Hack Day Guide:

Processing Live Tweets

datasciencedojo

Table of Contents

datasciencedojo

Lab Prerequisites

1) An Azure 30-day free trial account, or an Azure account with administrative access rights
   a) A free trial account can be found here:
      http://azure.microsoft.com/en-us/pricing/free-trial/
2) Text Editor (also called IDE; below are two recommendations, only one of them)
   a) Notepad++: http://notepad-plus-plus.org/download/v6.7.5.html
   b) Sublime Text 2: http://www.sublimetext.com/2
3) A Twitter Account

datasciencedojo

Chapter 1: Overview

This lab will show you how to work with a Twitter stream. Specifically to save the data from the Twitter stream and to feed it to a live dashboard. The business applications for real-time Twitter data is quite versatile. Some examples include tracking tornados, real-time sentiment analysis by geographic location, and brand management portals.

Below is a picture of the end result, a dashboard that updates Tweets, live.

created_at	tweet_body
Wed May 06 00:56:38 +0000 2015	@Dragonostic yeah
Wed May 06 00:56:38 +0000 2015	@joshdreyer thanks Josh, life is good
Wed May 06 00:56:38 +0000 2015	RT @DJMARCD: ;;; #NewMusic #RapVideos BO...
Wed May 06 00:56:38 +0000 2015	RT @KING5Seattle: Please RT >> #voicesaveki...
Wed May 06 00:56:38 +0000 2015	RT @nocontrol91: Remember when i accidental...
Wed May 06 00:56:38 +0000 2015	RT @wantAfernplant: And one vote from my gr...
Wed May 06 00:56:38 +0000 2015	RT @xtinaNOW: She has so much more to give!...
Wed May 06 00:56:38 +0000 2015	This movie pretty bad ass
Wed May 06 00:56:37 +0000 2015	#LisaRinna Belle Gray Lisa Rinna Fly Front Pock...
Wed May 06 00:56:37 +0000 2015	#VoiceSaveIndia win win win
Wed May 06 00:56:37 +0000 2015	@CanadianPaleo Prize patrol is giving 1 Austral...
Wed May 06 00:56:37 +0000 2015	@natemaloley MARIA UR SO CUTE
Wed May 06 00:56:37 +0000 2015	A nigga cant impress me with shit

datasciencedojo

Technology and Workflow:

The pipeline that you will build in this guide will span the entire big data processing (ETL) spectrum.
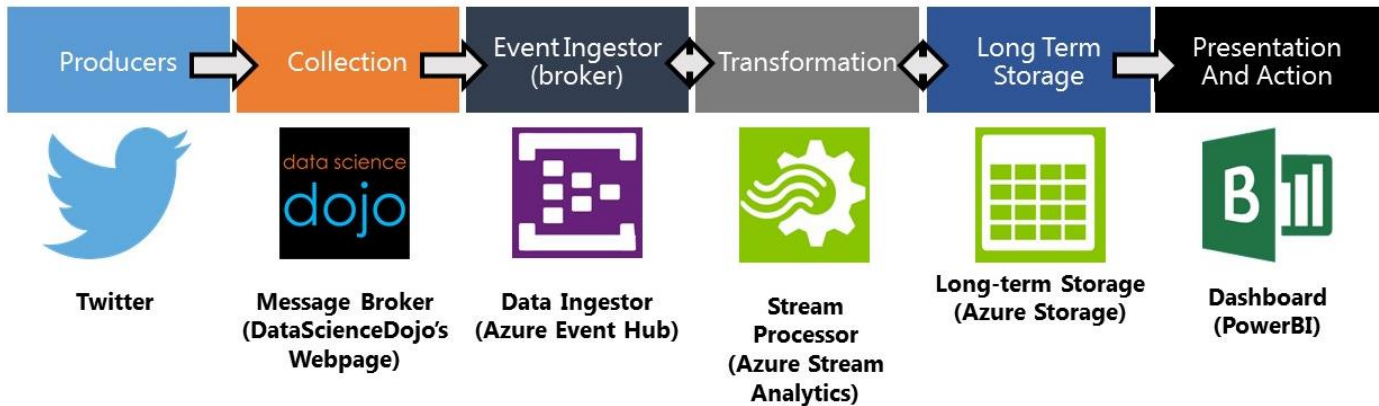
First we will turn your Twitter account into a web app that is capable of pushing live Tweets. Then we will use a message broker designed by DataScienceDojo to request Tweets from the Twitter account and push the Tweets into your own Azure Event Hub. From there we will have Stream Analytics stream processors read, filter, and aggregate the Tweets that go through. The Stream processors will output to both a dashboard for real time analysis, and an Azure Storage Account (blob) for long-term storage.
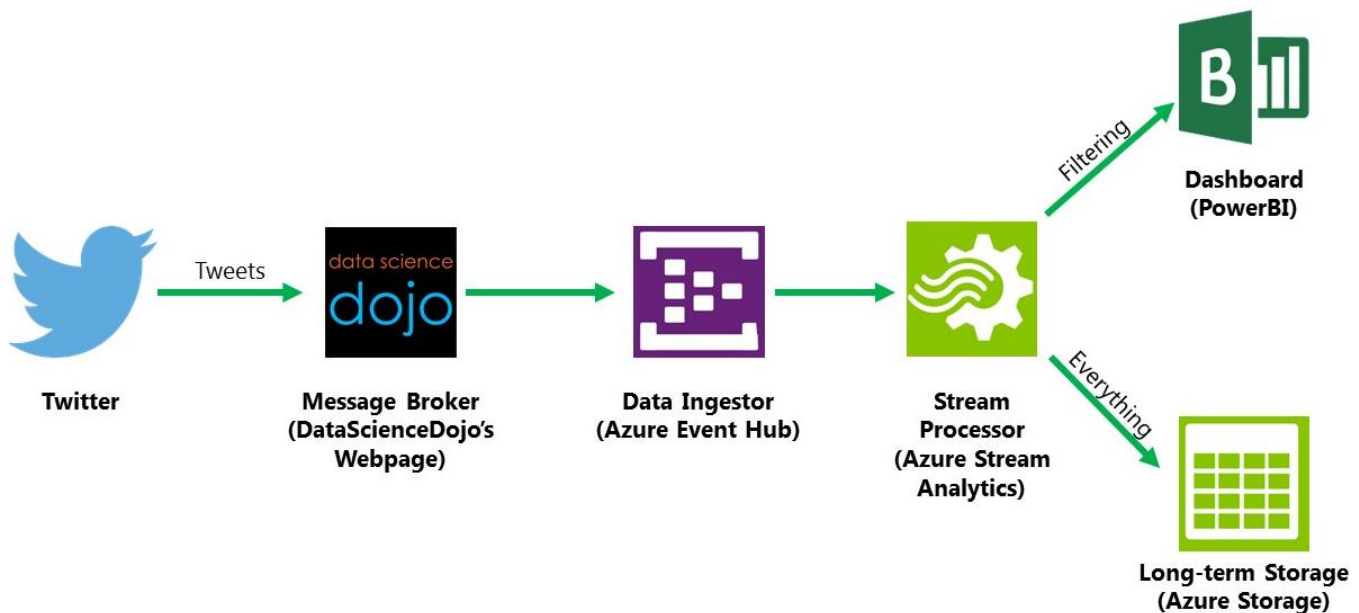
Chapter 2: Twitter

Twitter is a good source for public and abundant amounts of information. About 4500 Tweets are generated every second. Twitter opens up a free and throttled down API for anyone who has a Twitter account, which will grab 1~40% of the actual live tweets that are being generated (about 1-3 Tweets a second). Which means anyone with some programming knowledge can obtain at least 86,400 Tweets per day per account using this free and public API. We will setup your Twitter account to be able to do this.

Exercise 1: Getting a Twitter Account

We will be taking a conventional Twitter account and turning it into a web app that will live stream incoming tweets from around the world. To do this, you must first have a Twitter account. **If you already have a Twitter account, skip to Exercise 2.**
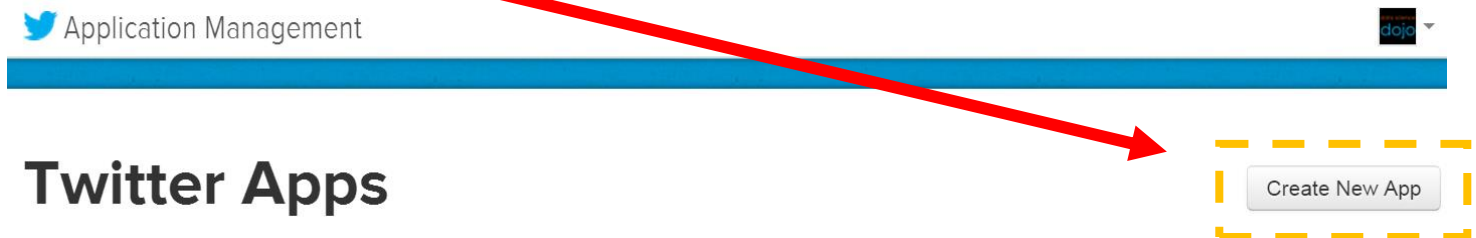
1. Go to http://twitter.com and find the sign up box, or go directly to https://twitter.com/signup

2. Enter your **full name**, **email address**, and a **password**

3. Click "**Sign up for Twitter"**

4. On the next page, you can select a **username** (usernames are unique identifiers on Twitter) — type your own or choose one Twitter suggests. Twitter will tell you if the username you want is available.

