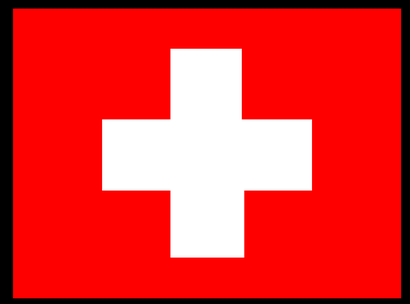
MAX

# Adobe Presentation

Daniel Wanja | Flex, Ruby on Rails, in the cloud

# Hi...I'm Daniel!

- @danielwanja
- onrails.org
- flexonrails.com
- appsden.com
- d@n-so.com



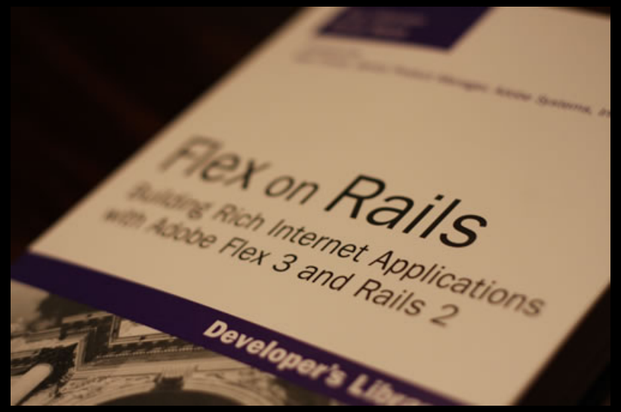
+

=

#todo: add more info?

# Some of my work/portfolio

MAX
CONNECT. DISCOVER. INSPIRE.













Adobe

# Thanks to Pinnacol

- Done a lot of work for them with many teams
- Part of the tech team, Cool stuff
- Flex + Rails
- #todo: get few screen shots

# What's this about?

- Well Flex is cool, Rails is Cool, and the Cloud is awesome...
- .. So let's check out how we can use it together
- Also,  the tools used in this presentation are used more and more on many new websites...

#todo: get concrete examples, wooga.com (amf, rails, redis)

# Agenda...crazy...crazy



1. Intro
2. Anatomy of a Rails app
3. Different Options to Connect Flex to a Server
4. Flex and Vanilla Rails #todo: maybe I skip that one
5. a) Flex app that use Bulk API #todo: pick a or b
   or b) RestfulX
6. Flex Gotcha's
7. Redis a Key/Value store and more
8. WebSockets and Pusher
9. MongoDb a nosql schema less database
10. Ruby and AMF
11. Server side compression using Zlib

# 4

## Anatomy of a Rails app

# Anatomy of a Rails app...in the cloud



- #todo: provide diagram
- #todo: explain Rails is the guardian, last line of defense,
- #todo: this is the why Rails picture

- #todo: why in the cloud
- convenience, managed for you, scale, by the hour?
- everything we'll show in the cloud can be developed on your Mac locally or on you Linux box
- Windows? (I don't know)

# Different Options to Connect Flex to a Server

- HTTPService: XML/JSON/AMF
- Sockets/WebSockets

# Flex and Vanilla Rails



If you don't have rails yet

```
$ gem install rails
...
Successfully installed rails-3.0.8
22 gems installed
...
Installing ri documentation for rails-3.0.8...
...
Installing RDoc documentation for rails-3.0.8...
```

# Flex and Vanilla Rails - create an empty rails app

```
$ rails new adobemax-first-project
create
create  README
create  Rakefile
create  config.ru
create  .gitignore
create  Gemfile
create  app
create  app/controllers/application_controller.rb
create  app/helpers/application_helper.rb
...
```

# Flex and Vanilla Rails - setup and add to git

change the Gemfile

replace gem 'sqlite3' by gem 'pg'

$ brew install postgresql

$ env ARCHFLAGS="-arch x86_64" gem install pg Note: install's postgresql using homebrew

$ bundle install

$ git add . $ git commit -m "First checkin

brew install only needed if you want to run it locally...

