Javascript á vefþjónum……

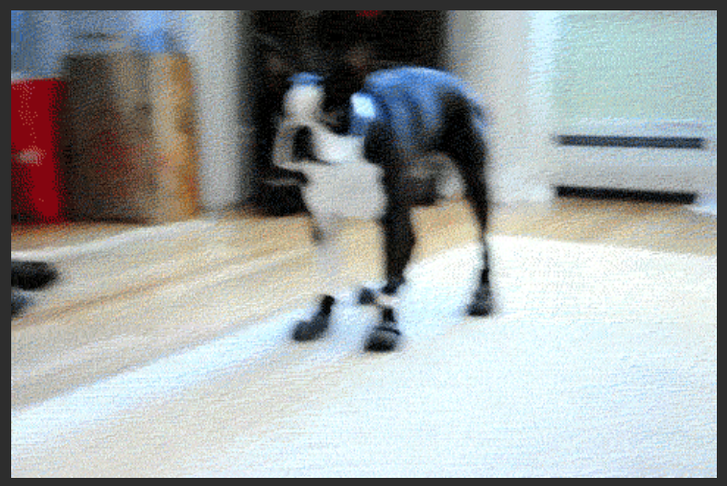
```
$('server')
  .use('react')
  .use('uglify')
  .use('bootstrap')
  .on('request', $.route)
  .on('ddos', $.reject)
  .end('you just won 1.000.000$');
```

Javascript á vefþjónum…… hversu brjálaðslega mikil snilld er það? Ég nota bara jquerykóða til að stjórna öllu ooog þetta verður örugglega fullt af onelineerum

Það var mjög mörgum sem leið akkúrat svona þegar þeir heyrðu að það væri kominn góður platform til að keyra javascript á servernum

(Andrew Garfield & Emma stone)