5. **Double-check** your name, email address, password, and username.

6. Click "**Create my account"**

7. Twitter will send a **confirmation email** to the email address you entered. Click the link in that email to confirm your email address and account.

8. You will need to add your phone number to the account under settings in order to receive a Twitter app from Twitter. Twitter will send you a confirmation text to the phone number via text messaging. Please read this tutorial on how to add a mobile phone number to your account: https://support.twitter.com/articles/110250-adding-your-mobile-number-to-your-account-via-web

datasciencedojo

Exercise 2: Provisioning a Twitter Streaming App

The Twitter Streaming APIs use [OAuth](#) to authorize requests. The first step to use OAuth is to create a new application on the Twitter Developer site. Make sure you have a mobile phone number synced to your twitter account or else you will not be provisioned a stream app from Twitter.
How to: https://support.twitter.com/articles/110250-adding-your-mobile-number-to-your-account-via-web

1. Sign in to https://apps.twitter.com/
2. Click **Create New App**.

3. Enter **Name**, **Description**, **Website**.
   a) The Website field is not really used. It does not have to be a valid website but does have to look like a valid website, so make sure it starts with **"http://www"** and ends with **".com"**.
   b) The description is for your own documentation.
   c) You may leave the call back URL blank.
   d) Agree to the developer agreement and create.

datasciencedojo

4. Within your provisioned Twitter app, click on **"Keys and Access Tokens"**.

5. Scroll to the bottom and click **"create my access token"** to generate security tokens for your Twitter app.

datasciencedojo

6. Click on "**Test OAuth**" on the top right-hand corner to get a summarization of your 4 security keys.

7. Keep this page open, you will need these 4 security keys to create a Twitter stream. Alternatively you may save them to a text editor like notepad/word/notepad++/sublime. You now have a Twitter API that is capable of pushing out Tweets upon request.

Exercise 3: Inputting Your Twitter API Credentials to the Broker

Data Science Dojo has built a WebApp that will take in a Twitter API, then listen in for live Tweets, and send those Tweets into an Azure Event Hub, given the Event Hub connection credentials. The WebApp is built in Django and utilizes a Twitter streaming package called Tweepy. If you wish to build a brokering app like this yourself, that maintains a Twitter stream connection, we also recommend C# and a package called Tweetini. For Twitter packages in any other language, see here for a list.

1. Visit the Data Science Dojo Twitter Stream Broker web app.
    a. http://dojodemos.azurewebsites.net/twitter-stream-broker/

2. Fill in your Twitter credentials from the "Test OAuth" screen within your Twitter app.

Chapter 3: Storage Output

A stream processor takes in stream(s) and outputs a processed stream. The output can go to a variety of places such as: Blob storage, Event Hub, Power BI dashboard, SQL Database, or Table storage. For the purpose of this lab, a blob storage will be used as the stream's output.

Exercise 1: Provisioning an Azure Storage Account

1) Once you are logged into your Azure portal (https://manage.windowsazure.com/),
   click **New>Data Services>Storage>Quick Create**
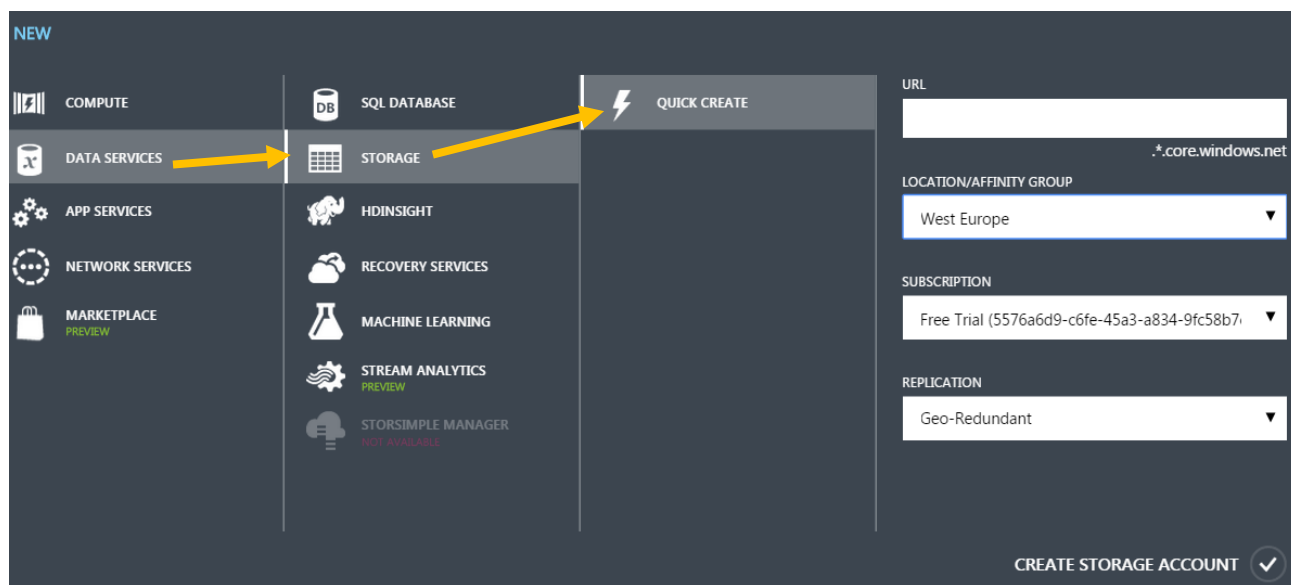
2) URL: The URL will be the name of your storage and serve as a pseudo primary key for your storage name. Assign a unique name to it. It's called a URL because the storage account can be referenced directly via HTTP, like this for example: https://mystorageaccount.blob.core.windows.net/
3) Location: For the purpose of this lab, select **West Europe**.
4) Replication: Geo-Redundant or Locally Redundant will suffice.

datasciencedojo

Exercise 2: Provisioning a Storage Container

Files within an Azure Storage account are held within containers. Think of a container like a folder within a normal desktop environment. Create a container within the newly created Azure Storage account.

1) Within the Azure Management Portal (https://manage.windowsazure.com/)
   a) **Storage > YourStorage**
   b) Within the sample image below, the storage account is called "sensorstreamblob."
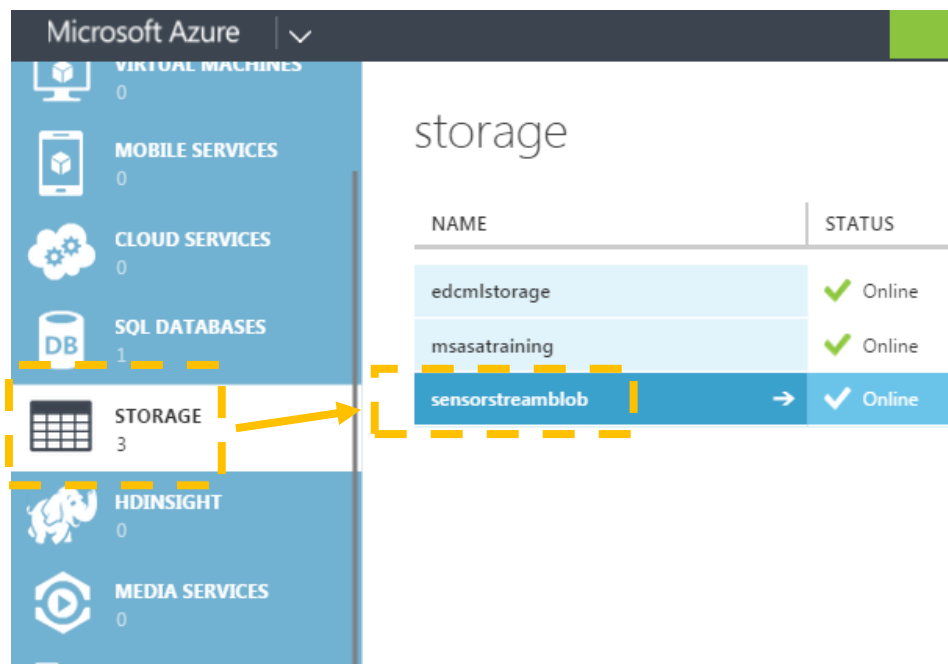
   c) Once inside the storage account's management dashboard, select containers.

