




Heroku

“A platform that runs your HTTP code
(pushed through Git)”

~ 5 seconds ago

“A scalable platform that runs your HTTP code
(pushed through Git)”

Feedback


nswrailcorp-web 

Run Production Check







- 
Resources
- 
Activity
- 
Access
- 
Settings

Dynos

web node server.js

1X  1

\$0.00

-   **1X standard dynos**
512 MB \$0.05/dyno hour
-  **2X standard dynos**
1 GB \$0.10/dyno hour
- PERFORMANCE DYNOS 
-  **Performance dynos**
Superior Quality-of-Service
Low latency, high throughput
6 GB / 8 CPUs \$0.80/dyno hour
-  Info on dyno management

Est. Monthly Cost
\$0.00

Feedback

- Resources
- Activity
- Access
- Settings

Dynos

web node server.js

1X 2 \$34.50

- 1X standard dynos
512 MB \$0.05/dyno hour
- 2X standard dynos
1 GB \$0.10/dyno hour
- PERFORMANCE DYNOS
- Performance dynos
Superior Quality-of-Service
Low latency, high throughput
6 GB / 8 CPUs \$0.80/dyno hour
- Info on dyno management

Est. Monthly Cost
\$34.50

nswrailcorp-web Run Production Check

Feedback

- Resources
- Activity
- Access
- Settings

Dynos

web node server.js

1X 3 \$70.50

- 1X standard dynos
512 MB \$0.05/dyno hour
- 2X standard dynos
1 GB \$0.10/dyno hour
- PERFORMANCE DYNOS
- Performance dynos
Superior Quality-of-Service
Low latency, high throughput
6 GB / 8 CPUs \$0.80/dyno hour
- Info on dyno management

Est. Monthly Cost
\$70.50

“Heroku dynos are single threaded, a single request at a time. Say it took 10 ms to respond then a single dyno could deal with 100 requests per second. Adding more dynos increases concurrency not performance.”

~ now

HTTP Reverse Proxy

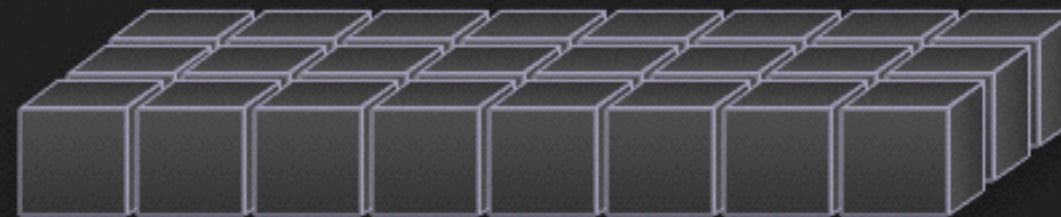
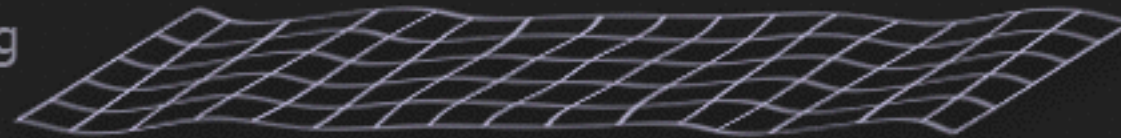
HTTP Cache