En margir voru yfir sig hræddir við þessa hugmyndafræði

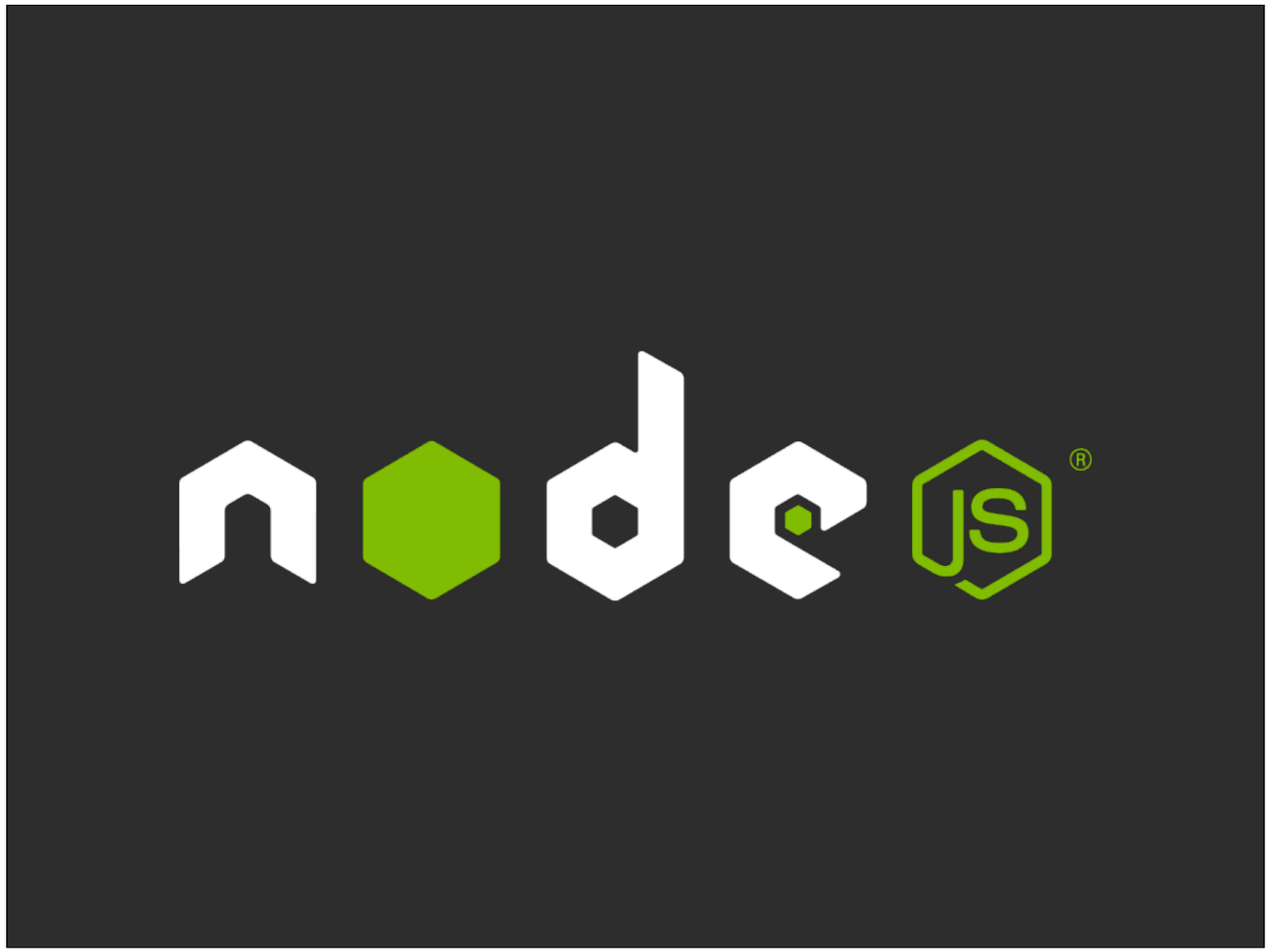


Og þvertóku fyrir að hoppa út í djúpu laugina og prufa þetta

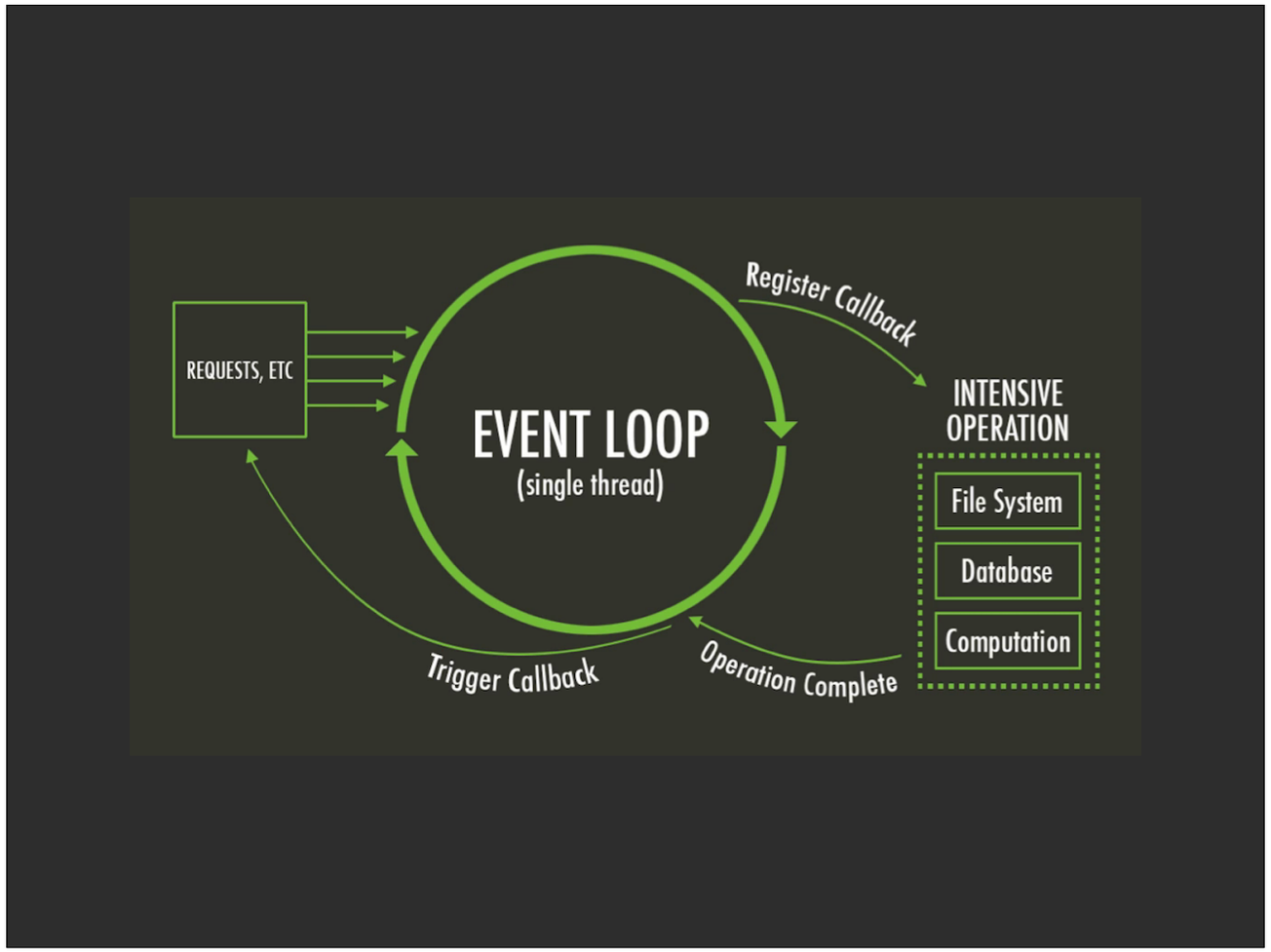
Því að node.js átti það til að springa ansi vel framan í fólk

```
1 //The promise of a short talk
2 new Promise(function(resolve,reject) {
3   if(Math.random() <= 0.1){
4     return reject('>20 minutes');
5   }
6   return resolve('18 minutes');
7 });
```

En hvað um það, ég er ekki kominn hingað til að tala um allt það versta í node.js og ég var einnig búinn að lofa að tala bara í 20 mínútur og ég ætla að reyna að halda mig innan þeirra marka



En hvað er eiginlega node.js?
node.js notar Javascript sem forritunarmál, en Ryan Dahl skapara node.js fannst það passa fullkomlega við asynchrous módelið sem hann hafði í huga af því að það var eitt af fáu vinsælu málunum sem ekki hafði I/O aðgerðir og hann gat því fundið þær upp frá grunni. Asynchrous þýðir í raun að margir hlutir geta verið að keyrast á sama tíma óháð hver öðrum. Ryan Dahl fannst einnig mjög sterkt hversu mikil þróun hefði átt sér stað í keyrsluumhverfum fyrir Javascript og sérstaklega V8 vélinni frá Google sem hann gat nýtt sér við gerð node.js.



Node.js keyrir eitt process á einum kjarna oftast á einum þræði sem styðst við eina event loopu. Event loopan er betur skýrð á myndinni en hún er stanslaus keyrsla á atburðahring sem fylgir First in First out aðferðafræðinni.

Hægt er að keyra node.js á öllum helstu stóru umhverfunum en þau eru: Linux, Windows, OSX og FreeBSD. Það er mjög einfallt að setja node.js upp og það þarf ekki risa vefþjóna eins og apache til að setja fyrir framan það. Fyrir ykkur sem þekkja Ruby og Python þá er node.js mjög líkt Event Machine í Ruby og Twisted eða Tornado í Python

Node.js er nú þegar búið að ná talsverðri útbreiðslu og eru mjög stór fyrirtæki farin að nota það. Yahoo, Paypal, linkedin og svo mætti lengi telja hafa öll sömu sögu að segja.

Með því að nota Node.js hafa þau aukið afköst forritara, gert hugbúnaðinn sinn hraðari og minnkað kostnaður við að reka vefþjóna sína

Aukin afköst forritara lýsa sér best hjá Yahoo! en hjá þeim kunnu margir forritarar Javascript og allir vefir þeirra eru skrifaðir í javascript. Hins vegar eru bakendarnir oftast í öðrum tungumálum og því var mjög mikið um það sem kallast context switching hjá forriturunum þeirra eða þegar maður þarf að skipta oft milli tungumála

Sem dæmi um hraðaukningu og kostnaðarminnkun þá var grunnkerfi sem var skipt út hjá paypal hraðað um 200ms með því að skipta yfir í node og það fáránlegasta í þessu var að node.js notaði bara 1 kjarna meðan java clusterið notaði 5!

```
1  //Hlöðum inn http pakkanum
2  var http = require('http');
3
4  var server = http.createServer(function (
                  request, response) {
5    response.end('Virkar!');
6  });
7
8  server.listen(13337, function(){
9    console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
10 });
```

En nú langar mig að sýna ykkur nokkur kóðabrot!

