salsa digital

Introduction to Amazon Web Services

Rob Amos - rob@salsadigital.com.au

@bok_

r.bok.im/5

- Owning the Stack
- Amazon Web Services Overview
- AWS SDK for iOS
- Alternative SDK

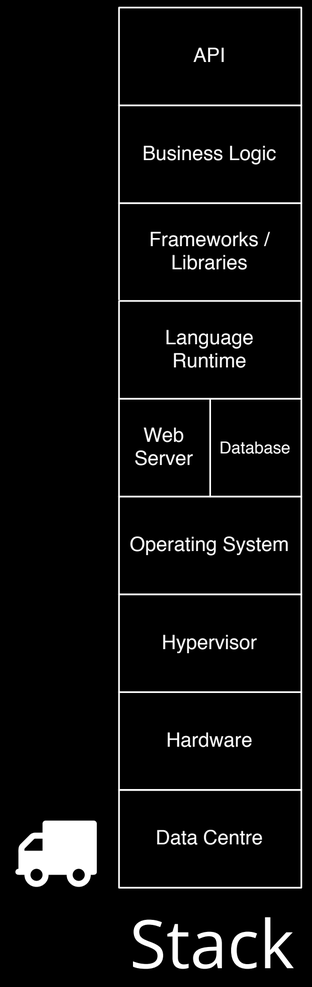
salsa digital

“The strategy today is simple: In order to move fast, build what you can't buy or risk losing control of your fate.”