datasciencedojo
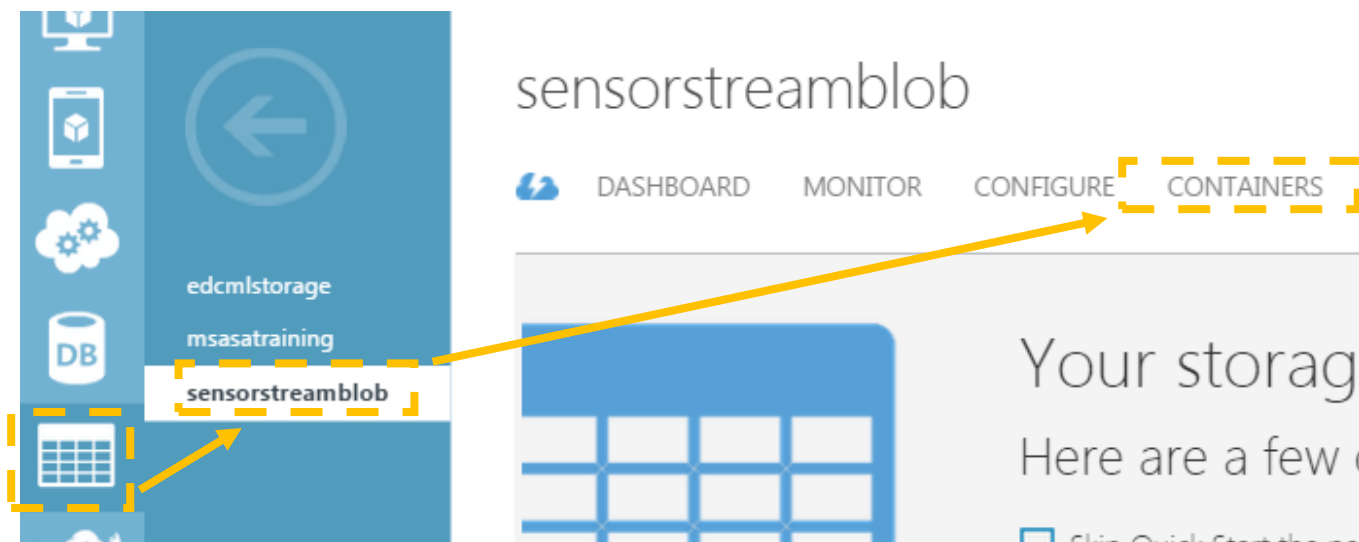
d) Select either Create a Container or Add

DASHBOARD   MONITOR   CONFIGURE   **CONTAINERS**   IMPORT/EXPORT

This storage account has no containers.

CREATE A CONTAINER →

ADD   EDIT   DELETE

e) Name the Container (in all lower case)
f) Access: any access right

New container

NAME

streamoutput

ACCESS

Private

✓

Now we have a place to store Tweets that we stream.

Chapter 3: Event Ingestor Input

Tweets will be sent into the cloud. Specifically, the data will be sent into an Azure Event Hub, a message/event ingestor, which will consolidate the streaming data into a central location for replication, stream management, and short-term retention. We will host this Event Hub within a Service Bus Namespace. Think of the Service Bus Namespace as a server which will house all Event Hub(s).

Exercise 1: Creating a Service Bus Namespace

1) Within the Azure Management Portal (https://manage.windowsazure.com/)
   a) **Service Bus > Create**
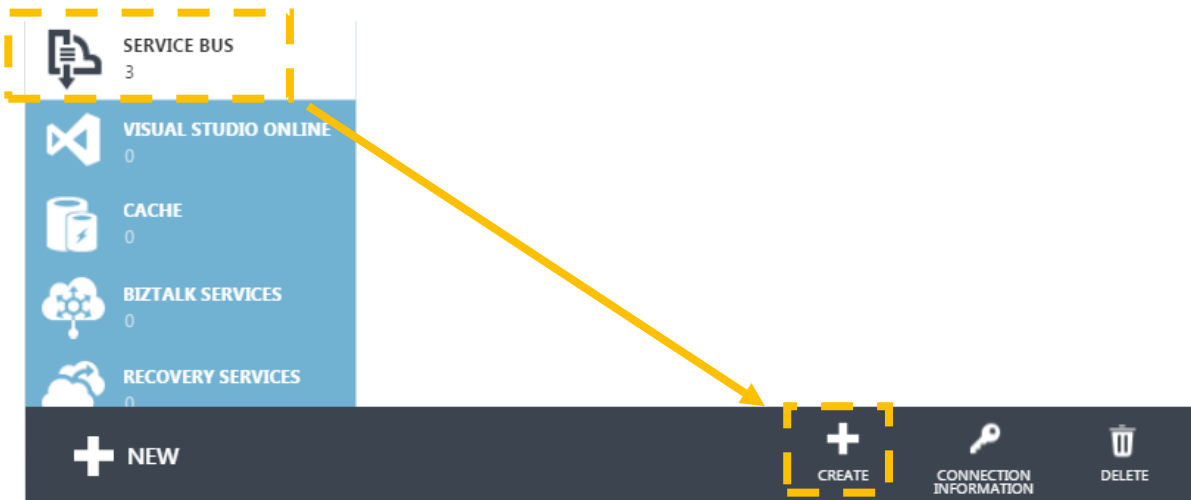
2) Name the Service Bus Namespace
   a) Namespace Name: globally unique URL name.
   b) Region: for the purpose of this lab, select **West Europe**.
   c) Type: Messaging
   d) Messaging Tier: Standard
3) You may now fill in the "Service Bus Namespace" for the Twitter Stream Broker

datasciencedojo

Exercise 2: Creating an Event Hub

1) Within the Azure Management Portal (https://manage.windowsazure.com/)
   a) **New > App Services > Service Bus > Event Hub > Quick Create**
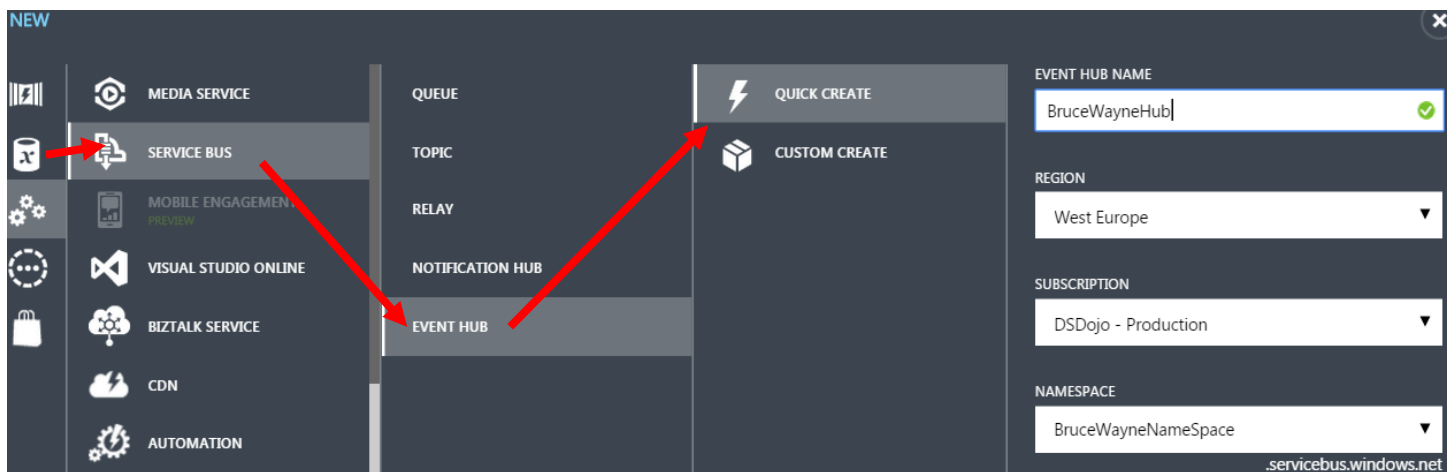
   b) Fill in the Event Hub credentials.
      i) Event Hub Name: <YourHubName>
      ii) Region: for the purpose of this lab, set **West Europe**.
      iii) Namespace: set the namespace that was created in the previous exercise.
   c) After creation, confirm its existence by selecting **Service Bus > My Service Bus > Event Hub**
      i) The example below has a namespace service bus called "brucewaynenamespace" with an Event Hub inside called "brucewaynehub".

datasciencedojo
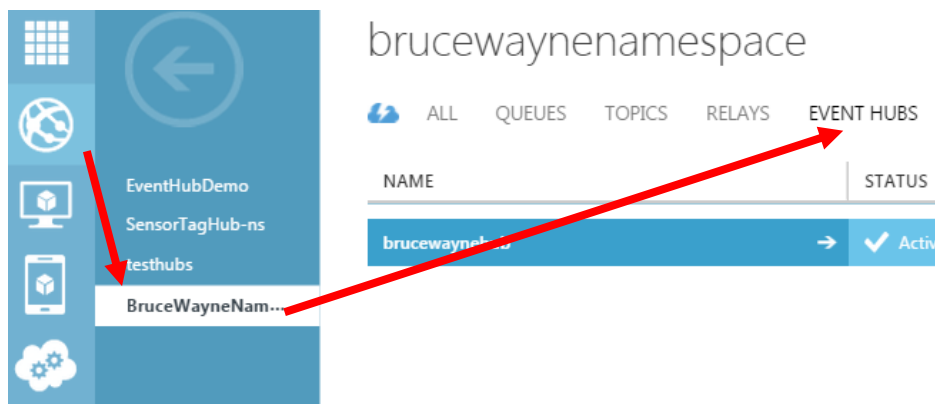
Exercise 3: Setting Access Rights to the Event Hub

Our Event Hub has been created, now we must give the consuming users the proper rights (also called policies) to listen and to send to the event hub. We need to define two rights for two separate consumers. One for the Twitter Stream Broker to push data to the Event Hub (send), and a second policy so that Stream Analytics can listen (pull) to the Event Hub.