Ég vil byrja á að taka dæmi um lítinn og sætan vefþjón sem svarar fyrirspurnum sem berast gegnum http.

```
1  //Hlöðum inn http pakkanum
2  var http = require('http');
3
4  var server = http.createServer(function (
                  request, response) {
5    response.end('Virkar!');
6  });
7
8  server.listen(13337, function(){
9    console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
10 });
```

Í það verkefni getum við nýtt okkur http pakkann sem er hluti af standard library-inu. Við hlöðum honum inn með CommonJS fallinu require.

```
1  //Hlöðum inn http pakkanum
2  var http = require('http');
3
4  var server = http.createServer(function (
                  request, response) {
5    response.end('Virkar!');
6  });
7
8  server.listen(13337, function(){
9    console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
10 });
```

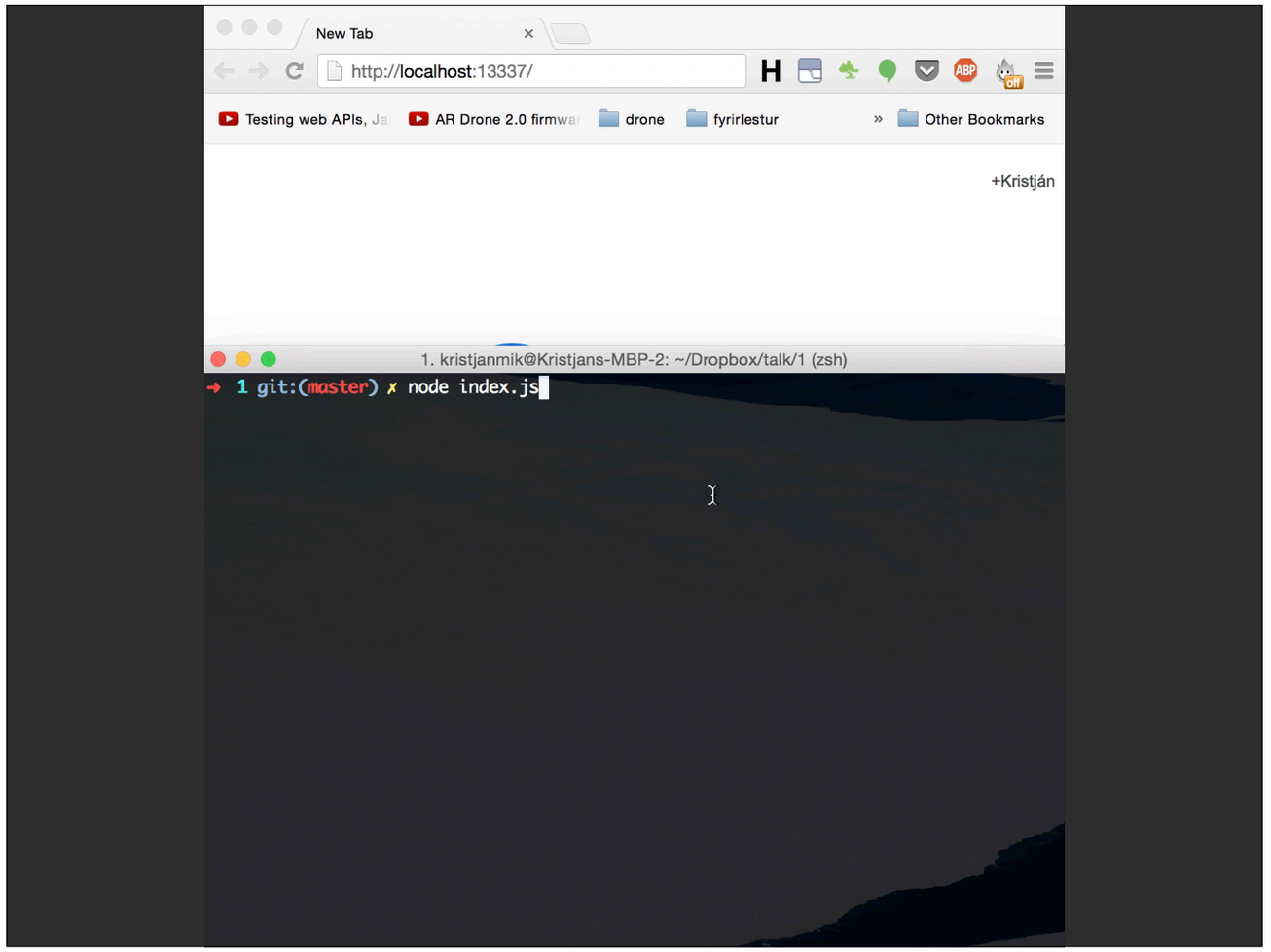
Við setjum upp serverinn með því að kalla á createServer fallið á http pakkanum ->

```
1  //Hlöðum inn http pakkanum
2  var http = require('http');
3
4  var server = http.createServer(function (
                  request, response) {
5    response.end('Virkar!');
6  });
7
8  server.listen(13337, function(){
9    console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
10 });
```

Sem svarar alltaf með textanum Virkar þegar hann fær til sín beiðni

```
1  //Hlöðum inn http pakkanum
2  var http = require('http');
3
4  var server = http.createServer(function (
                  request, response) {
5    response.end('Virkar!');
6  });
7
8  server.listen(13337, function(){
9    console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
10 });
```