–Om Malik, Masters of their Own Destiny - 2014-02-10

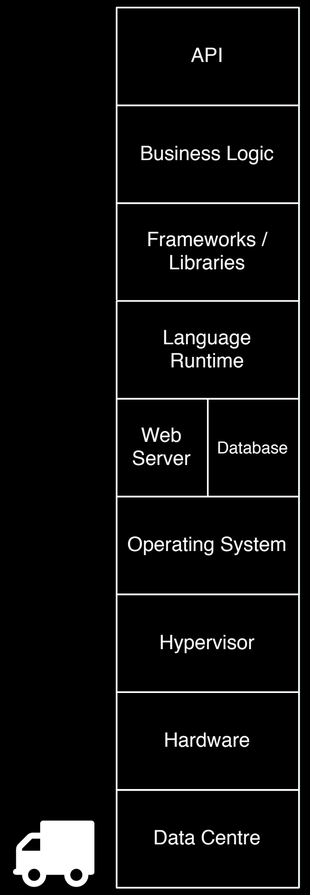
r.bok.im/4

Owning the Stack



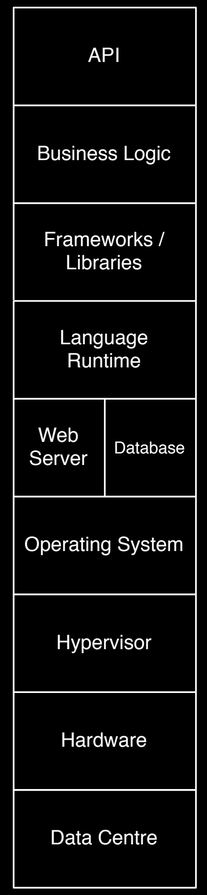
Stack

Owning the Stack

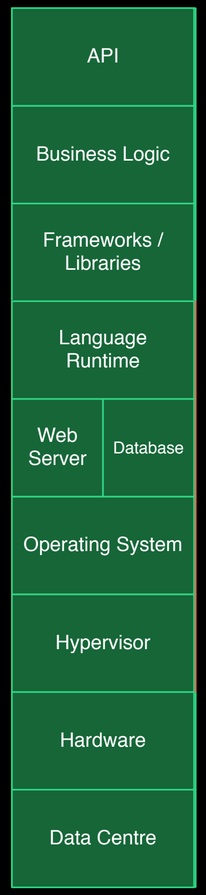


Stack

Owning the Stack



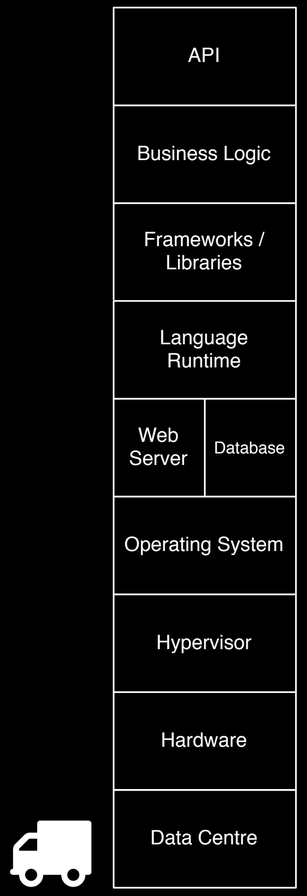
Stack



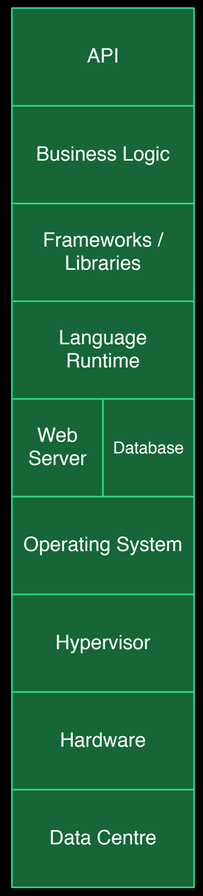
Google

Owning the Stack

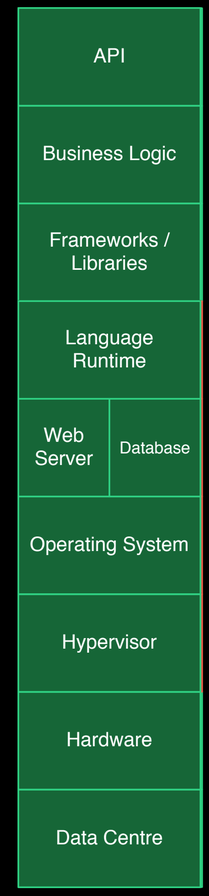
salsa digital



Stack

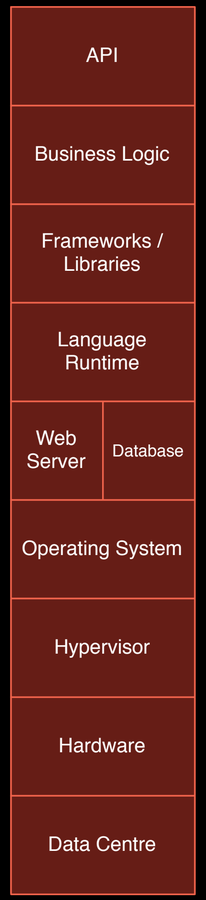


Google



Facebook

Owning the Stack



Backend as a Service

Owning the Stack

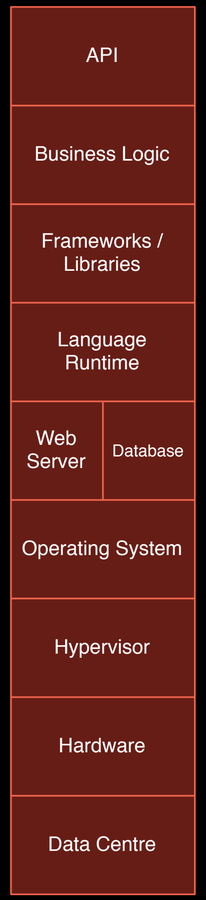
salsa digital

StackMob

Powering Mobile Applications

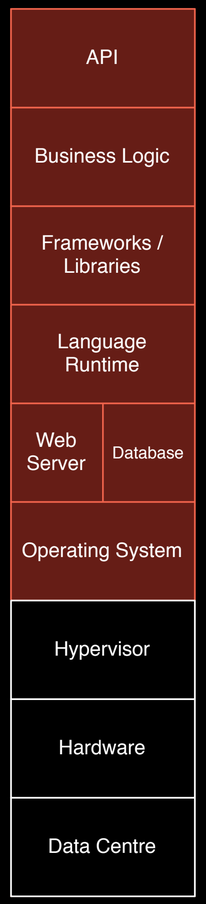
r.bok.im/6

Owning the Stack

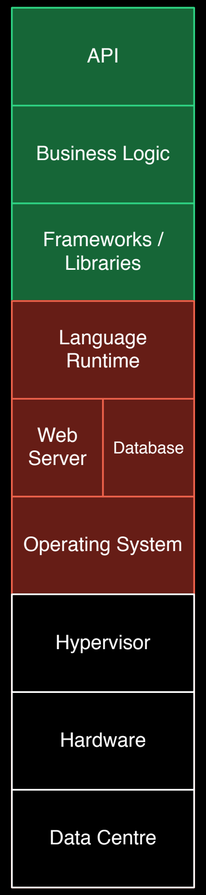


Backend as a Service

Owning the Stack

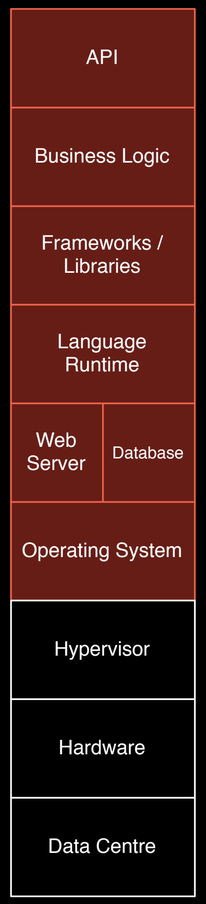


Parse / Kinvey

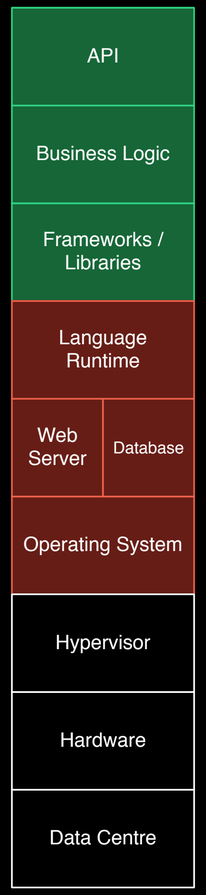


Heroku

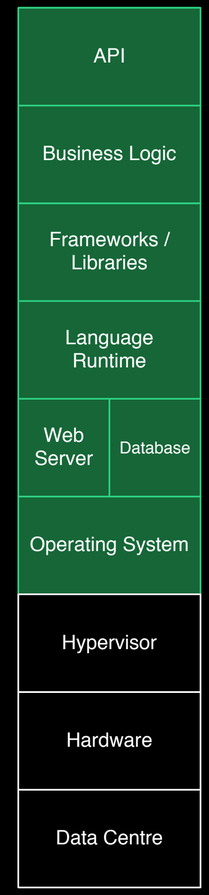
Owning the Stack



Parse / Kinvey



Heroku



AWS

Amazon Web Services

Overview

Why AWS?

salsa digital

- Consumption-based billing.
- Economies of scale.
- Improve availability, durability and performance.
- Reduce overall costs.
- Self-service infrastructure.
- Easily scale up and down.
- Improve agility and time to market.