# Flex and Vanilla Rails - send to heroku

MAX CONNECT. DISCOVER. INSPIRE.

```
$ git push heroku master
-----> Heroku receiving push
-----> Rails app detected
-----> Detected Rails is not set to serve static_assets
       Installing rails3_serve_static_assets... done
-----> Configure Rails 3 to disable x-sendfile
       Installing rails3_disable_x_sendfile... done
-----> Configure Rails to log to stdout
       Installing rails_log_stdout... done
-----> Gemfile detected, running Bundler version 1.0.7
       Unresolved dependencies detected; Installing...
...
-----> Compiled slug size is 3.9MB
-----> Launching... done, v4
       http://adobemax-first-project.heroku.com deployed to Heroku
```



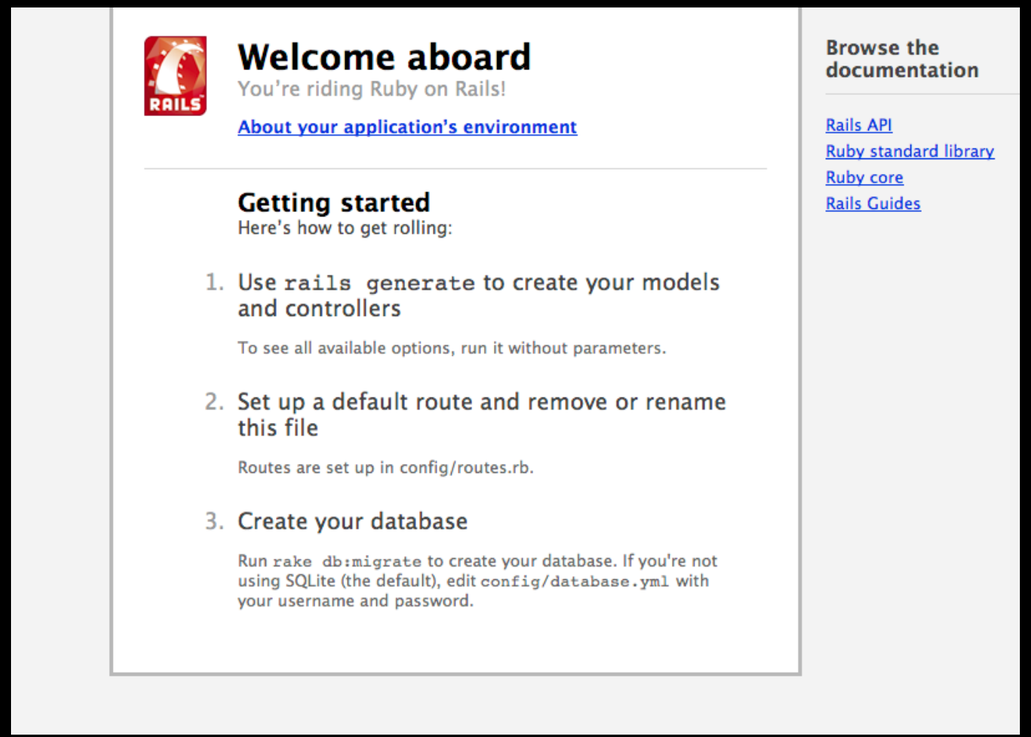


Adobe

# Flex and Vanilla Rails - The App is Live!

http://adobemax-first-project.heroku.com

http://freezing-autumn-489.herokuapp.com/ #todo: get app with proper name



# Flex and Vanilla Rails - other heroku commands

```
$ heroku ps
Process      State              Command
------------ ------------------ ------------------------------
web.1        up for 1m          thin -p $PORT -e $RACK_ENV -R $HER..
```

# Flex and Vanilla Rails - other heroku commands

MAX CONNECT. DISCOVER. INSPIRE.

