confidence</b>	A decimal percentage that represents the confidence that Watson has in this answer. Higher values represent higher confidences.
<b>/question/answers/answer/evidence</b>	The container for supporting passages that are returned from an evidence request. The number of passages is never more than the value of <b>/question/evidenceRequest/@items</b> in the request. However, no evidence passages might be returned, as for example when an answer came from the title of a document. <b>Important:</b> Do not confuse this value with the <b>/question/evidencelist</b> .
<b>/question/answers/answer/evidence/passage</b>	The container for a supporting passage for an answer and its metadata. The element has one attribute:  <b>@id</b> An integer that uniquely identifies a passage within the set of supporting passages for an answer.
<b>/question/answers/answer/evidence/passage/copyright</b>	The copyright holder for the document that contains the evidence passage. If there is no copyright information, the element is empty.

Table 7. Elements of the response payload (continued)

Element Name	Description
<code>/question/answers/answer/evidence/passage/document</code>	<p>The URL to a document passage for an answer.</p> <p>In release 2 and later, the document element contains a URL to the document service. The content of that URL is an HTML representation of the document that contains the supporting passage. The URL includes the document id and the begin and end marks for the span of text that marks the passage in the document. For example:</p> <pre>&lt;document&gt;http://myhost.example.com:8080/   deepqa/v1/question/document/1400795/23   /55 &lt;/document&gt;</pre> <p>In releases before 2.0, the document element contains a URL that points to the document that contains the supporting passage. The document can be retrieved by using an HTTP GET to this URL. The document is returned in compressed or uncompressed format, depending on the HTTP Accept-Encoding header that is sent with the HTTP GET. If HTTP compression requested, the response is a compressed document.</p>
<code>/question/answers/answer/evidence/passage/termsOfUse</code>	A URL that points to the license that describes the terms of use of the document.
<code>/question/answers/answer/evidence/passage/text</code>	The passage text, essentially one or more sentences.
<code>/question/answers/answer/evidence/passage/title</code>	The passage title, a string of text.
<code>/question/answers/answer/evidenceProfile</code>	The container for the evidence profile for this answer. The evidence profile indicates where the evidence came from.
<code>/question/answers/answer/evidenceProfile/featureGroup</code>	The container for feature groups. A feature group is a name-value pair that represents a combined summary of a set of evidence features.
<code>/question/answers/answer/evidenceProfile/featureGroup/name</code>	A string that represents the feature group.
<code>/question/answers/answer/evidenceProfile/featureGroup/value</code>	The summary value for the feature group. <b>value</b> is a decimal value in the range 0 - 1. The values of all groups sum to one so that they can be compared as relative percentages.
<code>/question/answers/answer/formattedText</code>	The HTML-formatted version of the answer text that is returned when you set <b>formattedAnswer</b> to true when you submit a question. The formatted answer includes content from the <code>&lt;body&gt;</code> tag of the answer.
<code>/question/category</code>	The category of the question that was submitted with the question. When no category was submitted with the question, an empty <b>category</b> element is returned in the response.
<code>/question/errorNotifications</code>	The container for the set of recoverable errors that occurred during question processing. It is only included when the value of <b>/question/status</b> is <b>Complete</b> .
<code>/question/errorNotifications/error</code>	<p>The container for message information about a recoverable error. The notification can contain more than one message. The element contains one attribute:</p> <p><b>@id</b> A string that uniquely identifies the IBM Watson component that generated the error.</p>
<code>/question/errorNotifications/text</code>	A string that describes the error.



Table 7. Elements of the response payload (continued)

Element Name	Description
<code>/question/evidencelist</code>	The container for a ranked list of passages that Watson used to generate the answer. Each piece of evidence contains passage text and other relevant information. Do not confuse this value with the <code>/question/answers/answer/evidence</code> <b>Restriction:</b> This element is returned only when the IBM Watson processing pipeline is configured to support it. In a default Watson system, this element is not returned.
<code>/question/evidencelist/evidence</code>	The container for information about the evidence passage. The element contains one attribute:  <b>@id</b> A string that uniquely identifies a passage within the list of evidence passages.
<code>/question/evidencelist/evidence/copyright</code>	The copyright holder for the document that contains the evidence passage. If there is no copyright information, the element is empty.
<code>/question/evidencelist/evidence/metadataMap</code>	The container for the metadata from the document that contains the evidence passage. This element is returned only when the IBM Watson processing pipeline is configured to support metadata.
<code>/question/evidencelist/evidence/termsOfUse</code>	A URL that points to the license that describes the terms of use of the document that contains the evidence passage. If there is no information about terms of use, the element is empty.
<code>/question/evidencelist/evidence/text</code>	The evidence passage text.
<code>/question/evidencelist/evidence/title</code>	The title of the document for the evidence passage.
<code>/question/evidencelist/evidence/document</code>	The URL to a document passage for an answer. For more information, see the information for the <code>/question/answers/answer/evidence/passage/document</code> element.
<code>/question/evidencelist/evidence/value</code>	A decimal percentage that represents the confidence that Watson has in this evidence. Higher values represent higher confidences.
<code>/question/focuslist</code>	The container for a list of focus elements that are determined by the pipeline for the final answer.
<code>/question/latlist</code>	The container for lexical answer types (LATs) that the pipeline determined for the final answer. The <code>/question/lat</code> is submitted in the POST when the question was submitted. The <code>/question/latlist</code> contains the LATs that were determined by the pipeline when it processed the answer.
<code>/question/pipelineid</code>	The internal ID that is assigned for the final answer CAS. This element contains the internal CAS ID that is assigned after the question is answered. You can use this ID to identify the question with the internal data structures that Watson uses. <b>Important:</b> Do not confuse this value with the <code>/question/id</code> , which is the external-facing ID used to identify a question.
<code>/question/qclasslist</code>	The container for a list of question classes that are determined by the pipeline for the final answer.
<code>/question/status</code>	The response status of the request. For more information about the values of the response status, see Table 6 on page 32
<code>/question/supplemental</code>	Contains more information about the answers for a customization of the IBM Watson processing pipeline. In a Watson system that is not customized, this element is not returned. For more information about this element, see the documentation for your IBM Watson solution.

Table 7. Elements of the response payload (continued)

Element Name	Description
<code>/question/synonymList</code>	<p>The container for the set of synonyms. The list contains synonyms for words in the question. When no synonyms are found, an empty <b>synonymList</b> element is returned in the response. You can provide feedback to Watson through an updated list of synonyms.</p> <p>In the default settings, Watson returns the following types of synonyms:</p> <p><b>Numbers</b> For example, 3,000 for 3000.</p> <p><b>Abbreviations</b> For example, UK for United Kingdom and USA for United States of America.</p> <p><b>Dates</b> For example, 4th of July for July 4.</p> <p><b>Periods in names</b> For example, john q adams is returned for John Q. Adams.</p> <p>To return a more extensive list of synonyms, the aggressive query expansion option must be enabled in the Watson backend.</p>
<code>/question/synonymList/term</code>	<p>The original term value with its part of speech, such as noun, adj, adv, and verb.</p> <p>The value can be any of the part of speech elements that are defined in slot grammar lexicons. The element has the following attributes:</p> <p><b>@lemma</b> The lemma form of term. This attribute is new in 2.1.2.</p> <p><b>@partOfSpeech</b> The part of speech for the term, such as noun or verb.</p> <p><b>@value</b> The value, or actual content, of the term.</p>
<code>/question/synonymList/term/synSet</code>	<p>The set that represents the source (such as wordnet or wikiredirect) of the synonyms. The element has one attribute:</p> <p><b>@name</b> The name of the synSet.</p>
<code>/question/synonymList/term/synSet/synonym</code>	<p>A synonym that belongs to a synSet. The synonym is represented by its value and whether it is used. All synonyms are returned in lowercase text. The element has two attributes:</p> <p><b>@isChosen</b> Whether the synonym is valid. Attribute values: true   false. A true value indicates that the synonym was chosen by the pipeline as valid. With the false value, the information is not returned.</p> <p><b>@weight</b> The weight of the synonym. This attribute is new in 2.1.2.</p>

## Example request with optional fields

This example passes in the text of the question in the **questionText** section. The **lat** element identifies the lexical answer type as **Animal** and the question category as **Mammals**. This request is for no more than three answers, with both formatted and unformatted answer text. The **items** attribute of the **evidenceRequest** element

specifies returning up to two supporting passages pieces of evidence. The **profile** attribute is set to Yes, so the response includes an evidence profile.