r.bok.im/7

Global Infrastructure

r.bok.im/8

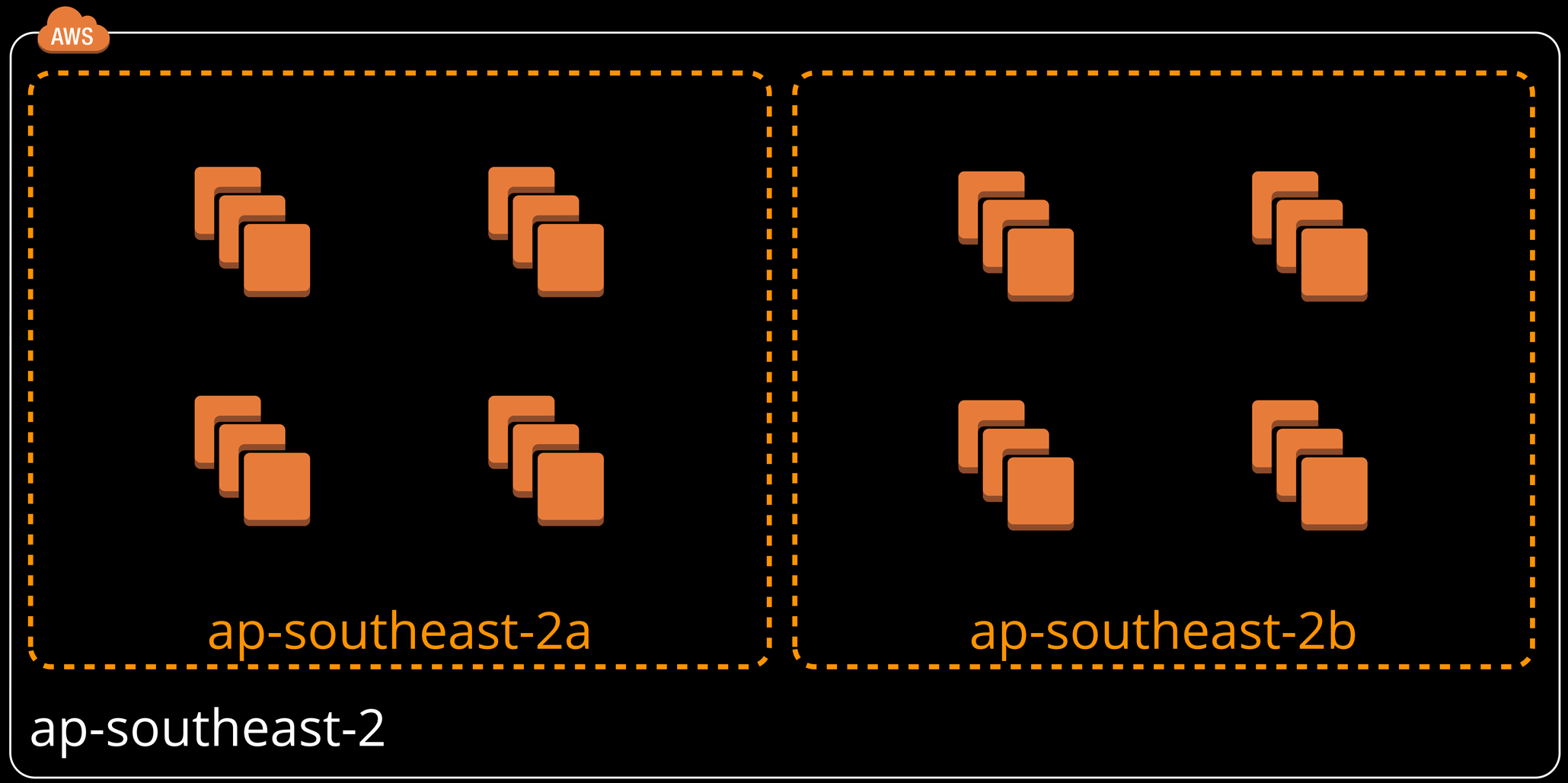
Global Infrastructure

r.bok.im/8

Global Infrastructure

r.bok.im/8

Global Infrastructure



Security

salsa digital

r.bok.im/9

Products

salsa digital

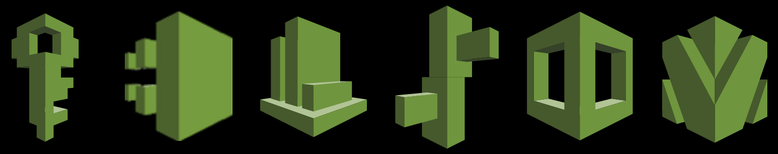
Compute and Networking

Application Services

Analytics

Storage and Content Delivery

Deployment and Management



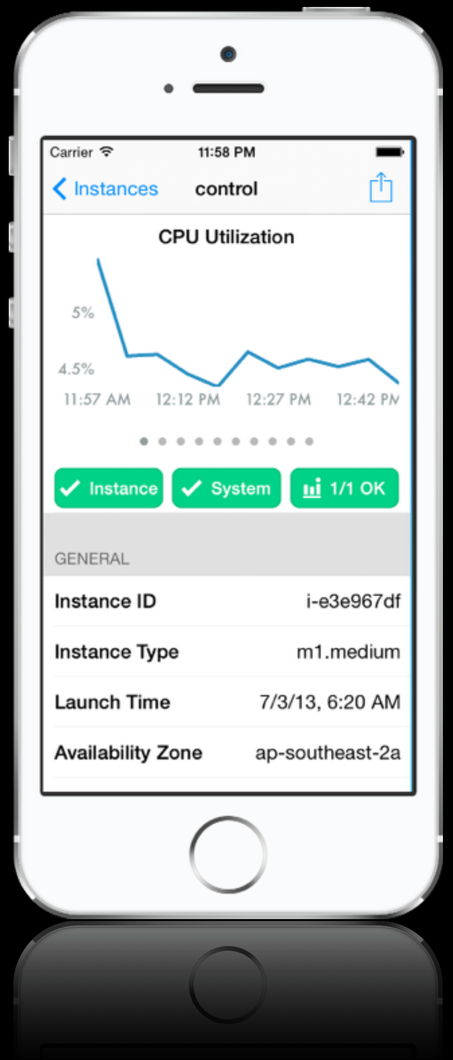
Database

r.bok.im/a

Elastic Compute Cloud

salsa digital

- “Virtual Servers” in the cloud, pay per hour for what you use.
- From 1 CPU with 630 MiB of RAM ($0.02 per hour).
- To 32 CPUs with 244 GiB of RAM ($8.14 per hour).
- Automatically add/remove instances to meet demand.
- Backed by the Elastic Block Store - drives from 1GB to 1TB, as many as you can handle.
- Free tier: Linux/Windows Micro instance.



r.bok.im/b

Simple Queue Service

- Guaranteed “at least once” message delivery.
- De-couple iPhone and backend.
- Asynchronous backend? Don’t pay for backend to be up all the time.
- $0.50 per 1 million SQS requests.
- Free tier: first 1 million SQS requests.

r.bok.im/c

Kinesis

salsa digital

- Brand new service.
- Real-time processing of information streams.
- Useful for streaming sensor data, logs, analytics.
- Backend can scale processing as required.
- Process the entire Twitter firehose for $0.15 per hour.

r.bok.im/d

Simple Notification Service

- Supports Push Notifications, emails, and SMS (US only).
- APNS, APNS_Sandbox, GDM, ADM.
- Subscribe to topics.
- Suitable for notifications and push messages.
- $1 per 1 million push notifications.
- Free tier: first 1 million are free.

