

AWS re:Invent

Writing JavaScript Applications with the AWS SDK

by Loren Segal

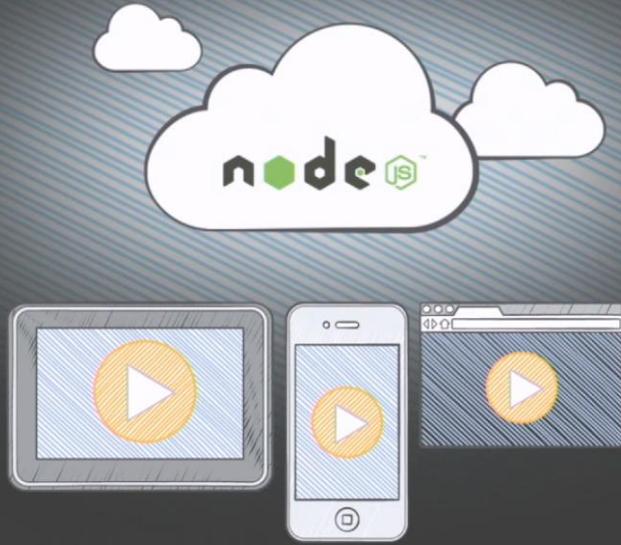
November 15th, 2013



© 2013 Amazon.com, Inc. and its affiliates. All rights reserved. May not be copied, modified, or distributed in whole or in part without the express consent of Amazon.com, Inc.

We give a guided tour of using the AWS SDK for JavaScript to create powerful web applications. Learn the best practices for configuration, credential management, streaming requests, as well as how to use some of the higher level features in the SDK. We also show a live demonstration of using AWS services with the SDK to build a real-world JavaScript application.

JavaScript
is everywhere.



AWS SDK for JavaScript in Node.js

Almost a year old!



Goals

1. Learn about AWS SDK for Node.js
2. Introduce AWS SDK for JavaScript in the Browser
3. Write a two-tiered web application using JavaScript, HTML, and CSS

AWS SDK for Node.js

Full Service Coverage

Support for over 30  services

Extensible Clients

Customize any part of the request cycle

Standard Node.js Idioms

Streams, EventEmitter, Domains

Open Source

Apache License, Version 2.0

<http://github.com/aws/aws-sdk-js>

Getting Started

AWS SDK for Node.js

Installing

Bash

```
$ npm install aws-sdk
```

Loading

JS

```
var AWS = require('aws-sdk');
```

Configuring the SDK

AWS.config

Credentials *
Region *
Extras

* Required by the SDK

Configuring Credentials

IAM roles for EC2 Instances

Environment Variables
File System (outside source control)

You can use one of the above options

Do Not Hardcode
Credentials

Unless they are read-only and scoped to specific resources.

IAM Roles for EC2 Instances

=

Zero Configuration

Environment Variables

AWS_ACCESS_KEY_ID

AWS_SECRET_ACCESS_KEY

AWS_REGION*

If you are going to be using environment variables, these are some of the environment variables that you can use to set up the SDK.

Configuring

Region and Extras

JS

```
AWS.config.update({
  region: 'us-west-2', // AWS_REGION
  maxRetries: 10,      // default: 3
  logger: process.stdout,
  // ... more options ...
});
```

This is how you can configure the SDK programmatically,

Config From a File

JS

```
AWS.config.loadFromPath('./config.json');
```

If this file contains credentials, keep it out of source control

Working with Services

Service Objects

- AWS.S3
- AWS.EC2
- AWS.DynamoDB
- AWS.SQS
- AWS.SNS

...

Constructing a Service Object

JS

```
var ec2 = new AWS.EC2([config]);
```

You can also pass in an optional configuration as above to have that service be confined to that particular configuration object.

Calling an Operation

JS

```
ec2.describeInstances(params, callback);
```

You can call an operation once you have the service object you want to use as above

The Callback

JS

```
function (err, data) { ... }
```

The Callback is a standard NodeJS function with the error and data that needs to be sent back if call is successful.

Getting a Request Object

JS

```
var req = ec2.describeInstances(params);
```

Sending the Request Object

JS

```
var resp = req.send(callback);
```

Adding Listeners to the Request Object

JS

```
req.on('complete', function(resp) { ... });
```

This can be used in an eventing system as shown later

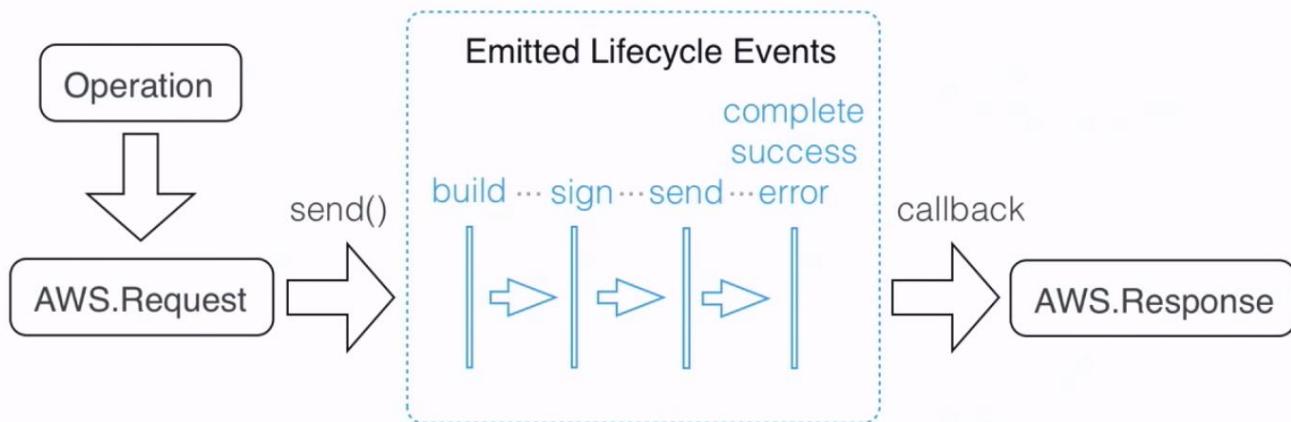
The Request Cycle

The SDK architecture is mainly made of up requests and responses

Send AWS.Request

Get AWS.Response

Request Lifecycle



You call an operation that returns to you an **AWS.Request** object, you can then send the request object to a server. This effectively starts the event lifecycle and events start getting emitted for the stages of progress. If you passed a callback in the request object, it will get triggered at the end of the request lifecycle and returns an **AWS.Response** object.

AWS.Request

`.send(callback)`
`.on(event, callback)`
`.httpRequest`
...

These are some properties on the **AWS.Request** object, the `.httpRequest` helps you see the request headers and so on

AWS.Response

`.error`
`.data`
`.retryCount`
`.httpResponse`
...

Request Lifecycle Recap

Send AWS.Request
Emits Lifecycle Events
Callback with AWS.Response

This is how the Request architecture works in the SDK

Features of the SDK

SDK Features

Global Configuration Object
Bound Parameters
Response Pagination
Event Listeners (Per-Service and Global)
API Version Locking
Secure Credential Management

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> var AWS = require('aws-sdk');
undefined
> process.env.AWS_REGION
'us-east-1'
> AWS.config
{ credentials:
  { expired: false,
    expireTime: null,
    accessKeyId: 'AKIAIDH30YK74N5KSKNA',
    sessionToken: undefined,
    envPrefix: 'AWS' },
  credentialProvider:
  { providers:
    [ [Function],
      [Function],
      [Function] ] },
  region: 'us-east-1',
  logger: null,
  apiVersions: {},
  apiVersion: null,
  endpoint: undefined,
  httpOptions: {},
  maxRetries: undefined,
```

Log in to our terminal and start NodeJS

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
expireTime: null,
accessKeyId: 'AKIAIDH30YK74N5KSKNA',
sessionToken: undefined,
envPrefix: 'AWS' },
credentialProvider:
{ providers:
  [ [Function],
    [Function],
    [Function] ] },
region: 'us-east-1',
logger: null,
apiVersions: {},
apiVersion: null,
endpoint: undefined,
httpOptions: {},
maxRetries: undefined,
maxRedirects: 10,
paramValidation: true,
sslEnabled: true,
s3ForcePathStyle: false,
computeChecksums: true,
dynamoDbCrc32: true }
```

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> AWS.config.logger = process.stdout;
```

We then set up our *logger* and set it to *stdout* in this case

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
sync: false,
bufferProcessing: false,
onwrite: [Function],
writecb: null,
writelen: 0,
buffer: [] },
writable: true,
allowHalfOpen: false,
onend: null,
destroyed: false,
errorEmitted: false,
bytesRead: 0,
_bytesDispatched: 937,
_pendingData: null,
_pendingEncoding: '',
columns: 79,
rows: 23,
_type: 'tty',
fd: 1,
_isStdio: true,
destroySoon: [Function],
destroy: [Function] }
>
```



```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> s3 = new AWS.S3
```

