

ABCs of APIs with Node.js

wifi: Village Events

password: atlantatechvillage

Greg Rewis

grewis@apigee.com

@garazi

#apigee

I work for Apigee

Walgreens

ebay



FT
FINANCIAL
TIMES



Chegg VIACOM

redbox

swisscom



MORNINGSTAR

CITRIX



Earth Networks



INNOTAS
On Time. On Plan. On Demand.

infogroup
NONPROFIT SOLUTIONS

meredith

PEARSON



TRADEKING

Autodata



IQT
IN-Q-TEL



mBlox

MARKS & SPENCER

Digital River

LIVE NATION



Telefonica



EQUIFAX

HCSC
Health Care Service Corporation

SAMTRAFIKEN
SAMTRAFIKEN I SVERIGE AB

verizon

kt

dish
NETWORK

AT&T U-verse

European Patent Office

apigee

Why do we do these workshops?
Why free?



Apigee is always free for developers

Free Hosted Accounts

25GB storage limit,

10M push notifications/month

no API/bandwidth limit

Commercial use OK

Paid plans available for large companies if you need to **deploy** on **your own servers** or SLAs, 4 nines, multi-region, phone support, more storage, etc.

Who are **you**?

Agenda

Section 1

Introduction to APIs

Section 2

Node.js Crash Course

Section 3

Best Practices in API Design

Section 4

Intro to Apigee127

Agenda

Section 5

Building API Specs/Docs

Section 6

Creating Controllers

Section 7

API Management

Section 8

Deploying the APIs



API Fundamentals



What is an API?

Application Programming Interface (abbr.: API)

noun

a system of tools and resources in an operating system, enabling developers to create software applications

What is an API?

A service that:

- specifies how one application can talk to another
- exposes some of an app's internal functions or logic to the outside world
- makes it possible for applications to share data
- is comprised of a set of HTTP/ HTTPS endpoints



Applications use APIs...



...like humans use websites

Node.js Crash Course

Who is using Node.js?

The New York Times

Medium

PayPal™

Microsoft

LinkedIn

DOWJONES



A TimeWarner Company

Walmart



codenvy

yammer™



Cloud9 IDE
Your code anywhere, anytime



YAHOO!

NETFLIX

eBay™



Groupon®

Node.js success stories

- Groupon re-implemented their system in Node.js which resulted in page load times dropping by a whopping 50%.
- LinkedIn moved to Node.js from Rails for their mobile traffic, reducing the number of servers from 30 to 3 (90% reduction) and the new system was up to 20x faster.

Node.js success stories

- PayPal reported: double the number of requests per-second and reduced response time by 35% or 200 milliseconds.
- In addition, PayPal measured a 2x increase in developer productivity, where it took half the number of developers to deliver an application when compared to Java, and it was delivered in less time