r.bok.im/e

Simple Storage Service

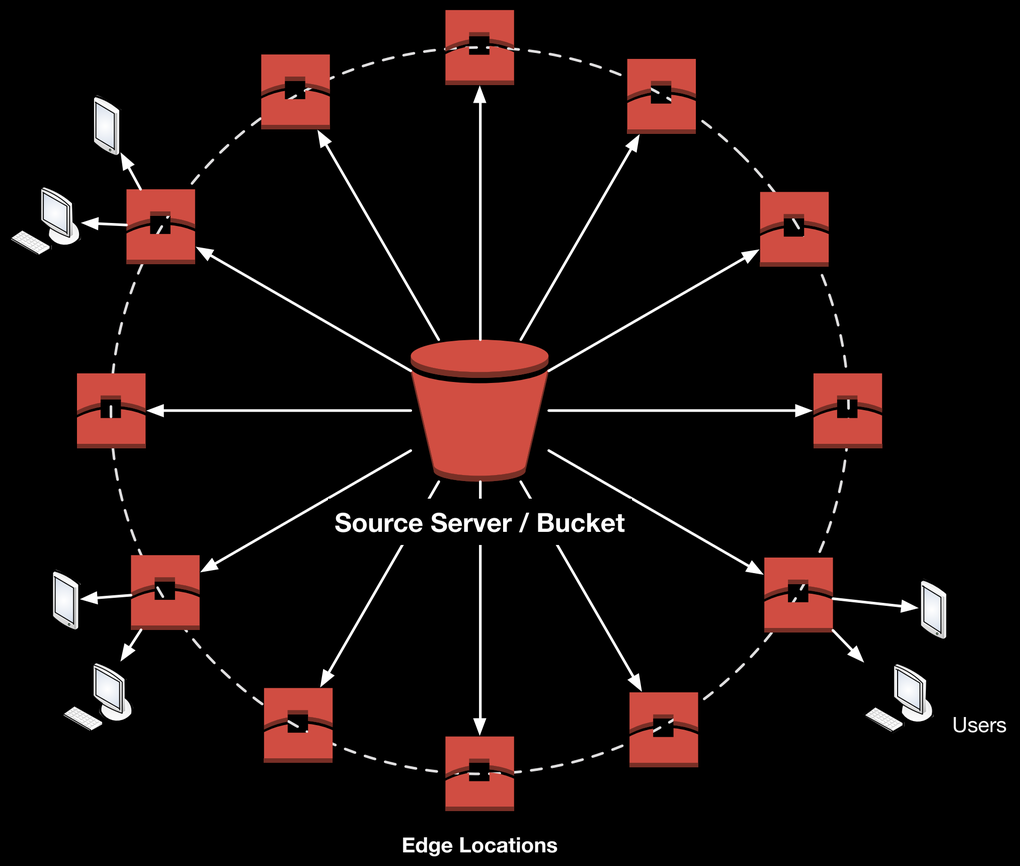
salsa digital

- Secure file-based storage in the cloud.
- 99.999999999% durability.
- From 1 byte to 5TB per file. No account limit.
- Services like Dropbox, Netflix run off S3.
- Great for backups - archive to Glacier. (Arq)
- Run static websites from S3.
- Widely supported by FTP clients like Transmit.
- $0.094 per GB. $0.012 for Glacier.
- Free tier: first 5GB is free.

r.bok.im/f

CloudFront

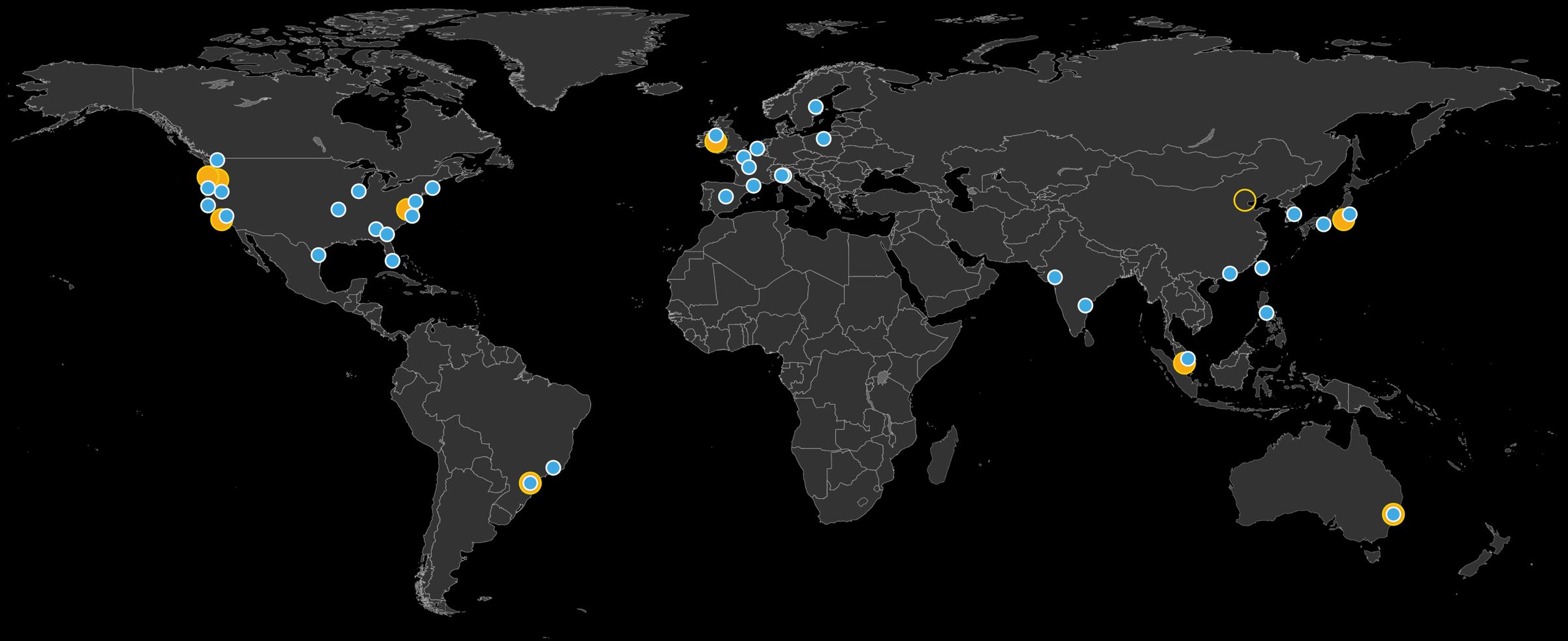
salsa digital



- Content Distribution Network
- Similar to Akamai, Cachefly, etc.
- Content is reverse-proxied and stored close to the end user.
- Use with S3 for ultimate easy, fast static website.
- Handles dynamic content and streaming media.
- Used by IMDB, NASA/JPL, etc.
- $0.12 - $0.25 per GB out.
- Free tier: none.