Next, we create an S3 object,

```
Terminal Shell Edit View Window Help Fri 10:26 AM Isegal — node — 79x23
_isStdio: true,
destroySoon: [Function],
destroy: [Function] },
apiVersions: {},
apiVersion: null,
endpoint: undefined,
httpOptions: {},
maxRetries: undefined,
maxRedirects: 10,
paramValidation: true,
sslEnabled: true,
s3ForcePathStyle: false,
computeChecksums: true,
dynamoDbCrc32: true },
endpoint:
{ protocol: 'https:',
host: 's3.amazonaws.com',
port: 443,
hostname: 's3.amazonaws.com',
pathname: '/',
path: '/',
href: 'https://s3.amazonaws.com/' } }
> 
```

We now have an S3 object

```
Terminal Shell Edit View Window Help Fri 10:27 AM Isegal — node — 79x23
> resp = s3.listBuckets().send(); null
null
> [AWS s3 200 1.329s 0 retries] listBuckets({})
> 
```

We then send a simple **listBuckets** request to the server as above. There is no callback here and we have used a null to limit the response displayed

```
> resp = s3.listBuckets().send(); null
null
> [AWS s3 200 1.329s 0 retries] listBuckets({})
resp.data
{ Buckets:
[ { Name: 'aws-reinvent-2013-test-bucket0',
CreationDate: Thu Nov 14 2013 11:29:14 GMT-0800 (PST) },
{ Name: 'aws-reinvent-2013-test-bucket1',
CreationDate: Thu Nov 14 2013 11:29:15 GMT-0800 (PST) },
{ Name: 'aws-reinvent-2013-test-bucket10',
CreationDate: Thu Nov 14 2013 11:29:29 GMT-0800 (PST) }, 
```

We now can do a **resp.data** to see the data returned from the **listBuckets** command

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
{
  Name: 'aws-sdk-watchdog',
  CreationDate: Tue Nov 05 2013 17:57:44 GMT-0800 (PST) },
  { Name: 'beanstalk-assets',
  CreationDate: Tue Nov 05 2013 18:01:30 GMT-0800 (PST) },
  { Name: 'elasticbeanstalk-sa-east-1-579606425216',
  CreationDate: Tue Nov 05 2013 18:46:51 GMT-0800 (PST) },
  { Name: 'elasticbeanstalk-us-east-1-579606425216',
  CreationDate: Tue Nov 05 2013 18:49:04 GMT-0800 (PST) },
  { Name: 'elasticbeanstalk-us-west-2-579606425216',
  CreationDate: Tue Nov 05 2013 18:51:12 GMT-0800 (PST) },
  { Name: 'lorenfoo',
  CreationDate: Tue Nov 05 2013 19:39:28 GMT-0800 (PST) },
  { Name: 'reinvent2013-blog-demo',
  CreationDate: Thu Nov 14 2013 17:20:14 GMT-0800 (PST) },
  { Name: 'reinventfoo',
  CreationDate: Thu Nov 14 2013 12:36:00 GMT-0800 (PST) },
  { Name: 'ruby-beanstalk',
  CreationDate: Wed Oct 24 2012 16:46:36 GMT-0700 (PDT) } ],
Owner:
{ ID: '6a85738073641be06bba5c958299d11ff9fdc99721f8e2225889650c54996cee',
  DisplayName: 'lsegalaws' },
RequestId: '638A5BC91CDF10A7'
```

> █

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> res.error
null
> █
```

This is how we can use the **SDK** at a basic level.

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> s3bucket = new AWS.S3({params: {Bucket: 'reinventfoo'}})
> █
```

When you are working with services like **S3**, you are often sending a lot of the same **parameters** to the same thing like a bucket. The SDK allows you to use the concept of **bound parameters** to send commonly used data once.

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
    destroySoon: [Function],
    destroy: [Function] },
  apiVersions: {},
  apiVersion: null,
  endpoint: undefined,
  httpOptions: {},
  maxRetries: undefined,
  maxRedirects: 10,
  paramValidation: true,
  sslEnabled: true,
  s3ForcePathStyle: false,
  computeChecksums: true,
  dynamoDbCrc32: true,
  params: { Bucket: 'reinventfoo' } },
endpoint:
{ protocol: 'https:',
  host: 's3.amazonaws.com',
  port: 443,
  hostname: 's3.amazonaws.com',
  pathname: '/',
  path: '/',
  href: 'https://s3.amazonaws.com/' } }
> 
```

We now have a bucket created with some bounded parameters to it.

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> resp = s3bucket.listObjects().send(); null
null
> [AWS s3 200 0.437s 0 retries] listObjects({ Bucket: 'reinventfoo' })
resp.data
```

We can now just call the method against the resource we want as above without passing the parameters along anymore

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
Fri 10:28 AM
> [AWS s3 200 0.437s 0 retries] listObjects({ Bucket: 'reinventfoo' })
resp.data
{ Contents:
  [ { Key: 'key',
      LastModified: Thu Nov 14 2013 12:36:35 GMT-0800 (PST),
      ETag: '"acbd18db4cc2f85cedef654fcc4a4d8"',
      Size: 3,
      Owner: [Object],
      StorageClass: 'STANDARD' },
    { Key: 'key.txt',
      LastModified: Thu Nov 14 2013 23:40:42 GMT-0800 (PST),
      ETag: '"5d41402abc4b2a76b9719d911017c592"',
      Size: 5,
      Owner: [Object],
      StorageClass: 'STANDARD' } ],
  CommonPrefixes: [],
  Name: 'reinventfoo',
  Prefix: null,
  Marker: null,
  MaxKeys: 1000,
  IsTruncated: false,
  RequestId: 'A99B7B5D610421DF'
}
> 
```

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
Fri 10:29 AM
> resp = s3bucket.getObject({Key: 'key.txt'}).send(); null
null
> [AWS s3 200 0.793s 0 retries] getObject({ Key: 'key.txt', Bucket: 'reinventfo
o' })
> resp.data.Body.toString()
'hello'
> 
```

You can bind parameters to any service object available in the SDK

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
Fri 10:30 AM
> table = new AWS.DynamoDB({params: {TableName: 'sessions'}}); null
null
> 
```

Let us see response pagination feature SDK feature. This is useful when your request to services like S3 or DynamoDB is large and splitted over multiple parts or calls so you have to page repeatedly, the SDK simplifies this for you automatically. We have created a DynamoDB table object above to demonstrate this feature

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
Fri 10:31 AM
> table = new AWS.DynamoDB({params: {TableName: 'sessions'}}); null
null
> resp = table.scan({Limit: 10}).send(); null
null
> [AWS dynamodb 200 0.518s 0 retries] scan({ Limit: 10, TableName: 'sessions' })
)
resp.data
{ Count: 10,
  Items:
  [ { session_id: [Object], value: [Object] },
    { session_id: [Object], value: [Object] } ],
  LastEvaluatedKey: { session_id: { S: 'hello15' } },
  ScannedCount: 10
}
>
```

Sometimes there is a lot more data that we want back, so the SDK is going to help us out

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
Fri 10:31 AM
> resp.hasNextPage()
true
>
```

The **SDK** has a method on the response object called ***resp.hasNextPage()***, the above case shows that there are more pages to get for the call we made for the response object

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
Fri 10:31 AM
> resp.hasNextPage()
true
> resp = resp.nextPage().send(); null
null
> [AWS dynamodb 200 0.342s 0 retries] scan({ Limit: 10,
  TableName: 'sessions',
  ExclusiveStartKey: { session_id: [Object] } })
resp.data
{ Count: 10,
  Items:
    [ { session_id: [Object], value: [Object] },
      { session_id: [Object], value: [Object] } ],
  LastEvaluatedKey: { session_id: { S: 'hello48' } },
  ScannedCount: 10 }
>
```

This will submit a new request for the next page of results

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
Fri 10:31 AM
> resp = resp.nextPage().send(); null
null
> [AWS dynamodb 200 0.395s 0 retries] scan({ Limit: 10,
  TableName: 'sessions',
  ExclusiveStartKey: { session_id: [Object] } })
resp.data
{ Count: 10,
  Items:
    [ { session_id: [Object], value: [Object] },
      { session_id: [Object], value: [Object] } ],
  LastEvaluatedKey: { session_id: { S: 'hello19' } },
  ScannedCount: 10 }
>
```

We can continue to use the `resp.hasNextPage()` method on our own to get all the response data but the SDK has a way to help us get all the data also as shown below

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> var page = 1;
undefined
> table.scan({Limit: 10}).eachPage(function (err, data) {
... if (data) {
..... console.log("Page " + page++)
..... console.log(data.Items.length)
..... }
...}); null
```

The **eachPage()** does the **hasNextPage()** method call on our behalf to get all the data in one go.

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> var page = 1;
undefined
> table.scan({Limit: 10}).eachPage(function (err, data) {
... if (data) {
..... console.log("Page " + page++)
..... console.log(data.Items.length)
..... }
...}); null
null
> [AWS dynamodb 200 0.447s 0 retries] scan({ Limit: 10, TableName: 'sessions' })
)
Page 1
10
[AWS dynamodb 200 0.341s 0 retries] scan({ Limit: 10,
  TableName: 'sessions',
  ExclusiveStartKey: { session_id: [Object] } })
```