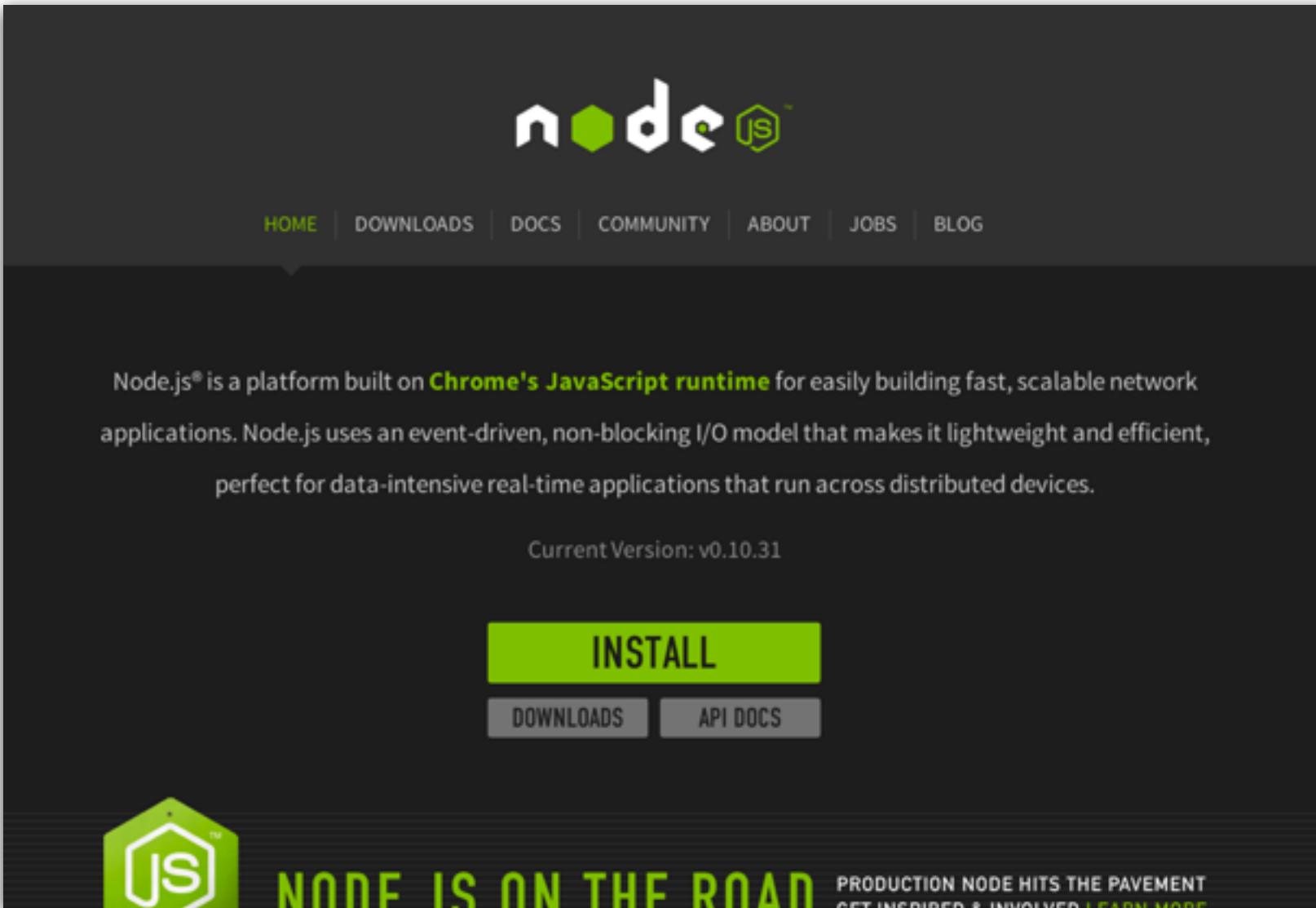
Node.js success stories

- WalMart Labs had a bumper launch with Node.js in 2013, where they put all of their Mobile traffic through Node.js on black-friday, the busiest shopping period of the year.
- The WalMart servers didn't go over 1% CPU utilization and the team did a deploy with 200,000,000 users online.

Why node.js?

- Javascript running on the server
- Asynchronous, event-driven
- HTTP is a first-class citizen of the Node world
- Exploding in popularity

Node.js installation



The screenshot shows the official Node.js website. At the top center is the Node.js logo. Below it is a navigation bar with links: HOME (highlighted in green), DOWNLOADS, DOCS, COMMUNITY, ABOUT, JOBS, and BLOG. The main content area contains a paragraph about Node.js, mentioning its event-driven, non-blocking I/O model and its use of Chrome's JavaScript runtime. Below this is a note about the current version (v0.10.31). A large green 'INSTALL' button is prominently displayed, with 'DOWNLOADS' and 'API DOCS' buttons positioned below it. At the bottom, there is a logo for 'NODE IS ON THE ROAD' and a call to action: 'PRODUCTION NODE HITS THE PAVEMENT GET INSPIRED & INVOLVED [LEARN MORE](#)'.

node.js™

HOME | DOWNLOADS | DOCS | COMMUNITY | ABOUT | JOBS | BLOG

Node.js® is a platform built on **Chrome's JavaScript runtime** for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Current Version: v0.10.31

INSTALL

DOWNLOADS API DOCS

NODE IS ON THE ROAD

PRODUCTION NODE HITS THE PAVEMENT
GET INSPIRED & INVOLVED [LEARN MORE](#)

Blocking vs. non-blocking

```
<html>
<head>
  <script>
    var test = document.getElementsByTagName('li');
    alert("There are " + test.length + " items");
  </script>
</head>
<body>
  <ul>
    <li>A list item</li>
    <li>Another list item</li>
    <li>A third item in the list</li>
  </ul>
</body>
</html>
```

Blocking vs. non-blocking

```
<html>
<head>
    <script>
        function countItems(el, callback) {
            setTimeout(function() {
                var test = document.getElementsByTagName(el);
                callback(test)
            }, 2000);
        }
        countItems('li', function(test) {
            alert("There are " + test.length + " items");
        });
        alert('Hello World');
    </script>
</head>
<body>
    <ul>
        <li>A list item</li>
        <li>Another list item</li>
        <li>A third item in the list</li>
    </ul>
</body>
</html>
```

The world's simplest web server

```
var http = require('http');

http.createServer(function(req, res) {
    res.end('Hello World');
}).listen(8888, '127.0.0.1');

console.log('Server running at http://127.0.0.1:8888/');
```

```
LapGARbage:api-workshop greg$ node server.js
Server running at http://127.0.0.1:8888/
```

The world of npm

The screenshot shows the npm homepage with the following content:

- Navigation:** HOME, API, BLOG, NODEJS, JOBS.
- Search:** Search Packages.
- User:** Create Account | Login.
- Title:** Node Packaged Modules.
- Total Packages:** 92 546.
- Downloads:** 14 258 858 downloads in the last day, 102 953 208 downloads in the last week, 421 148 617 downloads in the last month.
- Patches welcome!**
- Instructions:** Any package can be installed by using `npm install`. Add your programs to this index by using `npm publish`.
- Recently Updated:**
 - 3m generator-natural
 - 7m protagonist-experimental
 - 8m ah-dashboard-plugin
 - 8m gammaultils
 - 11m gulp-rework
 - 11m koa-rroute
 - 14m gulp-esnext
 - 14m skynet-heartbeat
 - 15m raptor-optimizer
 - 15m sassr
 - More...
- Most Depended Upon:**
 - 6941 underscore
 - 6350 async
 - 5460 request
 - 4775 lodash
 - 3574 commander
 - 3474 express
 - 2696 optimist
 - 2583 coffee-script
 - 2574 colors
 - 2196 mkdirp
 - More...
- Most Starred:**
 - 504 express
 - 294 async
 - 246 grunt
- Most Prolific Recently:**
 - 199 requirify-bot
 - 76 sindresorhus
 - 62 scottcorgan