#### Example XML request

```
<?xml version="1.0" encoding="UTF-8"?>
<question xmlns="http://www.ibm.com/xmlns/deepqa/question">
  <questionText>What is the largest species of bear found in the world</questionText>
  <lat>Animal</lat>
  <category>Mammals</category>
  <context>Fauna</context>
  <formattedAnswer>true</formattedAnswer>
  <items>3</items>
  <evidenceRequest items="2" profile="Yes"/>
  <passthru></passthru>
</question>
```

This example shows the request in JSON format:

```
{
  "question": {
    "category": "Mammals",
    "context": "Fauna",
    "evidenceRequest": {
      "items": "2",
      "profile": "Yes"
    },
    "formattedAnswer": "true",
    "items": "3",
    "lat": "Animal",
    "questionText": "What is the largest species of bear found in the world"
    "passthru":
  }
}
```

#### Example response

##### XML response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<question id="1421218" xmlns="http://www.ibm.com/xmlns/deepqa/question" xmlns:qa="http://www.ibm.com/xmlns/deepqa/question">
  <category>Mammals</category>
  <evidenceRequest profile="Yes" items="2"/>
  <lat>Animal</lat>
  <questionText>What is the largest species of bear found in the world</questionText>
  <status>Complete</status>
</question>
```

##### JSON response

```
{
  "question": {
    "category": "Mammals",
    "evidenceRequest": {
      "items": 2,
      "profile": "Yes"
    },
    "id": 1421217,
    "lat": "Animal",
    "questionText": "What is the largest species of bear found in the world",
    "status": "Complete"
  }
}
```

#### Related reference:

“Pipeline messages and HTTP codes” on page 21

A request to the Question and Answer REST service might result in a warning or error before Watson processes the request. Use the error response information that is returned by the pipeline to diagnose and resolve the issue.

## POST /deepqa/v1/question: Asynchronous (deprecated)

Post a question in asynchronous mode.

### Purpose

Post a question for Watson so that you can use a question ID to retrieve an answer.

**Important:** With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. If you implemented the Question and Answer REST service using the asynchronous mode, update your code to use the synchronous mode instead.

From release 2.0 until release 2.8, the Question and Answer REST service supported two modes, asynchronous and synchronous. In asynchronous mode, you submit a question and Watson returns a unique identifier URL in the response. By using the ID, you poll the server with a GET request to determine when the question processing completes. When processing is complete, the successful response contains the answers and other information.

### Request syntax

The input and output message document formats are described with an informal syntax. Because of the formal nature of XPath notation, the principle description is in XML format. The values indicate XML schema data types. For JSON, the XML element names correspond to JSON object names.

The special characters that follow the elements denote usage. Each element without one of these appended characters is a required element. For JSON, elements that are shown with an asterisk (\*) or plus sign (+) correspond to JSON arrays, and the names are not used. The following special characters are used in the syntax:

- ?        0 or 1 occurrence is allowed. The element can be included with a null or empty text.
- \*        0 or more occurrences are allowed. The element can be included with a null or empty text.
- +        1 or more occurrences are allowed.

```
<question xmlns="Host:Port/Context root/deepqa/v1/question"
  xmlns:qa="Host:Port/Context root/deepqa/v1/question">
  <category>String</category>?
  <context>String</context>?
  <lat>String</lat>?
  <questionText>String</questionText>
  <items>Integer</items>?
  <evidenceRequest items="Integer" profile="String"/>?
  <answerAssertion>String</answerAssertion>?
  <formattedAnswer>true|false</formattedAnswer>?
  <passthru>String</passthru>?
</question>
```

### Request header

Table 8. Request header elements

Name	Value	Required or optional
Accept	Specifies the format of the response. The values of Content-Type and Accept must be the same <code>application/json</code>   <code>application/xml</code>	Optional

Table 8. Request header elements (continued)

Name	Value	Required or optional
Content-Type	Specifies the format of the input message. The values of Content-Type and Accept must be the same. <code>application/json application/xml</code>	Required

## Parameters

The parameters in asynchronous mode match the synchronous mode. For more information about the parameters, see `POST /deepqa/v1/question`.

## Request elements

The request elements in asynchronous mode match the synchronous mode. For more information about the request elements, see `POST /deepqa/v1/question`.

## Asynchronous response outcomes

The response contains an HTTP code, an X-Deprecated header, a response status, and links to read the status and answer and to provide feedback about the answer. The value of the X-Deprecated header is the date of deprecation. The date follows the format `dd mmm yyyy`. The `dd` variable represents the two-digit day, `mmm` represents the three-character month name (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec), and `yyyy` represents the four-digit year.

When the question is submitted and there are no errors, HTTP response code 201 and a response status of Accepted are returned to indicate that Watson has the question. On subsequent poll requests, the status is Queued while the question is processed. When the final answer is returned, the status is Complete. The following table shows the successful HTTP response code for asynchronous mode.

HTTP Response code	Description
201	Created

Other response status values are listed in the following table.

Table 9. Response status in asynchronous mode

Asynchronous mode response status	Description
Accepted	The question is accepted for processing. This status is returned only as a response to the POST of a new question.
Queued	The question is queued until an answer is returned, at which point the status changes to Complete.
Complete	The answer is returned.
Failed	An error is returned while Watson processed the question.

## Response syntax

For more information about the syntax notations, see “Request syntax” on page 38.

```

<question xmlns="Watson REST server URI">
  <link href="http://myhost.example.com:8080/deepqa/v1/question{QID}" rel="self" />
  <link href="http://myhost.example.com:8080/deepqa/v1/question{QID}/feedback" rel="feedback" />
  <category>String</category>
  <lat>String</lat>
  <questionText>String</questionText>
  <evidenceRequest items="Integer" profile="String"/>
  <status>String</Status>
</question>

```

## Response elements

In addition to the elements shown in the following table, all input elements are returned as they were submitted on the POST.

Table 10. Elements of the response payload

Element name	Description
<b>/question/link</b>	<p>Provides a URL to use with a GET operation. The element has two attributes:</p> <p><b>@href</b> The URL.</p> <p><b>@rel</b> The type of link. The attribute has two possible values:</p> <p><b>self</b> Use this value to get the initial and updated response status.</p> <p><b>feedback</b> Use this value to submit feedback on the question.</p>
<b>/question/status</b>	The response status of the request. For more information about the syntax notations, see “Request syntax” on page 38.

## Example request with optional fields

This example passes in the text of the question in the **questionText** section. The **lat** element identifies the lexical answer type as **Animal** and the question category as **Mammals**. This request is for no more than three answers, with both formatted and unformatted answer text. The **items** attribute of the **evidenceRequest** element specifies returning up to two supporting passages pieces of evidence. The **profile** attribute is set to **Yes**, so the response includes an evidence profile.

### Example XML request

```

<?xml version="1.0" encoding="UTF-8"?>
<question xmlns="http://www.ibm.com/xmlns/deepqa/question">
  <questionText>What is the largest species of bear found in the world</questionText>
  <lat>Animal</lat>
  <category>Mammals</category>
  <context>Fauna</context>
  <formattedAnswer>true</formattedAnswer>
  <items>3</items>
  <evidenceRequest items="2" profile="Yes"/>
  <passthru></passthru>
</question>

```

This example shows the request in JSON format:

```
{
  "question": {
    "category": "Mammals",
    "context": "Fauna",
    "evidenceRequest": {
      "items": "2",
      "profile": "Yes"
    },
    "formattedAnswer": "true",
    "items": "3",
    "lat": "Animal",
    "questionText": "What is the largest species of bear found in the world"
    "passthru":
  }
}
```

## Example asynchronous response

### Asynchronous XML response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<question id="1421218" xmlns="http://www.ibm.com/xmlns/deepqa/question" xmlns:qa="http://www.ibm.com/xmlns/deepqa/question">
  <category>Mammals</category>
  <evidenceRequest profile="Yes" items="2"/>
  <lat>Animal</lat>
  <questionText>What is the largest species of bear found in the world</questionText>
  <status>Accepted</status>
  <link href="http://myhost.example.com:8080/deepqa/v1/question/1421218/feedback" rel="feedback"/>
  <link href="http://myhost.example.com:8080/deepqa/v1/question/1421218" rel="self"/>
</question>
```

### Asynchronous JSON response

```
{
  "question": {
    "category": "Mammals",
    "evidenceRequest": {
      "items": 2,
      "profile": "Yes"
    },
    "id": 1421217,
    "lat": "Animal",
    "links": {
      "feedback": "http://myhost.example.com:8080/deepqa/v1/question/1421217/feedback",
      "self": "http://myhost.example.com:8080/deepqa/v1/question/1421217"
    },
    "questionText": "What is the largest species of bear found in the world",
    "status": "Accepted"
  }
}
```

## GET /deeppa/v1/question/{QID} (deprecated)

Poll the server to return status information about the processing of a question or to retrieve the answer.