Page 2
10

```
Terminal Shell Edit View Window Help
Fri 10:33 AM
lsegal — node — 79x23
Page 7
10
[AWS dynamodb 200 0.348s 0 retries] scan({ Limit: 10,
  TableName: 'sessions',
  ExclusiveStartKey: { session_id: [Object] } })
Page 8
10
[AWS dynamodb 200 0.368s 0 retries] scan({ Limit: 10,
  TableName: 'sessions',
  ExclusiveStartKey: { session_id: [Object] } })
Page 9
10
[AWS dynamodb 200 0.356s 0 retries] scan({ Limit: 10,
  TableName: 'sessions',
  ExclusiveStartKey: { session_id: [Object] } })
Page 10
10
[AWS dynamodb 200 0.344s 0 retries] scan({ Limit: 10,
  TableName: 'sessions',
  ExclusiveStartKey: { session_id: [Object] } })
Page 11
0
```

It **scanned** and ran through all the available for us automatically.

```
Terminal Shell Edit View Window Help
Fri 10:34 AM
lsegal — node — 79x23
> req = s3bucket.listObjects();
```

```
Terminal Shell Edit View Window Help Isegal — node — 79x23
> 
extractData:
  [ [Function: SETUP_DATA],
  [Function: extractData],
  [Function: extractData] ],
send: [ [Function: SEND] ],
httpHeaders: [ [Function: HTTP_HEADERS] ],
httpData: [ [Function: HTTP_DATA] ],
httpDone: [ [Function: HTTP_DONE] ],
httpError: [ [Function: HTTP_ERROR] ],
retry:
  [ [Function: FINALIZE_ERROR],
  [Function: INVALIDATE_CREDENTIALS],
  [Function: REDIRECT],
  [Function: RETRY_CHECK],
  [Object],
  [Object] ],
build:
  [ [Function: buildRequest],
  [Function: addContentType],
  [Function: populateURI],
  [Function: computeContentMd5] ],
complete: [ [Function: LOG_REQUEST] ] } }
```

```
> 
Terminal Shell Edit View Window Help Isegal — node — 79x23
> req.on('success', function (resp) { console.log(resp.data) })
```

```
Terminal Shell Edit View Window Help Isegal — node — 79x23
> 
[ [Function: SETUP_DATA],
  [Function: extractData],
  [Function: extractData] ],
send: [ [Function: SEND] ],
httpHeaders: [ [Function: HTTP_HEADERS] ],
httpData: [ [Function: HTTP_DATA] ],
httpDone: [ [Function: HTTP_DONE] ],
httpError: [ [Function: HTTP_ERROR] ],
retry:
  [ [Function: FINALIZE_ERROR],
  [Function: INVALIDATE_CREDENTIALS],
  [Function: REDIRECT],
  [Function: RETRY_CHECK],
  [Object],
  [Object] ],
build:
  [ [Function: buildRequest],
  [Function: addContentType],
  [Function: populateURI],
  [Function: computeContentMd5] ],
complete: [ [Function: LOG_REQUEST] ],
success: [ [Function] ] }
```

```
Terminal Shell Edit View Window Help Fri 10:34 AM ⓘ Isegal — node — 79x23
> req.send(); null
null
>
```

```
Terminal Shell Edit View Window Help Fri 10:34 AM ⓘ Isegal — node — 79x23
null
> { Contents:
  [ { Key: 'key',
      LastModified: Thu Nov 14 2013 12:36:35 GMT-0800 (PST),
      ETag: '"acbd18db4cc2f85cedef654fccc4a4d8"',
      Size: 3,
      Owner: [Object],
      StorageClass: 'STANDARD' },
    { Key: 'key.txt',
      LastModified: Thu Nov 14 2013 23:40:42 GMT-0800 (PST),
      ETag: '"5d41402abc4b2a76b9719d911017c592"',
      Size: 5,
      Owner: [Object],
      StorageClass: 'STANDARD' } ],
  CommonPrefixes: [],
  Name: 'reinventfoo',
  Prefix: null,
  Marker: null,
  MaxKeys: 1000,
  IsTruncated: false,
  RequestId: '4F762B2F3B9F9A7A' }
[AWS s3 200 37.64s 0 retries] listObjects({ Bucket: 'reinventfoo' })
```

```
Terminal Shell Edit View Window Help Fri 10:35 AM ⓘ Isegal — node — 79x23
AWS.events
{ domain: null, _events: {} }
>
```

The SDK also has a global events object that is used every time you create a new service object.

```
Terminal Shell Edit View Window Help Fri 10:35 AM ⓘ Isegal — node — 79x23
AWS.events
{ domain: null, _events: {} }
> AWS.events.on('success', function (resp) { console.log(resp.data) })
{ domain: null,
  _events: { success: [ [Function] ] } }
>
```

We can now attach a success event to the global object as above.

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
AWS.events
{ domain: null, _events: {} }
> AWS.events.on('success', function (resp) { console.log(resp.data) })
{ domain: null,
  _events: { success: [ [Function] ] } }
> s3 = new AWS.S3();null
null
> s3.listBuckets().send(); null
null
>
```

We then recreate the S3 object too

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
  CreationDate: Tue Nov 05 2013 17:57:44 GMT-0800 (PST) },
  { Name: 'beanstalk-assets',
    CreationDate: Tue Nov 05 2013 18:01:30 GMT-0800 (PST) },
  { Name: 'elasticbeanstalk-sa-east-1-579606425216',
    CreationDate: Tue Nov 05 2013 18:46:51 GMT-0800 (PST) },
  { Name: 'elasticbeanstalk-us-east-1-579606425216',
    CreationDate: Tue Nov 05 2013 18:49:04 GMT-0800 (PST) },
  { Name: 'elasticbeanstalk-us-west-2-579606425216',
    CreationDate: Tue Nov 05 2013 18:51:12 GMT-0800 (PST) },
  { Name: 'lorenfoo',
    CreationDate: Tue Nov 05 2013 19:39:28 GMT-0800 (PST) },
  { Name: 'reinvent2013-blog-demo',
    CreationDate: Thu Nov 14 2013 17:20:14 GMT-0800 (PST) },
  { Name: 'reinventfoo',
    CreationDate: Thu Nov 14 2013 12:36:00 GMT-0800 (PST) },
  { Name: 'ruby-beanstalk',
    CreationDate: Wed Oct 24 2012 16:46:36 GMT-0700 (PDT) } ],
Owner:
{ ID: '6a85738073641be06bba5c958299d11ff9fdc99721f8e2225889650c54996cee',
  DisplayName: 'lsegala' },
RequestId: '135CBB67886B30C7'
[AWS s3 200 0.752s 0 retries] listBuckets({})
```

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> AWS.events.removeAllListeners('success')
{ domain: null, _events: {} }
>
```

This clears the events object as above.

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> AWS.config.credentials = new AWS.TemporaryCredentials
```

Next, let us see how to manage your credentials properly in your application using some new helpers added to the SDK. The AWS.TemporaryCredentials() method fetches temporary credentials from IAM and STS that will eventually expire after a short time period.

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
region: 'us-east-1',
logger: [Object],
apiVersions: {},
apiVersion: null,
endpoint: undefined,
httpOptions: {},
maxRetries: undefined,
maxRedirects: 10,
paramValidation: true,
sslEnabled: true,
s3ForcePathStyle: false,
computeChecksums: true,
dynamoDbCrc32: true },
endpoint:
{ protocol: 'https:',
host: 'sts.amazonaws.com',
port: 443,
hostname: 'sts.amazonaws.com',
pathname: '/',
path: '/',
href: 'https://sts.amazonaws.com/' } },
params: {} }
>
```

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> s3 = new AWS.S3
```

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
_isStdio: true,
destroySoon: [Function],
destroy: [Function] },
apiVersions: {},
apiVersion: null,
endpoint: undefined,
httpOptions: {},
maxRetries: undefined,
maxRedirects: 10,
paramValidation: true,
sslEnabled: true,
s3ForcePathStyle: false,
computeChecksums: true,
dynamoDbCrc32: true },
endpoint:
{ protocol: 'https:',
host: 's3.amazonaws.com',
port: 443,
hostname: 's3.amazonaws.com',
pathname: '/',
path: '/',
href: 'https://s3.amazonaws.com/' } }
> 
```

We now have a newly created S3 object that has that temporary credentials set up

```
Terminal Shell Edit View Window Help
Isegal — node — 79x23
> s3.listBuckets().send(); null
null
> [AWS sts 200 0.427s 0 retries] getSessionToken({})
[AWS s3 200 0.808s 0 retries] listBuckets({}) 
```

SDK Features

- Global Configuration Object
- Bound Parameters
- Response Pagination
- Event Listeners (Per-Service and Global)
- API Version Locking
- Secure Credential Management

AWS SDK for JavaScript in the Browser

Developer Preview

Looking for Feedback

Getting the SDK