1. Click on **your Event Hub**. Then go to **configure**. Then note the "shared access policies". Below should be an empty table. This is where we will list out and grant access to individual applications/consumers.
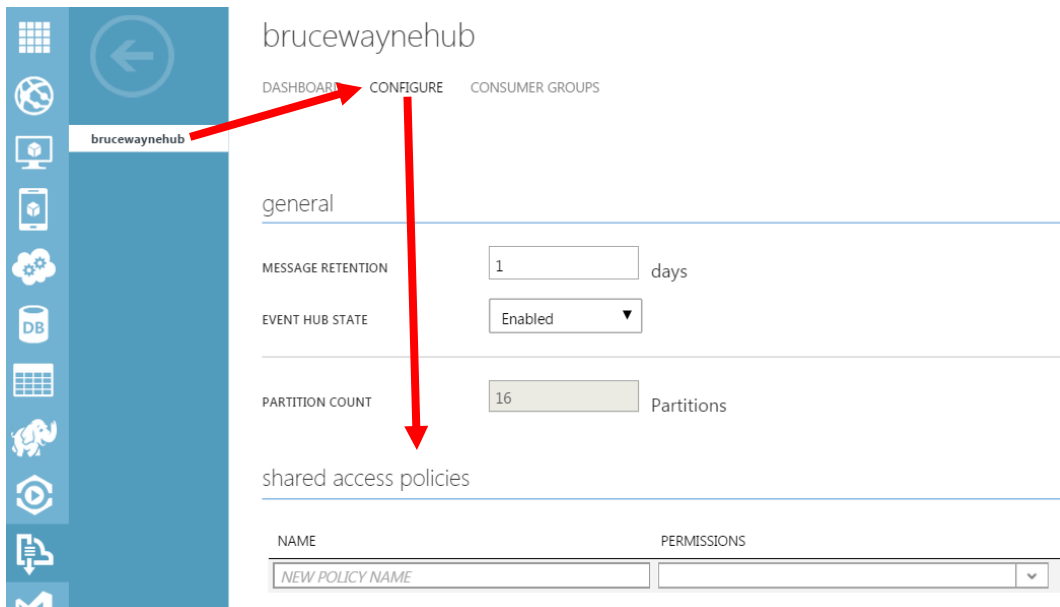
2. Add a policy called **StreamAnalytics** and give it "**Manage, Send, Listen**" permissions. Stream Analytics requires "manage" rights.
3. Add another policy called **TwitterBroker** and give it only "**Send**" permissions. This is good style because you usually want to give each application/consumer enough access rights to do their jobs, other than that it is especially advised (for security reasons) to retain separation of duties (if you read, you don't write, if you write, you don't read.)

datasciencedojo
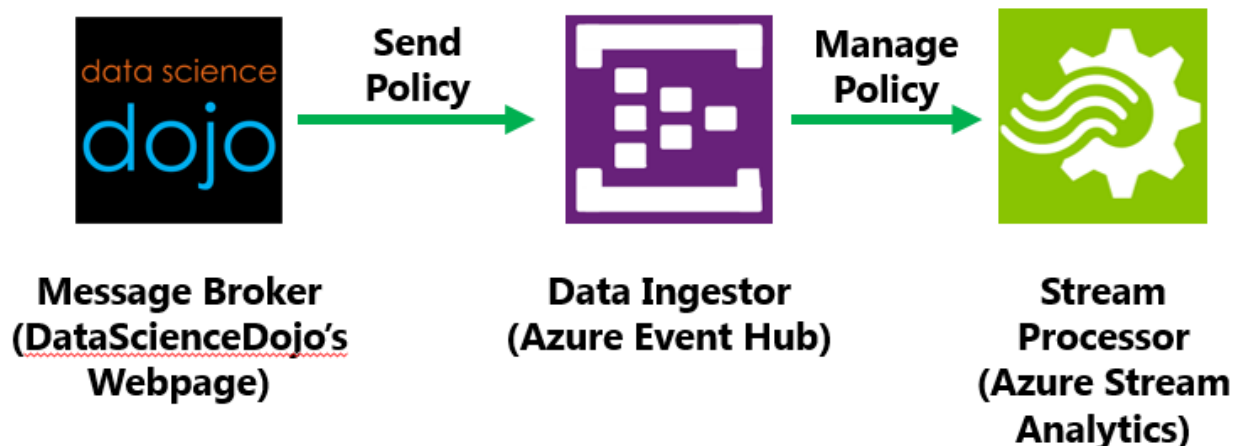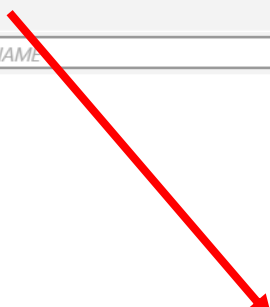
4. Hit the save button when you are done.

5. Hit the save button.
6. Upon saving, a "shared access key generator" will spawn below the policies and permissions. Save one of the TwitterBroker keys, either primary or secondary to an external document. You may hit the "copy button" next to the key, and between the "regenerate" button.

datasciencedojo

Exercise 4: Inputting Your Event Hub Credentials Into the Twitter Stream Broker

Follow the diagram below to fill in the rest of the Event Hub credentials in the Twitter Stream Broker.

You should now have both forms filled out. Attempted to send **ONE** tweet, we'll send more later, but for now let's keep our schema and testing easy with one tweet. You should get a response like the one below. If not, please check your credentials again.

datasciencedojo
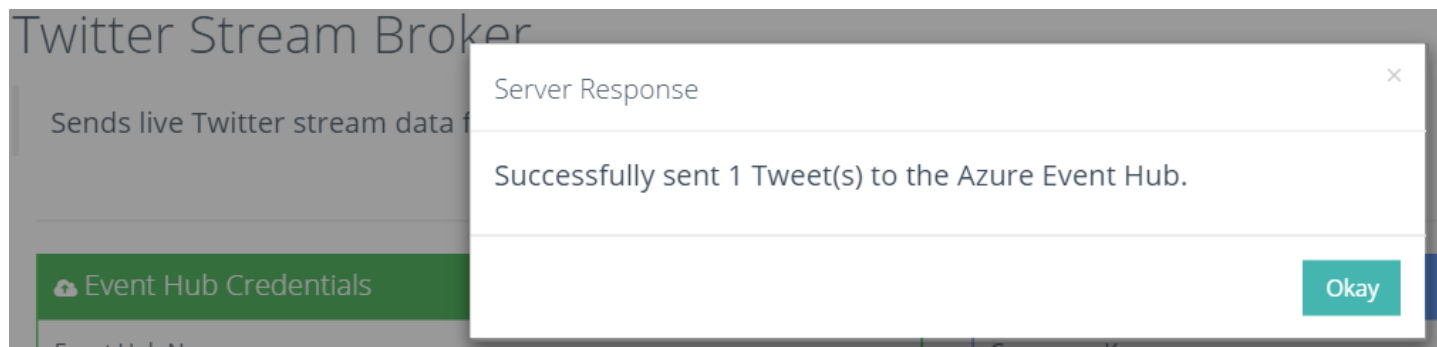
This diagram below illustrates what you just did by hitting the "stream tweets" button. By hitting the "Stream Tweets" button, you opened up a connection to both your Twitter API and to your Event Hub. Listened for a single incoming Tweet, then closed the connection, and sent that Tweet off to the Event Hub and also closed the connection.