### Purpose

**Important:** This operation requires asynchronous mode, which is deprecated.. With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. If you implemented the Question and Answer REST service using the asynchronous mode, update your code to use the synchronous mode instead.

In asynchronous mode, Watson returns a unique identifier URL in the response to submitting a question. You poll the server with a GET request that includes the identifier to determine when Watson finishes processing the question. When processing is done, the successful response contains the answers and other information.

## Request header

Table 11. Request header elements

Name	Description	Required or optional
Accept	Specify the format of the response. <code>application/json</code> or <code>application/xml</code>	Optional  JSON is the default. You must specify <code>application/xml</code> to receive an XML response.

## Parameters

The Question and Answer service supports these parameters in the resource URL.

Table 12. Request parameters

Name	Description	Required or optional
alt	Specifies whether XML or JSON is used for the request and accepted for the response. The <b>alt</b> parameter takes precedence when both the <b>alt</b> parameter and the HTTP Accept header are specified. <code>application/json</code> or <code>application/xml</code> <b>Tip:</b> Use the <b>alt</b> parameter during only development and testing.	Optional
{QID}	The question id	Required

## Successful asynchronous response outcomes

If the operation is successful, the HTTP status code is 200 (OK).

**Important:** With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. All asynchronous responses include the X-Deprecated header. The value of the header is the date of deprecation. The dates follow the format *dd mmm yyyy*. The *dd* variable represents the two-digit day, *mmm* represents the three-character month name (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec), and *yyyy* represents the four-digit year.

Response code	Description
200	OK

## Response syntax

The input and output message document formats are described with an informal syntax. Because of the formal nature of XPath notation, the principle description is in XML format. The values indicate XML schema data types. For JSON, the XML element names correspond to JSON object names.

The special characters that follow the elements denote usage. Each element without one of these appended characters is a required element. For JSON, elements that



are shown with an asterisk (\*) or plus sign (+) correspond to JSON arrays, and the names are not used. The following special characters are used in the syntax:

- ? 0 or 1 occurrence is allowed. The element can be included with a null or empty text.
- \* 0 or more occurrences are allowed. The element can be included with a null or empty text.
- + 1 or more occurrences are allowed.

In addition to the elements that are shown here, the response includes all elements that are submitted with the question.

```
<question xmlns="Watson REST server URI" id="Integer">
  <answers>
    <answer id="Integer">*
      <text>String</text>
      <confidence>Decimal</confidence>
      <evidence>
        <passage id=Integer>*
          <copyright>String</copyright>
          <metadataMap />
          <text>String</text>
          <termsOfUse>url</termsOfUse>
          <title>String</title>
          <document>url</document>
        </passage>
      </evidence>
      <evidenceProfile>
        <featureGroup name="String">*
          <name>String</name>
          <value>Integer</value>
        </featureGroup>
      </evidenceProfile>
      <supplemental />*
    </answer>
  </answers>
  <category>String</category>
  <errorNotifications>*
    <error id="String">*
      <text>String</text>
    </error>
  </errorNotifications>
  <evidencelist>*
    <evidence id="String">*
      <copyright>String</copyright>
      <metadataMap />
      <termsOfUse>url</termsOfUse>
      <text>String</text>
      <title>String</title>
      <document>url</document>
      <value>Decimal</value>
    </evidence>
  </evidencelist>
  <evidenceRequest items="Integer" profile="String" />?
  <focuslist>
    <focus value="String" />*
  </focuslist>
  <latlist>
    <lat value="String" />*
  </latlist>
  <pipelineid>Integer</pipelineid>
  <qclasslist>
    <qclass value="String" />*
  </qclasslist>
  <status>String</status>
  <synonymlist>
    <term lemma="String" partOfSpeech="String" value="String">*
      <synSet name="String">
        <synonym isChosen="Boolean" weight="Decimal">String</synonym>*
      </synSet>
    </term>
  </synonymlist>
  <link href="Host:Port/Context root/deepqa/v1/question{QID}" rel="self" />
  <link href="Host:Port/Context root/deepqa/v1/question{QID}/feedback" rel="feedback" />
</question>
```

## Response elements

The response elements in asynchronous mode match the synchronous mode. For more information about the request elements, see `POST /deepqa/v1/question`.

## Example response

### XML response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<question xmlns:ns2="http://www.ibm.com/xmlns/deepqa/question" id="545706C1A3084322946C0B885A4289F0">
  <answers>
    <answer id="0">
      <text>Roanoke Colony</text>
      <confidence>0.28744</confidence>
      <evidenceProfile>
        <featureGroup>
          <name>FeatureGroup_Linguistic</name>
          <value>0.9999948206932325</value>
        </featureGroup>
        <featureGroup>
          <name>FeatureGroup_Type</name>
          <value>2.0069991645870886E-6</value>
        </featureGroup>
        <featureGroup>
          <name>FeatureGroup_Spatio-temporal</name>
          <value>1.3500703232240038E-6</value>
        </featureGroup>
        <featureGroup>
          <name>FeatureGroup_Relevance</name>
          <value>1.8222372797419442E-6</value>
        </featureGroup>
      </evidenceProfile>
      <evidence>
        <passage id="9316">
          <copyright>http://en.example.org/awiki/Copyrights</copyright>
          <text>In 1585, the first English colony in the Americas was founded in Virginia by Walter Raleigh.
The colony, named Roanoke, failed and became known as the lost colony.</text>
          <termsOfUse>http://en.example.org/awiki/License</termsOfUse>
          <title>England</title>
          <document>http://myhost.example.com:8080/deepqa/v1/question/document/9316/11/52</document>
        </passage>
        <passage id="22416349">
          <copyright>http://en.example.org/awiki/Copyrights</copyright>
          <text>The commission included a requirement to return with a lump of gold to prove
that he had reached the South Sea, or a member of Raleigh's colony.</text>
          <termsOfUse>http://en.example.org/awiki/License</termsOfUse>
          <title>History of a Settlement (1607-1699)</title>
          <document>http://myhost.ibm.com:8080/deepqa/v1/question/document/22416349/10/85</document>
        </passage>
      </evidence>
      <formattedText><p><b>The commission included a requirement to return with a lump of gold to prove
that he had reached the South Sea, or a member of Raleigh's colony.</b></p></formattedText>
    </answer>
    <answer id="1">
      <text>Massachusetts Bay Colony</text>
      <confidence>0.15163</confidence>
      <evidenceProfile>
        <featureGroup>
          <name>FeatureGroup_Linguistic</name>
          <value>0.9999961944410577</value>
        </featureGroup>
        <featureGroup>
          <name>FeatureGroup_Type</name>
          <value>1.3965711656993506E-6</value>
        </featureGroup>
        <featureGroup>
          <name>FeatureGroup_Spatio-temporal</name>
          <value>1.0809663024571708E-6</value>
        </featureGroup>
        <featureGroup>
          <name>FeatureGroup_Relevance</name>
          <value>1.3280214741508218E-6</value>
        </featureGroup>
      </evidenceProfile>
      <evidence>
```

```

<passage id="292285">
  <copyright>http://en.example.org/awiki/Copyrights</copyright>
  <text>The Rhode Island and Providence colony granted greater freedoms than other American
  colonies.</text>
  <termsOfUse>http://en.example.org/awiki/License</termsOfUse>
  <title>State religion</title>
  <document>myhost.example.com:8080/deepqa/v1/question/document/292285/10/85</document>
</passage>
<passage id="218110">
  <copyright>http://en.example.org/awiki/Copyrights</copyright>
  <text>Although Puritan and Pilgrim did not describe the same groups, both were protestants.
  </text>
  <termsOfUse>http://en.example.org/awiki/License</termsOfUse>
  <title>Massachusetts Bay Colony</title>
  <document>http://myhost.ibm.com:8080/deepqa/v1/question/document/218110/10/85</document>
</passage>
</evidence>
<formattedText><p>Although Puritan and Pilgrim did not describe the same groups, both were
protestants.</p></formattedText>
</answer>
</answers>
<category></category>
<errorNotifications/>
<evidenceRequest profile="yes" items="2"/>
<focuslist>
  <focus value="This lost colony"/>
</focuslist>
<items>2</items>
<latlist>
  <lat value="colony"/>
</latlist>
<passthru></passthru>
<pipelineid>251817795</pipelineid>
<qclasslist>
  <qclass value="FACTOID"/>
</qclasslist>
<questionText>This lost colony was never found?</questionText>
<status>Complete</status>
<synonymList>
  <term lemma="television" partOfSpeech="noun" value="television">
    <synSet name="Wordnet_television-noun-1">
      <synonym isChosen="true" weight="1.0">tv</synonym>
      <synonym isChosen="true" weight="1.0">telecasting</synonym>
      <synonym isChosen="true" weight="1.0">video</synonym>
    </synSet>
    <synSet name="Wordnet_television+receiver-noun-1">
      <synonym isChosen="false" weight="1.0">telly</synonym>
    </synSet>
  </term>
</synonymList>
<link href="http://myhost.example.com:8080/deepqa/v1/question/545706C1A3084322946C0B885A4289F0/feedback" rel="feedback"/>
<link href="http://myhost.example.com:8080/deepqa/v1/question/545706C1A3084322946C0B885A4289F0" rel="self"/>
</question>

```

### JSON response