r.bok.im/g

Edge Locations



r.bok.im/8

DynamoDB

salsa digital

- Managed NoSQL Database
- Super fast: <10ms latency. SSD-based.
- Eventually consistent writes.
- Guaranteed scale/throughput.
- From tiny one row tables to hundreds of GB
- $0.0074 per 10 writes/second. $0.0074 per 50 reads/second. $0.285 per GB.
- Free Tier: 100MB plus 5 writes/second, 10 reads/second.

r.bok.im/h

Elastic Beanstalk

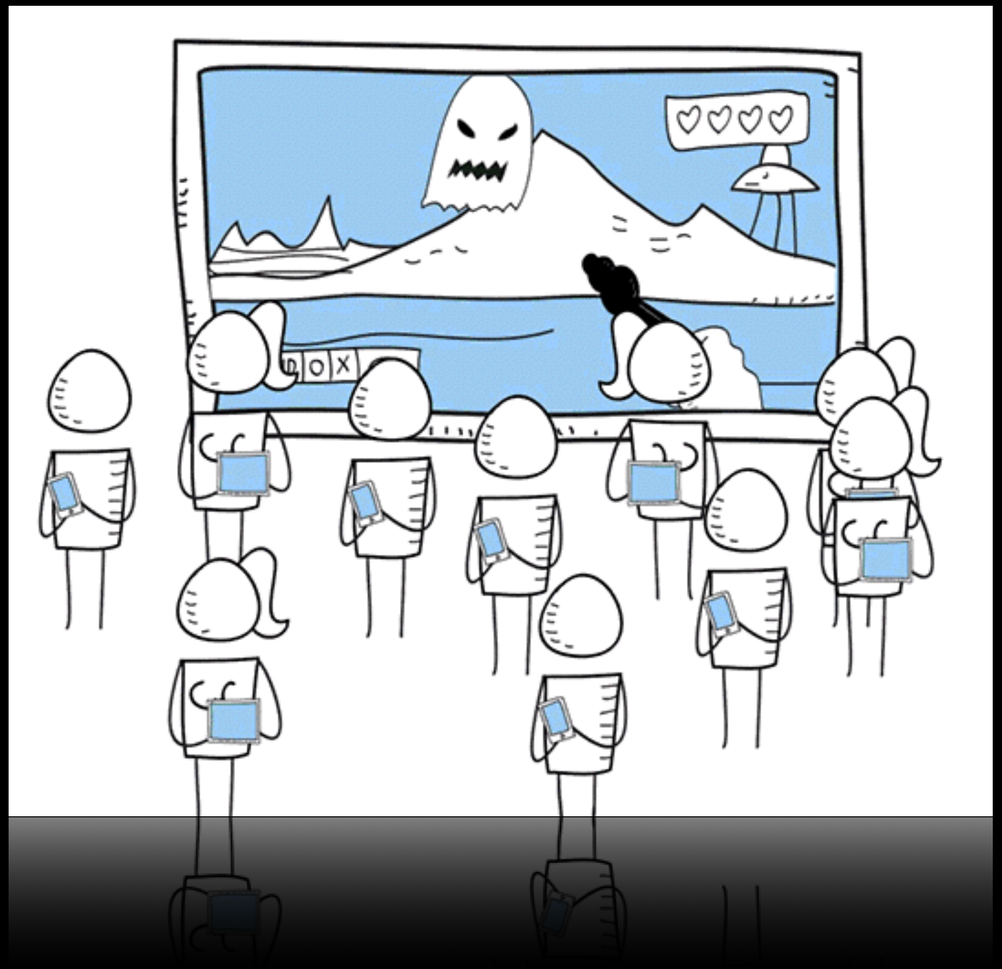
salsa digital

- Rapid deployment Application Container or Platform-as-a-Service solution.
- No loss in flexibility.
- Supports Node.js, PHP, Python, Ruby, .NET and Java.
- Point it at your git repository, configure environment and go.
- Beanstalk will configure EC2 Instances, Load Balancers, Databases, Auto Scaling, Self Healing and deployment for you.
- No cost for Elastic Beanstalk, pay only for resources consumed.

r.bok.im/i

AppStream

salsa digital



- Stream resource intensive apps and games from the cloud.
- App runs in the cloud, H.264 video is streamed to the device in real-time - throw computing power at it.
- Great for photo editing, 3D rendering (render in the cloud and stream the results).
- Limited preview release only.
- Powers the EVE Online Character Generator.