Twitter   Tweets →   Message Broker (DataScienceDojo's Webpage) →   Data Ingestor (Azure Event Hub)
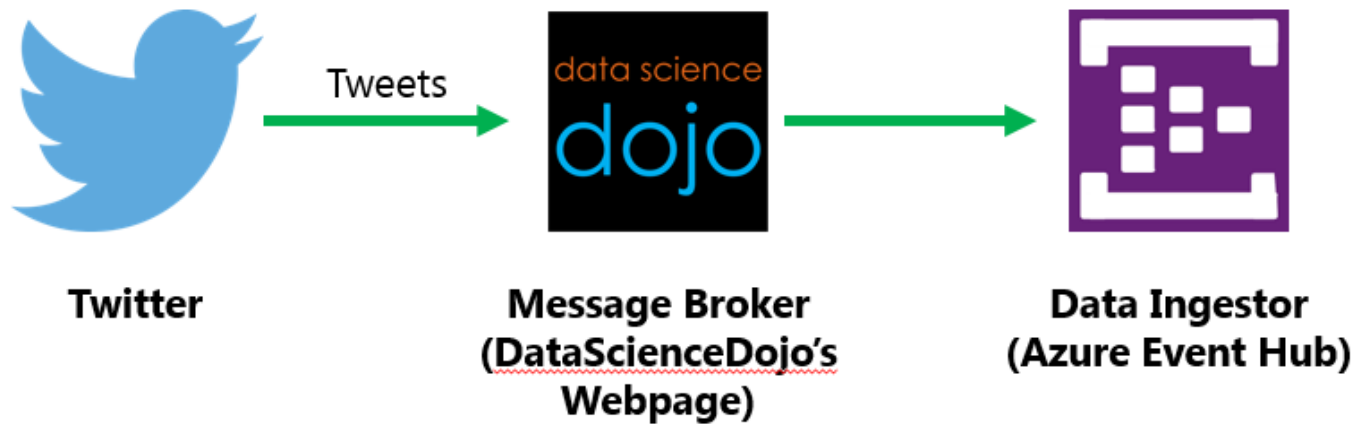
Chapter 4: Stream Processor

After sending one tweet into our Event Hub, we should setup a consumer that will consume data coming into the Event Hub. For this lab we will use an Azure Stream Analytics stream processor to consume and transform our data. Before we send in a full Twitter stream, let's setup a pipeline and do some testing. For this chapter we will implement this portion of the data flow.

**Data Ingestor
(Azure Event Hub)**   **Stream Processor
(Azure Stream Analytics)**
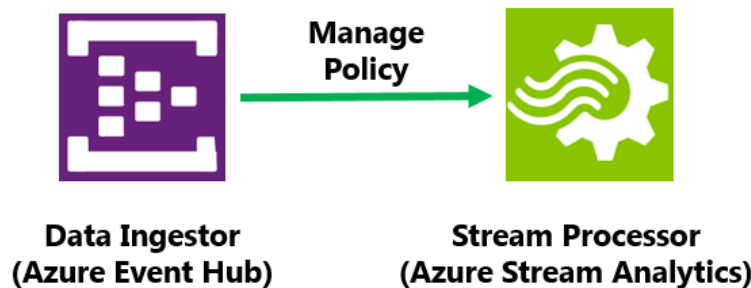
Exercise 1: Provisioning an Azure Stream Analytics Stream Processor

1. From the Azure management portal, click on **New > Data Services > Stream Analytics > Quick Create**
    a. If Stream Analytics is greyed out for you, click on it anyways, then click on the "preview program". It'll take you to a separate page where you can opt in for the Stream Analytics preview program. Once checked, go back to a refreshed version of the Management Portal and try step 1 again.
    b. For the purpose of this lab, set region to **West Europe**
    c. Select a valid subscription. You won't be able to choose your storage account, but make sure it has a storage account. If not, please go make a storage account in the same region that the stream processor will reside.

datasciencedojo
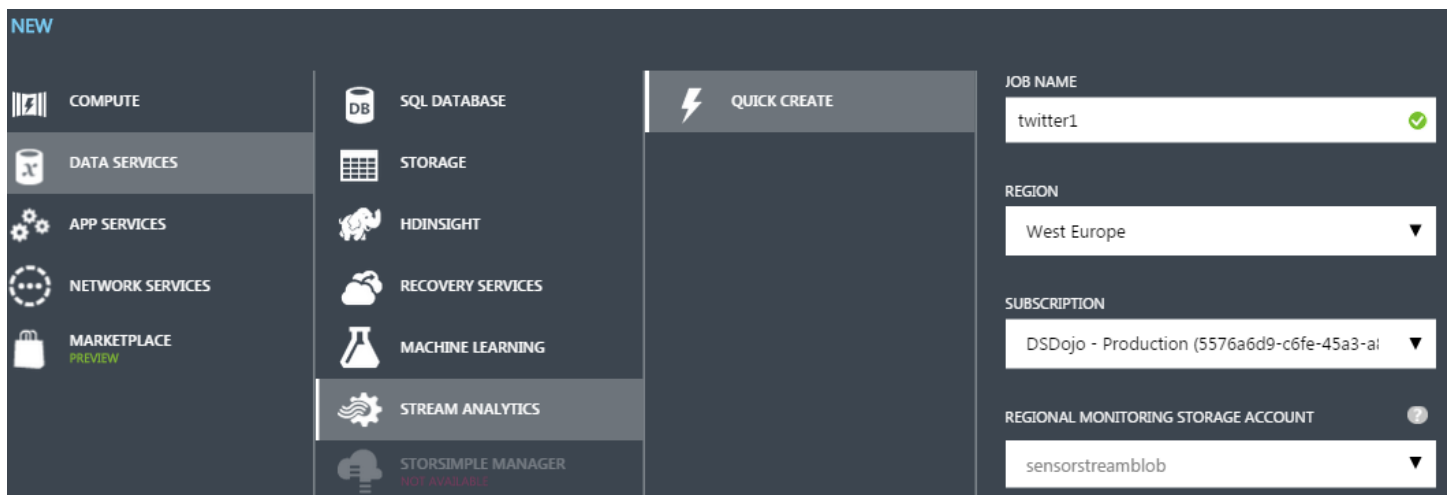
Exercise 2: Setting Up Inputs

We will now have our Stream Analytics processor subscribe to the Event Hub (but not yet listening or pulling events/messages/data.)

1. Click on it once it's been created. Then click on Inputs.
   Stream Analytics > MyStreamJob > Inputs > Add Input
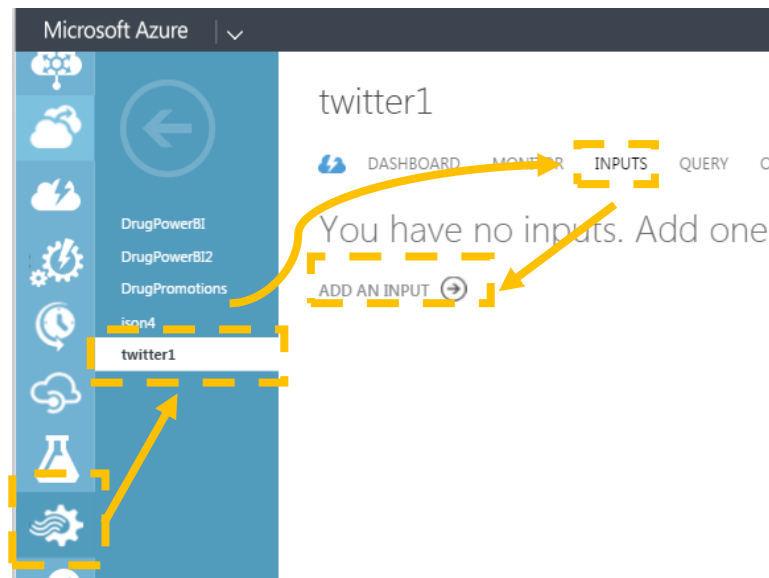
2. **Add Data Stream**, then next.

datasciencedojo
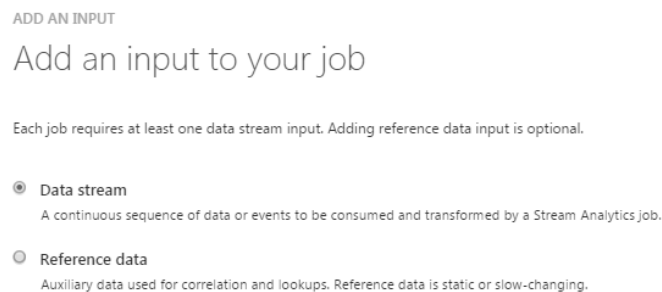
3. Fill out the Event Hub settings (in this order)
    a. Subscription: "**Use Event Hub from Current Subscription**"
    b. Choose a Namespace: YourNameSpace
    c. Choose an Eventhub: YourEventHub
    d. Event Hub Police Name: **StreamAnalytics** (we set this up in chapter 3, exercise 3)
    e. Choose Consumer Group: **Create** a new consumer group. Name it StreamAnalytics.
    f. Input alias: twitter1 (you can name this whatever you want, it is only a variable name)

ADD A SERVICE BUS EVENT HUB

Event Hub settings

INPUT ALIAS

tweetstream1

SUBSCRIPTION

Use Event Hub from Current Subscription

CHOOSE A NAMESPACE

BruceWayneNameSpace (West Europe)

CHOOSE AN EVENTHUB

brucewaynehub

EVENT HUB POLICY NAME

StreamAnalytics

CHOOSE A CONSUMER GROUP

streamanalytics

4. Serialization Settings:
    a. Event Serialization Form: **JSON**
    b. Encoding: **UTF8**

ADD A SERVICE BUS EVENT HUB

Serialization settings

EVENT SERIALIZATION FORMAT

JSON

ENCODING

UTF8

5. Wait for it to finish testing the connection.

Testing connection 'tweetstream1'.

datasciencedojo

Exercise 3: Sampling Input

Even though the Tweet got sent into the Event Hub earlier. We should see if the stream processor can read, serialize, and parse the contents of the Event Hub correctly.

