ASYNC/AWAIT

JANUARY 2018 @ NORFOLK.JS

STANLEY ZHENG

WE ALL LIKE JAVASCRIPT

JAVASCRIPT IS ASYNCRONOUS

ASYNCRONOUS CODE IS HARD

I WANT MY BRAIN TO WORK LESS

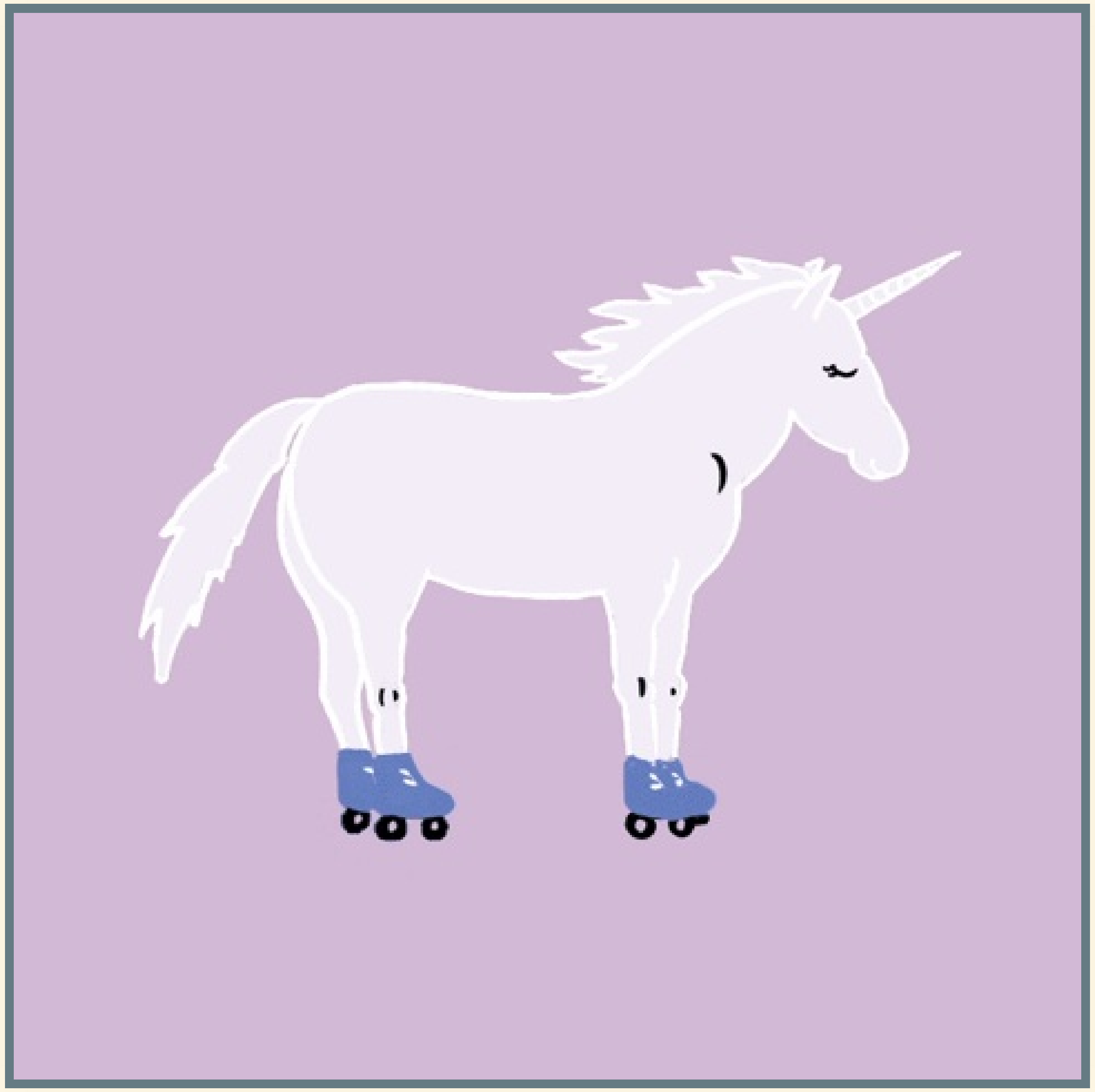
NODE 6

```
FROM node:6

# if you're doing anything beyond your local machine, please pin
FROM node:6

RUN mkdir -p /opt/app

# default to port 3000
ARG API_PORT=3000
ENV API_PORT $API_PORT
EXPOSE $API_PORT

# check every 30s to ensure this service returns HTTP 200
# HEALTHCHECK CMD curl -fs http://localhost:$PORT/healthz || exit
```

NODE 8

```
FROM node:8

# if you're doing anything beyond your local machine, please pin
FROM node:6

RUN mkdir -p /opt/app

# default to port 3000
ARG API_PORT=3000
ENV API_PORT $API_PORT
EXPOSE $API_PORT

# check every 30s to ensure this service returns HTTP 200
# HEALTHCHECK CMD curl -fs http://localhost:$PORT/healthz || exit
```

LEVEL UP!



NODE 8 LTS

- Node.js API (N-API)
- V8 5.8
- Buffer Improvements
- Async/Await (7.5+)
- NPM 5
- Better Promise API
- HTTP/2

ADDS TWO KEYWORDS

ASYNC

AWAIT

WHAT IS ASYNC/AWAIT?

"Async/Await keywords allows us to pause the execution of functions and this in turn writes asynchronous code that reads like synchronous code." - MPJ, FunFunFunction

EXAMPLE

```
async function foo () {
    const p1 = await bar()
    const p2 = await baz("World", 2000)
    return await p1 + p2 + await Promise.resolve('!');
}

function bar () {
    return new Promise((resolve)=>setTimeout(
        function(){resolve("Hello ")}, 500))
}

function baz(str, interval) {
    return new Promise((resolve)=>setTimeout(
        function(){resolve(str)}, interval))
}
```

http://jsbin.com/nufumiz/edit?js,console

EVALUATE IT IN AN IIFE

```
(async function main() {
        console.log(await foo())
})()
# > "Hello World!"
```

PERFORM IT IN PARALLEL EXAMPLE

```
var process = (x) => new Promise(resolve => setTimeout(() => reso
async function foo() {
  var s = new Date().getTime();
  var result1 = await process("1");
  var result2 = await process("2");
  console.log("foo: " + result1 + " " + result2 + ": time " + (ne
}
foo();

async function bar() {
  var s = new Date().getTime();
  var result1 = process("1");
  var result2 = process("2");
  console.log("bar: " + (await result1) + " " + (await result2) +
}
bar();
```

- @zirman

NOTES OF ASYNC AWAIT

- you can't await a non-top level function
- if you await something that doesn't return you'll block your application run
- Async await is learned from C# (2012) and probably in your other favorite language

References

- https://www.youtube.com/watch?v=568g8hxJJp4
- Building Iterators https://github.com/getify/You-Dont
JS/blob/master/async%20%26%20performance/ch4.r
aware-generator-runner
- Iterators https://medium.freecodecamp.org/demystify
iterables-iterators-4bdd0b084082 Credit
- Made with https://github.com/webpro/reveal-md