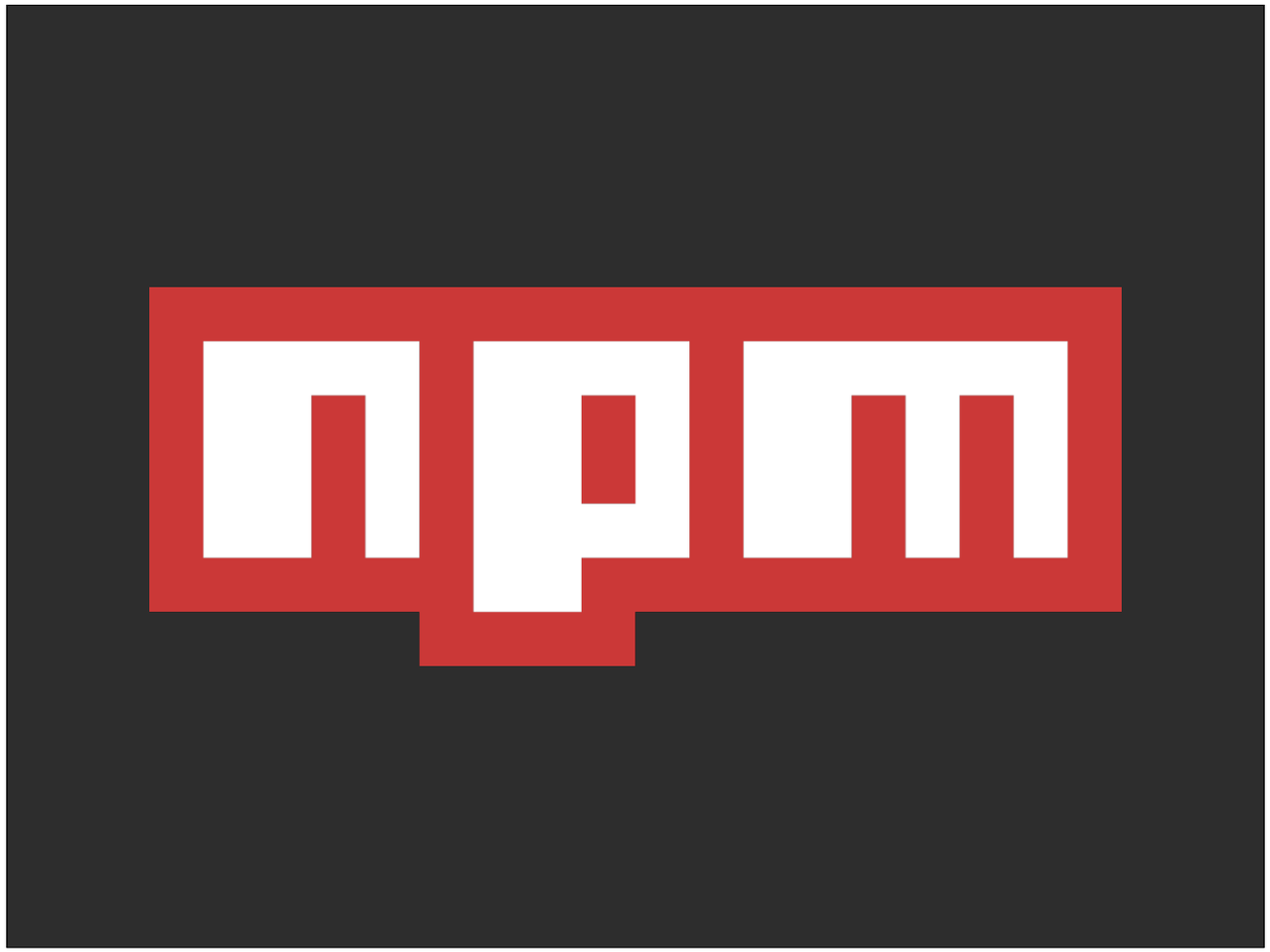
Og að lokum bindum við okkur á port 13337. Við getum bundið okkur eða hlustað á hvaða port sem er og port 80 er það sem þið munduð hlusta á til að keyra vefsíðu ef þið hafið ekkert millilag á milli



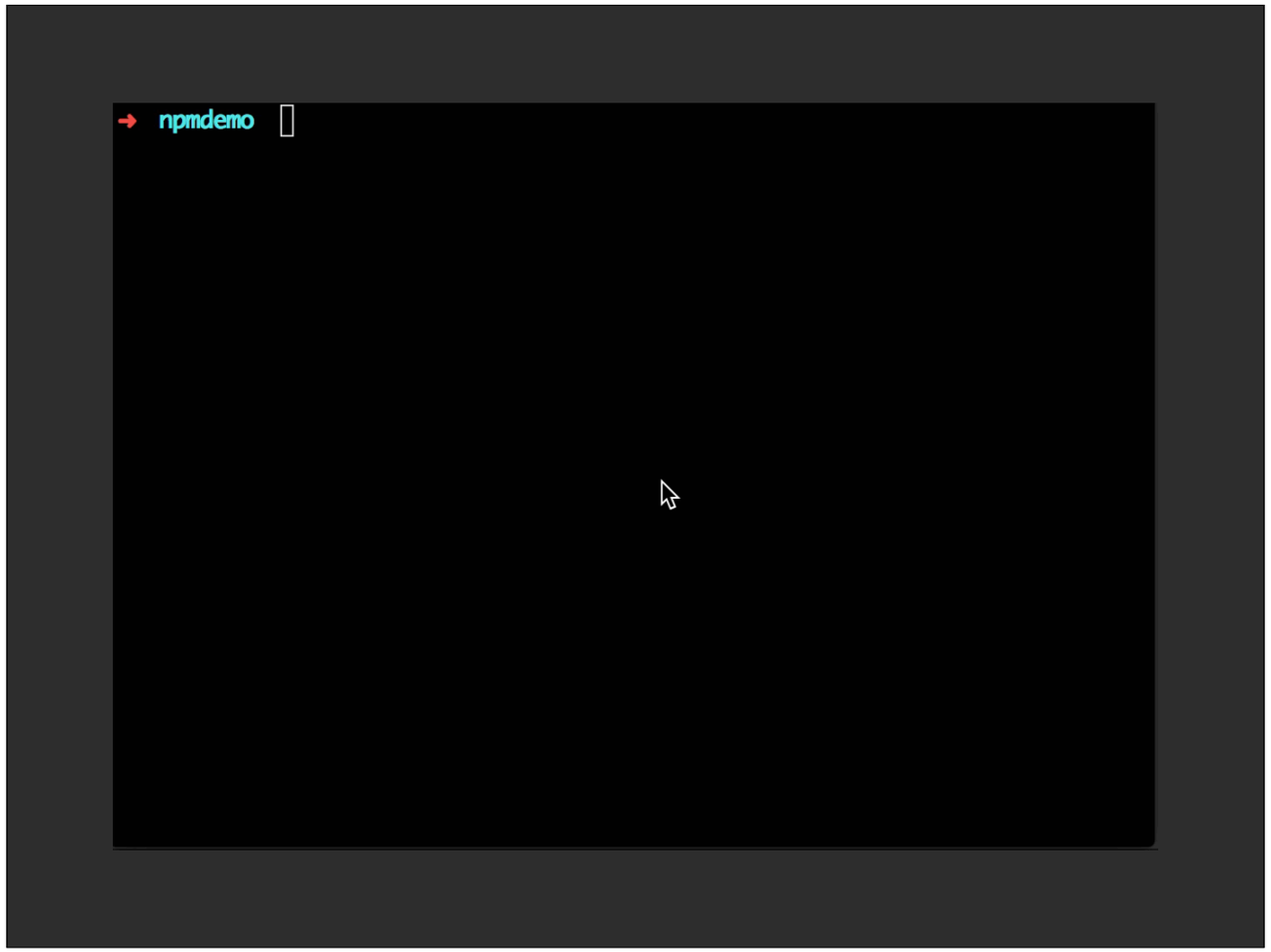
Eins og þið sjáið keyrir þetta kóðann og allt svínvirkar þegar við sláum inn slóðina þá birtist textinn virkar

```
1  //Hlöðum inn express stoðgrindinni
2  var app = require('express')();
3
4  //Við getum notað innbyggða vegvísun
5  app.get('/beep', function (req, res) {
6    res.send('Virkar!');
7  });
8
9  app.listen(13337, function () {
10   console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
11 });
```