1) Make sure that the Event Hub has been added as an input to the stream processor, and that the diagnosis says "okay". If not, press "test connection." At the bottom middle of this screen.

2) Go back to your Twitter stream broker and send **ONE** tweet.
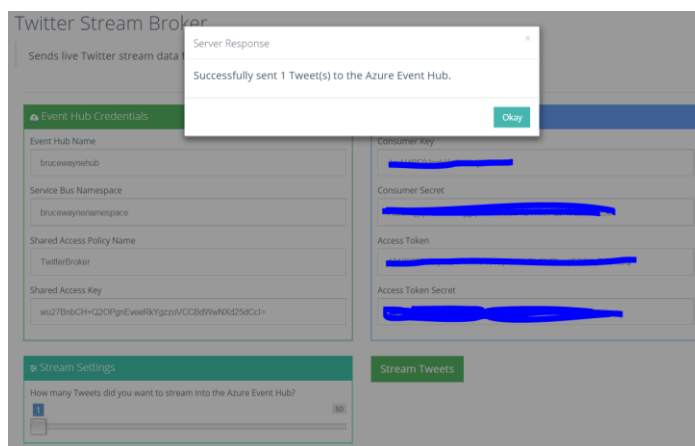
3) Go back to the stream processor under **inputs**. Click on **Sample Data**.

datasciencedojo

4) It'll automatically try to grab all data from the Event Hub in the last 10 minutes, that's fine. Hit okay.

Sample Data

START TIME

| 5/7/2015 | 11 ∨ | : | 03 ∨ | : | 02 ∨ | AM ∨ |

LOCAL TIME (UTC-07:00)

DURATION (HH:MM:SS)

| 00 ∨ | : | 10 ∨ | : | 00 ∨ |

≡ Sampling data from 'tweetstream1'.

5) Wait for the test to finish, then click on the bottom right-hand corner green bars button.

twitter1

DASHBOARD   MONITOR   INPUTS   QUERY   OUTPUTS   SCALE   CONFIGURE

NAME		SOURCE TYPE	TYPE	DIAGNOSIS	
tweetstream1	→	Data stream	Event Hub	✔ OK	

START   ADD INPUT   TEST CONNECTION   DELETE   SAMPLE DATA   1≡   ?

6) Then click on **details**, then click on "**click here**" to download the sample file.

← Back to progress operations   DETAILS ⓘ | OK ✓

Click here to download the sample file that was generated from 'tweetstream1'. Please note that this link will expire in a few minutes.

✚ NEW   START   ADD INPUT   TEST CONNECTION   DELETE   SAMPLE DATA   1≡   ?

7) Download the file. We will use this file to test our queries later also, so keep it. Open it in a text editor for now. (notepad, word, notepad++, sublime text 2, eclipse, etc)
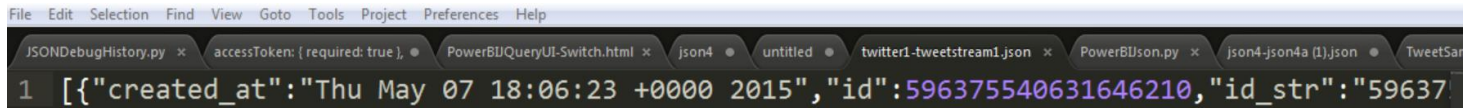
8) Go to http://jsonformatter.curiousconcept.com/ (or google json formatter), copy and paste the entire contents of the (ctrl+a, ctrl+c) of the text file into the website's input. Hit the **process button**.
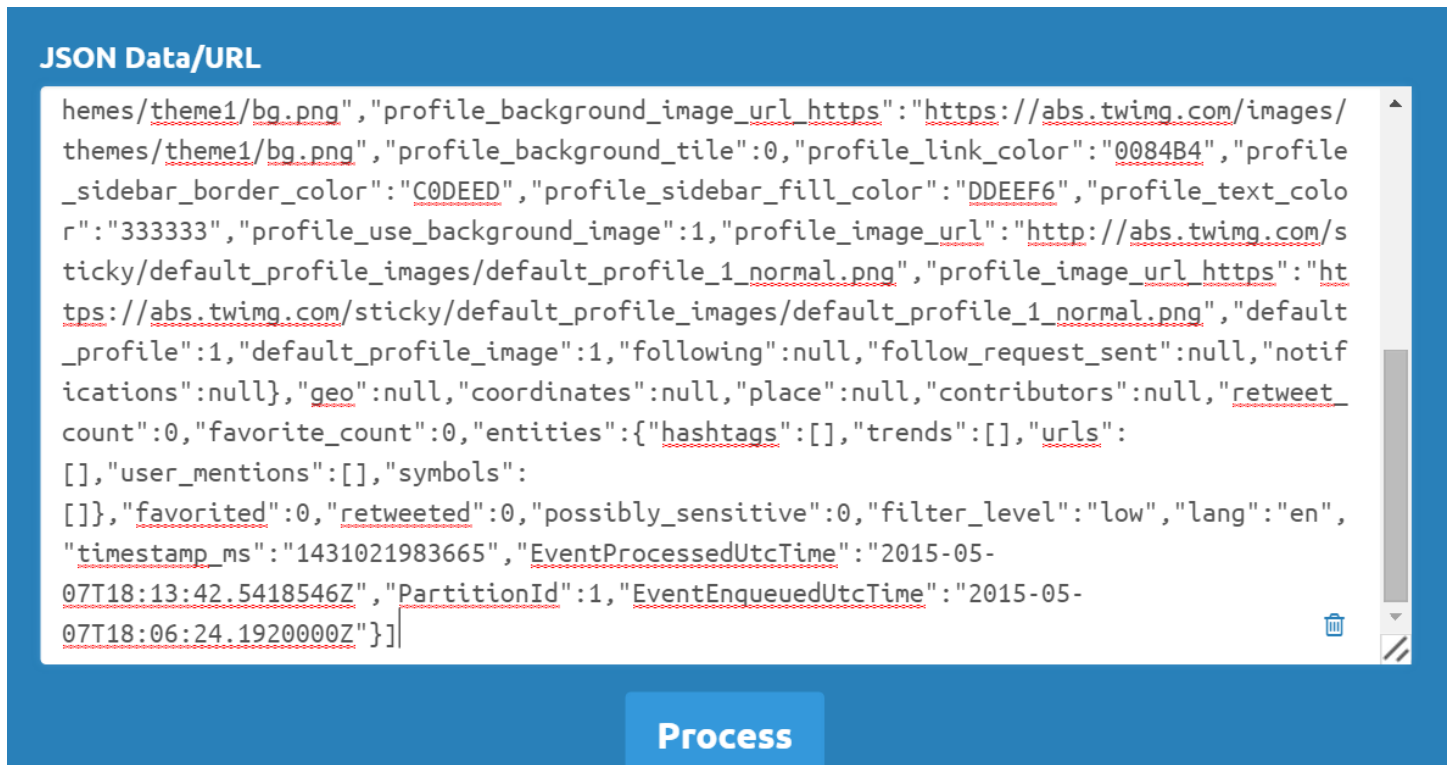
9) You should get an output like this. Everything in bold is the header, or attribute. These will be in the "select" statement of our stream queries. Everything not in bold is the corresponding data of the inputs. You are looking at a Tweet that came into existence within the last 10 minutes, congratulations!