$ heroku logs

```
2011-06-10T02:07:03+00:00 heroku[api]: Release v2 created by d@n-so.com
2011-06-10T02:07:08+00:00 heroku[api]: Add-on update shared-database
2011-06-10T02:07:08+00:00 heroku[api]: Release v3 created by d@n-so.com
2011-06-09T20:57:23-07:00 heroku[slugc]: Slug compilation started
2011-06-10T03:58:11+00:00 heroku[api]: Deploy 3897c3b by d@n-so.com
2011-06-10T03:58:11+00:00 heroku[api]: Release v4 created by d@n-so.com
2011-06-10T03:58:11+00:00 heroku[web.1]: State changed from created to starting
2011-06-09T20:58:11-07:00 heroku[slugc]: Slug compilation finished
2011-06-10T03:58:15+00:00 heroku[web.1]: Starting process with command: `thin -p 12924 -e production -
R /home/heroku_rack/heroku.ru start`
2011-06-10T03:58:17+00:00 app[web.1]: >> Thin web server (v1.2.6 codename Crazy Delicious)
2011-06-10T03:58:17+00:00 app[web.1]: >> Maximum connections set to 1024
2011-06-10T03:58:17+00:00 app[web.1]: >> Listening on 0.0.0.0:12924, CTRL+C to stop
```





Adobe

# Flex and Vanilla Rails - other heroku commands

```
$ heroku run console
Running console attached to terminal... up, run.2
Loading production environment (Rails 3.0.8)
irb(main):001:0> Post.count
=> 0
```

# Let's add some useful code

```
$ rails generate scaffold post title:string body:text
invoke  active_record
create    db/migrate/20110610040608_create_posts.rb
create    app/models/post.rb
invoke    test_unit
create      test/unit/post_test.rb
create      test/fixtures/posts.yml
 route  resources :posts
invoke  scaffold_controller
create    app/controllers/posts_controller.rb
invoke    erb
create      app/views/posts
create      app/views/posts/index.html.erb
...
```

```
$ heroku run rake db:migrate
==  CreatePosts: migrating
=============================================
-- create_table(:posts)
   -> 0.0133s
==  CreatePosts: migrated (0.0134s)
=============================================
```