Það er mjög vinsælt að nota framework eða stoðgrindur á íslensku til að létta undir með manni og express er einmitt fullkomið í það. Það er mjög létt í keyrslu og þvælist lítið fyrir manni. —>



Express pakkinn er ekki innbyggður inn í node.js og því þarf að sækja hann annarstaðar frá en til þess getum við notað npm. npm er pakkakerfi rétt eins og pip í python og rubygems í ruby. Sem dæmi um hversu brilliant þetta pakkakerfi er þá er npm sjálft pakki í npm. Kerfið er bæði hratt og stöðugt og eru komnir um 105þúsund pakkar sem hafa verið sóttir 600miljón sinnum síðustu 30 daga. Í node heiminum segir fólk yfirleitt “there is a module for that” eins og apple segir alltaf “there is an app for that” af því að magnið af pökkum er svo gífurlega mikið



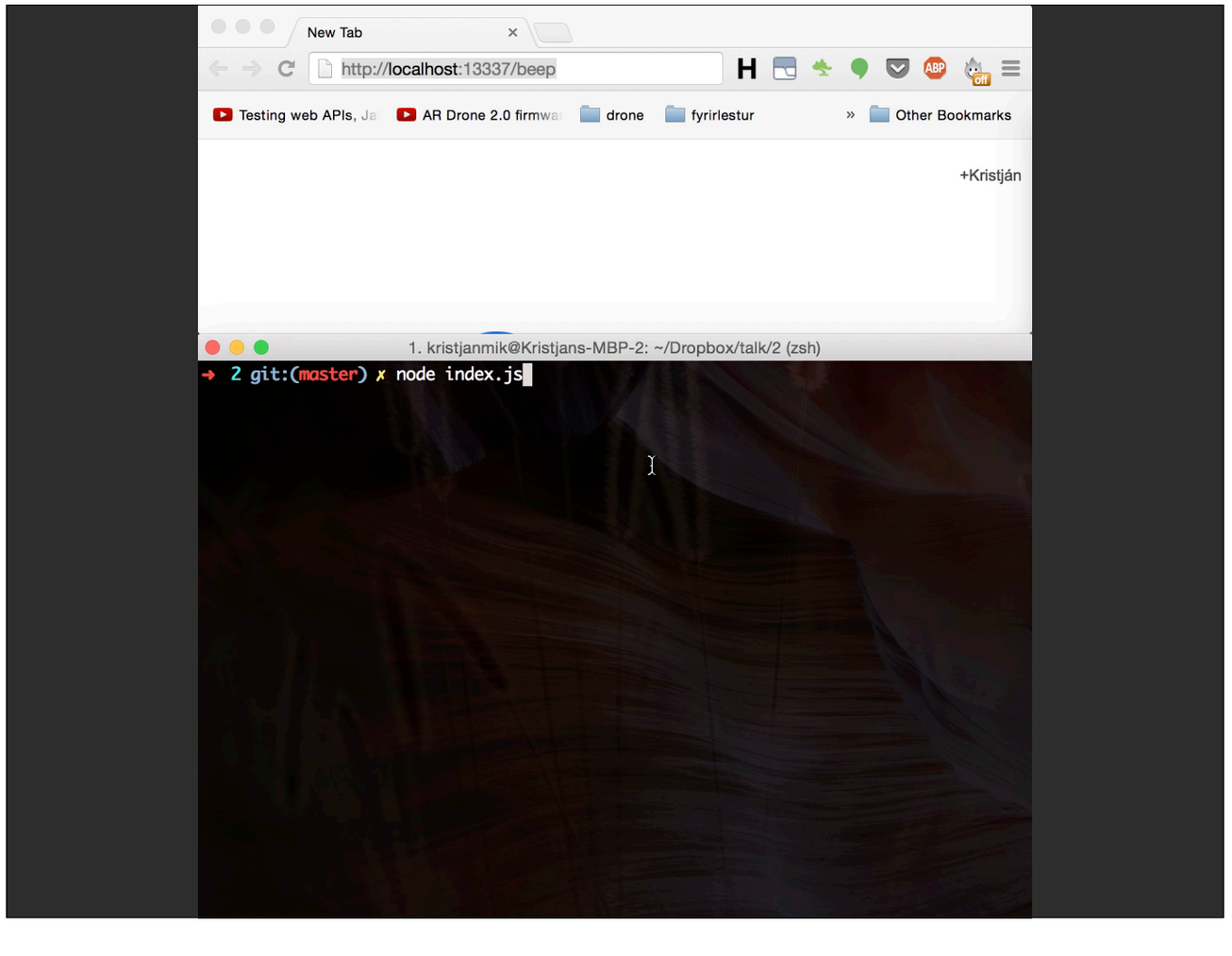
Til að byrja á því að nota npm þurfum við fyrst að segja því að við séum með verkefni í þessari möppu með því að keyra npm init. Að því loknu setjum við upp pakkann með npm install express og setjum flaggið save með sem vistar npm express í projectið þannig að næst þegar við keyrum npm install þá kemur sá pakki með. Að þessu loknu er pakkinn aðgengilegur með require. og við getum notað hann á einfaldan hátt.

```
1  //Hlöðum inn express stoðgrindinni
2  var app = require('express')();
3
4  //Við getum notað innbyggða vegvísun
5  app.get('/beep', function (req, res) {
6    res.send('Virkar!');
7  });
8
9  app.listen(13337, function () {
10   console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
11 });
```

Aftur að dæminu okkar. Hérna notum við express pakkann en að öðru leiti er þetta alveg sami kóði og í fyrsta dæminu ->

```
1  //Hlöðum inn express stoðgrindinni
2  var app = require('express')();
3
4  //Við getum notað innbyggða vegvísun
5  app.get('/beep', function (req, res) {
6    res.send('Virkar!');
7  });
8
9  app.listen(13337, function () {
10   console.log('Vefþjónn keyrir á:
                  http://localhost:13337');
11 });
```

fyrir utan að við notum skástrik beep sem slóð. Við erum að nota innbyggða vegvísunarkerfið í express, á ensku routing til að kalla fram þessa virkni