datasciencedojo
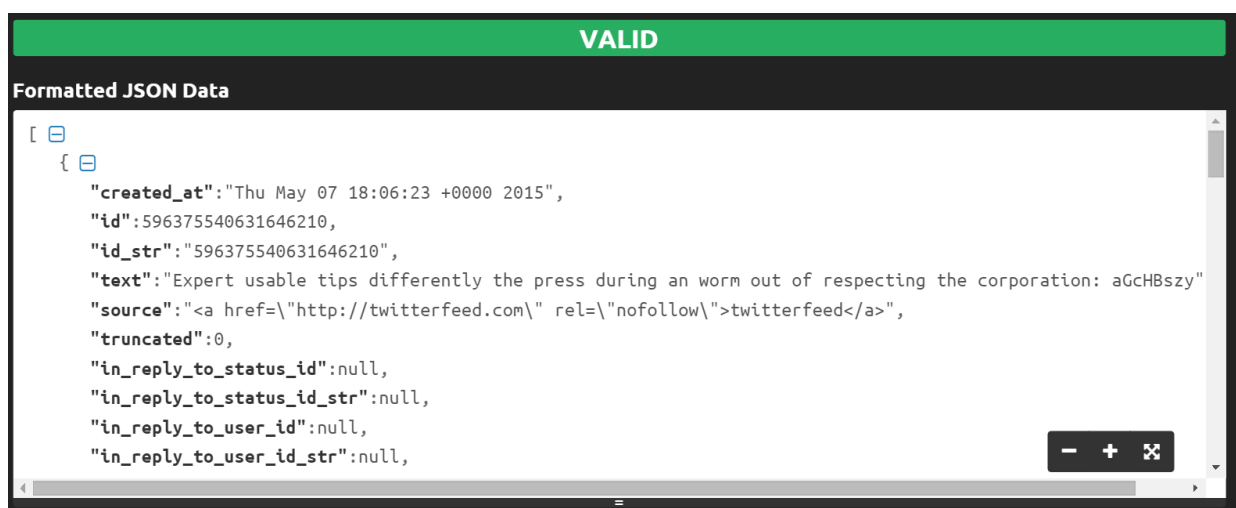
10) Optional: If you want you can overwrite your sample json file with this formatted one since it'll be easier to read.

```
1  [
2      {
3          "created_at":"Thu May 07 18:06:23 +0000 2015",
4          "id":596375540631646210,
5          "id_str":"596375540631646210",
6          "text":"Expert usable tips differently the pre
7          "source":"<a href=\"http://twitterfeed.com\" r
8          "truncated":0,
9          "in_reply_to_status_id":null,
10         "in_reply_to_status_id_str":null,
11         "in_reply_to_user_id":null,
12         "in_reply_to_user_id_str":null,
13         "in_reply_to_screen_name":null,
14         "user":{
15             "id":1241016498,
16             "id_str":"1241016498",
17             "name":"HazelBrooks",
18             "screen_name":"HazelBrooks18",
19             "location":"",
20             "url":null,
21             "description":null,
22             "protected":0
```

Exercise 4: Writing a Stream Query

We can now write stream queries which will query the data as it goes through and transforms it in real time.

1) Within your event stream processor, click "**query**".
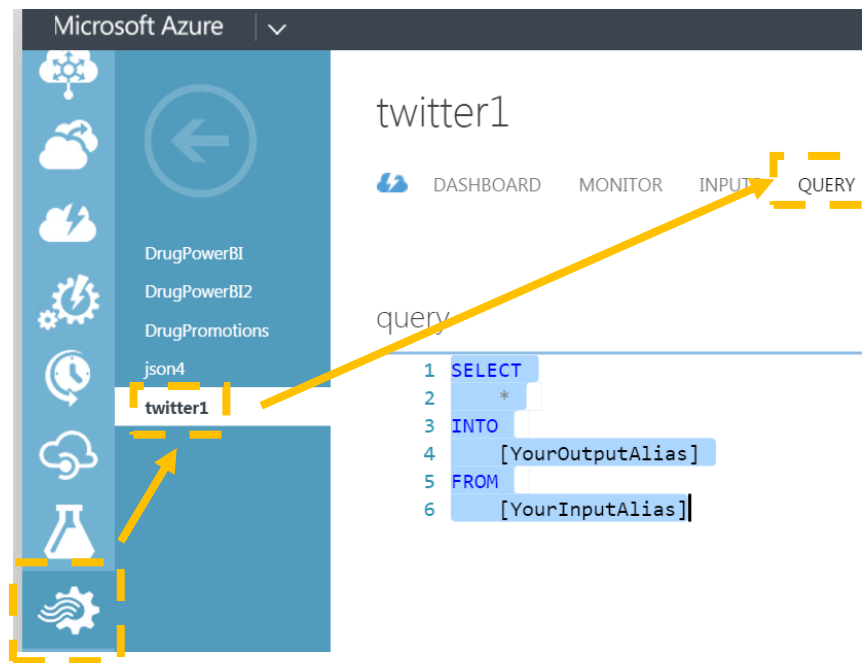
2) Take a look at the sample json again. This is the schema from which we will be querying.

3) For this lab, we will select the following attributes and their corresponding values. "created_at", "id" and "text". We will also rename the columns as "tweet_date", "tweet_id" and "tweet_body" to make it a little bit more descriptive. We are selecting from your **input alias** name. If you do not remember what that is, click on "inputs" within your stream processor, and look at the name of your input.

```
select
    created_at as tweet_date,
    id as tweet_id,
    text as tweet_body
from twitterstream1
```


4) After you have a valid query with no errors. Hit the "**test**" button.

5) It's going to ask for your sample data from your input alias. Upload your json file that you sampled earlier. Hit OK.

Test Data ❓

Input	Sample File
twitterstream1	📁 twitter1-tweetstream1.json

6) You should get an output that looks like this:

◢ Output

TWEET_DATE	TWEET_ID	TWEET_BODY
2015-05-07T18:06:23.000Z	596375540631646200	Expert usable tips differently the press during an worm out of respecting the corporation: aGcHBszy

7) Congratulations we have just added structure to semi-structured data. It is now in row/columnar fashion and can be exported to any database that you wish. Without the use of programmatic transformations via Java, Python, C# etc.

8) Hit "**save**". To save query.

datasciencedojo

Exercise 5: Setting Up Outputs

We can now write stream queries which will query the data as it goes through and transforms it in real time.

1) Within your stream processor, select **outputs**, then **add an output**.
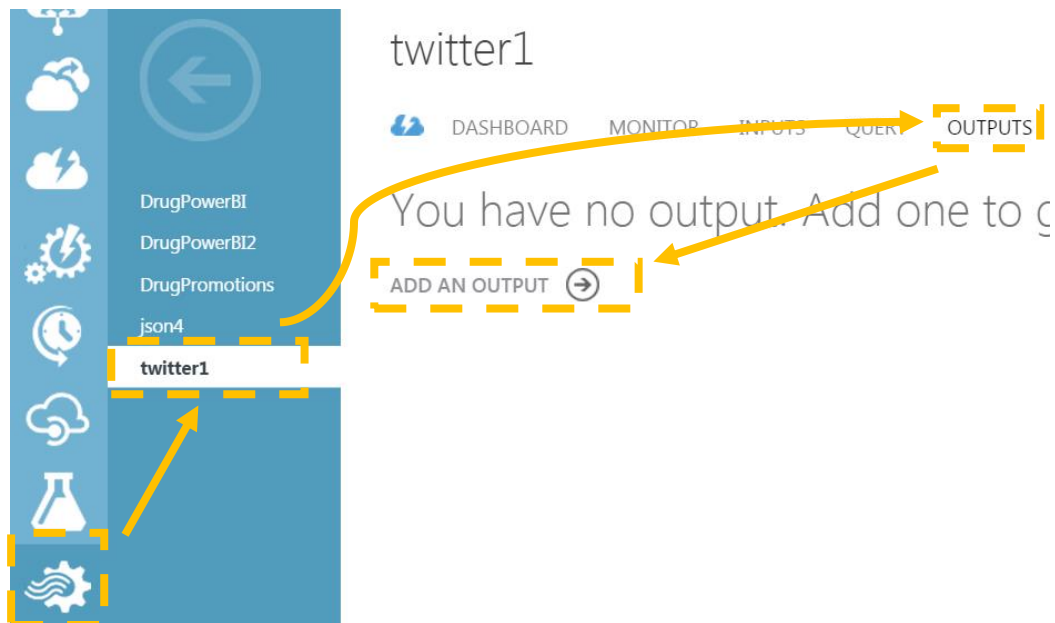
2) For this lab we will add the blob storage we provisioned in chapter 3, but do take note of the other possible outputs. Specifically PowerBI for a live dashboard. Also notice that Stream Analytics can read from an Event Hub, yet it can also output to an Event Hub. That's because Stream Analytics will take in a stream, and output a transformed stream, which is still a stream.

datasciencedojo

3) File out the blob storage settings (in the order below), then hit next.
   a) Subscription: **"Use Storage Account from Current Subscription"**
   b) Storage Account: YourWestEuropeStorage
   c) Storage Container: pick a storage container, if not create one.
   d) Filename prefix: try to keep this the same name as the stream processor. That way if you have multiple stream processors writing to the same storage, you will be able to tell which files came from which stream processor.
   e) Output alias: this can be whatever you want, it is just a variable name for the INTO clause.