```
<script src="https://sdk.amazonaws.com/js/aws-sdk-2.0.0-rc1.min.js" />
```

5 Supported
Services

Amazon S3
Amazon DynamoDB
Amazon SQS
Amazon SNS
STS

All Modern Browsers



28.0+



23.0+



10+



5.1+

Usage is the same.

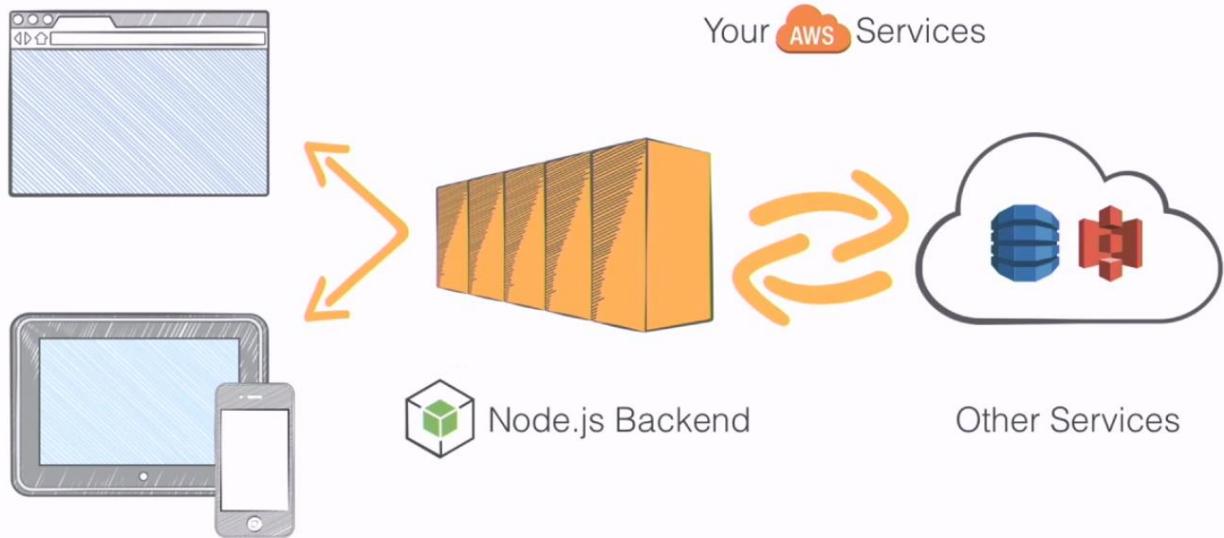
But in your browser or mobile device

Configuration is Different

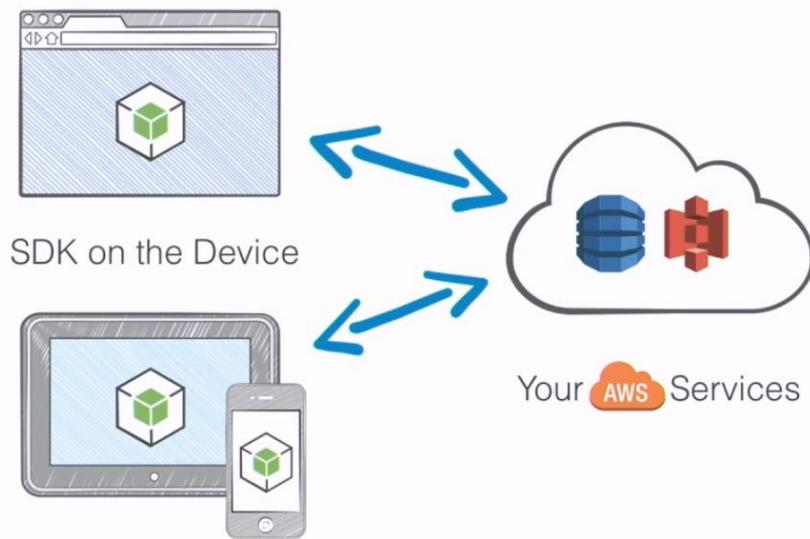
Why is it different?

Two-Tier Web Applications

Traditional Application Architecture



Two-Tier Application Architecture



Benefits

Fewer moving parts
Easy prototyping
Deploying as simple as
copying files to Amazon S3
Fully dynamic app for
pennies a month

Next Level Web Apps

App Ideas

Forum Software
Blog Commenting Service
Blogging Platform
Firefox/Chrome Extensions
WinRT (Metro Style) Apps
Any Mobile App!

Let's Look at a Web Application

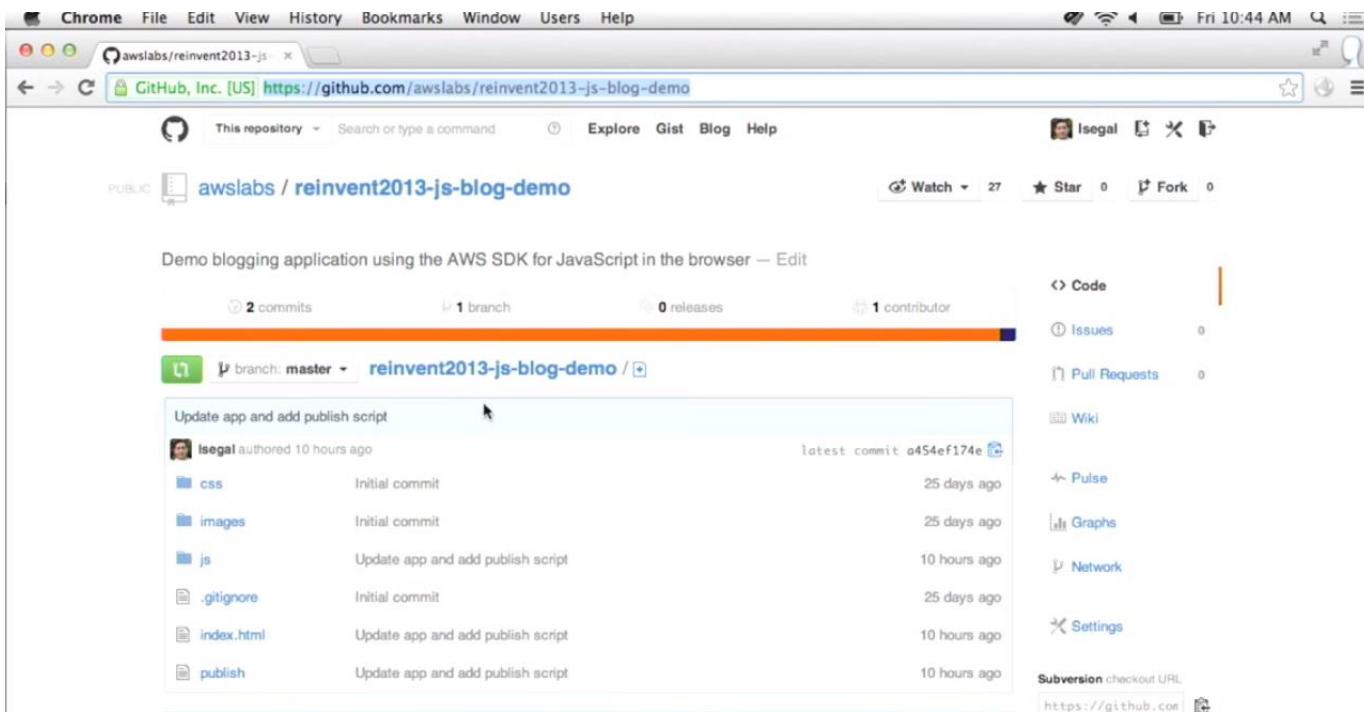
Using nothing but
HTML, CSS, and JavaScript

A Simple Blog

Content stored in Amazon DynamoDB



Assets in Amazon S3



The screenshot shows a GitHub repository page for 'awslabs/reinvent2013-js-blog-demo'. The repository is public and has 2 commits, 1 branch, 0 releases, and 1 contributor. The master branch is selected. The commit history shows:

File	Message	Time
css	Initial commit	25 days ago
images	Initial commit	25 days ago
js	Update app and add publish script	10 hours ago
.gitignore	Initial commit	25 days ago
index.html	Update app and add publish script	10 hours ago
publish	Update app and add publish script	10 hours ago

On the right side of the page, there are links for Code, Issues, Pull Requests, Wiki, Pulse, Graphs, Network, and Settings. A Subversion checkout URL is also provided.

GitHub, Inc. [US] | https://github.com/aws-samples/reinvent2013-js-blog-demo

aws-samples / reinvent2013-js-blog-demo

Code Issues 0 Pull requests 0 Projects 0 Insights

Demo blogging application using the AWS SDK for JavaScript in the browser

5 commits 1 branch 0 releases 2 contributors Apache-2.0

Branch: master New pull request Create new file Upload files Find file Clone or download

hyandell Adding template

Latest commit 319d859 10 days ago

.github Adding template 10 days ago

css Initial commit 4 years ago

images 4 years ago

js 4 years ago

.gitignore 4 years ago

CODE_OF_CONDUCT.md 10 days ago

CONTRIBUTING.md 10 days ago

LICENSE 2 years ago

NOTICE Noting licensing 2 years ago

index.html Update app and add publish script 4 years ago

publish Update app and add publish script 4 years ago

© 2018 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub API Training Shop Blog About

Sublime Text 2 File Edit Selection Find View Goto Tools Project Window Help

index.html — reinvent2013-blog-demo

FOLDERS

- reinvent2013-blog-demo
 - css
 - images
 - js
 - app.js
 - appinfo.js
 - appinfo.js.sample
 - Markdown.Converter.js
 - Markdown.Editor.js
 - Markdown.Sanitizer.js
 - .gitignore
 - index.html
 - publish