Eins og þið sjáið þá keyrir þetta alveg eins, við förum inn á slóðina skástrik beep og fáum textan virkar

```
var fs = require('fs'); //Skráarkerfis
         einingin
var app = require('express')();

app.get('/straumur', function (request,
        response) {
  //Straumur án þess að halda öllum
                  gögnunum í vinnsluminni
  fs.createReadStream('./index.html')
                          .pipe(response);
});

app.listen(13337, function () {
  console.log('Vefþjónn keyrir á:
                http://localhost:13337');
});
```

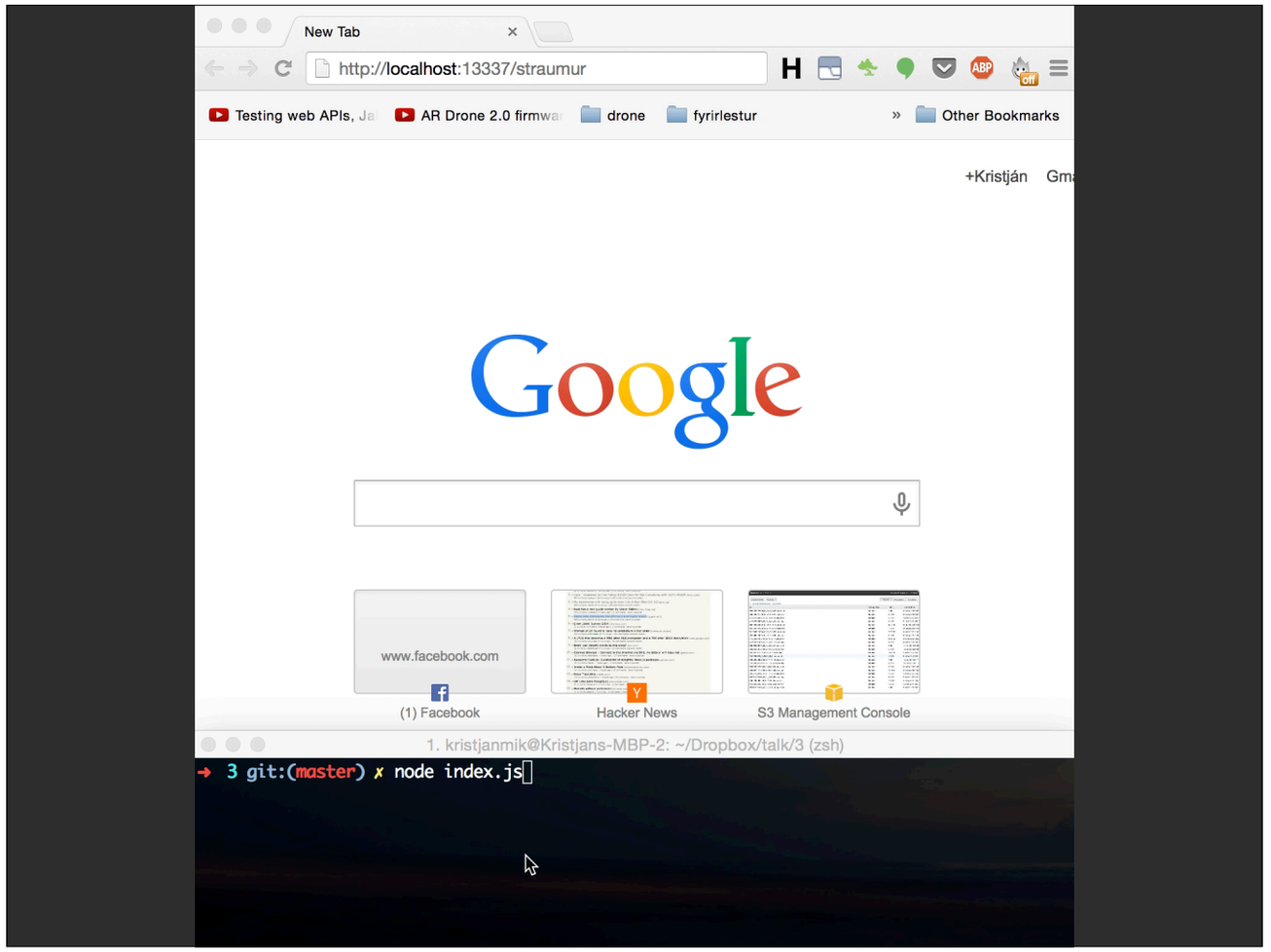
Mig langar að sýna ykkur hvernig straumar í node.js eru af því að þeir eru svo fáránlega öflugir. Straumarnir virka þannig að forritið þarf að geyma mjög lítinn hluta gagnanna eða í þessu tilviki innihald skráar í minni því að gögnin flæða einfaldlega inn og út. Hérna streymum við html skjá úr sömu möppu og notum hana sem svar fyrir beiðninni þegar farið er inn á localhost skástrik straumur.

```
1  var fs = require('fs'); //Skráarkerfis
2           einingin
3  var app = require('express')();
4
5  app.get('/straumur', function (request,
6          response) {
7    //Straumur án þess að halda öllum
                    gögnunum í vinnsluminni
8    fs.createReadStream('./index.html')
                              .pipe(response);
9  });
10
11 app.listen(13337, function () {
12   console.log('Vefþjónn keyrir á:
13                  http://localhost:13337');
14 });
```

Fyrst náum við í skráarkerfispakkann

```
1  var fs = require('fs'); //Skráarkerfis
2           einingin
3  var app = require('express')();
4
5  app.get('/straumur', function (request,
6          response) {
7    //Straumur án þess að halda öllum
                    gögnunum í vinnsluminni
8    fs.createReadStream('./index.html')
                              .pipe(response);
9  });
10
11 app.listen(13337, function () {
12   console.log('Vefþjónn keyrir á:
13                  http://localhost:13337');
14 });
```

Og þegar requestið kemur þá þá búum við til straum og sendum hann sem svar með .pipe fallinu, en .pipe er eitt aðal fallið í heimi node.js strauma.