```

{
  "question": {
    "answers": [
      {
        "confidence": 0.28743999999999997,
        "evidence": [
          {
            "copyright": "http://en.example.org/awiki/Copyrights",
            "document": "http://myhost.ibm.com:8080/deepqa/v1/question/document/9316/11/52",
            "id": "9316",
            "termsOfUse": "http://en.example.org/awiki/License",
            "text": "In 1585, the first English colony in the Americas was founded in Virginia by
            Walter Raleigh.
            The colony, named Roanoke, failed and became known as the lost colony.",
            "title": "England"
          },
          {
            "copyright": "http://en.example.org/awiki/Copyrights",
            "document": "http://myhost.example.com:8080/deepqa/v1/question/document/22416349/10/85",
            "id": "22416349",
            "termsOfUse": "http://en.example.org/awiki/License",
            "text": "The commission included a requirement to return with a lump of gold to prove
            that he had reached the South Sea, or a member of Raleigh's colony ",

```

```

        "title" : "History of a Settlement (1607-1699)"
    },
    ],
    "evidenceProfile" : [ { "name" : "FeatureGroup_Linguistic",
        "value" : "0.9999948206932325"
    },
    { "name" : "FeatureGroup_Type",
        "value" : "2.0069991645870886E-6"
    },
    { "name" : "FeatureGroup_Spatio-temporal",
        "value" : "1.3500703232240038E-6"
    },
    { "name" : "FeatureGroup_Relevance",
        "value" : "1.8222372797419442E-6"
    }
    ],
    "id" : 0,
    "text" : "Roanoke Colony",
    "formattedText": "<html>Roanoke Colony</html>"
},
{
    "confidence" : 0.15162999999999999,
    "evidence" : [ { "copyright" : "http://en.example.org/awiki/Copyrights",
        "document" : "http://myhost.example.com:8080/deepqa/v1/question/document/292285/15/44",
        "id" : "292285",
        "termsOfUse" : "http://en.example.org/awiki/License",
        "text" : "The Colony of Rhode Island and Providence Plantations, founded by religious dissenters forced to flee the Massachusetts Bay colony, is widely regarded as the first polity to grant religious freedom to all its citizens, although Catholics were barred intermittently. ",
        "title" : "State religion"
    },
    { "copyright" : "http://en.example.org/awiki/Copyrights",
        "document" : "http://myhost.example.com:8080/deepqa/v1/question/document/218110/10/24",
        "id" : "218110",
        "termsOfUse" : "http://en.example.org/awiki/License",
        "text" : "The Pilgrims, like the Puritans that would later found Massachusetts Bay Colony to the north, were a Protestant group that closely followed the teachings of John Calvin. ",
        "title" : "Plymouth Colony"
    }
    ],
    "evidenceProfile" : [ { "name" : "FeatureGroup_Linguistic",
        "value" : "0.9999961944410577"
    },
    { "name" : "FeatureGroup_Type",
        "value" : "1.3965711656993506E-6"
    },
    { "name" : "FeatureGroup_Spatio-temporal",
        "value" : "1.0809663024571708E-6"
    },
    { "name" : "FeatureGroup_Relevance",
        "value" : "1.3280214741508218E-6"
    }
    ],
    "id" : 1,
    "text" : "Massachusetts Bay Colony",
    "formattedText": "<html>Massachusetts Bay Colony</html>"
}
],
"category" : "",
"errorNotifications" : [ ],
"evidenceRequest" : { "items" : 2,
    "profile" : "yes"
},
"focuslist" : [ { "value" : "This lost colony" } ],
"id" : "545706C1A3084322946C0B885A4289F0",
"items" : 2,
"latlist" : [ { "value" : "colony" } ],
"links" : { "feedback" : "myhost.ibm.com:8080/deepqa/v1/question/545706C1A3084322946C0B885A4289F0/feedback",
    "self" : "http://myhost.example.com:8080/deepqa/v1/question/545706C1A3084322946C0B885A4289F0"
},
"passthru" : "",
"pipelineid" : "251817795",
"qclasslist" : [ { "value" : "FACT0ID" } ],
"questionText" : "This lost colony was never found?",
"status" : "Complete"
"synonymList" : [ {
    "lemma" : "television",

```

```
"partOfSpeech" : "noun",
"value" : "television",
"synSet" : [
    {
        "name" : "Wordnet_television-noun-1",
        "synonyms" : [
            {
                "isChosen" : true,
                "value" : "tv"
            },
            {
                "isChosen" : false,
                "value" : "telecasting"
            },
            {
                "isChosen" : true,
                "value" : "video"
            }
        ]
    },
    {
        "name" : "Wordnet_television+receiver-noun-1",
        "synonyms" : [
            {
                "isChosen" : false,
                "value" : "telly"
            }
        ]
    }
]
}
```

**Related concepts:**

“Synonym feedback example” on page 19

Watson uses synonyms to improve answer recall. In release 2.2 and later, you can provide the list of synonyms that you want Watson to use when it answers a question, which might improve retrieval results.

**Related reference:**

“Pipeline messages and HTTP codes” on page 21

A request to the Question and Answer REST service might result in a warning or error before Watson processes the request. Use the error response information that is returned by the pipeline to diagnose and resolve the issue.

"Messages returned with an answer response" on page 25

When the Question and Answer service returns an answer and issues exist, information about the issue is returned with the response.

### POST /deeqpa/v1/question/{QID}/feedback (deprecated)

Add feedback to the answer feedback collection.

## Purpose

**Important:** This operation requires asynchronous mode, which is deprecated.. With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. If you implemented the Question and Answer REST service using the asynchronous mode, update your code to use the synchronous mode instead.

The Question and Answer REST service supports providing feedback on the quality of an answer and the relevance and authority for supporting passages. For example, users might enter feedback when they are viewing the answers. The feedback is saved with the input and output for later analysis.

## Request syntax

The input and output message document formats are described with an informal syntax. Because of the formal nature of XPath notation, the principle description is in XML format. The values indicate XML schema data types. For JSON, the XML element names correspond to JSON object names.

The special characters that follow the elements denote usage. Each element without one of these appended characters is a required element. For JSON, elements that are shown with an asterisk (\*) or plus sign (+) correspond to JSON arrays, and the names are not used. The following special characters are used in the syntax:

?        0 or 1 occurrence is allowed.

\*        0 or more occurrences are allowed.

+        1 or more occurrences are allowed.

```
<?xml version="1.0"? xmlns="Watson REST server feedback URI"
  xmlns:qa="QA REST server URL">
  <feedback id="String">
    <answer id="String">+
      <rankRating>Integer</rankRating>
      <passage id="{PID}">+
        <relevanceRating>Integer</relevanceRating>
        <authorityRating>Integer</authorityRating>
      </passage>
    </answer>
  </feedback>
```

## Request header

Table 13. Request header elements

Name	Value	Required or optional
Accept	Specify the format of the response. The values of Content-Type and Accept must be the same. application/json   application/xml	Optional
Content-Type	Specifies the format of the input message. The values of Content-Type and Accept must be the same. application/json   application/xml	Required

## Parameters

The service supports these parameters in the resource URL.

Table 14. Resource parameters

Name	Value	Required or optional
alt	Specifies whether XML or JSON is used for the request and accepted for the response. The <b>alt</b> parameter takes precedence when both the <b>alt</b> parameter and the HTTP Accept header are specified. application/json   application/xml <b>Tip:</b> Use the <b>alt</b> parameter during only development and testing.	Optional

## Request elements

Table 15. Elements of the request payload

Element Name	Description	Required or optional
/feedback	The container for feedback on the answers and evidence for a question. The element has one attribute: <b>@id</b> The integer value of /question/@id in the response for <b>POST /v1/question</b> . The ID represents the target question for the feedback.	Required
/feedback/answer	The container for feedback for a particular answer. The element has one attribute: <b>@id</b> The integer value of /question/answers/answer/@id in the response for <b>GET v1/question/{QID}</b> . The ID represents a particular answer.	Required
/feedback/answer/passage	The container for information about a supporting passage. The element contains one attribute: <b>@id</b> An integer that uniquely identifies this passage within the set of supporting passages for this answer.	Optional
/feedback/answer/rankRating	Indicates how good this answer is. An integer in the range 1 - 5. Higher values represent better answers.	Optional
/feedback/answer/passage/relevanceRating	Indicates how relevant the passage is for providing support for the answer. An integer in the range 1 - 5. Higher values represent higher relevancy.	Optional
/feedback/answer/passage/authorityRating	Indicates a measure of the authority of the document that contains the passage when used to support the answer. An integer in the range 1 - 5. Higher values represent greater authority.	Optional

## Response outcomes

**Important:** With release 2.8, the asynchronous mode of the Question and Answer service is deprecated. All asynchronous responses include the X-Deprecated header. The value of the header is the date of deprecation. The dates follow the format *dd mmm yyyy*. The *dd* variable represents the two-digit day, *mmm* represents the three-character month name (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec), and *yyyy* represents the four-digit year.

## Example request

These examples provide feedback about the quality of two answers. They also include information about the relevance and authority for a passage in each answer.

### Example XML request