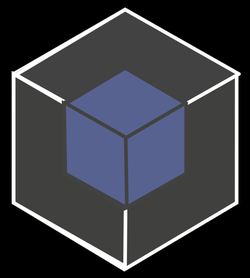
r.bok.im/k

AWS SDK and Toolkits

salsa digital

Java

Python



PHP

.NET

Ruby

NodeJS

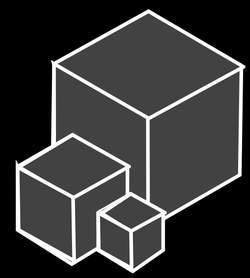
iOS

Android

Visual Studio

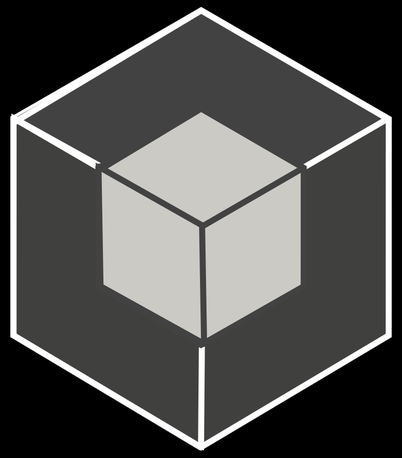
Ellipse

Powershell



CLI

r.bok.im/l



AWS SDK for iOS

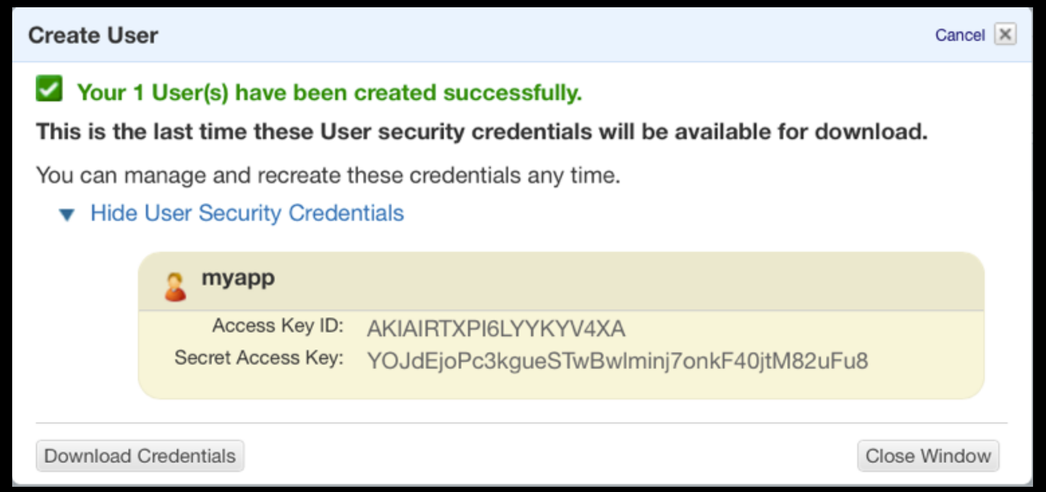
salsa digital

r.bok.im/m

Getting Setup

salsa digital

- Sign up to AWS.
- Create IAM User.
- Save Access/Secret Keys.
- Install the SDK:
  - CocoaPods.
  - Static Library.
  - Pull it in from GitHub.



Example: EC2

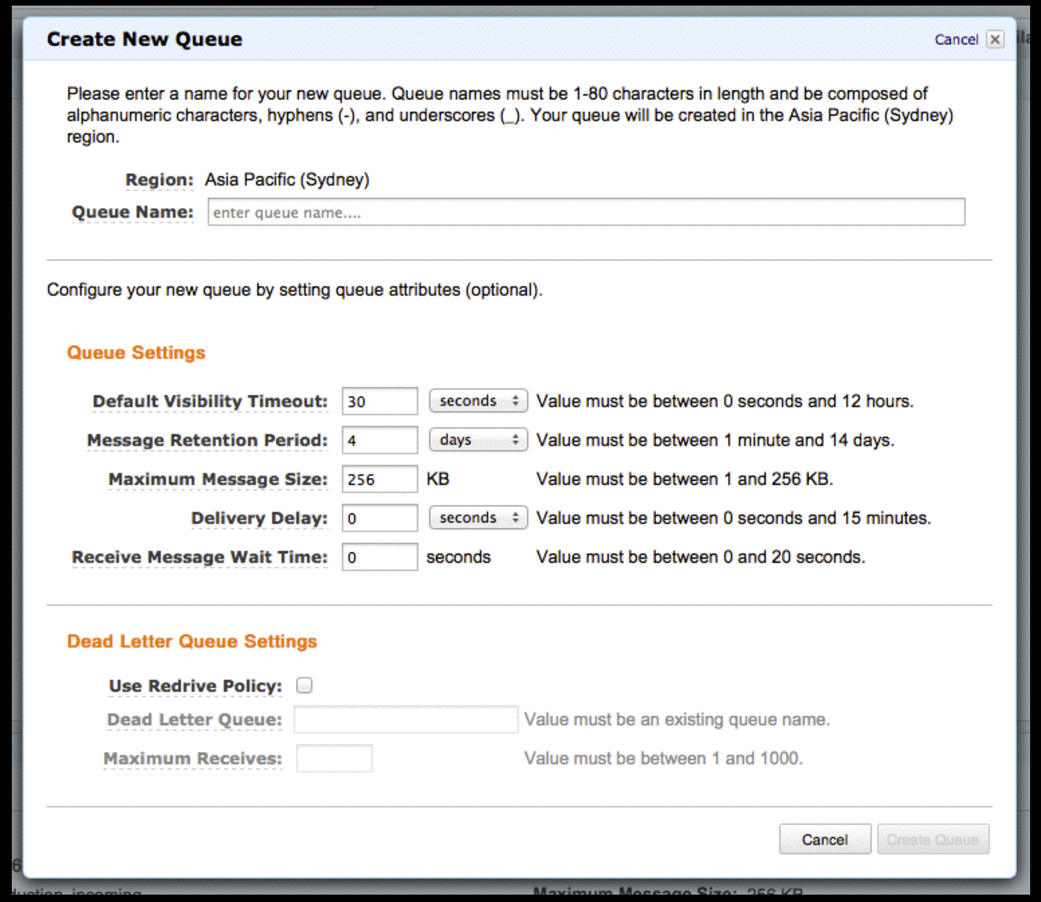
salsa digital

```objc
#import <AWSiOSSDK/EC2/AmazonEC2Client.h>
#import <AWSiOSSDK/AWSRuntime.h>

- (void)getRegions
{
    // Create the client
    AmazonEC2Client *client = [[AmazonEC2Client alloc] initWithAccessKey:@"xxxx" withSecretKey:@"yyyy"];
    [client setEndpoint:[AmazonEndpoints ec2Endpoint:AP_SOUTHEAST_2 secure:YES]];

    // Create the request
    EC2DescribeRegionsRequest *request = [[EC2DescribeRegionsRequest alloc] init];

    // Execute the request
    EC2DescribeRegionsResponse *response = [client describeRegions:request];

    if (response.error != nil)
    {
        // error
    } else
    {
        //
        // response.regions                NSArray of EC2Region Objects
        //
        // Each EC2Region has regionName and endpoint properties.
        //
    }
}
```

Example: EC2 (Asynchronous)


```objc
#import <AWSiOSSDK/EC2/AmazonEC2Client.h>
#import <AWSiOSSDK/AWSRuntime.h>

- (void)getRegions
{
    // Create the client
    AmazonEC2Client *client = [[AmazonEC2Client alloc] initWithAccessKey:@"xxxx" withSecretKey:@"yyyy"];
    [client setEndpoint:[AmazonEndpoints ec2Endpoint:AP_SOUTHEAST_2 secure:YES]];

    // Create the request
    EC2DescribeRegionsRequest *request = [[EC2DescribeRegionsRequest alloc] init];
    [request setDelegate:self];

    // Execute the request
    [client describeRegions:request];
}

- (void)request:(AmazonServiceRequest *)request didFailWithError:(NSError *)error
{
    // error
}

- (void)request:(AmazonServiceRequest *)request didCompleteWithResponse:(AmazonServiceResponse *)response
{
   //
   // response.regions                NSArray of EC2Region Objects
   //
   // Each EC2Region has regionName and endpoint properties.
   //
}
```