Þetta virðist hlaðast í einni hendingu en í rauninni er um straum að ræða og vefþjónninn byrjar að svara um leið og hann fær fyrstu gögnin úr skránni. Því má segja að hann sé í rauninni að skila skránni í bútum til vafrarans.

```
var fs = require('fs');
var app = require('express')();
var request = require('request');
 //Breytir PNG í ASCII fyrir terminal
var pictureTube = require('picture-tube');

app.get('/', function (req, res) {
  console.log('Kallað á /');
  var tube = pictureTube();
  // req | tube | process.stdout
  req.pipe(tube).pipe(process.stdout);
});

app.listen(13337, function () {
  console.log('Vefjónn keyrir á:
                    http://localhost:13337');
  request.get('http://kristjanmik.is/
                    content/images/node.png')
    .pipe(request('http://localhost:
                                    13337'));
});
```

Til að gera þetta aðeins flóknara þá getum við líka breytt straumum og tengt fleiri strauma saman að vild. Hérna náum við í mynd og sendum hana viðstöðulaust áfram á express vefþjóninn okkar sem birtir myndina sem ASCII art í terminalinum.

```
1  var fs = require('fs');
2  var app = require('express')();
3  var request = require('request');
4   //Breytir PNG í ASCII fyrir terminal
5  var pictureTube = require('picture-tube');
6
7  app.get('/', function (req, res) {
8    console.log('Kallað á /');
9    var tube = pictureTube();
10   // req | tube | process.stdout
11   req.pipe(tube).pipe(process.stdout);
12 });
13
14 app.listen(13337, function () {
15   console.log('Vefjónn keyrir á:
                    http://localhost:13337');
16   request.get('http://kristjanmik.is/
                    content/images/node.png')
17     .pipe(request('http://localhost:
                                    13337'));
18 });
```

Við nýtum okkur pakkann request úr NPM sem getur gert köll yfir http.

```
1  var fs = require('fs');
2  var app = require('express')();
3  var request = require('request');
4   //Breytir PNG í ASCII fyrir terminal
5  var pictureTube = require('picture-tube');
6
7  app.get('/', function (req, res) {
8    console.log('Kallað á /');
9    var tube = pictureTube();
10   // req | tube | process.stdout
11   req.pipe(tube).pipe(process.stdout);
12 });
13
14 app.listen(13337, function () {
15   console.log('Vefjónn keyrir á:
                    http://localhost:13337');
16   request.get('http://kristjanmik.is/
                    content/images/node.png')
17     .pipe(request('http://localhost:
                                    13337'));
18 });
```

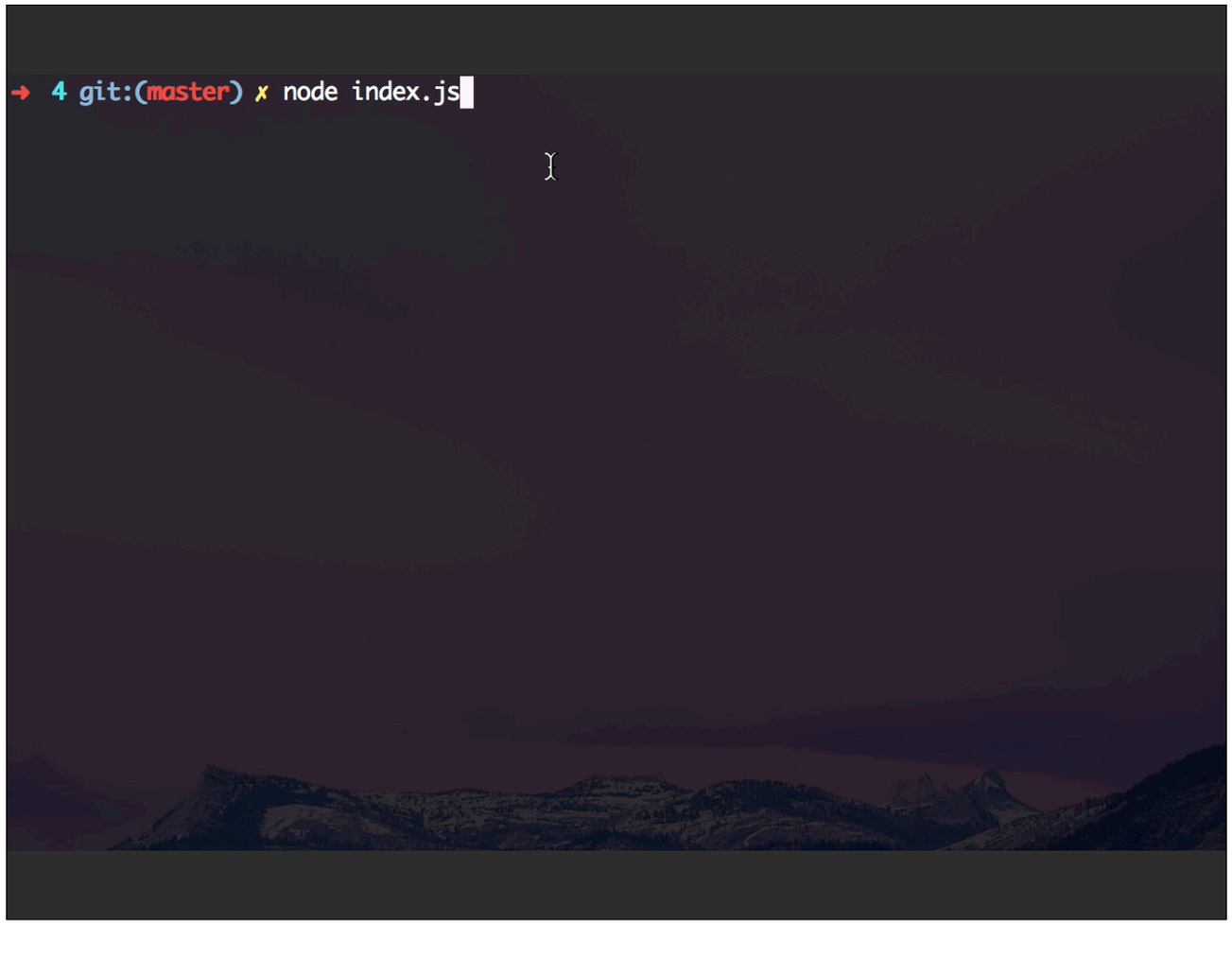
Og auðvitað er til pakki í npm til að breyta PNG straum í ASCII straum.

```
1  var fs = require('fs');
2  var app = require('express')();
3  var request = require('request');
4   //Breytir PNG í ASCII fyrir terminal
5  var pictureTube = require('picture-tube');
6
7  app.get('/', function (req, res) {
8    console.log('Kallað á /');
9    var tube = pictureTube();
10   // req | tube | process.stdout
11   req.pipe(tube).pipe(process.stdout);
12 });
13
14 app.listen(13337, function () {
15   console.log('Vefjónn keyrir á:
                    http://localhost:13337');
16   request.get('http://kristjanmik.is/
                    content/images/node.png')
17     .pipe(request('http://localhost:
                                    13337'));
18 });
```