Modules must be installed

```
var express = require('express');
var app = express();

app.get('/', function (req, res) {
  res.send('Hello World');
}).listen(8888);

console.log('Server running at http://127.0.0.1:8888/');
```

```
LapGARbage:api-workshop greg$ node server.js

module.js:340
    throw err;
    ^
Error: Cannot find module 'express'
    at Function.Module._resolveFilename (module.js:338:15)
    at Function.Module._load (module.js:280:25)
    at Module.require (module.js:364:17)
    at require (module.js:380:17)
```

Installing modules

- Individual modules must be installed using the NPM command line

```
LapGARBage:api-workshop greg$ npm install express
npm http GET https://registry.npmjs.org/express
npm http 200 https://registry.npmjs.org/express
npm http GET https://registry.npmjs.org/express/-/express-4.8.7.tgz
npm http 200 https://registry.npmjs.org/express/-/express-4.8.7.tgz
npm http GET https://registry.npmjs.org/debug/1.0.4
npm http GET https://registry.npmjs.org/domed/0.4.1
```

- Once installed (node_modules folder), they are added to the app using require('module name')

```
var express = require('express');
```

Installing multiple modules

- When a Node.js app launches, it looks for a file, package.json, in the same folder as the main JS file to determine if all of its ‘pieces’ are present

```
{  
  "name": "sample-node-app",  
  "version": "0.0.0",  
  "description": "Sample Node Application",  
  "main": "server.js",  
  "dependencies": {  
    "express": "3.x.x",  
    "usergrid": "x.x.x",  
    "request": "x.x.x"  
  }  
}
```

- Using npm install will install all of the listed dependencies at one time.

Express routes

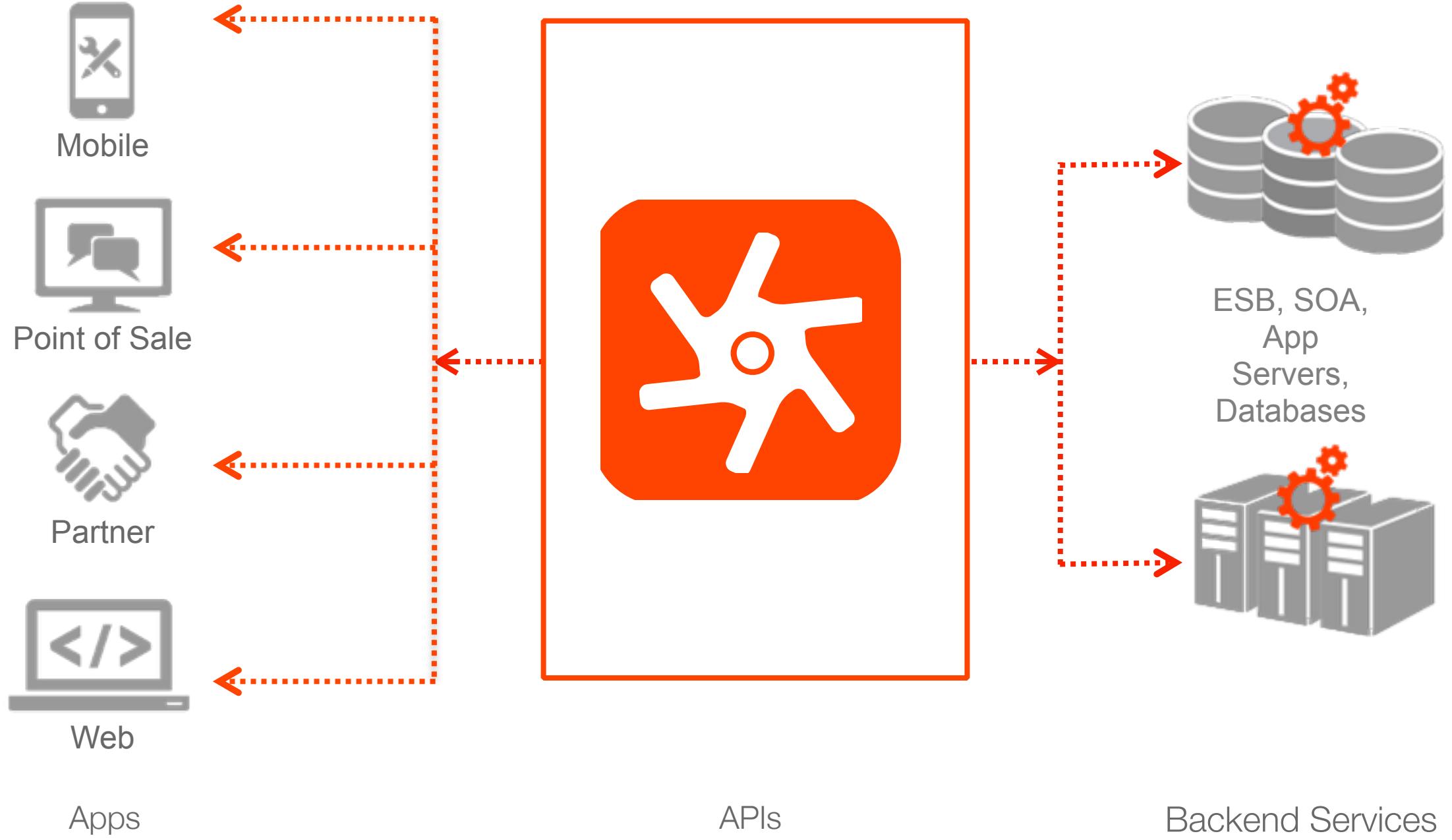
```
var express = require('express');
var app = express();

app.get('/hello', function (req, res) {
  res.send('Hello World');
});

app.use('/', express.static(__dirname));

app.listen(8888, function (req, res) {
  console.log('Server running at http://127.0.0.1:8888/');
});
```