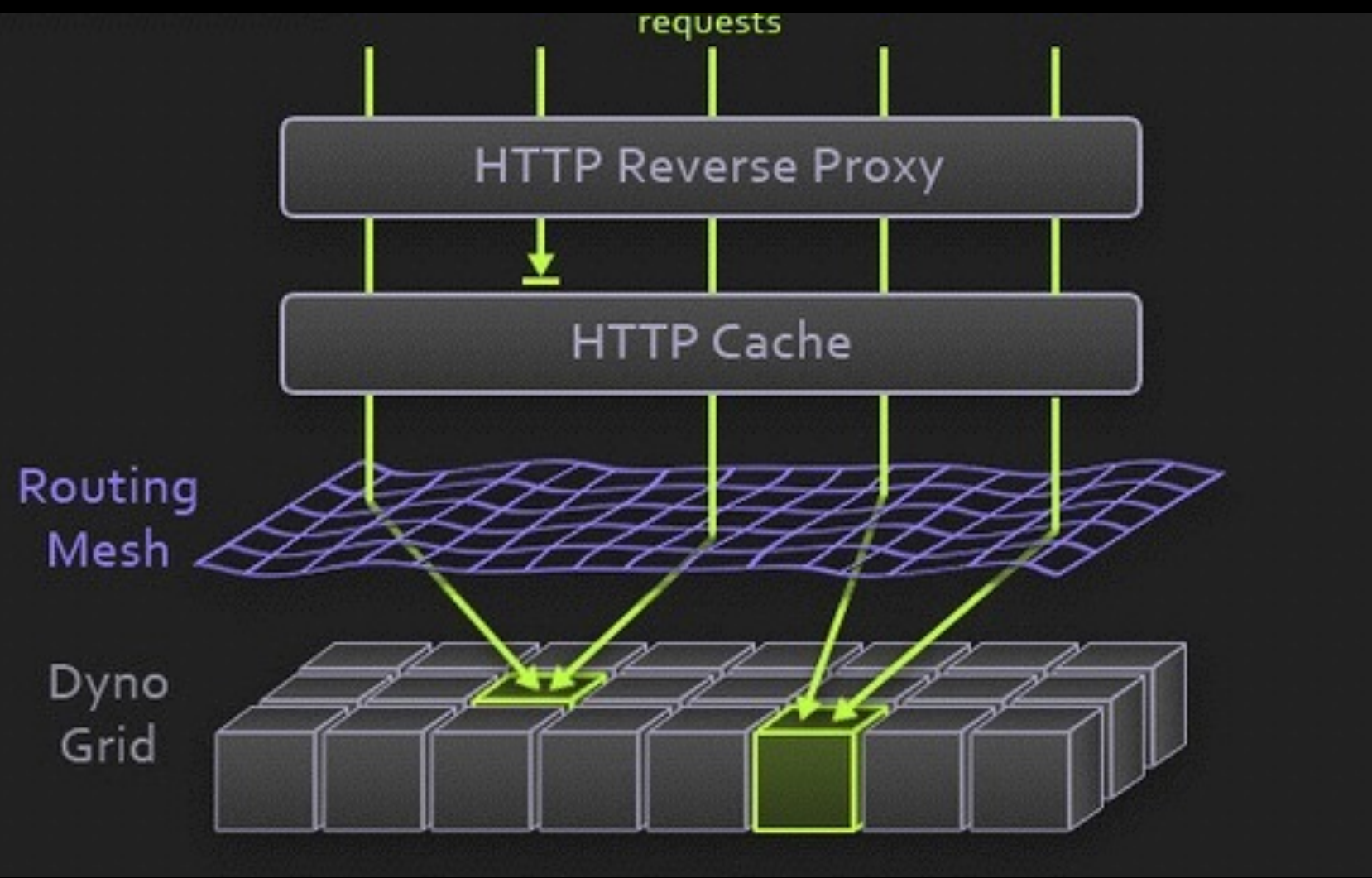
Routing
Mesh

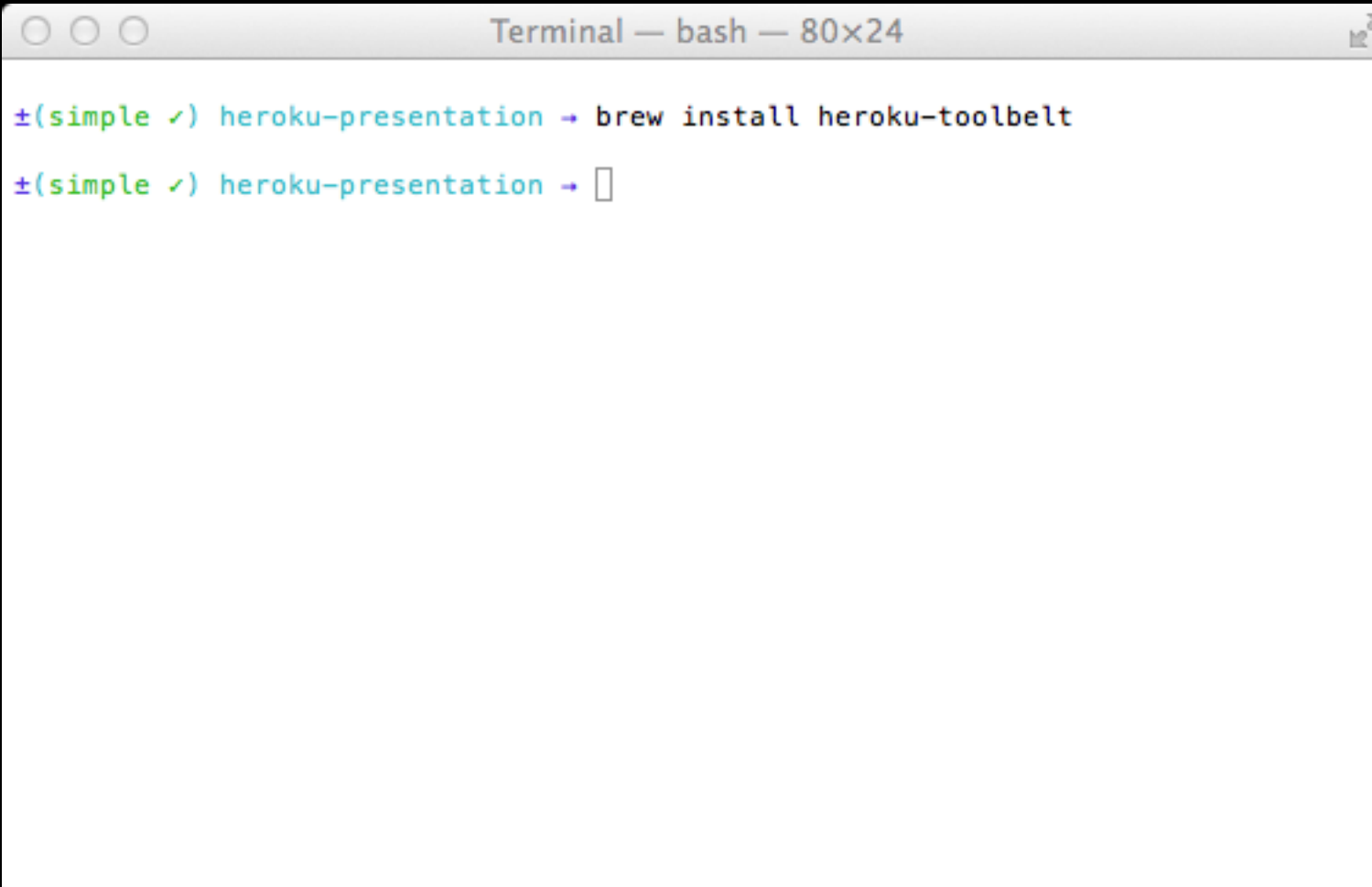
Dyno
Grid

SQL
Database

Memory
Cache







A terminal window titled "Terminal — bash — 80x24" with standard macOS window controls. The terminal displays a command prompt with a green prompt character, a green "(simple ✓)" status, and a blue "heroku-presentation" label. The command "brew install heroku-toolbelt" is entered. A second prompt line is visible below it, with a blue "heroku-presentation" label and a cursor.

```
±(simple ✓) heroku-presentation → brew install heroku-toolbelt
±(simple ✓) heroku-presentation →
```



Terminal — bash — 80x24



```
±(simple ✓) heroku-presentation → brew install heroku-toolbelt
```

```
±(simple ✓) heroku-presentation → heroku apps:create intu-simple  
Creating intu-simple... done, stack is cedar  
http://intu-simple.herokuapp.com/ | git@heroku.com:intu-simple.git  
Git remote heroku added
```

```
±(simple ✓) heroku-presentation →
```