SQS Example



r.bok.im/n

SQS: Send Message

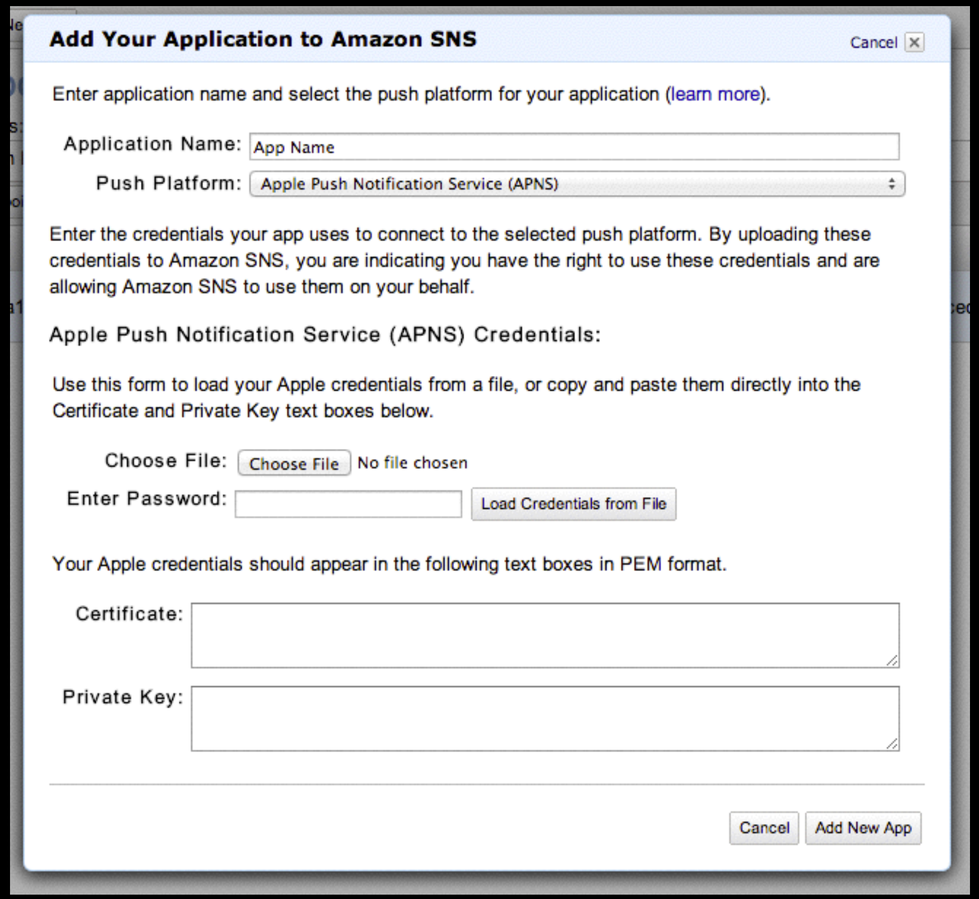

```
#import <AWSiOSSDK/SQS/AmazonSQSClient.h>
#import <AWSiOSSDK/AWSRuntime.h>


// Create the client
AmazonSQSClient *client = [[AmazonSQSClient alloc] initWithAccessKey:@"xxxx" withSecretKey:@"yyyy"];
[client setEndpoint:[AmazonEndpoints sqsEndpoint:AP_SOUTHEAST_2 secure:YES]];

// Create the request
SQSSendMessageRequest *request = [[SQSSendMessageRequest alloc] init];
[request setQueueUrl:@"https://sqs.ap-southeast-2.amazonaws.com/account/queuename"];
[request setMessageBody:@"Put your message body here."];

// Execute the request
SQSSendMessageResponse *response = [client sendMessage:request];

if (response.error != nil)
{
    // error

} else
{
    // response.mD5OfMessageBody              // MD5 digest of the message body
    // response.messageId                     // Unique identifier for the message
}
```

SQS: Receive Message


```objc
// Create the request
SQSReceiveMessageRequest *request = [[SQSReceiveMessageRequest alloc] init];
[request setQueueUrl:@"https://sqs.ap-southeast-2.amazonaws.com/account/queuename"];
[request setMaxNumberOfMessages:@1];

// Execute the request
SQSReceiveMessageResponse *response = [client receiveMessage:request];

if (response.error != nil)
{
    // error

} else
{
    // response.messages                NSArray of SQSMessage objects
    //
    // SQSMessage:
    //      .messageId                  Unique identifier for message
    //      .receiptHandle              Used for deleting the message
    //      .mD5OfBody                  MD5 Digest of message body
    //      .body                       Actual message body
}
```

SNS Example

salsa digital



r.bok.im/o

SNS: Subscribe to Push Notifications


```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    [application registerForRemoteNotificationTypes:UIRemoteNotificationTypeAlert|UIRemoteNotificationTypeBadge|
UIRemoteNotificationTypeSound];
    return YES;
}

- (void)application:(UIApplication *)application didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)deviceToken
{
    AmazonSNSClient *client = [[AmazonSNSClient alloc] initWithAccessKey:@"xxx" withSecretKey:@"yyy"];
    SNSCreatePlatformEndpointRequest *request = [[SNSCreatePlatformEndpointRequest alloc] init];
    NSString *stringToken = [deviceToken hexadecimalRepresentation];
    [request setToken:stringToken];
    [request setCustomUserData:[[[UIDevice currentDevice] identifierForVendor] UUIDString]];
    [request setPlatformApplicationArn:@“arn:aws:sns:us-east-1:123456789012:app/APNS/App-Name"];

    [client createPlatformEndpoint:request];
}
```

SNS: Send Push Notification

salsa digital

```
// Create the request
SNSPublishRequest *request = [[SNSPublishRequest alloc] init];
[request setTargetArn:@“arn:aws:sns:us-east-1:262192482026:endpoint/APNS_SANDBOX/Kestrel-Sandbox/
5910b677-612b-3cec-a3a3-e476fe5e73d1"];
[request setMessage:@"Message body."];

// Execute the request
SNSPublishResponse *response = [client publish:request];

if (response.error != nil)
{
    // error

} else
{
    // response.messageId              Unique identifier for message
}
```