Best Practices in API Design



GREAT DESIGN



ALL THE WORK
YOU DON'T
ASK PEOPLE
WHO USE
YOUR PRODUCT
to **DO**

REBEKAH
COX

Source: <http://stylebizz.com/design-quotes/>





Reluctant API Design

/getDog
/getAllDogs
/petDog
/feedDog
/createRecurringDogWalk
/giveCommand
/healthCheck
/getRecurringDogWalkSchedule
/getLocation
/teachTrick

Reluctant API Design

/getAllDogs	/getNewDogsSince	/get Squirrels ChasingPuppies
/petDog	/getSittingDogs	/newDogFor Owner
/feedDog	/setDogStateTo/saveDog	/getNewDogsAt Kennel Since
/createRecurringWakeUp	/getAll Leashed Dogs	/getSittingDogsAt Park
/giveCommand	/verify Veterinarian Location	/set Leashed DogStateTo
/checkHealth	/createRecurring Medication	/save MommaDogs Puppies
/getRecurringWakeUpSchedule	/doDirect Owner Discipline	
/getLocation	/doCheckupWith Veterinarian	
/getDog	/getRecurring Feeding Schedule	
/newDog	/get Hunger Level	

It happens in the real world



API	NVP	SOAP
AddressVerify	NVP	SOAP
BillOutstandingAmount	NVP	SOAP
Callback	NVP	
CreateRecurringPaymentsProfile	NVP	SOAP
DoAuthorization	NVP	SOAP
DoCapture	NVP	SOAP
DoDirectPayment	NVP	SOAP
DoExpressCheckoutPayment	NVP	SOAP
DoNonReferencedCredit	NVP	SOAP
DoReauthorization	NVP	SOAP
DoReferenceTransaction	NVP	SOAP
DoVoid	NVP	SOAP
GetBalance	NVP	SOAP
GetBillingAgreementCustomerDetails	NVP	SOAP
GetExpressCheckoutDetails	NVP	SOAP
GetRecurringPaymentsProfileDetails	NVP	SOAP
GetTransactionDetails	NVP	SOAP
ManageRecurringPaymentsProfileStatus	NVP	SOAP
ManagePendingTransactionStatus	NVP	SOAP
MassPayment	NVP	SOAP
RefundTransaction	NVP	SOAP
SetCustomerBillingAgreement	NVP	SOAP
SetExpressCheckout	NVP	SOAP
TransactionSearch	NVP	SOAP
UpdateRecurringPaymentsProfile	NVP	SOAP

Design for adoption



A resource is something that lives somewhere

IP: 173.194.70.102 → google.com

Twitter: 14352786 → @garazi

<https://github.com/apigee>

Verbs are an action

Operation	SQL	HTTP
Create	INSERT	POST
Read (Retrieve)	SELECT	GET
Update (Modify)	UPDATE	PUT / PATCH
Delete (Destroy)	DELETE	DELETE

Let's start by applying some of the best-practice principles from REST.

Resource modeling

We only need two base URLs for a resource

A collection

```
/dogs
```

A resource

```
/dogs/1234
```

Collection names – singular or plural?

Foursquare

```
/checkins
```

Groupon

```
/deals
```

Zappos

```
/Product
```

Collection names – abstract or concrete?

Super High

/things

too few things?
go lower

High

/animals



Medium

/dogs



Low

/beagles

too many things?
go higher

Three's company

```
/owners/5678/dogs
```

Sweep complexity behind the “?”



Sweep complexity behind the “?”

```
/dogs?color=red&state=running&location=park
```

Give developers what they want

```
/dogs?fields=name,color,location.postalCode
```

Putting it all together

Resource	POST create	GET read	PUT update	DELETE delete
/dogs	create a new dog	list dogs	replace dogs with new dogs	delete all dogs
/dogs/1234	treat as a collection create new dog in it	show 1234	if exists update 1234 if not create 1234	delete 1234

Actions are collections

(hypothetical)

```
/convert?from=EUR&to=CNY&amount=100
```

DigitalOcean

```
/droplets/{droplet_id}/reboot
```

Facebook

```
/search?q=watermelon&type=post
```

How many versions?



How many versions?

- Maintain at least one version back, to give developers at least one cycle to react before obsoleting a version.
- The amount of time that passes until an old version is deprecated can vary, but 6-12 months is a good rule-of-thumb.