```

index.html
1 <html>
2   <head>
3     <title>AWS SDK for JavaScript Blog</title>
4     <link rel="stylesheet" type="text/css" href="css/style.css" />
5   </head>
6   <body>
7     <div id="fb-root"></div>
8     <nav>
9       <a id="new-post-button" class="admin-button" href="#"> Post</a>
10      <a id="login-button" href="#">Login</a>
11    </nav>
12    <div id="content">
13      <h1>re:Invent 2013 Blog Demo</h1>
14
15      <!-- Article editor markup -->
16      <div id="article-editor">
17        <h2>Create a New Article</h2>
18        <p>
19          <input id="article-asset" type="file" />
20          <input id="article-date" type="hidden" value="" />
21          Title: <input id="article-title" type="text" size="100" />
22          <button id="cancel-publish-button">Cancel</button>
23          <button id="publish-button">Publish</button>
24        </p>
25        <div class="wmd-panel">
26          <div id="wmd-button-bar"></div>
27          <textarea class="wmd-input" id="wmd-input"></textarea>
28        </div>
29        <div id="wmd-preview" class="wmd-panel wmd-preview"></div>
30      </div>
31
32      <!-- Placeholder markup for articles -->
33      <div id="articles">Loading articles...</div>
34    </div>
35
36      <!-- HTML template for displaying an individual article -->
37      <script id="article-template" type="text/template">
38        <article id="{{slug}}">
39          <header>
40            <a href="#" class="admin-button delete-button">Delete</a>
41            <a href="#" class="admin-button edit-button">Edit</a>

```

git branch: master, index: ✓, working: ✓, Line 1, Column 1

Spaces: 2 HTML

```

22      <button id="cancel-publish-button">Cancel</button>
23      <button id="publish-button">Publish</button>
24    </p>
25    <div class="wmd-panel">
26      <div id="wmd-button-bar"></div>
27      <textare class="wmd-input" id="wmd-input"></textare>
28    </div>
29    <div id="wmd-preview" class="wmd-panel wmd-preview"></div>
30  </div>
31
32  <!-- Placeholder markup for articles -->
33  <div id="articles">Loading articles...</div>
34 </div>
35
36  <!-- HTML template for displaying an individual article -->
37  <script id="article-template" type="text/template">
38    <article id="{{slug}}">
39      <header>
40        <a href="#" class="admin-button delete-button">Delete</a>
41        <a href="#" class="admin-button edit-button">Edit</a>
42        <h2 class="title">{{title}}</h2>
43        <h3 class="publishDate">Published on {{publishDate}}</h3>
44      </header>
45      <section class="body">{{body}}</section>
46    </article>
47  </script>
48
49  <!-- Markdown editor scripts -->
50  <script type="text/javascript" src="js/Markdown.Converter.js"></script>
51  <script type="text/javascript" src="js/Markdown.Sanitizer.js"></script>
52  <script type="text/javascript" src="js/Markdown.Editor.js"></script>
53
54  <!-- Load the SDK -->
55  <script type="text/javascript" src="https://sdk.amazonaws.com/js/aws-sdk-2.0.0-rc1.min.js"></script>
56
57  <!-- Load our application -->
58  <script type="text/javascript" src="js/appinfo.js"></script>
59  <script type="text/javascript" src="js/app.js"></script>
60 </body>
61 </html>
62

```

git branch: master, index: ✓, working: ✓, 7 lines, 282 characters selected Spaces: 2 HTML

We load the SDK at the bottom of the UI page as above

```

1 appInfo = {
2   admin: {
3     appId: '173506512855005',
4     providerId: 'graph.facebook.com',
5     roleArn: 'arn:aws:iam::579606425216:role/reinvent2013-blog-demo-admin'
6   },
7   db: {
8     region: 'us-west-2',
9     tableName: 'reinvent2013-blog-demo',
10    readCredentials: {
11      accessKeyId: 'AKIAJR7Q4N5WRHHVMIDQ',
12      secretAccessKey: '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'
13    }
14  },
15  s3: {
16    region: 'us-east-1',
17    bucket: 'reinvent2013-blog-demo',
18    prefix: 'assets/'
19  }
20};
21

```

This file contains all of our SDK configuration details, we are using a very scoped credentials that only has read-only access to the relevant DynamoDB table only. We also have the details for the public S3 bucket where our app is going to be getting the static asset files from.

Sublime Text 2 window showing the app.js file. The code sets up basic AWS configuration, including the region, credentials, and logger. It also initializes service objects for DynamoDB, S3, and md2html converters.

```
1 (function() {
2
3     // Setup basic AWS configuration
4     AWS.config.update({
5         region: appInfo.db.region,
6         credentials: appInfo.db.readCredentials,
7         logger: console
8     });
9
10    var md2html = Markdown.getSanitizingConverter();
11    var articleOrder = {};
12    var articleData = {};
13    var adminLoggedIn = false;
14    var adminCredentials = new AWS.WebIdentityCredentials({
15        RoleArn: appInfo.admin.roleArn,
16        ProviderId: appInfo.admin.providerId
17    });
18
19    // Setup some service objects
20    var dbReader = new AWS.DynamoDB({params: {TableName: appInfo.db.tableName}});
21    var dbWriter = new AWS.DynamoDB({
22        params: {TableName: appInfo.db.tableName},
23        credentials: adminCredentials
24    });
25    var s3Bucket = new AWS.S3({
26        paramValidation: false,
27        computeChecksums: false,
28        params: {Bucket: appInfo.s3.bucket},
29        region: appInfo.s3.region,
30    });
31
32    // ...
33
34    // ...
35
36    // ...
37
38    // ...
39
40    // ...
41
42    // ...
43
44    // ...
45
46    // ...
47
48    // ...
49
50    // ...
51
52    // ...
53
54    // ...
55
56    // ...
57
58    // ...
59
60    // ...
61
62    // ...
63
64    // ...
65
66    // ...
67
68    // ...
69
70    // ...
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85}
```

Sublime Text 2 window showing the app.js file. The code includes a function to generate a GUID, and another function to upload an asset to S3. The uploaded URL is then appended to the article body.

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
```

This is where we are uploading the assets into S3

This screenshot shows the Sublime Text 2 interface with three tabs open: index.html, appinfo.js, and app.js. The app.js tab contains the following JavaScript code:

```
function publishArticle() {
    var slug = articleTitle.value.toLowerCase().replace(/[^a-z0-9]/ig, '-');
    var params = {
        Item: {
            type: {S: 'article'},
            publishDate: {N: articleDate.value || new Date().getTime().toString()},
            title: {S: articleTitle.value},
            body: {S: articleBody.value},
            slug: {S: slug}
        }
    };
    var slugElement = document.getElementById(slug);
    if (slugElement) {
        var titleEl = document.querySelector('#' + slug + '.title');
        var bodyEl = document.querySelector('#' + slug + '.body');
        titleEl.innerText = params.Item.title.S;
        bodyEl.innerHTML = md2html.makeHtml(params.Item.body.S);
        dbWriter.putItem(params).send();
    } else {
        dbWriter.putItem(params, loadArticles);
    }
    cacheArticle(params.Item);
    hideEditor();
}
```

The status bar at the bottom indicates: git branch: master, index: ✓, working: ✓, 14 lines, 463 characters selected. The bottom right corner shows Spaces: 2 and JavaScript.

This is the code that interacts with DynamoDB

This screenshot shows the Sublime Text 2 interface with three tabs open: index.html, appinfo.js, and app.js. The app.js tab contains the following JavaScript code:

```
function deleteArticle(article) {
    var params = {
        Key: {
            type: {S: 'article'},
            publishDate: {N: article.publishDate.getTime().toString()}
        }
    };
    dbWriter.deleteItem(params, loadArticles);
}

function cacheArticle(item) {
    var slug = item.slug.S;
    var timestamp = parseInt(item.publishDate.N);
    articleOrder[slug] = timestamp;
    articleData[slug] = {
        slug: slug,
        publishDate: new Date(timestamp),
        title: item.title.S,
        body: item.body.S
    };
}

function showEditor(article) {
    articleTitle.value = article.title ? article.title : '';
    articleDate.value = article.publishDate ? article.publishDate.getTime().toString() : '';
    articleBody.value = article.body ? article.body : '';
    articleEditor.style.display = 'block';
}
```

Sublime Text 2 window showing the file structure of the reinvent2013-blog-demo project. The current file is app.js, which contains the following code:

```
147     function loadArticleEditor() {
148         return new Markdown.Editor(md2html);
149     }
150
151     function loadArticles() {
152         articleOrder = {};
153         articleData = {};
154
155         // Clear div
156         articles.innerHTML = '';
157
158         var params = {
159             Limit: 20,
160             ScanIndexForward: false,
161             KeyConditions: {
162                 type: {
163                     AttributeValueList: [{S: "article"}],
164                     ComparisonOperator: "EQ"
165                 },
166                 publishDate: {
167                     AttributeValueList: [{N: "0"}],
168                     ComparisonOperator: "GE"
169                 }
170             }
171         };
172
173         dbReader.query(params).eachPage(function (err, data) {
174             if (data) {
```

