I want my ES6, like ES.NOW

Ken Rimple Mentor, Trainer, Consultant Chariot Solutions

Slides - http://1drv.ms/1DJQ4RO <-- that's not a zero

Topics

- What is ES6?
- Where are we now?
- How can we code ES6 now and run in ES5?
  - Client-side
  - Server-side

What is ES6?

ECMASCRIPT 6

- The next evolution of JavaScript (ECMASCRIPT === ES)
- Being embedded in browsers within the next 18 months
- Lots of new features

Major new features

- Classes, extends, constructors and member functions

```
class Klingon extends Warrior {
  constructor(name) { }
  methodA();
  methodB();
}
```

Local variables, constants

```
function foo() {

  // function-scoped
  var x = 234;

  if (x === 234) {
    let b = 24;  // scoped to if
  }
}
```

Arrow functions and string magic

```
myArray.forEach((val) =>
  console.log(`${val.a} - ${val.b}`);

var longStr = `
This is a long
string with
multiple lines`;
```

Default and "rest" parameters to functions

```
function foo(a = 234, ...b) {
  console.log(a);
  for (ele of b) {
     console.log(ele);
  }
}
```

Destructuring

Pull out parts of an object into variables

```
let a = { b: 'c', d: 'e' };
let {b, d} = a;
```

Better collections

```
let keys = new Set();
keys.add(1); keys.add(2);
keys.has(3) // false
keys.has(1) // true

let map = new Map();
map.set('a', 123);
map.set('b', 'foobarbaz');
map.has('a') // true
map.get('a') // 123
```

Promises (native to JS!)

```
Called API:

return new Promise((resolve, reject) => {
  resolve(answer);
  // error condition
  reject(errorData);
}

Caller:
apiCall.then(
  (answer) => { ... },
  (error) => { ... }
});
```

Generators

```
function *shoppingList() {
  yield 'apples';
  yield 'oranges';
  yield 'cereal';
}

let iterator = shoppingList();
while(true) {
  var obj = iterator.next();
  if (obj.done) break;
  console.log(obj.value);
}
```

What is supported now?

- See http://kangax.github.io/compat-table/es6/
  - Browser-side - not everything natively - unless you're on Firefox Nightly
  - Or you can use transpilers, shims and/or a build process
  - Server-side - depends on engine

ES6 in the Browser...

What about today?

Let's move the enterprise to that NOW!

How else? Two major themes

- Dynamic run-time translation
- Transpiling (translation compiling) to generate source (or source + library references)

Transpilers

Babel

- Has many flags and options
- Very configurable

Babel interactive API

```
< script src="lib/browser.js"></ script>
< script src="lib/browser-polyfill.js"></ script>
< script src="lib/shim.js"></ script>
```

Now you can mount scripts with type of `text/babel` and they will be transpiled

Approach not quite recommended...

- Babel is meant to be run as part of a build
- Not all features work in a dynamic mode (modules, etc.)

Babel command line

```
$ npm install -g babel

$ babel foo.js
// output returned inline
```

Better yet, you should use a build system

Build options

- Build-based transformer plugin (gulp-babel)
- Browserify with babel plugin
- Karma with pre-processor

Babel Gulp project demo

Code transformers supported

Use `--help` to return list of transformers

```
$ babel --help

 Transformers:

    - [asyncToGenerator]
    - [bluebirdCoroutines]
    - es3.memberExpressionLiterals
    - es3.propertyLiterals
    - es5.properties.mutators
    ...
```

Adding or removing transformers

- Babel has a set of default and optional transformers
- Blacklist normally included transformers to remove

```
babel --blacklist=es6.spread,
      es6.regex.unicode foo.js
```

- Whitelist - only these transformers are executed

```
babel --whitelist=es6.classes,es6.forOf
```

- Concept - remove transpiler features as browsers support them - reality? Maybe not or complex build targeting each browser supported...

Traceur

Using Traceur in the browser

```
< script src=".../traceur.js">< /script>
< script src=".../bootstrap.js">< /script>
< script type="module">
      Your ES6 code here...
< /script>
```

Traceur in a build

- You can use
  - tracer as a command-line utility
  - traceur-gulp (|grunt|whatever)
  - 6to5ify with Browserify
  - Karma traceur preprocessor

Some challenges

- Generated ES5 code is not always correct
- The transpiler cannot know all API changes by itself
  - Example - Array methods

```
Array.from({length: 5}, (v, k) => k);
```

- `from` is an ES6 feature in `Array`, but was not included in the transpiler directly.
- Babel provides a polyfill for this - `polyfill.js` - in the library

Transpiling is not a panacea

- You need to pick a module loader strategy to transpile down to
- You need to re-test once browsers begin to support features
- Some ES6 features (tail-recursion) will require binary browser changes
- Some APIs won't be available without polyfills
- Check sites like Kangax

Where are we heading?

- Angular 2.0 - will use the traceur compiler + AtScript (===> Typescript 1.5)
- Typescript 1.5 - will generate ES5 AND ES6, contain AtScript annotationes
- Ember 2 - will use ES6 coding features (with Babel)
- EVENTUALLY - ES6 rules them all
- But we thought that with HTML5 too

Server-side ECMAScript 6

Several Options here

- NodeJS with a transpiler to ES5 (see prior examples)
- NodeJS with a wrapper such as `node-babel`
- NodeJS activating Harmony features directly
- io.js (fork of NodeJS)

NodeJS and ECMAScript 6 features w/Babel

- Just install Babel or
- Run `node-babel`

NodeJS and ECMAScript 6

- You can enable ES6 features
- Much of ES6 is implemented
- You must turn on special flags

```
$ node --v8-options | grep "harmony"
  --harmony_scoping (enable harmony block scoping)
  --harmony_modules (enable harmony modules
      (implies block scoping))
  --harmony_generators (enable harmony generators)
  --harmony_strings (enable harmony string)
  --harmony_arrays (enable harmony arrays)
  ...
```

Welcome `babel-node`

- Runs Node with a Babel shim

```
babel-node
> myfunc = () => console.log('hiya!');
> myfunc();
hiya!
```

- All ES6 harmony flags supported as of 5.0.8

ES6 Node server example

Alternative to NodeJs - io.js

- A fork of `node.js`
- Updated status `https://github.com/iojs/io.js/issues/1336`
- Controversial, but has a number of developers focused on moving the platform forward
- Already supports a number of ES6 features out of the box, can turn on the rest

ES6-ready framework example

- Koa - developer of Express - uses `--harmony` flag

Should you do this now?

- Up to you
- Specs will be tweaked
- Do NOT dive in and use every feature without weighing it
  - You can still code "Good Parts" style
  - But Crockford has a video on the newer "Good Parts" of ES6
  - Link: http://goo.gl/eD4UB5

Resources

- Traceur - https://github.com/google/traceur-compiler
- Babel - https://babeljs.io
- Slides - http://1drv.ms/1DJQ4RO <-- that's not a zero