Versioning schemes

Twilio

```
/2010-04-01/Accounts/
```

salesforce.com

```
/services/data/v30.0/limits
```

Facebook

```
/v2.0/me
```

Foursquare

```
/v2/users/{userid}/checkins
```

Intro to Apigee 127

Apigee 127 on the web

The screenshot shows the GitHub interface. At the top, there is a navigation bar with a GitHub icon, a search bar containing "Search GitHub", and links for "Explore", "Gist", "Blog", and "Help". Below the navigation bar is the repository card for "apigee-127". The repository card features a pink and white checkered logo, the repository name "apigee-127", and a small icon indicating 40 contributors. Below the repository card is a search bar with a "Filters" dropdown and a "Find a repository..." input field. A horizontal line separates this from the list of repositories.

a127-samples

Updated 3 days ago

JavaScript ★1 4

<https://github.com/apigee-127>

Installing Apigee 127

- Apigee 127 is installed using the command line – note that you may have to use sudo on a Mac

```
npm install -g apigee-127
```

```
LapGARBage:api-workshop greg$ sudo npm install -g apigee-127
npm http GET https://registry.npmjs.org/apigee-127
npm http 200 https://registry.npmjs.org/apigee-127
npm http GET https://registry.npmjs.org/apigee-127/-/apigee-127-0.1.3.tgz
npm http 200 https://registry.npmjs.org/apigee-127/-/apigee-127-0.1.3.tgz
npm http GET https://registry.npmjs.org/lodash
npm http GET https://registry.npmjs.org/async
npm http GET https://registry.npmjs.org/commander
npm http GET https://registry.npmjs.org/apigee-access
```

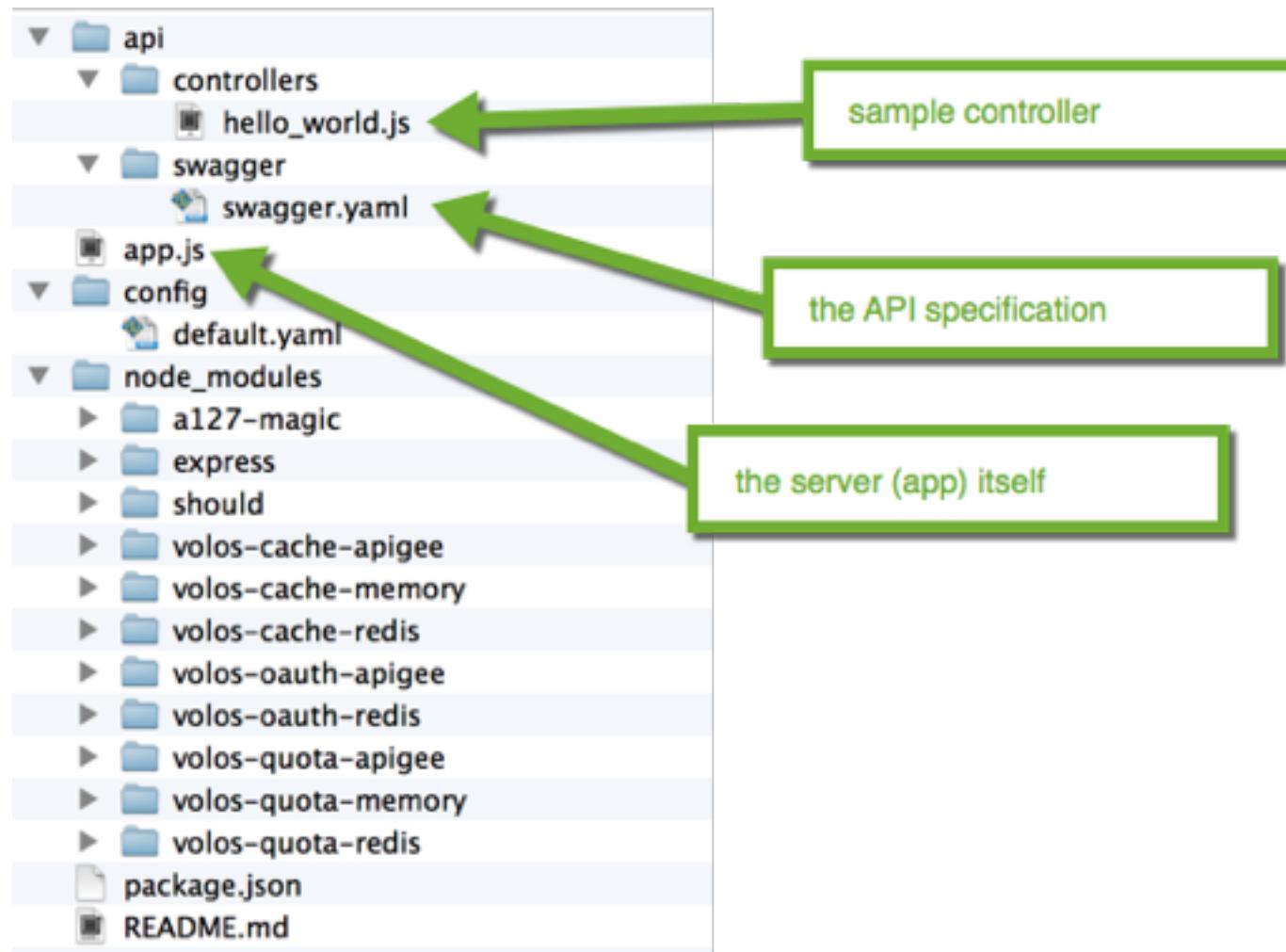
- Using the -g option installs Apigee 127 as a globally available app – this is recommended

Creating an Apigee 127 Project

- Using the command line, navigate to the place where you want to create a new project – note, this will create a folder where you will be working throughout the workshop

```
LapGARbage:Development greg$ a127 project create api-workshop
creating: .npmignore
creating: README.md
creating: api
creating: app.js
creating: config
creating: package.json
creating: api/controllers
creating: api/swagger
creating: config/default.yaml
creating: api/controllers/hello_world.js
creating: api/swagger/swagger.yaml
```