```
<?xml version="1.0" encoding="UTF-8"?>
<feedback xmlns="http://www.ibm.com/xmlns/deepqa/question/feedback"
xmlns:qa="http://www.ibm.com/xmlns/deepqa/question" id="1547593">
  <answers>
    <answer id="1">
      <rankRating>2</rankRating>
      <passage id="1">
        <relevanceRating>3</relevanceRating>
        <authorityRating>4</authorityRating>
      </passage>
    </answer>
    <answer id="3">
      <rankRating>2</rankRating>
      <passage id="1">
        <relevanceRating>3</relevanceRating>
        <authorityRating>4</authorityRating>
      </passage>
    </answer>
  </answers>
</feedback>
```

### Example JSON request

```
{
  "feedback": {
    "id": "1547592",
    "answer": [
      {
        "id": "1",
        "rankRating": "2",
        "passage": {
          "id": 1,
          "relevanceRating": "3",
          "authorRating": "4"
        }
      },
      {
        "id": "3",
        "rankRating": "2",
        "passage": {
          "id": 1,
          "relevanceRating": "3",
          "authorRating": "4"
        }
      }
    ]
  }
}
```



**Related reference:**

“Pipeline messages and HTTP codes” on page 21

A request to the Question and Answer REST service might result in a warning or error before Watson processes the request. Use the error response information that is returned by the pipeline to diagnose and resolve the issue.



## Chapter 5. The JavaScript toolkit

The JavaScript toolkit provides files and widgets to help you create rich, web-based applications that are enabled for Watson. You can use these widgets in your own applications.

---

### JavaScript samples

Sample JavaScript files and widgets to create rich web-based applications that can be used in your own applications.

#### Running the samples

Before you run the samples, deploy the projects to your web application server. The JavaScript samples use the `com.ibm.watson.sdk.dojo`, `com.ibm.watson.sdk.wea.toolkit`, and `com.ibm.watson.sdk.wea.toolkit.samples` projects.

After you deploy the projects, you can view the samples in your web browser by opening the HTML files in the following sections. The HTML files are stored in the `com.ibm.watson.sdk.wea.toolkit.samples` project in the `WebContent` directory. The links assume that your web browser runs on localhost on port 8080.

#### Question controls

##### Question input

A Dojo-based question input widget that allows a user to enter a question.

##### Question input and followup

A Dojo-based question input widget that allows a user to enter a question and then specify more information for the question as a follow-up request.

#### Progress controls

##### Progress indicator with jQuery

A progress indicator that displays an animated IBM Watson graphic. It also includes a customizable status message with the percentage complete, a progress bar showing percent that is done in five stages, and a Cancel button.

##### Progress dialog control

A progress dialog that also displays an animated IBM Watson graphic.

##### Progress bar control

A progress bar control.

#### Confidence and evidence controls

##### Confidence bar

A bar that indicates the confidence in a particular piece of evidence along with a label and confidence percentage.

##### Evidence button

A styled button with associated label. The label can be displayed north, south, east or west of the Evidence icon.

**Evidence viewer**

A styled container to display evidence information and to define a title.

**User controls****Rating control**

A widget that provides a way for a user to rate an answer or a piece of evidence.

---

## Design patterns for an IBM Watson enabled application

The IBM Watson enabled application toolkit includes a collection of user interface elements to use in a customized IBM Watson enabled application.

The following list includes elements that are currently available as well as elements that are in the sample application and for preview only and not fully supported:

- “Progress Dialog and Progress Monitor controls” on page 64
- “Confidence Bar”
- “Evidence Button” on page 58
- “Evidence and Evidence Evaluation widgets” on page 60
- “Rating System” on page 57
- “Question input” on page 56
- “Question input with dialog” on page 57
- 
- Preview only:
  - “Watson Bar” on page 58
  - “Evidence Card” on page 61
  - “Reference Card” on page 62
  - “Panels specific to Watson” on page 58
  - “Question/Answer Set Pair” on page 63

**Confidence Bar**

- Widgets:
  - `com.ibm.watson.ui.ConfidenceBar`
- CSS: `ConfidenceBar.css`

The confidence widget displays a bar indicating the confidence in a particular piece of evidence, along with a label and confidence percentage.

There are two versions currently available.

- Displays five boxes for the confidence scale:



- Displays three boxes for the confidence scale:



The percentage and confidence level can each be positioned independently, specified by compass points (north, south, east or west).

The following parameters can be specified on the confidence bar control:

- `confidenceBarWidth`  
The width in pixels of the confidence bar to display. Optional. Default is 48.
- `confidenceBarHeight`  
The height in pixels for a confidence bar to display. Optional. Default is 16.
- `useSmallConfidenceBars`  
Set to true to show the smaller three box version of the confidence bar. Default is false.
- `useDarkBackground`  
Set to true if the confidence bar is being placed on a dark background and you want a darker grey displayed in the confidence bars. Default is false.
- `confidence`  
The percentage confidence level to display. Default is 0.
- `displayPercentage`  
Comma-separated list of compass points to indicate where the confidence percentage is shown. For example, "north,south,east,west" would display the percentage to the top, bottom, right and left of the confidence bar. The default value is east only.
- `displayLevel`  
Comma-separated list of compass points to indicate where the confidence level is shown. For example, "north,south,east,west" would display the confidence level to the top, bottom, right and left of the confidence bar. The default value is east only.
- `confidenceLevelStrings`  
Comma-separated list of confidence levels. The confidence bar supports five confidence levels. By default these are Very Low, Low, Medium, High, and Very High (or Low, Medium, and High for the three box version) but these display values can be changed using this attribute.  
  
The list must be in the form <Low Level String>,<Medium Level String>,<High Level String> for three bars or <Very Low String>,<Low String>,<Medium String>,<High String>,<Very High String> for five bars.  
  
If you are using the three bar version, you must specify three confidence levels or this value will be ignored. You must specify five levels for the larger five box bar.
- `confidenceLevelClippedStrings`  
Comma-separated list of clipped confidence levels. The confidence bar supports three or five confidence levels, whose display values are customizable using the `confidenceLevelStrings` attribute. If these string are too long to be displayed in the UI shorter (clipped) versions can be provided which will be displayed instead of the full version.  
  
The list must be in the form <Low Level Clipped String>,<Medium Level Clipped String>,<High Level Clipped String> for three levels or <Very Low Level Clipped String>,<Low Level Clipped String>,<Medium Level Clipped String>,<High Level Clipped String>,<Very High Level Clipped String> for five levels. These actual settings are the default values.
- `clipConfidenceLevelString`

Set `clipConfidenceLevelString` to `true` to display the clipped version of the confidence levels. Default is `false`.

- `confidenceLevels`

A comma-separated list of confidence levels. For a five box version of the confidence bar, four confidence levels are needed.

The confidence levels need to be set in the form `<very low cut off>,<low cut off>,<medium cut off>,<high cut off>`. Any value greater than the high cut off is considered very high. For a three box version of the confidence bar, two confidence levels are needed, in the form `<low cut off>,<medium cut off>`.

The default value is 10,40,70,80 for five boxes and for the three box version the default is 40,70.

- `confidenceLevelClasses`

A comma-separated list of confidence level classes to use. For a five box version of the confidence bar, five confidence classes are needed. The default is `very_low,low,moderate,high,very_high` For a three box version the default is `low,moderate,high`.

The confidence level classes need to be set in the form `<very low class>,<low class>,<medium class>,<high class>,<very high class>` for a five box version and for a three box version of the confidence bar, three confidence classes are needed, in the form `<low class>,<medium class>,<high class>`.

The following code example shows how to specify a confidence bar:

```
<div data-dojo-type="com.ibm.watson.ui.ConfidenceBar" data-dojo-props="confidence:10"></div>
```

A programmatically created confidence bar might look like this

```
require(["com/ibm/watson/ui/ConfidenceBar"], function (ConfidenceBar) {
    var aConfidenceBar = new ConfidenceBar({
        useDarkBackground:<useDarkBackground>,
        useSmallConfidenceBars:<showSmallBars>,
        confidence:<percentage>,
        displayPercentage:<displayPercentage>,
        displayLevel:<displayConfidence>,
        confidenceLevelStrings:<custom_confidence_levels>,
        confidenceLevelClippedStrings:<custom_clipped_confidence_levels>,
        clipConfidenceLevelString:<display_clipped>,
        confidenceBarWidth:<confidence_bar_width>,
        confidenceBarHeight:<confidence_bar_height>
    });
    dojo.place(aConfidenceBar.domNode, document.getElementById("displayConfidenceBarHere"),"only");
});
```

## Question input

This control is a Dojo-based question input widget that allows a user to enter a question and then click **Ask** to get an answer. For example, if a user enters "Who is the president?" the best first answer is shown, and any follow-up request or question is always treated as a new question.

## Question input with dialog

This control is a Dojo-based question input widget that allows a user to enter a question and then to specify more information for that question, as a follow-up request. For example, if a user asks "Who is the president?", Watson might send a best first answer but also make a request such as, "Please specify a country or organization." The user's next entry is considered part of that dialog interaction, so if they enter "USA" they are effectively asking "Who is the president [of the] USA?". The user can opt to abandon a line of questioning by clicking Cancel Question or New Question, and start over at any time.

## Rating System

- Widget: `com.ibm.watson.ui.RatingSystem`
- CSS: `RatingSystem.css`

The `RatingSystem` widget provides a way for a user to rate an answer or a piece of evidence returned by Watson. It is an extension of the `dojox Rating` widget. The `RatingSystem` widget provides a star-based rating system as well as a label that can be positioned either north, south, east or west. You can specify the number of stars to show, how many should be selected by default, what hover text to show for each star and so on.

The `RatingSystem` can be used to rate an answer or a piece of evidence from IBM Watson. The widget includes support for associating the button with a piece of data. A host application can be notified in the following ways of a rating being changed:

- A developer can connect their own function with the `onStarClicked` function of the widget.
- A developer can provide a callback by using a parameter that will be called when the rating is changed.
- A developer can specify a topic to be published to when the rating is changed by using `dojo publish` or `subscribe` methods.

The following parameters can be specified on the `RatingSystem` widget:

- `numStars`  
The number of stars to show for the ratings widget. Default is five.
- `label`  
The label that should appear along with the rating system. Default is blank ("").
- `hoverText`  
The hover text to appear for the stars. Must be in the form of a comma-separated list of hover text strings, one for each star.  
If the number of hover text strings does not match the number of stars, the hover text will not be displayed. Default is blank ("").

- `displayLabel`  
Compass point to indicate where the label is shown (that is, "north", "south", "east" or "west"). The default value is west.
- `ratingChangedTopic`  
The topic to publish to when the rating is changed. Default is `PubSubTopics.WATSON_RATING_VALUE_CHANGED`.
- `ratingChangedCallback`  
Callback function to call when the evidence button is clicked. Default is null.
- `evidenceData`  
A parameter to store general evidence or answer information. This argument value could be an ID to an evidence object or an evidence data source.  
The contents of this field will be added to the click event and returned to the rating changed callback, as well as published to the rating changed topic. Default is null.
- `value`  
The number of stars to have selected when the control is loaded on the page, that is, the initial rating for the widget. Default is 0.

## Watson Bar

The Watson Bar presents the navigation choices in a Watson application. In addition to the tabs that are specific to Watson, an Ask Watson button is always available on the leading edge of the Watson Bar.



Selecting the **Ask Watson** button (including the avatar) results in the Ask Watson panel sliding up with the focus on the entry field.

## Panels specific to Watson

Each tab that is specific to Watson is associated with a unique panel. All such panels have the following common characteristics:

- Selecting a Watson tab results in its associated panel sliding up while other Watson panels slide down at the same time. The slide duration is 600 msec.
- Watson panels are all the same height within an Advisor.
  - The panels are sized to reveal sufficient amount of the original application to be maintain the user context and to easily select (thereby collapsing an exposed Watson panel).
  - The panel chrome contains collapse and user assistance buttons.
  - Options can be listed within Watson panels. In each case, an Evidence button is positioned along the trailing edge of each option in the lists.

## Evidence Button

- Widget: `com.ibm.watson.ui.EvidenceButton`
- CSS: `EvidenceButton.css`



The evidence button is a styled button with associated label. The label can be



displayed north, south, east or west of the Evidence icon.

The evidence button can be used to display a piece of evidence for a particular answer from IBM Watson by associating the button with a piece of evidence.

A host application can be notified in the following ways through an evidence button being clicked by a user:

- A developer could associate their own function with the onClick event of the button.
- A developer can provide a callback by using a parameter that will be called when the button is clicked.
- A developer can specify a topic to be published to when the button is clicked by using dojo publish or subscribe methods.

The following parameters can be specified on the evidence button widget:

- `label`  
The label to display for this evidence button. Default is a blank label ("").
- `evidenceButtonClickedTopic`  
The topic to publish to when the button is clicked Default is `PubSubTopics.WATSON_EVIDENCE_BUTTON_CLICKED`.
- `displayLabel`  
Comma-separated list of compass points to indicate where the label is shown. For example, "north,south,east,west" would display the label to the top, bottom, right and left of the evidence button. The default value is south only.
- `evidenceBtnClickedCallback`  
Callback function to call when the evidence button is clicked. Default is null.
- `evidenceData`  
A parameter to store general evidence information. This could be an ID to an evidence object or an evidence data source. The contents of this field is added to the click event and returned to the button-clicked callback, as well as published to the button-clicked topic. Default is null.

Here is a simple example of putting an evidence button on a page:

```
<div data-dojo-type="com.ibm.watson.ui.EvidenceButton"></div>
```

The user of a Watson application can use the system for the evidence and reasoning underlying any information visible in the application. This is always enabled by the placement of an Evidence Button on the trailing edge of the option's row.


The following image shows an example of placing an Evidence button on a Watson application page.

<b>Treatment plan 1</b> Trastuzumab	Confidence <b>50%</b>	
<b>Treatment plan 2</b> Afatinib	Confidence <b>30%</b>	
<b>Treatment plan 3</b> Dacomitinib	Confidence <b>10%</b>	

## Evidence and Evidence Evaluation widgets

- Widgets:
  - com.ibm.watson.ui.Evidence
  - com.ibm.watson.ui.EvidenceEvaluation
- CSS:
  - Evidence.css
  - EvidenceEvaluation.css

The Evidence widget provides a styled container that should be used to display evidence information and allows you to define a title.

 Evidence

**PolicyGuideline ID:** RAD.00041      **Version:** 2010-05-10

**Title:** Intensity Modulated Radiation Therapy (IMRT)

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint o

**Policy and Guideline Evidence:**

IMRT of the prostate is considered **medically necessary** in individuals who meet the following:

- In hac habitasse platea dictumst
- Vivamus semper metus nunc, vitae viverra nulla.

The EvidenceEvaluation widget is another styled container that should be used to display any evaluation of that evidence to be returned to the middleware.

**Evaluate Recommendations**

Recommendation & Evidence ☐ Accept ☐ Reject ☐ Not Reviewed

Policy and Guideline Version Policy Guideline is correct ▼

Additional Comments

Submit

The two widgets can be displayed together, with zero padding or margins between them, but they can also be displayed independently.

- Here is the parameter that can be set on the Evidence widget:  
     evidence\_label: The title label to display in the Evidence viewer. Default is **Evidence**.
- Here is the parameter that can be set on the Evidence Evaluation widget:  
     label: The title label to display in the Evidence evaluation viewer. Default is **Evaluate Recommendations**.

Here is an example of adding an evidence widget and evidence evaluation widget to a page:

```
<div data-dojo-type="com.ibm.watson.ui.Evidence">
  ...some content...
</div>
<div data-dojo-type="com.ibm.watson.ui.EvidenceEvaluation">
  ...some content...
</div>
```

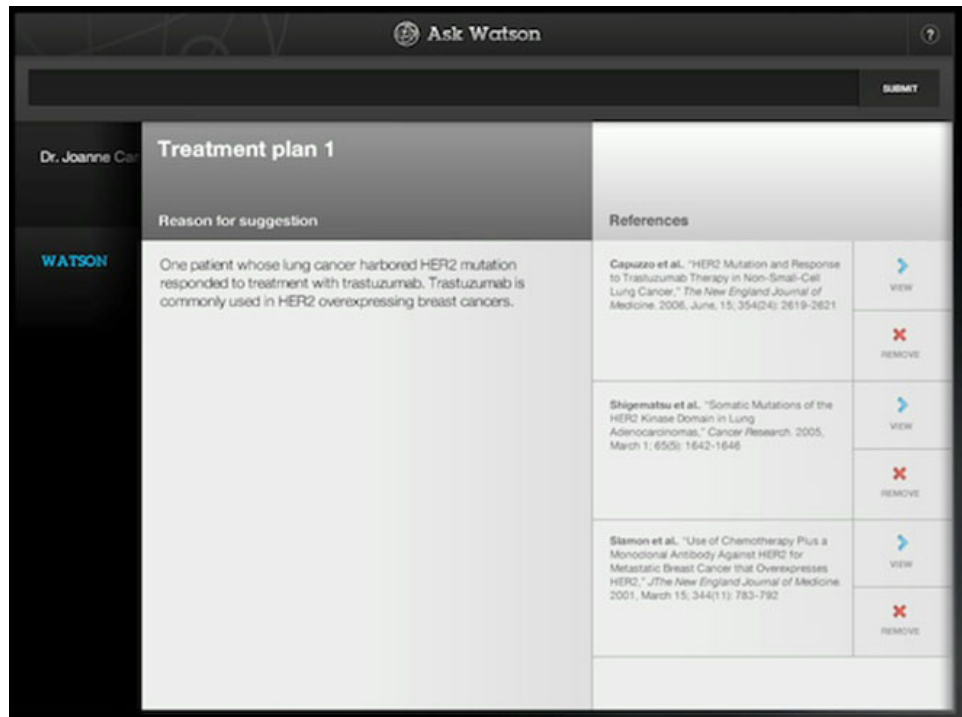
## Evidence Card

In response to the user selecting to review the evidence underlying information in the Watson application, the panel containing the evidence can slide from the trailing edge of the display space. For example, the Evidence Card could split into two columns:

- **Reasons**
- **References**

The former column describes the reasoning underlying the option while the latter provides access to the evidence underlying the reasoning.

Each item in the References column includes a View option which routes the user to the associated Reference Card.



In a mobile interface, a user can flick toward the trailing edge or tap on an underlying display on the leading edge to close the Evidence Card by sliding it out of view toward the trailing edge in 300 msec.

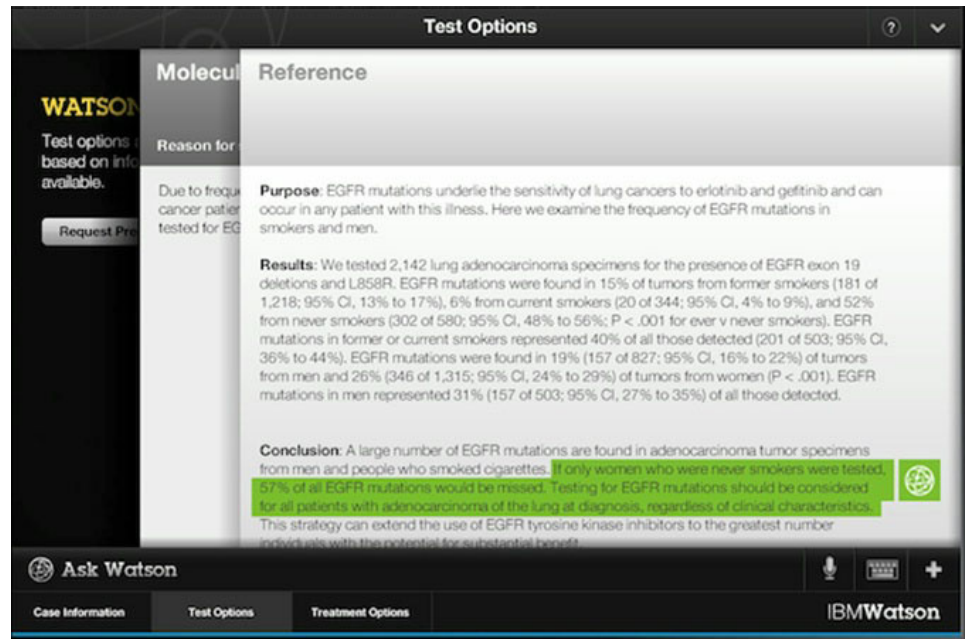
## Reference Card

In response to the user selecting to review a reference included on an Evidence Card, the panel containing the reference slides from the trailing edge of the display space. The Reference Card is split into two columns:

- Reference content
- IBM Watson indicator

All passages that Watson relies on are highlighted in the same way. Watson indicators are used to underscore that Watson is reasoning regarding the highlighted passages and are top-justified with each used passage. More than one line of continuous or discontinuous text may map onto a single Watson indicator because the design spans multiple rows. The latter column is used to highlight the reference content that led to the Watson-enabled option and associated reasoning.




The Evidence Card is designed to take up 75% of the display space width.



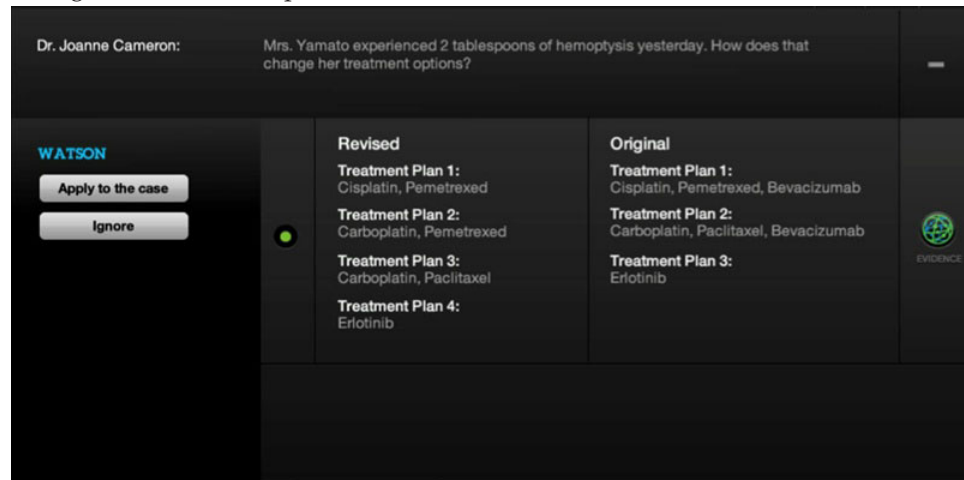
In a mobile interface, a user can flick toward the trailing edge or tap on an underlying display on the leading edge to close the Evidence Card by sliding it out of view toward the trailing edge in 300 msec.

## Question/Answer Set Pair

Question/Answer Set Pair is used for both explicit questions within standalone question answering applications and within larger IBM Watson Advisors. In the latter case, a user can choose to apply a candidate answer to the work item currently being addressed by the Advisor. The contents and formatting of candidate answers might be highly variable and based on specific domains and requirements that might require Candidate Answer patterns.

Dr. Joanne Cameron: How do I manage a lung cancer patient with a HER2 mutation?		—	
<b>WATSON</b>		There is some evidence to support using the following agents in patients with lung cancer harboring a HER2 mutation:	
Treatment plan 1 Trastuzumab		Confidence <b>50%</b>	
Treatment plan 2 Afatinib		Confidence <b>30%</b>	
Treatment plan 3 Dacomitinib		Confidence <b>10%</b>	

A larger Advisor example:



When presented in the context of a larger Advisor, a user can choose whether to apply the information associated with a candidate answer to the work item, such as an oncology treatment selection.

Since there might be multiple candidate answers from which to choose, a user must be able to select specific candidate answers to apply.

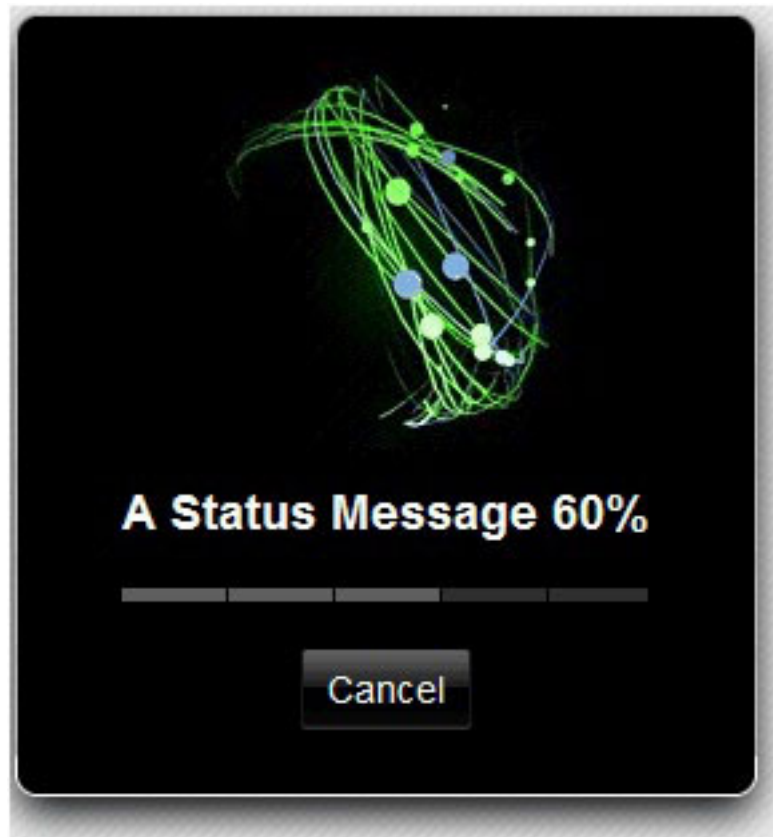
## Progress Dialog and Progress Monitor controls

- Widgets:
  - `com.ibm.watson.ui.ProgressMonitor`
  - `com.ibm.watson.ui.ProgressDialog`
- CSS: `ProgressDialog.css`

The progress monitor widget is a progress indicator that displays in a `<div>` section in a page.

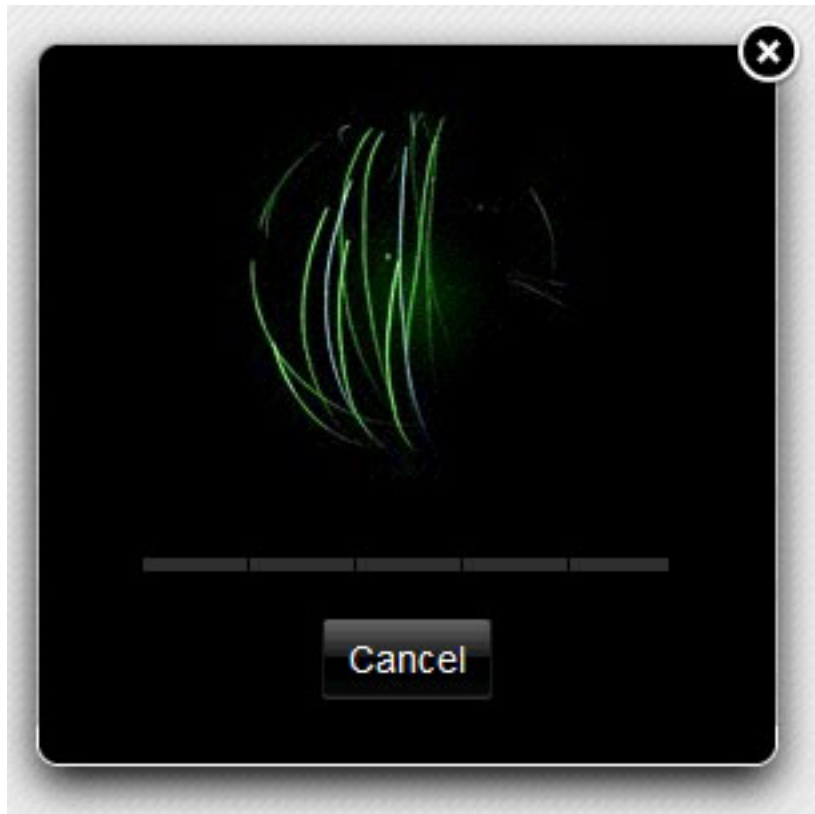
The progress dialog widget is an extension of the progress monitor widget, where the same progress indicator appears in a dialog.

When indicating that a process is being executed, the IBM Watson avatar animation is used to indicate the ongoing process. Displays the various steps in the process as they are being executed.



The progress indicator displays an animated IBM Watson graphic, a customizable status message with the percentage complete, a progress bar showing percent done in five stages, and a **Cancel** button.

The progress dialog is similar with the exception that it is a dialog and it has a **Close** button.



The progress monitor and progress dialog controls use dojo publish and subscribe calls to display or hide the monitor, cancel the progress task, and update the status message and the percent done message.

The following parameters can be specified to customize the widget:

- `openTopic`  
The topic to subscribe to which tells us when to display the progress monitor. Default is `PubSubTopics.WATSON_PROGRESS_OPEN`
- `statusTopic`  
The topic to subscribe to which tells us to update the progress status. Default is `PubSubTopics.WATSON_PROGRESS_STATUS_UPDATE`
- `closeTopic`  
The topic to subscribe to which tells us when to close the progress monitor. Default is `PubSubTopics.WATSON_PROGRESS_CLOSE`
- `progressTopic`  
The topic to subscribe to which tells us what to update the progress percentage to. Default is `PubSubTopics.WATSON_PROGRESS_UPDATE`
- `cancelTopic`  
The topic to publish to if a user clicks the Cancel button in the progress monitor. Default is null so that no action is taken when a user clicks the Cancel button.
- `loopProgress`  
Loop the progress bar in the progress monitor and don't wait for progress updates. Default value is false.
- `loopProgressSpeed`  
The speed, in milliseconds, to loop the progress bar. Default is 35 ms.
- `displayProgressPercent`



Specify whether or not you want to see the percent done alongside the progress status in the progress monitor. Default value is false.

Here is an example of adding a progress monitor to a page:

```
<div data-dojo-type="com.ibm.watson.ui.ProgressMonitor">
</div>
```

Or like this with some of the parameters set:

```
<div data-dojo-type="com.ibm.watson.ui.ProgressMonitor" data-dojo-props="openTopic:'anOpenTopicId',
statusTopic:'aStatusTopicId'cancelTopic:'aCancelTopicId',progressTopic:'progressTopicId'">
</div>
```

To make the progress monitor visible, a user would need to publish to the open topic for the monitor, specifying the initial status message to display. Like this

```
function displayProgressDiv(){
  require(["dojo/topic"],
    function(topic){
      topic.publish("openTopicId", "An Initial Status Message");
    }
  );
}
```

To change the status message in the monitor after it is visible, a user would need to do the following

```
function updateProgressDivStatus(){
  require(["dojo/topic"],
    function(topic){
      topic.publish("statusTopicId", "A new Status Message");
    }
  );
}
```

To update the progress of the progress monitor, a user would need to do the following, where aValue is a number (must be in the for '10' etc.. not '10%').

```
function updateProgressDivPercentDone(){
  require(["dojo/topic"],
    function(topic){
      topic.publish("progressTopicId", aValue);
    }
  );
}
```

and so on.

There is a distinct difference between Cancel and Close with the progress monitor and progress dialog. Closing the monitor removes it. Cancelling the monitor closes the monitor but it also publishes out to the cancelTopic so that any subscribed functions can do whatever actions are necessary to cancel the job associated with the monitor.



## Chapter 6. SDK PHP library overview

The SDK includes a PHP library that you can use to call services for Watson. You can use the Watson PHP library in your own PHP application.

### Before you begin

- Set up a PHP runtime environment that is compiled with cURL support and connected to a web server.
- Configure the connection to the REST server URL. Browse to the `com.ibm.watson.sdk/PHP/libs` directory in the SDK. Open the `question.html` file with a text editor and verify or change the server value.

### Features of the PHP library

For now, only the Watson Question and Answer service is supported, and you can post a question and retrieve an answer. The library handles formatting the question by adding the appropriate JSON and header information into a form that can be posted to Watson.

### Installing and configuring the SDK PHP library

The `com.ibm.watson.sdk/PHP/` directory contains a small Watson PHP library and a sample html page. Copy the contents of the directory to your web server and maintain the directory structure:

```
| - question.html  
| - question.php  
| - libs+  
|   |  
|   | - watson.php
```

### Running the samples

To run the sample, go to `http://localhost/question.html`, where *localhost* is the location of the SDK PHP library in your web server.



## Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
1623-14, Shimotsuruma, Yamato-shi  
Kanagawa 242-8502 Japan

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created

programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation  
Bldg. 101/4th Floor/Dept. MU5A/Div F6  
11301 Burnet Road  
Austin, TX 78758  
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

---

## Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, and service names may be trademarks or service marks of others.

---

## Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy> and the section titled "Cookies, Web Beacons, and Other Technologies" in IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details>. Also, see the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.









Product Number:

IBM Confidential  
Printed in USA