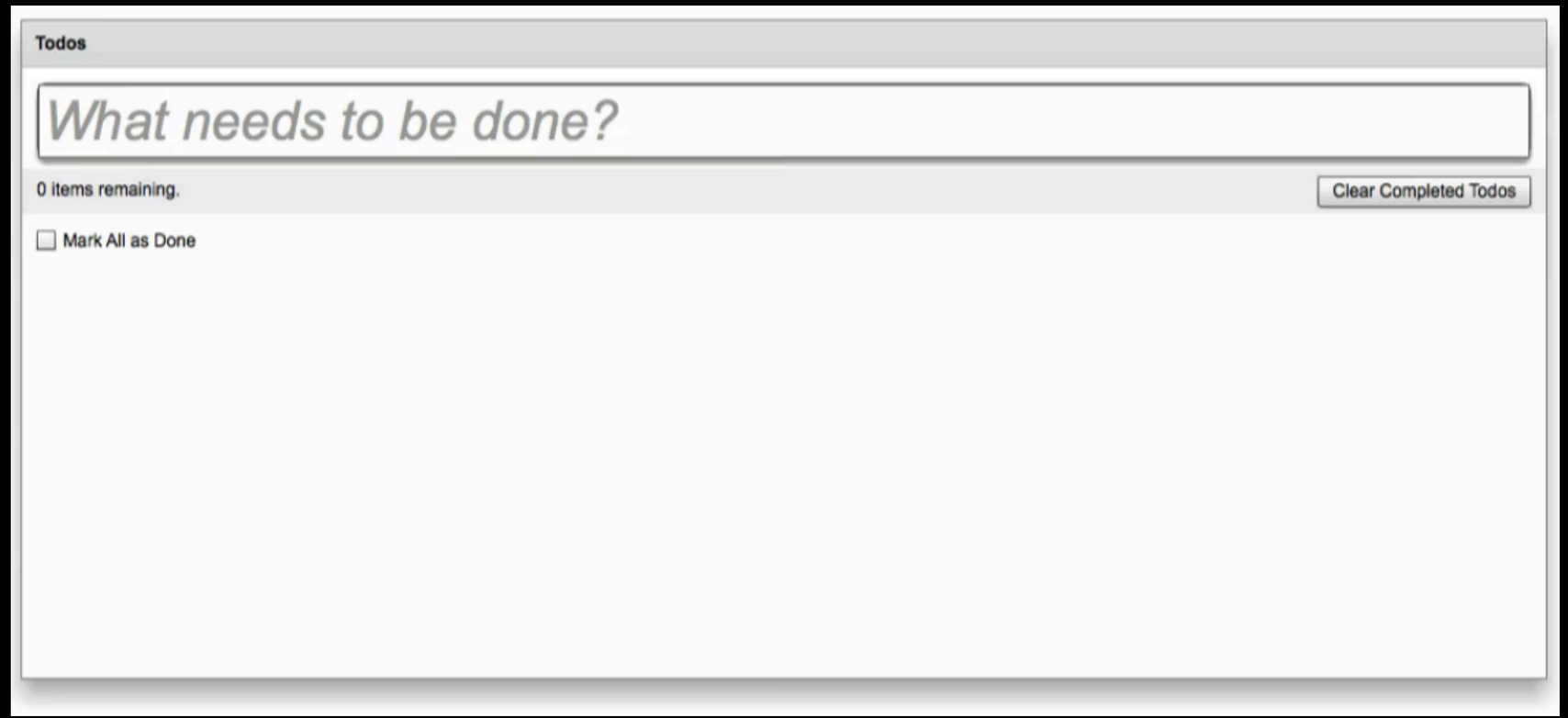
# 5

## Bulk API

# Flex app that use Bulk API

- Rails: https://github.com/drogus/bulk_api
- Flex: https://github.com/danielwanja/bulk_data_source_flex

# Bulk API demo



# Bulk API - Rails



Create the Rails application

```
$ rails new todos
```

Edit the Gemfile and add

```
gem 'bulk_api'
```

Setup the bulk_api, add a todo resource, migrate the database and start the server

```
$ bundle install
$ rails generate bulk:install
$ rails g scaffold todo title:string done:boolean
$ rake db:migrate
$ rails s
```

Ok, we are done on the Rails side.

# Bulk API - Flex

```
package resources
{
    import bulk_api.BulkResource;

    [RemoteClass(alias="Todo")]
    public dynamic class Todo extends BulkResource
    {
        public function Todo(attributes:Object=null) {
            super(attributes);
        }

        resource("todos", Todo)
    }
}
```

# Bulk API - Flex

MAX CONNECT. DISCOVER. INSPIRE.

## Create

```
var todo:Todo = new Todo({title:newTodo.text, done:false})
var call:AsyncToken = BulkResource.create(Todo, {todos:new ArrayCollection([todo])});
call.addResponder(new AsyncResponder(addTodoResult, faultHandler));

private function addTodoResult(event:ResultEvent, token:Object=null):void
{
  todos.addItem(event.result.todos.getItemAt(0));
  newTodo.text = "";
}
```

## Update

```
var call:AsyncToken = BulkResource.update(Todo, {todos:new ArrayCollection(todos)});
```

## Delete

```
var done:Array = [];
for each (var todo:Todo in todos) {
    if (todo.done) done.push(todo.id);
}
var call:AsyncToken = BulkResource.destroy(Todo, {todos:new ArrayCollection(done)});
```





Adobe

# 6

## Flex Gotcha's

# Flex Gotcha's

- authentication_token
- crossdomain.xml
- error_handling

#todo: expand

#todo: create flex_error_handling gem

# 7

Redis

# Redis - uses?

- Message Passing
- Logging
- Social Graphs
- Asset Caching

# Redis

- sets
- sorted sets
- counters
- lists
- publish-subscribe

# Redis

- keys i.e. "user:1000:password" or "comment:1234:reply.to"
- lists: LPUSH, RPUSH, LRANGE
- sets: SADD, SUNION, SINTER, SDIFF
- hash: HMSET, HGETALL

# Redis - Enable on Heroku

```
$ heroku addons:add redistogo:nano
Adding redistogo:nano to freezing-autumn-489... done, v14 (free)
```

# Redis - config your Rails app

- Add to you gem file: gem 'redis'
- Add config/initializers/redis.rb

```
uri = URI.parse(ENV["REDISTOGO_URL"])
REDIS = Redis.new(:host => uri.host, :port => uri.port, :password => uri.password)
```