Sublime Text 2 window showing the same file structure and current file (app.js). The code has been scrolled down to show more of the function body:

```
// Clear div
articles.innerHTML = '';

var params = {
    Limit: 20,
    ScanIndexForward: false,
    KeyConditions: {
        type: {
            AttributeValueList: [{S: "article"}],
            ComparisonOperator: "EQ"
        },
        publishDate: {
            AttributeValueList: [{N: "0"}],
            ComparisonOperator: "GE"
        }
    }
};

dbReader.query(params).eachPage(function (err, data) {
    if (data) {
        for (var i = 0; i < data.Items.length; i++) {
            cacheArticle(data.Items[i]);
        }
        if (!this.hasNextPage()) renderArticles();
    }
});
```

We are doing a simple query here and using response pagination to get each available page

The screenshot shows the Sublime Text 2 interface with the following details:

- File Menu:** Sublime Text 2, File, Edit, Selection, Find, View, Goto, Tools, Project, Window, Help.
- Status Bar:** Fri 10:47 AM, git branch: master, index: ✓, working: ✓, 14 lines, 463 characters selected.
- Code Editor:** The app.js file is open, showing code for Facebook login. The code includes logic for handling the 'auth.authResponseChange' event, initializing the Facebook SDK, and updating UI elements like a login button based on the user's status.
- Folders:** The project structure shows folders for css, images, and js, with app.js being the active file.
- Bottom Status:** Spaces: 2, JavaScript.

```
217 }
218
219 // Facebook login
220 window.fbAsyncInit = function() {
221     FB.init({appId: appInfo.admin.appId});
222
223     FB.Event.subscribe('auth.authResponseChange', function(response) {
224         if (response.status === 'connected') {
225             adminCredentials.params.WebIdentityToken =
226                 response.authResponse.accessToken;
227             adminCredentials.refresh(function (err) {
228                 if (err) {
229                     console.log("Error logging into application", err.message);
230                     body.className = '';
231                     loginButton.innerText = 'Login';
232                     adminLoggedIn = false;
233
234                 } else {
235                     console.log("Logged into application as administrator");
236                     body.className = 'admin-logged-in';
237                     loginButton.innerText = 'Logout';
238                     adminLoggedIn = true;
239                 }
240             });
241         } else {
242             console.log("Logged out");
243             body.className = '';
244             loginButton.innerText = 'Login';
245             adminLoggedIn = false;
246         }
247     });
248
249     FB.getLoginStatus();
250 };
251
252 // Load the SDK asynchronously
253 (function(d, s, id){
254     var js, fjs = d.getElementsByTagName(s)[0];
255     if (d.getElementById(id)) {return;}
256     js = d.createElement(s); js.id = id;
257     js.src = "//connect.facebook.net/en_US/all.js";
258     fjs.parentNode.insertBefore(js, fjs);
259 }(document, 'script', 'facebook-jssdk'));
260
261 loadArticles();
262 publishButton.addEventListener('click', publishArticle, false);
263 cancelPublishButton.addEventListener('click', hideEditor, false);
264 postButton.addEventListener('click', showEditor, false);
265 loginButton.addEventListener('click', adminLogin, false);
266 chooseImageButton.addEventListener('click', function() { articleAsset.click()});
267 articleAsset.addEventListener('change', uploadAsset, false);
268 })();
269
270
```

This is the OAuth login

The screenshot shows the Sublime Text 2 interface with the following details:

- File Menu:** Sublime Text 2, File, Edit, Selection, Find, View, Goto, Tools, Project, Window, Help.
- Status Bar:** Fri 10:47 AM, git branch: master, index: ✓, working: ✓, 14 lines, 463 characters selected.
- Code Editor:** The app.js file is open, showing code for OAuth login. It includes logic for loading the Facebook SDK asynchronously and setting up event listeners for various UI buttons.
- Folders:** The project structure shows folders for css, images, and js, with app.js being the active file.
- Bottom Status:** Spaces: 2, JavaScript.

```
242     console.log("Logged out");
243     body.className = '';
244     loginButton.innerText = 'Login';
245     adminLoggedIn = false;
246 }
247 );
248
249 FB.getLoginStatus();
250 };
251
252 // Load the SDK asynchronously
253 (function(d, s, id){
254     var js, fjs = d.getElementsByTagName(s)[0];
255     if (d.getElementById(id)) {return;}
256     js = d.createElement(s); js.id = id;
257     js.src = "//connect.facebook.net/en_US/all.js";
258     fjs.parentNode.insertBefore(js, fjs);
259 }(document, 'script', 'facebook-jssdk'));
260
261 loadArticles();
262 publishButton.addEventListener('click', publishArticle, false);
263 cancelPublishButton.addEventListener('click', hideEditor, false);
264 postButton.addEventListener('click', showEditor, false);
265 loginButton.addEventListener('click', adminLogin, false);
266 chooseImageButton.addEventListener('click', function() { articleAsset.click()});
267 articleAsset.addEventListener('change', uploadAsset, false);
268 })();
269
270
```

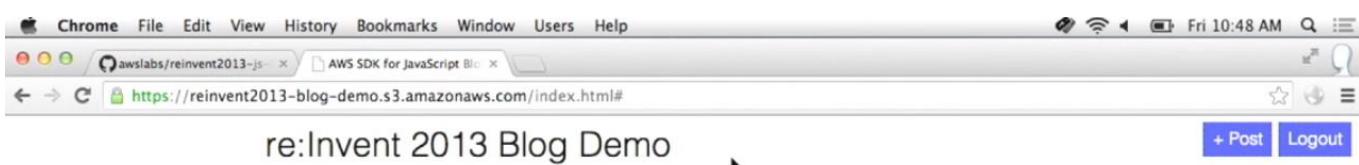
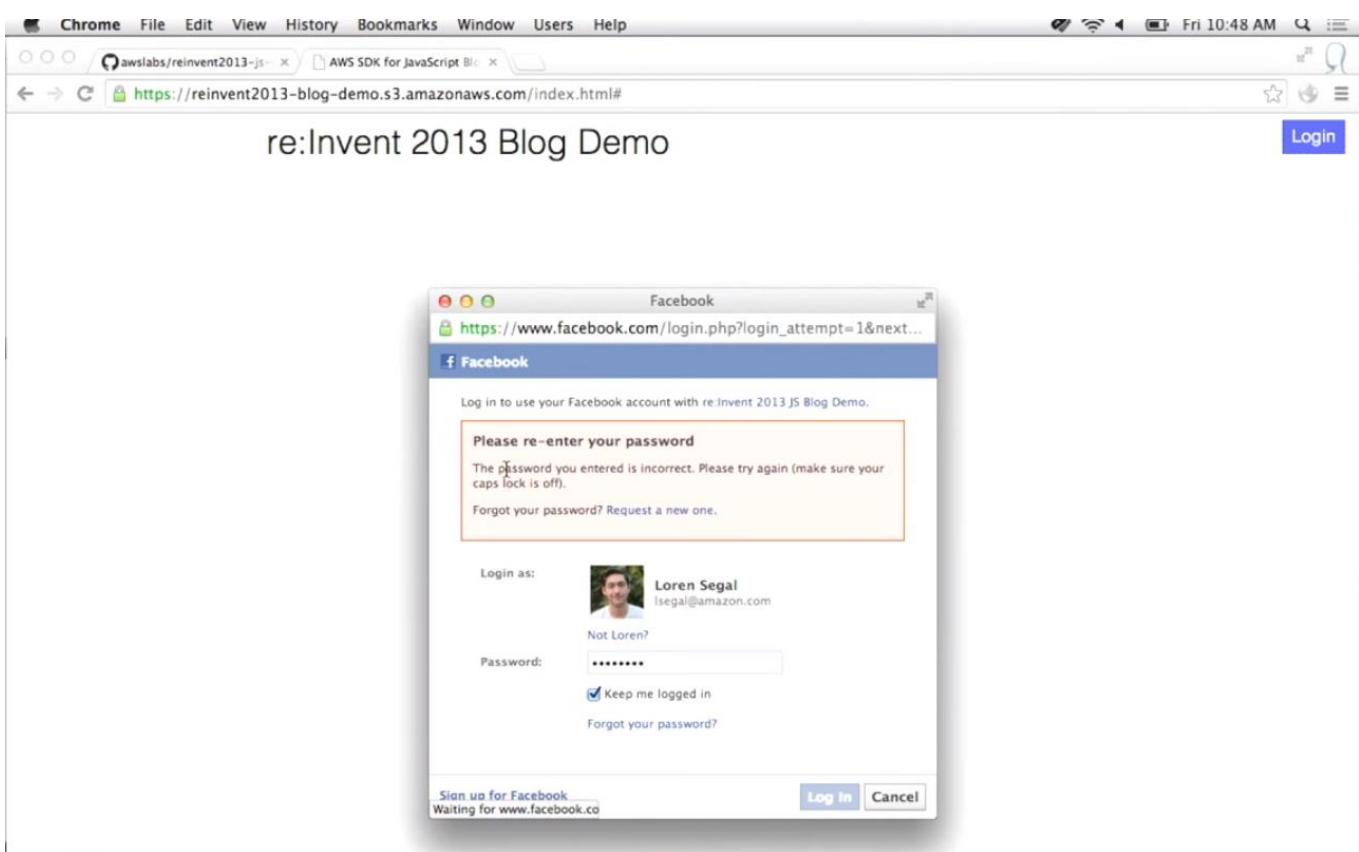
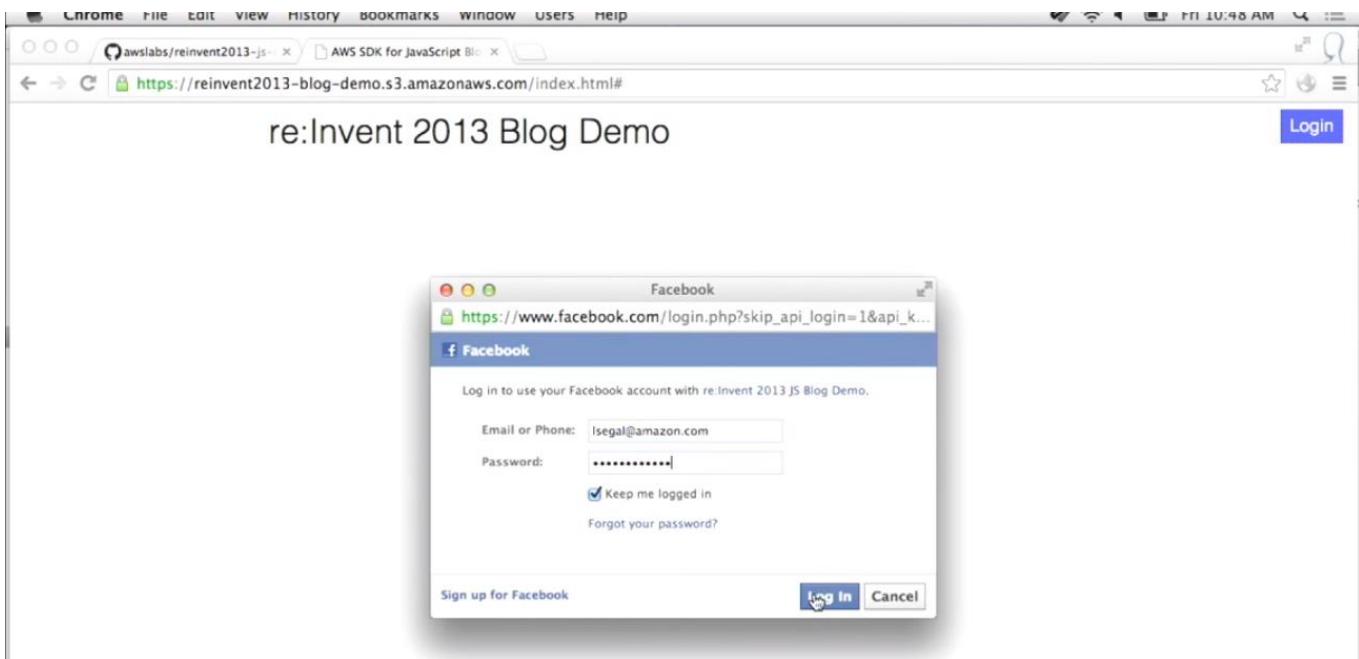
```
#!/bin/sh
BUCKET=reinvent2013-blog-demo
aws s3 sync . s3://$BUCKET \
--exclude '*' --include '*.js' --include '*.html' --include '*.css' --include '*.*' \
--region us-east-1 --acl public-read
```