Apigee 127 App Structure



Try it out

- Navigate inside the project folder, then execute the command

a127 project start

```
LapGARbage:api-workshop greg$ a127 project start
starting with account settings: workshop
deployment files will remain after exiting: /Users/greg/Development/api-workshop
/config/.a127_env,/Users/greg/Development/api-workshop/config/.a127_secrets
Starting: /Users/greg/Development/api-workshop/app.js...
    project started here: http://localhost:10010
    project will restart on changes.
    to restart at any time, enter 'rs'
connect: multipart: use parser (multiparty, busboy, formidable) directly
connect: limit: Restrict request size at location of read
usergrid
try this:
curl http://localhost:10010/hello?name=Scott
```

- Open a browser window and navigate to:

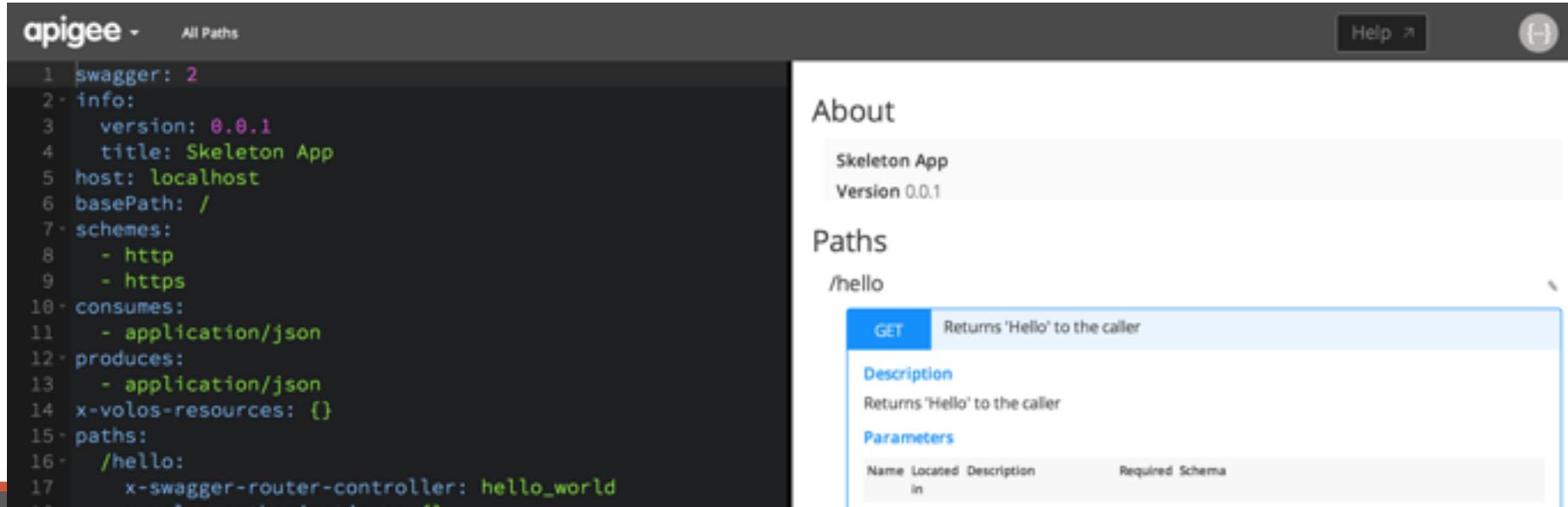
http://localhost:10010/hello?name=YOUR-NAME-HERE



Launching the Editor

- Launch the local Swagger editor by running
a127 project edit

```
LapGARBage:api-workshop greg$ a127 project edit
Swagger editor running: http://localhost:64268/
opening browser to: http://localhost:64268/
```



The screenshot shows the Apigee Swagger editor interface. On the left, there is a code editor displaying a JSON configuration file:

```
1 swagger: 2
2   info:
3     version: 0.0.1
4     title: Skeleton App
5   host: localhost
6   basePath: /
7   schemes:
8     - http
9     - https
10  consumes:
11    - application/json
12  produces:
13    - application/json
14  x-volos-resources: []
15  paths:
16    /hello:
17      x-swagger-router-controller: hello_world
18      summary: Returns 'Hello' to the caller
19      description: Returns 'Hello' to the caller
20      parameters:
21        - name: query
22          type: string
23          required: true
24          description: The value to say hello to
25          schema:
26            type: string
27            example: world
28      responses:
29        '200':
30          description: Returns 'Hello' to the caller
31          schema:
32            type: string
33            example: Hello world!
```

On the right, the API definition is displayed under the "About" section:

About

Skeleton App
Version 0.0.1

Paths

/hello

GET Returns 'Hello' to the caller

Description

Returns 'Hello' to the caller

Parameters

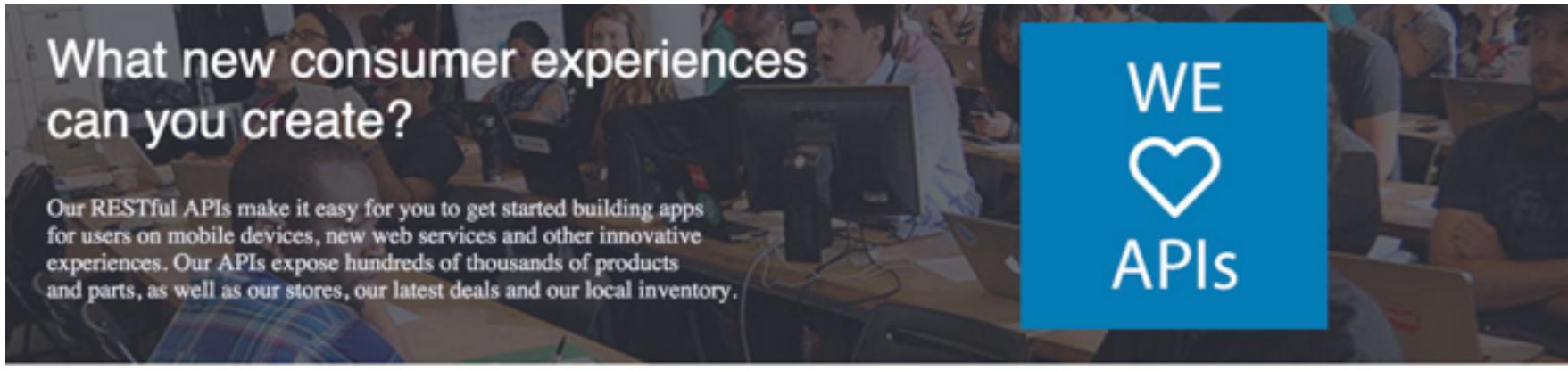
Name	Located	Description	Required	Schema
query	in			

apigee

Building an API Proxy

Sears Developer Portal

- For this example, we'll use the Product Search API



[Home](#)

Sears APIs

Product Search API

Use the Product Search API to find products based on keyword and filtered by other parameters, and to fetch product category information.

[Learn More](#)

Product Details API

Use the Product Details API to retrieve detailed information on products from colors and brands, to collections and lines.

[Learn More](#)

Top Sellers API

Use the Top Sellers API to retrieve product information on the top sellers at Sears and Kmart Stores.

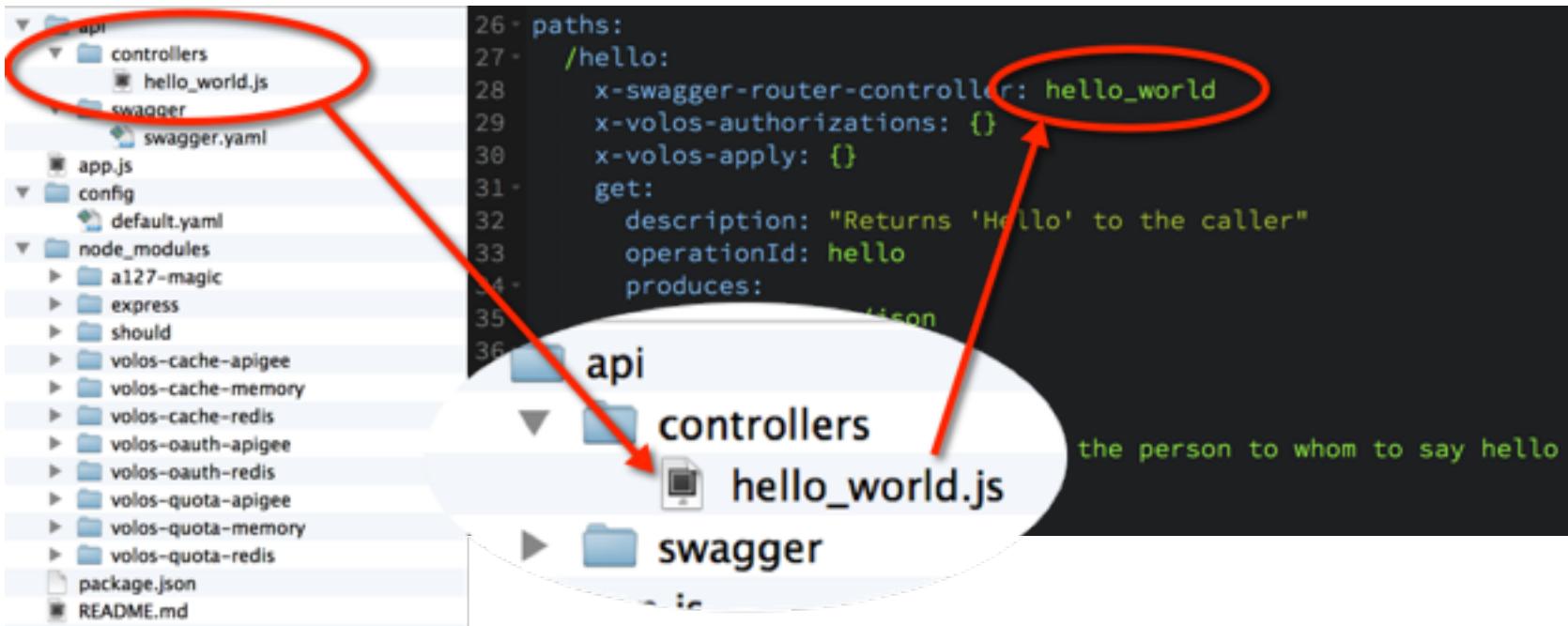
[Learn More](#)

Create a definition “/products” in the Swagger doc

- Copy/paste the /hello definition
- Change the route, x-swagger-router-controller and operationId to something that makes sense for you

<http://j.mp/products-definition-v1>

How Do Controllers Work?