# Flex to Redis direct?

- Flex to Redis directly is not a good idea
- But possible, i.e. https://github.com/danielwanja/redis_flex
- Better approach is used Rails as a wrapper to provide security
- Flex --Http-->Rails--Socket-->Redis

# Redis from Rails

```
$ heroku run console
Running console attached to terminal... up, run.5
Loading production environment (Rails 3.0.8)
irb(main):001:0> REDIS
=> #<Redis client v2.2.1 connected to redis://bass.redistogo.com:9581/0
(Redis v2.2.5)>
irb(main):002:0> REDIS.set("a", 42)
=> "OK"
irb(main):003:0> REDIS.get "a"
=> "42"
```

# Redis - install on OSX

```
  $ brew install redis
$ redis-server
$ redis-cli    # command line
```

# Demo Time!



- Twitter Like ... server
- #todo: show example use case (i.e. set and intersection for social network)
- note: see http://forrst.com/posts/SocialGraphingwithRedisSetsandPython-Rkd note: see http://devblog.bu.mp/how-we-use-redis-at-bump
- note: see http://flazz.me/redis-the-ak-47-of-database

# 8

## WebSockets and Pusher

# WebSockets and Pusher - Enable on Heroku

```
$ heroku addons:add pusher:test
Adding pusher:test to freezing-autumn-489... done, v7 (free)
```

See http://pusher.com/docs/heroku

Signup at http://pusher.com/

# Publish/Subscribe uses

- messaging
- screen synchronization
- notification
- inbox/queue count update
- system wide alarms

# Pusher and Rails?

- Rails is not needed
- Can be Flex-->Pusher-->Flex
- However, Rails can be useful

# Pusher and Flex?

- https://github.com/smakinson/Pusher-ActionScript-Library
- https://github.com/y8/websocket-as

# Add Pusher to your Rails app

- Add gem 'pusher' to your Gemfile
- `$ bundle install`
- `Get your pusher api key`
-

# config/environments/development.rb

```
require 'pusher'
Pusher.app_id = 6182
Pusher.key = 'a0b74a20a5d8df2db432'
Pusher.secret = '057c1aac0e15defa3a55'
```

# Rails: add message controller

```
  $ rails g controller Message hello
create  app/controllers/message_controller.rb
 route  get "message/hello"
invoke  erb
create    app/views/message
create    app/views/message/hello.html.erb
invoke  test_unit
create    test/functional/message_controller_test.rb
invoke  helper
create    app/helpers/message_helper.rb
invoke    test_unit
create      test/unit/helpers/message_helper_test.rb
```

# Rails: send

```
class MessageController < ApplicationController

  def hello
    Pusher['test_channel'].trigger('greet', {
      :greeting => "Hello there!"
    })
  end

end
```

# Flex: receive

```
private var pusher:Pusher;
private var channel:Channel;
protected function setup():void
{
    pusher = new Pusher('a0b74a20a5d8df2db432', "pusherexample" );
    channel = pusher.subscribe('test_channel');
    channel.bind('greet', gotData);
}

protected function gotData(data:Object):void {
    Alert.show(ObjectUtil.toString(data));
}
```

# Flex: send (not yet supported)

```
channel.trigger('greet', {greeting: message.text});
```

# 9

MongoDb

# MongoDb - a schema less, nosql, database...say what?



- #todo: provide context why it could be useful
- #todo: provide example of who uses it
- #todo: provide list of alternatives (Kyoto Cabinet, Casendra, ...)

# MongoDb - on Heroku

```
$ heroku addons:add mongohq:free
Adding mongohq:free to freezing-autumn-489... done, v17 (free)
```

# MongoDb and Ruby



Multiple gems

- http://mongoid.org/
- http://rubygems.org/gems/mongo
- http://mongomapper.com/
- https://github.com/mongoid/mongoid

We'll pick mongoid for this demo

# MongoDb - console

```
   $ mongo        # command line
> db.foo.save( { a : 1, children: [{a: 1.1}, {a: 1.2}] } )
> db.foo.find()
{ "_id" : ObjectId("4df6b7a6a170e0529dc98e42"), "a" : 1 }
> db.foo.remove({a:1})
```