We can then use the CLI to deploy the application using the command above. We are using the AWS S3 sync command to publish all our assets to the S3 bucket.

```
~/aws/js/reinvent2013-blog-demo (master)$ ./publish
upload: ./index.html to s3://reinvent2013-blog-demo/index.html
upload: js/Markdown.Sanitizer.js to s3://reinvent2013-blog-demo/js/Markdown.Sanitizer.js
upload: css/style.css to s3://reinvent2013-blog-demo/css/style.css
upload: images/wmd-buttons.png to s3://reinvent2013-blog-demo/images/wmd-buttons.png
upload: js/appinfo.js to s3://reinvent2013-blog-demo/js/appinfo.js
upload: js/app.js to s3://reinvent2013-blog-demo/js/app.js
upload: js/Markdown.Converter.js to s3://reinvent2013-blog-demo/js/Markdown.Converter.js
upload: js/Markdown.Editor.js to s3://reinvent2013-blog-demo/js/Markdown.Editor.js
~/aws/js/reinvent2013-blog-demo (master)$
```

The CLI has now deployed our app into the S3 bucket successfully.





We can now make a post as an admin

Chrome File Edit View History Bookmarks Window Users Help

Fri 10:49 AM

awslabs/reinvent2013-javascript AWS SDK for JavaScript Blog

<https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#>

re:Invent 2013 Blog Demo

+ Post Logout

Create a New Article

Title: Welcome to re:Invent 2013

Cancel Publish

B I # Hello everyone! Image Ctrl+G

Hello everyone!

Chrome File Edit View History Bookmarks Window Users Help

Fri 10:49 AM

awslabs/reinvent2013-javascript AWS SDK for JavaScript Blog

<https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#>

Desktop

FAVORITES

- Applications
- Downloads
- Documents
- Desktop
- Isegal

DEVICES

- Isegal-amazon

MEDIA

- Music
- Photos
- Movies

Name Date Modified Size Kind

logo-dark.png Yesterday 5:38 PM 9 KB Portable image

reinvent Nov 13, 2013 8:36 PM -- Folder

Cancel Open

Hello everyone!

Chrome File Edit View History Bookmarks Window Users Help Fri 10:49 AM

awslabs/reinvent2013-j... AWS SDK for JavaScript Blog https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#

re:Invent 2013 Blog Demo

Create a New Article

Title: Welcome to re:Invent 2013

B I Cancel Publish

Hello everyone!

Hello everyone!

Chrome File Edit View History Bookmarks Window Users Help Fri 10:50 AM

awslabs/reinvent2013-j... AWS SDK for JavaScript Blog https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#

re:Invent 2013 Blog Demo

Create a New Article

Title: Welcome to re:Invent 2013

B I Cancel Publish

Hello everyone!

Elements Resources Network Sources Timeline Profiles Audits Console

```
[AWS sts 200 0.638s 0 retries] assumeRoleWithWebIdentity({ RoleArn: 'arn:aws:iam::579606425216:role/reinvent2013-blog-demo-admin',
  ProviderId: 'graph.facebook.com',
  RoleSessionName: 'web-identity',
  WebIdentityToken:
    'CAACdzaIQn90BAAGZCnizPMZAnWMDhrok1LDcNLcSBLh5YTN6JKnvZBGKZARFzn8CZCMiH9tZBWv970aLLpYL5rEglnl5msMTrVZCk2rY4uY2ET2nmZADu
us0VMTHnnh9bXCcX6TSn7rScs1aGhAad27zZCjsZCfpv3ZBk75TDkG09fZA04ZCv7qaU2GBwi5IJZBmmmmJhvY1pkYcbv9zAZDZD' })
aws-sdk-2.0.0-rc1.min.js:7
Logged into application as administrator
app.js:235
Logged out
app.js:242
[AWS sts 200 0.404s 0 retries] assumeRoleWithWebIdentity({ RoleArn: 'arn:aws:iam::579606425216:role/reinvent2013-blog-demo-admin',
  ProviderId: 'graph.facebook.com'.
```

Chrome File Edit View History Bookmarks Window Users Help Fri 10:50 AM

awslabs/reinvent2013-js AWS SDK for JavaScript Blog https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#

re:Invent 2013 Blog Demo

Create a New Article

Title: Welcome to re:Invent 2013

Hello everyone!

Elements Resources Network Sources Timeline Profiles Audits Console

```
WebIdentityToken:  
'CAACdzaIQn90BA0GFN71AVZCTBxwIUY53PTCCj28uj71ElvJ2Jvh3rP8FF8b0Ih8TFT5BS9d5ZCoMN7UrWr8060lUzHJYmlW4kLQYUPbMhk7LtpWhkZAdM  
m27MkHLjt0d7pxRoW8ovP1o163twJdLyNR0mPZAohF0oZCQVTvc0NzYmmhGyqa65PtLSbA0ZC7Ker0B2HgpwmnwZDZD' }  
aws-sdk-2.0.0-rc1.min.js:7
```

Logged into application as administrator app.js:235

```
[AWS s3 200 0.581s 0 retries] putObject({ Key: 'assets/9a9179b1-d6ca-1808-a6f0-33f3d78e15ea.png',  
  ContentType: 'image/png',  
  Body:  
    { webkitRelativePath: '',  
      lastModifiedDate: Thu Nov 14 2013 17:38:00 GMT-0800 (PST),  
      name: 'logo-dark.png',  
      type: 'image/png',  
      size: 8504 },  
    ACL: 'public-read'
```

< top frame > <page context> All Errors Warnings Logs De ⚠ 1

Chrome File Edit View History Bookmarks Window Users Help Fri 10:49 AM

awslabs/reinvent2013-js AWS SDK for JavaScript Blog https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#

re:Invent 2013 Blog Demo

Create a New Article

Title: Welcome to re:Invent 2013

Hello everyone!