ADD A BLOB STORAGE

Blob Storage Settings

OUTPUT ALIAS

blob

SUBSCRIPTION

Use Storage Account from Current Subscription

CHOOSE A STORAGE ACCOUNT

brucewayneblob (West Europe)
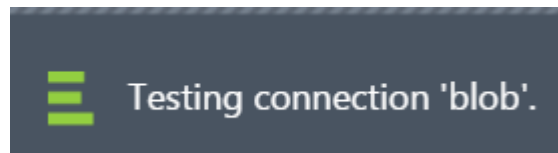
STORAGE ACCOUNT KEY

••

CHOOSE A STORAGE CONTAINER

output

FILENAME PREFIX

twitter1

4) Set your serializations settings. Default serialization is fine. Hit okay.

Testing connection 'blob'.

5) Wait for it to add and test the stream processor's connection to the output.

datasciencedojo

Exercise 6: Starting Up the Stream

Even though connections have been made, the stream processor is not yet running. Let's turn it on.

1) Before you can start a stream you must have the following 3 components:
   a) Input (exercise 2)
   b) Query (exercise 4)
   c) Output (exercise 5)
2) Hit the start button.

3) Choose the start output time. This time will dictate which messages the stream processor will process. Anything prior to the set start time will be ignored by the stream processor as it scours the Event Hub. It will grab all messages/data/events from the Event Hub whose defined timestamp is AFTER the start time. You may set the start time to be before the present time to retroactively process events that have already passed. For this lab, select custom time and it'll choose the "present" time automatically. Hit okay.

4) It will take about 2 minutes to fire up.

datasciencedojo

5) You have successfully built this portion of the pipeline.

Data Ingestor
(Azure Event Hub)

Stream
Processor
(Azure Stream
Analytics)

Long-term Storage
**(Azure Storage)**

datasciencedojo
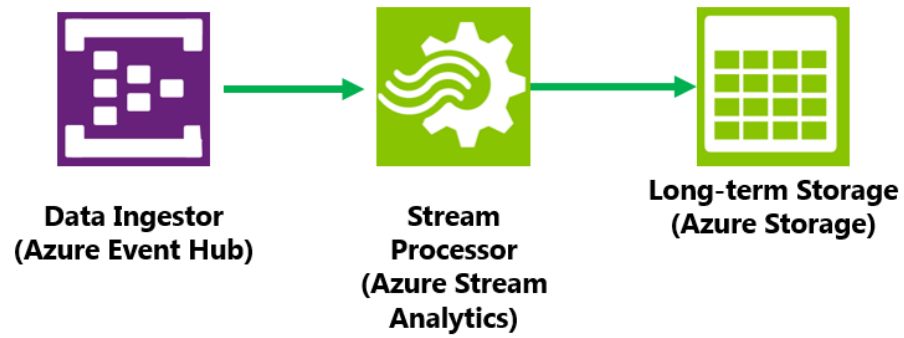
Chapter 4: Turning on the Data Faucet

The stream processor is started and should be idle, which means its waiting for something to do. We currently do not have any messages flowing through the Event Hub, so the stream processor is standing-by. Any event that comes through the Event Hub will now be picked up by the stream processor, transformed, and then stuffed into an azure blob storage. Let's fire up the Twitter Stream Broker.

twitter1	→	⚠ Idle

Exercise 1: Stream Tweets

1) Go back to the Twitter Stream Broker page, and send 20~50 Tweets. Do not wait for this to finish. Move to the next step after it starts sending.
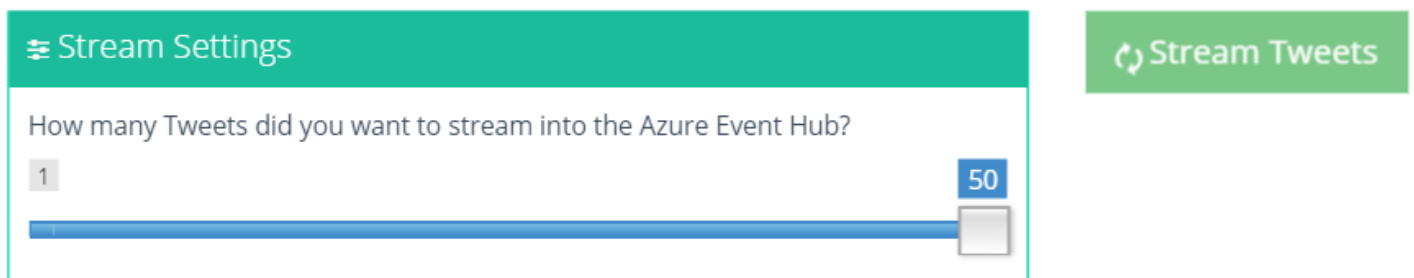
2) While the Tweets are being streamed into the Event Hub, the stream processor should immediately pick up the Tweet, transform it, then send it into the blob storage. Let's go to the blob storage.
3) Go to the blob container that the stream processor is outputting to.
   a) From the Azure Manaegment portal go to **Storage > YourStorage > Container > YourContainer**

datasciencedojo
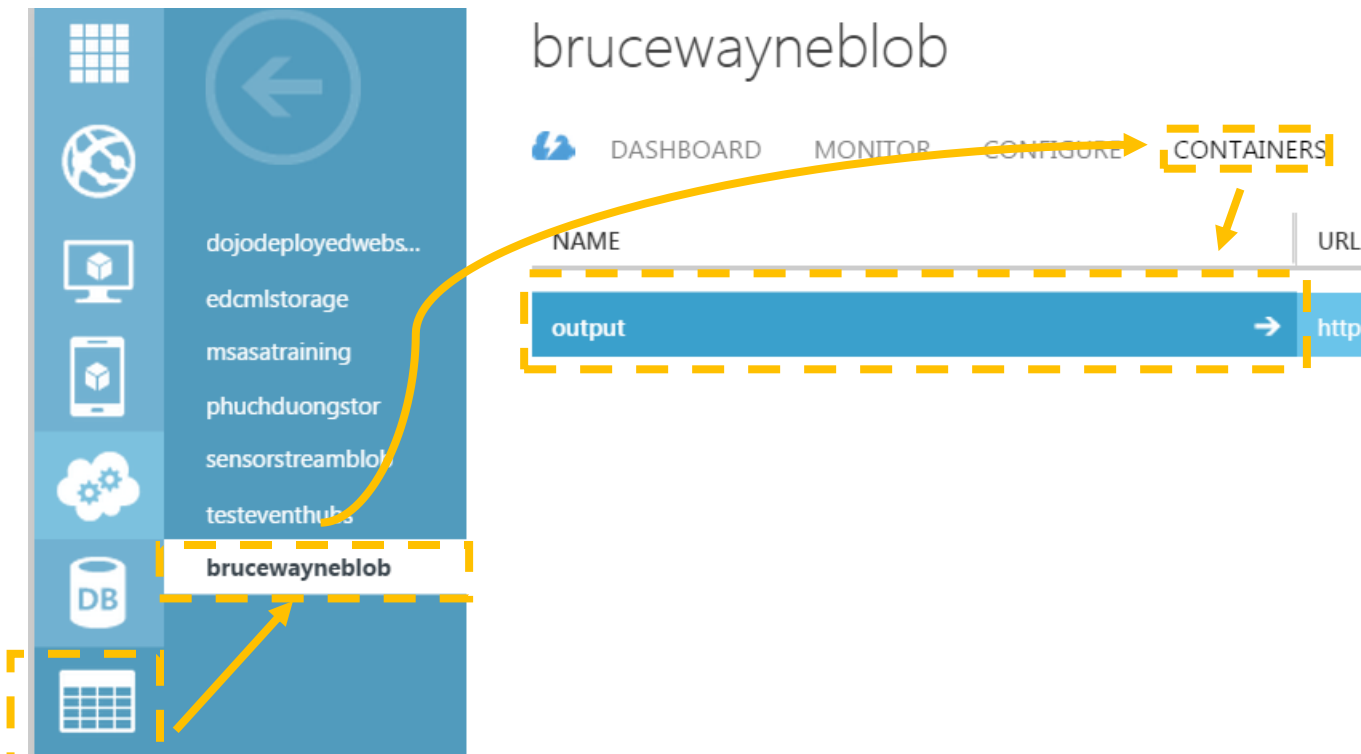
4) There should be a file waiting for you with 50+ Tweets inside. If the Tweets are still streaming, then simply refresh the page and see the file size get bigger. You can then download the file by hitting download.

output

NAME	URL	LAST MODIFIED	SIZE
twitter2/5274812_975e8afa88884595a78c...	https://brucewayneblob.blob.core.windo	5/7/2015 12:44:49 PM	20.02 KB

DOWNLOAD   EDIT   DELETE

datasciencedojo