## Create a new Rails app

```
$ rails new documents
```

## Update the Gemfile

```
# gem 'sqlite3'
gem "mongoid", "~> 2.0"
gem "bson_ext", "~> 1.3"
```

## Configure MongoDb

```
$ rails g mongoid:config
```

# MongoDb - let's add an article 'table' ...

```
 $ rails g scaffold article name:string content:text
invoke  mongoid
create    app/models/article.rb
invoke    test_unit
create      test/unit/article_test.rb
create      test/fixtures/articles.yml
 route  resources :articles
invoke  scaffold_controller
create    app/controllers/articles_controller.rb
...
```

# MongoDb - ... and some comments

```
  $ rails g model comment name:string content:text
invoke  mongoid
create    app/models/comment.rb
invoke    test_unit
create      test/unit/comment_test.rb
create      test/fixtures/comments.yml
...
```

# MongoDb - article.rb and comment.rb

```
class Article
  include Mongoid::Document
  field :name, :type => String
  field :content, :type => String
  embeds_many :comments
end
```

```
class Comment
  include Mongoid::Document
  field :name, :type => String
  field :content, :type => String
  embedded_in :article, :inverse_of => :comments
end
```

# MongoDb

- Similar to Rails: Article.all or Article.fist
- but more flexible: Article.first.as_document
- #todo: explain more flexible (i.e dynamic attributes, different attribute per version, no migrations, ...)

# MongoDb - similar to ActiveRecords

- Article.count #=> 3
- Article.first
  # => #<Article _id: 4df6ba248a45571e1e000002, _type: nil, name: "Using MongoDb", content: "...with mongoid">
- Article.create :name => "new entry", :content => "...with new content"

# MongoDb - but also more flexible (i.e. bypass mongoing)

Article.first.as_document

=> {"_id"=>BSON::ObjectId('4df6ba248a45571e1e000002'), "name"=>"Using MongoDb", "content"=>"...with mongoid"}

ruby-1.9.2-p180 :011 > Article.first.as_document.class

=> BSON::OrderedHash

```
Mongoid::Factory.build Article,
    { name: "Using MongoDb",
      content: "...with mongoid",
      comments: [  { :name=>"Daniel",
                     :content=>"this is very flexible"}],
      other: "Something else"
    }
```

# MongoDb - #todo

- Build example app that has a "dynamic" section
- I.e. Dynamic i.e. check "custom" attributes and allow
- to attach anything: expand row columns
- Add a finder on custom attributes

# MongoDb - install on OSX

```
   $ brew install mongodb
$ mongod
$ mongo        # command line
```

# 10

## Ruby and AMF

# Ruby and AMF

- #todo: build demo with
- https://github.com/warhammerkid/rails3-amf
- https://github.com/warhammerkid/rocket-amf

# 11

Zlib

# Zlib

- #todo: explain usage for http://vault.ncaa.com
- Show ratio: 5Mb -> 130Kb, 4.1Mb -> 20Kb

# Zlib - Ruby

```
require 'zlib'
File.open('data_to_compress.zlib','w') do |f|
    z = Zlib::Deflate.new(9)   # 9 = Z_BEST_COMPRESSION
    f.write z.deflate(s.to_xml, Zlib::FINISH)
    z.close
end
```