S3: Upload Example


```objc
- (void)takeAndUploadPhoto
{
    UIImagePickerController *controller = [[UIImagePickerController alloc] init];
    [controller setSourceType:UIImagePickerControllerSourceTypeCamera];
    [controller setDelegate:self];
    [self.navigationController presentViewController:controller animated:YES completion:NULL];
}

- (void)imagePickerController:(UIImagePickerController *)picker didFinishPickingMediaWithInfo:(NSDictionary *)info
{
    AmazonS3Client *client = [[AmazonS3Client alloc] initWithAccessKey:@"xxx" withSecretKey:@"yyy"];
    UIImage *image = [info objectForKey:UIImagePickerControllerEditedImage] ?: [info
objectForKey:UIImagePickerControllerOriginalImage];

    if (image != nil) {
        S3PutObjectRequest *request = [[S3PutObjectRequest alloc] initWithKey:@"photo.jpg" inBucket:@"MyS3Bucket"];
        [request setContentType:@"image/jpeg"];
        [request setData:UIImageJPEGRepresentation(image, 0.9f)];

        S3PutObjectResponse *response = [client putObject:request];
        if (response.error != nil) {
            // error handling
        }
        else {
            // success handling
        }
    }
    [self.presentedViewController dismissViewControllerAnimated:YES completion:NULL];
}
```

DynamoDB NSIncrementalStore


```
- (NSPersistentStoreCoordinator *)persistentStoreCoordinator
{
    if (_persistentStoreCoordinator != nil) {
        return _persistentStoreCoordinator;
    }

    [NSPersistentStoreCoordinator registerStoreClass:[AWSPersistenceDynamoDBIncrementalStore class]
                                        forStoreType:AWSPersistenceDynamoDBIncrementalStoreType];

    NSDictionary *tables = @{ @"Entity1": @"table1", @"Entity2": @"table2" };
    NSDictionary *hashKeys = @{ @"Entity1": @"hashKeyAttribute1", @"Entity2": @"hashKeyAttribute2" };
    NSDictionary *versionKeys = @{ @"Entity1": @"versionAttribute", @"Entity2": @"versionAttribute" };
    AmazonDynamoDBClient *client = [[AmazonDynamoDBClient alloc] initWithCredentials:nil];

    NSDictionary *options = @{
                              AWSPersistenceDynamoDBTableMapper: tables,
                              AWSPersistenceDynamoDBHashKey: hashKeys,
                              AWSPersistenceDynamoDBVersionKey: versionKeys,
                              AWSPersistenceDynamoDBClient: client
                              };

    NSError *error = nil;
    _persistentStoreCoordinator = [[NSPersistentStoreCoordinator alloc] initWithManagedObjectModel:[self managedObjectModel]];
    if (![_persistentStoreCoordinator addPersistentStoreWithType:AWSPersistenceDynamoDBIncrementalStoreType configuration:nil URL:nil
options:options error:&error]) {
        // error handling
    }

    return _persistentStoreCoordinator;
}
```

Official SDK Support

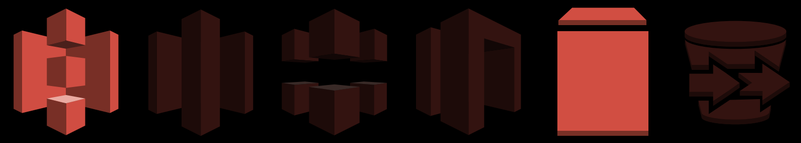
salsa digital

Compute and Networking

Application Services

Analytics

Storage and Content Delivery



Deployment and Management

Database

salsa digital

AWS SDK for iOS

Unsigned Apps Edition

r.bok.im/p

Benefits

salsa digital

- Fully asynchronous.
- Completion blocks instead of delegates.
- Strongly Typed. Enums instead of NSStrings.
- Built out of Mantle. Supports <NSCopying> and <NSCoding> of all objects. Serialisable to JSON.
- Uses KissXML and MantleXMLAdapter. Serialisable to XML.
- Built on NSOperations and NSURLSessions. Use familiar tools to control communications with AWS.

Unsigned Apps SDK Support

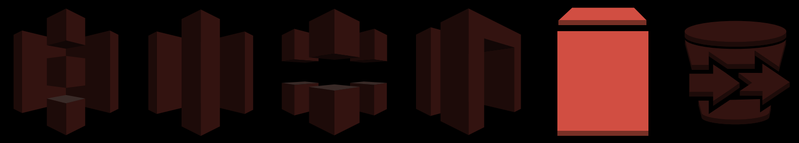
salsa digital

Compute and Networking

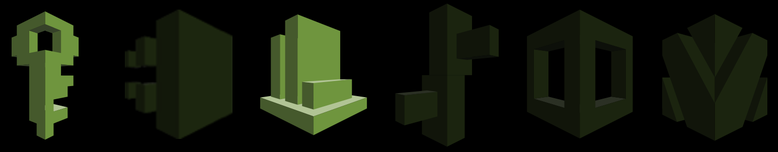
Application Services

Analytics

Storage and Content Delivery



Deployment and Management



Database

Example: Starting an EC2 Instance


```objc
#import <UAAWSSDK/UAEC2.h>


// Create the request
UAEC2StartInstancesRequest *request = [[UAEC2StartInstancesRequest alloc] init];
[request addInstanceID:@"i-12345678"];

// Execute the request
[request invokeWithOwner:self completionBlock:^(UAEC2StartInstancesResponse *response, NSError *error) {

    if (error != nil) {
        // error handling
    }
    else {
        // do something with response
    }
}];
```

Example: Waiting for Instance to Start


```objc
// Create the request
UAEC2DescribeInstancesRequest *request = [[UAEC2DescribeInstancesRequest alloc] init];
[request addInstanceID:@"i-12345678"];

// Execute the request, wait until the instance has started
[request waitWithOwner:self
   untilValueAtKeyPath:@"reservations.@unionOfObjects.instances.@unionOfObjects.state"
             isInArray:@[ @(UAEC2InstanceStateRunning) ]
       completionBlock:^(UAEC2DescribeInstancesResponse *response, NSError *error) {
    if (error != nil) {
        // do something with the error
    }
    else {
        // do something with the instance
    }
}];
```

salsa digital

Questions?

Rob Amos - rob@salsadigital.com.au