Við erum í rauninni að taka myndina, pípa henni í tube pakkann sem pípar outputinu í stdout. Eins og þið sjáið í commentinu á línu 10 þá er þetta svipað og pipe virknin í UNIX

```
1  var fs = require('fs');
2  var app = require('express')();
3  var request = require('request');
4   //Breytir PNG í ASCII fyrir terminal
5  var pictureTube = require('picture-tube');
6
7  app.get('/', function (req, res) {
8    console.log('Kallað á /');
9    var tube = pictureTube();
10   // req | tube | process.stdout
11   req.pipe(tube).pipe(process.stdout);
12 });
13
14 app.listen(13337, function () {
15   console.log('Vefjónn keyrir á:
                  http://localhost:13337');
16   request.get('http://kristjanmik.is/
                  content/images/node.png')
17     .pipe(request('http://localhost:
                                  13337'));
18 });
```

Til að kveikja á öllu ferlinu sendum við beiðni með request pakkanum á vefþjóninn. Það sem er sérstakt við þessa beiðni er að hún er sjálf straumur af vefsíðunni sem hýsir node.png myndina



Hérna sjái þið þetta keyra og myndin birtist okkur sem ascii mynd.

```
app.post('/upload/:key', function(req, res) {
  req.pause(); // Stöðvum gagnastrauminn

  //mongodb módel
  UploadKey.findOne({
    key: req.params.key
  }, function(err, key) {
    if(err) return res.status(500).end();
    if(!key) return res.status(404).end();

    var path = 'lifetimr/' + uuid.v4() + ".mp4";
    //Straumur í skráarkerfi í skýinu,
                                  Manta hjá Joyent
    var up = manta.createWriteStream(path,{});

    up.once('close', function() {
      res.status(200).end(); //Árangur!
    });
    req.pipe(up); //Streymum til Manta
    req.resume(); //Hefjum gagnastrauminn
  });
});
```

Mig langar líka að sýna ykkur hluta af kóða úr appi sem við erum að smíða af því það er mjög gott dæmi um hvernig hægt er að búa til API í node.js. Ég tók út mikið af kóða til einföldunar en hugmyndafræðin er sú sama. Einnig erum við ekki að gera ráð fyrir neinum villum því eins og þið hafið séð í kynningunni hingað til er node.js fullkomið, haha nei djók eins og ég sagði, bara til einföldunnar!

```
app.post('/upload/:key', function(req, res) {
  req.pause(); // Stöðvum gagnastrauminn

  //mongodb módel
  UploadKey.findOne({
    key: req.params.key
  }, function(err, key) {
    if(err) return res.status(500).end();
    if(!key) return res.status(404).end();

    var path = 'lifetimr/' + uuid.v4() + ".mp4";
    //Straumur í skráarkerfi í skýinu,
                                  Manta hjá Joyent
    var up = manta.createWriteStream(path,{});

    up.once('close', function() {
      res.status(200).end(); //Árangur!
    });
    req.pipe(up); //Streymum til Manta
    req.resume(); //Hefjum gagnastrauminn
  });
});
```

Við byrjum á því að samþykkja post beiðnir á slóðinni upload plús staðfestingarlykill.

```
app.post('/upload/:key', function(req, res) {
  req.pause(); // Stöðvum gagnastrauminn

  //mongodb módel
  UploadKey.findOne({
    key: req.params.key
  }, function(err, key) {
    if(err) return res.status(500).end();
    if(!key) return res.status(404).end();

    var path = 'lifetimr/' + uuid.v4() + ".mp4";
    //Straumur í skráarkerfi í skýinu,
                                  Manta hjá Joyent
    var up = manta.createWriteStream(path,{});

    up.once('close', function() {
      res.status(200).end(); //Árangur!
    });
    req.pipe(up); //Streymum til Manta
    req.resume(); //Hefjum gagnastrauminn
  });
});
```

Síðan leitum við að því hvort lykillinn sé til í mongodb en á meðan segjum við þeim sem er að tengjast okkur að bíða með gagnastrauminn sinn.

```
app.post('/upload/:key', function(req, res) {
  req.pause(); // Stöðvum gagnastrauminn

  //mongodb módel
  UploadKey.findOne({
    key: req.params.key
  }, function(err, key) {
    if(err) return res.status(500).end();
    if(!key) return res.status(404).end();

    var path = 'lifetimr/' + uuid.v4() + ".mp4";
    //Straumur í skráarkerfi í skýinu,
                                  Manta hjá Joyent
    var up = manta.createWriteStream(path,{});

    up.once('close', function() {
      res.status(200).end(); //Árangur!
    });
    req.pipe(up); //Streymum til Manta
    req.resume(); //Hefjum gagnastrauminn
  });
});
```

Þegar lykillinn er fundinn búum við til straum í skráarskýjaþjónustu sem virkar svipað og Amazon S3 og heitir Manta.

```
app.post('/upload/:key', function(req, res) {
  req.pause(); // Stöðvum gagnastrauminn

  //mongodb módel
  UploadKey.findOne({
    key: req.params.key
  }, function(err, key) {
    if(err) return res.status(500).end();
    if(!key) return res.status(404).end();

    var path = 'lifetimr/' + uuid.v4() + ".mp4";
    //Straumur í skráarkerfi í skýinu,
                                  Manta hjá Joyent
    var up = manta.createWriteStream(path,{});

    up.once('close', function() {
      res.status(200).end(); //Árangur!
    });
    req.pipe(up); //Streymum til Manta
    req.resume(); //Hefjum gagnastrauminn
  });
});
```

Að lokum getum við svo pípað straumnum til Manta og á sama tíma leyft þeim sem er að tengjast að senda okkur gögn að nýju. Þetta er mjög skilvirk leið til að taka á móti uploadi og þessi aðferð getur auðveldlega tekið við hundruðum uploada á hverri sekúntu.

Kristján Ingi Mikaelsson
@kristjanmik

Þetta er allt sem ég hef að þessu sinni. Ég vil þakka ykkur öllum fyrir góða áheyrn og ég vona að þið kíkið öll á node.js eftir þessa ráðstefnu. Ef þið viljið finna mig á internetinu þá er ég @ kristjanmik allstaðar
Takk fyrir mig!