Terminal — bash — 80x24



```
±(simple ✓) heroku-presentation → brew install heroku-toolbelt

±(simple ✓) heroku-presentation → heroku apps:create intu-simple
Creating intu-simple... done, stack is cedar
http://intu-simple.herokuapp.com/ | git@heroku.com:intu-simple.git
Git remote heroku added

±(simple ✓) heroku-presentation → git remote show -n heroku
* remote heroku
  Fetch URL: git@heroku.com:intu-simple.git
  Push URL: git@heroku.com:intu-simple.git
  HEAD branch: (not queried)
  Local ref configured for 'git push' (status not queried):
    (matching) pushes to (matching)

±(simple ✓) heroku-presentation →
```

```
Terminal — bash — 80x24

±(simple ✓) heroku-presentation → brew install heroku-toolbelt

±(simple ✓) heroku-presentation → heroku apps:create intu-simple
Creating intu-simple... done, stack is cedar
http://intu-simple.herokuapp.com/ | git@heroku.com:intu-simple.git
Git remote heroku added

±(simple ✓) heroku-presentation → git remote show -n heroku
* remote heroku
  Fetch URL: git@heroku.com:intu-simple.git
  Push URL: git@heroku.com:intu-simple.git
  HEAD branch: (not queried)
  Local ref configured for 'git push' (status not queried):
    (matching) pushes to (matching)

±(simple ✓) heroku-presentation → git push heroku master

±(simple ✓) heroku-presentation →
```

Heroku add-ons

- Add third-party services to your app
- For example: data storage, cache, email, logging, push, analytics, etc.
- The services expose their endpoints in environment variables accessible from the app code

heroku add-ons

How it Works

Pricing

Apps

Add-ons

Documentation

Support

Log in

Add powerful functionality to your apps with ease.

Search for add-ons

FEATURED



mongoHQ
Blazing-fast
MongoDB Hosting
Backed by SSDs

POPULAR



SendGrid

POPULAR



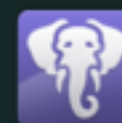
websolr

Data Stores

Choose from Postgres, Memcache, Mongo, Redis, Hadoop and more. Then forget doing database backups, restores, or wearing the pager ever again.



Treasure Data
Hadoop



Heroku Postgres



IronCache



RedisGreen

```
Terminal — bash — 80x24

±(jokes ✓) heroku-presentation → heroku addons --app intu-jokes
=== intu-jokes Configured Add-ons
mongohq:sandbox

±(jokes ✓) heroku-presentation → heroku addons:add sendgrid --app intu-jokes
Adding sendgrid on intu-jokes... done, v8 (free)
Use `heroku addons:docs sendgrid` to view documentation.

±(jokes ✓) heroku-presentation → heroku config --app intu-jokes
=== intu-jokes Config Vars
MONGOHQ_URL:      mongodb://heroku:06dc3b9c5c5c2a0a90610bbb8bfdc59f@troupe.mongohq.com:10085/app22290987
SENDGRID_PASSWORD: lxm6vdwl
SENDGRID_USERNAME: app22290987@heroku.com

±(jokes ✓) heroku-presentation → heroku addons:open mongohq --app intu-jokes
Opening mongohq:sandbox for intu-jokes... done

±(jokes ✓) heroku-presentation →
```


Service oriented architecture

- A single monolithic web app (one project on a single domain)
- Lots of small independent web services (each with their own codebase and URL endpoint)
- Benefits and drawbacks to each approach

Monolithic Pattern

- A lot easier to get started
- Simpler to refactor as the codebase grows
- Easier to debug (no external components, network, etc.)

Service Pattern

- Multiple small independent web services (eg. www, api, accounts, reports)
- They talk over HTTP
- Less code in the services (they do less things)
- Each has their own resources: web servers, db, message queues, etc

Service Pattern

- Maintainability
 - Easier over long term
 - Simpler logic
 - Easy to navigate code
- Scalability
 - Put more resources for frequently used services
 - Better resource utilization
 - Not affecting other services

Service Pattern

- Takes longer to build a working MVP (more upfront dev time)
- Authentication problems
- Cost and overdevelopment for simple apps
- Need to consider external factor (eg. network)
- Anti pattern, don't go swing in the other extreme and build hundreds of services