- All controllers are stored in the api/controllers folder
- The name of the controller (.js file) must match the **x-swagger-router-controller:** value in the swagger doc

How Do Controllers Work?

- Each controller exports an array of name/value pairings where the name corresponds to the **operationId**: in the swagger document and the value is the function within the controller

```
'use strict';

var util = require('util');

module.exports = {
  hello: hello
}

function hello(req, res) {
  var name = req.swagger.params.name.value;
  var person = name ? util.format('Hello, %s', name) : 'Hello, stranger!';
  res.json(person);
}

26- paths:
27-   /hello:
28-     x-swagger-router-controller: hello_world
29-     x-vолос-authorizations: {}
30-     x-vолос-apply: {}
31-     get:
32-       description: Returns 'Hello' to the caller
33-       operationId: hello
34-       produces:
35-         - application/json
36-       parameters:
37-         - name: name
38-           in: query
39-           description: The name of the person to whom to say hello
40-           required: false
41-           type: string
```

The products.js controller

- Create a file **products.js** (or whatever you called the x-swagger-router-controller) and save it in the **api > controllers** folder
- Require the **request** module
- Add a module.exports JSON object to map the operationID from the swagger file to an internal function in the controller

The products.js controller

- Install the request module and add it to the package.json file
npm install request --save
- Use the request module to make a call to the Sears Product Search API to search for a washer

<http://j.mp/sears-product-search-v1>

- Test your API by going to
http://localhost:10010/products

Make the API flexible

Currently, your API only returns washers. Let's make it a little more flexible by adding variables — or parameters as we call them

- Add a path parameter to /products
- Add a limit parameter to the GET request on /products
- Use `req.swagger.params.{Parameter}.value` to grab the variable in the controller

`http://j.mp/products-definition-v2`

`http://j.mp/sears-product-search-v2`

Test your API with different combinations

`http://localhost:10010/products/dryer?limit=5`

`http://localhost:10010/products/washer?limit=20`

Deploying the APIs

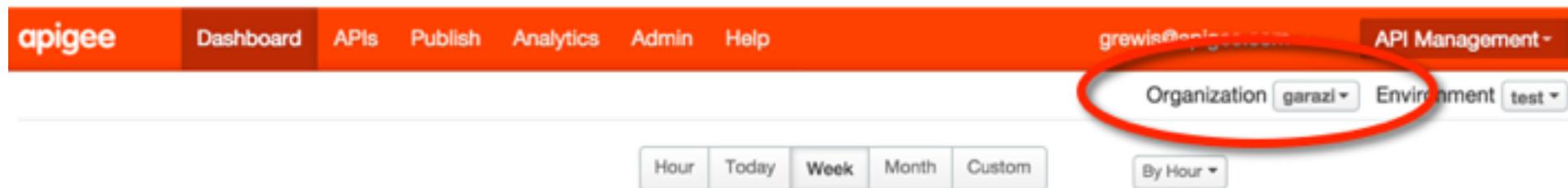
Deploying

An a127 account is needed to deploy to the Apigee Edge platform. You only need to create the account once, and it will be valid for all of your a127 projects.

- Create an a127 account
a127 account create {NAMEYOURACCOUNT}
- Select apigee as the provider and follow the prompts
- For the BaseURL and Virtual hosts options, simply hit Return

Deploying

- For organization, log into your Apigee account, click on the API Management panel. Your organization is in the upper righthand corner.
- For userID, use the email address that you used when you signed up for your Apigee account



Deploying

- To deploy to Apigee Edge
- a127 project deploy**
- After the upload is complete copy the URI of your APIs

```
LapGARbage:api-workshop greg$ a127 project deploy
Deploying project api-workshop to apigee...
name: api-workshop
environment: test
revision: 1
state: deployed
basePath: /
uris:
  - 'http://grewis-test.apigee.net/api-workshop'
```

Deploying

- If the upload times out — which can happen due to the server needing to run npm install — you can chose to have it upload your node_modules using the -u modifier

a127 project deploy -u

Congratulations!

- Open the API Management console and choose API Proxies from the API menu
- Click on your API to view it

The screenshot shows the Apigee API Management console interface. At the top, there's a navigation bar with links for apigee, Dashboard, APIs, Publish, Analytics, Admin, Help, and a user account (grewis@apigee.com). The 'API Management' dropdown is also visible. Below the navigation, the organization is set to 'grewis'. The main content area displays the 'api-workshop' API details. It includes a 'Revision 1 Summary' section with creation and update dates (Sep 5, 2014 2:32:43 PM). The 'Description' section specifies a Default Proxy Endpoint Base Path of '/api-workshop' and a Default Target Endpoint Node.js Script of 'node://app.js'. There's a button to 'Edit Revision Summary'. Below this is a 'Resources' table with columns for Name, Proxy Endpoint, Method, Path, URL, Policies, and Actions. A '+ Resource' button is available. The final section shown is 'Deployments', which lists a single entry for Environment 'test' and Revision '1' with the URL 'http://grewis-test.apigee.net/api-workshop'. This URL is circled in red.

Get more training

Advanced class?

Training for your company? Still free!

grewis@apigee.com

Congratulations!

I don't accept tips but I do accept
tweets!

@garazi #apigee

grewis@apigee.com

Thank you