Thanks to the @theaboutbox

# Zlib - Flex

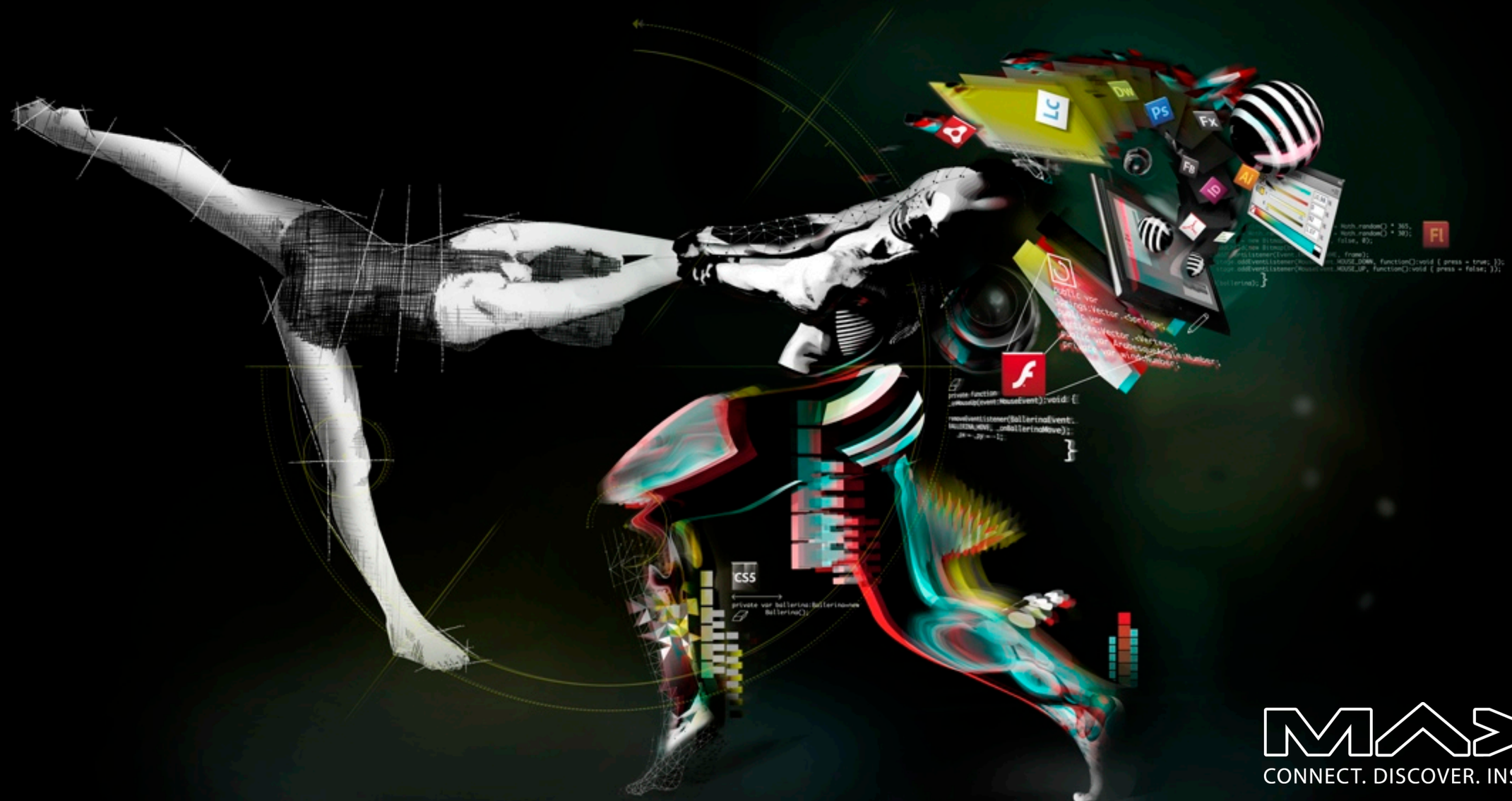
- http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/flash/utils/CompressionAlgorithm.html

```
import flash.utils.ByteArray;

private function loadData():void {
    var loader:URLLoader = new URLLoader();
    loader.dataFormat = "binary";
    loader.addEventListener(Event.COMPLETE, completeHandler);
    var request:URLRequest = new URLRequest("data.zlib");
    loader.load(request);
}
private function completeHandler(event:Event):void {
    var loader:URLLoader = URLLoader(event.target);
    var ba:ByteArray = loader.data;
    ba.uncompress();
    var s:String = ba.toString();
    var xml:XML = new XML(s);
}
```

# QA?

# Code Sample

```
package tests
{
    import org.flexunit.asserts.assertEquals;

    public class AdditionTest
    {
        [Test]
        public function testOnePlusOne():void
        {
            assertEquals(2, 1+1);
        }
    }
}
```




MAX
CONNECT. DISCOVER. INSPIRE.