Elements Resources Network Sources Timeline Profiles Audits Console

```
WebIdentityToken:  
'CAACdzaIQn90BA0GFN71AVZCTBxwIUY53PTCCj28uj71ElvJ2Jvh3rP8FF8b0Ih8TFT5BS9d5ZCoMN7UrWr8060lUzHJYmlW4kLQYUPbMhk7LtpWhkZAdM  
m27MkHLjt0d7pxRoW8ovP1o163twJdLyNR0mPZAohF0oZCQVTvc0NzYmmhGyqa65PtLSbA0ZC7Ker0B2HgpwmnwZDZD' }  
aws-sdk-2.0.0-rc1.min.js:7
```

Logged into application as administrator app.js:235

```
[AWS s3 200 0.581s 0 retries] putObject({ Key: 'assets/9a9179b1-d6ca-1808-a6f0-33f3d78e15ea.png',  
  ContentType: 'image/png',  
  Body:  
    { webkitRelativePath: '',  
      lastModifiedDate: Thu Nov 14 2013 17:38:00 GMT-0800 (PST),  
      name: 'logo-dark.png',  
      type: 'image/png',  
      size: 8504 },  
    ACL: 'public-read',  
    Bucket: 'reinvent2013-blog-demo' })  
aws-sdk-2.0.0-rc1.min.js:7
```

< top frame > <page context> All Errors Warnings Logs De ⚠ 1

Chrome File Edit View History Bookmarks Window Users Help

Fri 10:50 AM

awslabs/reinvent2013-js AWS SDK for JavaScript Blog

<https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#>

re:Invent 2013 Blog Demo

Create a New Article

Title: Welcome to re:Invent 2013

B I

Hello everyone!

Hello everyone!

AWS re:Invent
November 12-15, 2013 - Las Vegas

Chrome File Edit View History Bookmarks Window Users Help

Fri 10:50 AM

awslabs/reinvent2013-js AWS SDK for JavaScript Blog

<https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#>

re:Invent 2013 Blog Demo

Create a New Article

Title: Welcome to re:Invent 2013

B I

Hello everyone!

Hope you are are enjoying the sessions!

Hello everyone!

AWS re:Invent
November 12-15, 2013 - Las Vegas

Chrome File Edit View History Bookmarks Window Users Help Fri 10:51 AM

awslabs/reinvent2013-j5 AWS SDK for JavaScript Blog

https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#

re:Invent 2013 Blog Demo

Welcome to re:Invent 2013

Published on Fri Nov 15 2013 10:50:35 GMT-0800 (PST)

Hello everyone,

AWS re:Invent
November 12-15, 2013 - Las Vegas

Hope you are are enjoying the sessions!

Hope you enjoyed Deadmau5!

+ Post Logout

We can also add more posts

Chrome File Edit View History Bookmarks Window Users Help Fri 10:51 AM

awslabs/reinvent2013-j5 AWS SDK for JavaScript Blog

https://reinvent2013-blog-demo.s3.amazonaws.com/index.html#

re:Invent 2013 Blog Demo

Goodbye re:Invent

Published on Fri Nov 15 2013 10:51:30 GMT-0800 (PST)

Hope you enjoyed the conference!

Welcome to re:Invent 2013

Published on Fri Nov 15 2013 10:50:35 GMT-0800 (PST)

Hello everyone,

AWS re:Invent
November 12-15, 2013 - Las Vegas

Hope you are are enjoying the sessions!

Hope you enjoyed Deadmau5!

+ Post Logout

Key Differences

Three-Tier to Two-Tier

Browser Security

CORS in the browser
Credentials on device

Cross-Origin Resource Sharing

CORS

Browser sends pre-flight request to external host.
Host acknowledges browser.
Browser sends XHR request.

CORS
+ S3

CORS needs special configuration on Amazon S3.
Configure CORS with bucket policy.

Configuring CORS on Amazon S3



Getting Credentials Onto Your Device

Getting
Credentials
Onto Your
Device

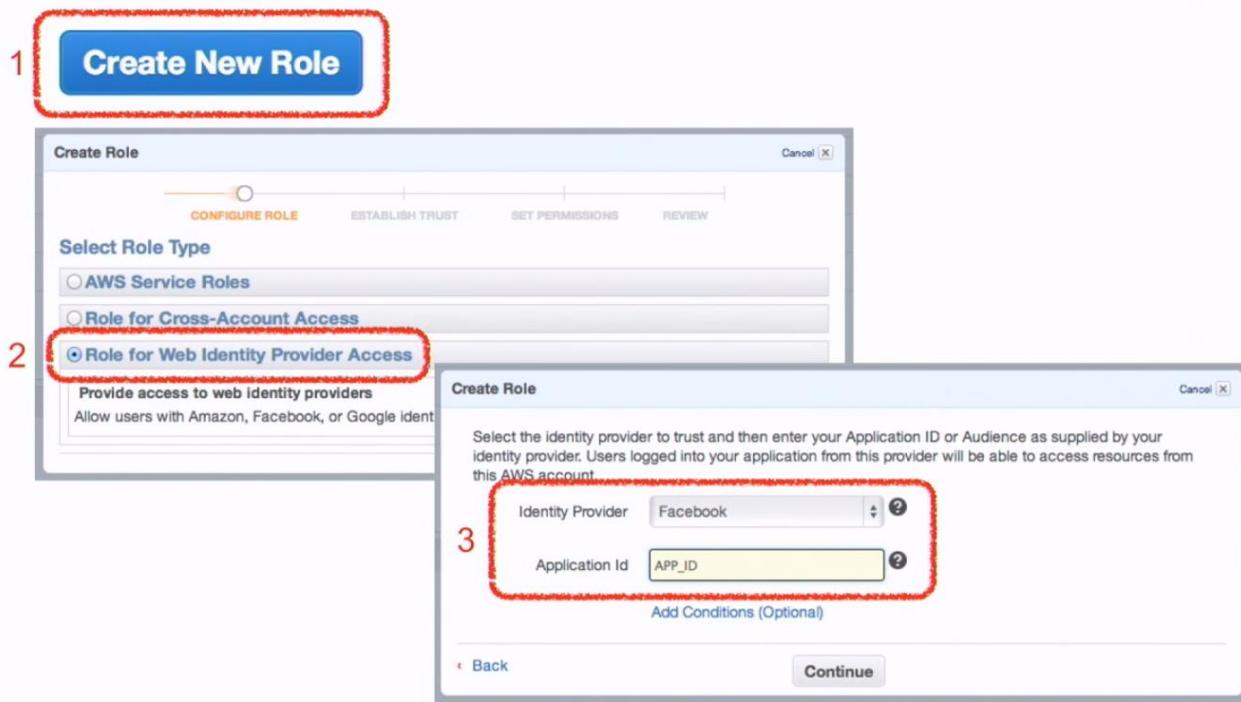
Never hardcode
credentials
Use Web Identity
Federation

Web Identity Federation

Use Facebook, Google, or Login with Amazon as third-party identity providers

Web Identity Federation

Set up IAM roles for these identity providers



Web Identity Federation

Set up permissions for IAM role

AWS.WebIdentityCredentials

JS

```
AWS.config.credentials = new AWS.WebIdentityCredentials({  
    RoleArn: 'arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>',  
    ProviderId: 'graph.facebook.com',  
    WebIdentityToken: fbAccessToken  
});
```

Get a Facebook Access Token

JS

```
// 1. Load the FB JS SDK  
// 2. Call FB.login()  
FB.login(function (response) {  
    if (response.authResponse) {  
        fbAccessToken = response.authResponse.accessToken;  
        AWS.config.credentials = new AWS.WebIdentityCredentials({...});  
    };
```

Same Concept

For other identity providers

Our Community

We ❤

Open Source

[https://github.com/
aws/aws-sdk-js](https://github.com/aws/aws-sdk-js)

Contributing
to the SDK

Improve Documentation
Report Issues
Submit Pull Requests
Third-Party Plugins

Third Party Plugins

A great way to add features

Node.js

The screenshot shows the npm package page for 'aws-sdk'. At the top is the npm logo and a search bar labeled 'Search Packages'. Below the search bar is the package name 'aws-sdk' in bold. A horizontal line follows, with the text 'AWS SDK for JavaScript' underneath. A command line interface (CLI) command '\$ npm install aws-sdk' is shown below. A red rectangular box highlights the 'Dependents' section, which lists 76 packages that depend on 'aws-sdk', including 'excess', 'caisson', 'dynamocmd', 'tadaa-elb', 's3http', 'ec2-instance-data', 'aws-command-stack', 'basin', 'parcel', 'dynamo-flu', 'aws-ses-mail', 'dynamite', 'mongodump-s3', 'iamhard', 'aws-curl', 'seslist', 'beetea-server', 'redshift-cli', 'gdoc-to-s3', 'cinovo-loganalyzer-aws', and 'and 56 more'. Below this is another red box containing the word 'DEPENDENCIES'.

The screenshot shows the Bower package manager landing page. It features a large, colorful cartoon bird logo on the left. To the right of the logo, the word 'Bower' is written in a large, bold, red font. Below 'Bower', the text 'A package manager for the web' is displayed in a smaller, dark gray font. Underneath that, the text 'By Twitter' is shown. At the bottom of the page is a green button with the text 'View on GitHub'.