THE PROJECTOR WORKS
MODULES EVERYWHERE
EV SSL:

- Barclays Bank PLC [GB] https://bank.barclays.co.uk
- Stripe, Inc. [US] https://stripe.com/gb
- CurrencyTransfer Ltd [GB] https://app.currencytransfer.com
- A Medium Corporation [US] https://medium.com

Actually checks who you are, shows that in browser
40X FASTER
HOW CERTSIMPLE IS BUILT
We use npm on the **backend**

We use npm on the **frontend**

**All our code** is npm modules
ALL THE THINGS WE WISH WE’D KNOWN EARLIER
Why we chose npm

How you can use npm

Things to avoid
More packages
Every frontend package CertSimple needed was officially maintained on npm

- stripe
- jQuery
- Ractive.js
- AGAVE.js
- tags-input
- selectize.js
Attitude to npm over time

selectize.js

Release on NPM #60

Closed andreypopp opened this issue on Aug 3, 2013 · 13 comments

brianreavis commented on Aug 3, 2013
This isn't a node package, so I'd really like to not clutter NPM with it.

brianreavis commented on Jan 30
Caved in and published it: https://www.npmjs.com/package/selectize
General JS things frontend needed also on npm

es6-shim

country-data

array-includes

alpha-sort

superagent
USE REAL PROJECTS, NOT STACKOVERFLOW
Other module formats?

- **window globals** are dead

- **RequireJS** never had a module repository - dead

- **UMD** only exists because of globals and Require - just dead code at this point.

- Very few **ES6** packages, and they’ll be on npm anyway
Building, & bundling npm modules

<table>
<thead>
<tr>
<th>Build Tasks</th>
<th>gulp</th>
<th>npm run-script build</th>
<th>webpack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bundle</td>
<td>browserify</td>
<td>browserify</td>
<td>webpack</td>
</tr>
</tbody>
</table>
Webpack?

- Can bundle npm packages
- Replaces build system as well
- 8-10% smaller modules + hot reload
- We use a bunch of existing gulp tools, so benefit not yet worth it
modules everywhere

Mike MacCana @mikemaccana
@izs didn't this used to have 'no lib dir, everything is a module?' in it? Does that still apply? docs.npmjs.com/misc/coding-st...

THAT isaac @izs
@mikemaccana I DO usually do that. But not all the time, and npm's initial design predates npm existing ;)
18/05/2015 16:21
Why just modules?

- Simple imports
- Modules are a reasonable unit of granularity
- Unlike /lib, every module has ‘test’, README.md, packages.json, etc.
- Easier to open source
Making your modules prominent

Automatically build a Sublime .project file that prioritises private npm modules
Bundling

This is the entire `js/src/index.js`

Makes a bundle in `js/dist/index.js`
The ‘frontend’ module:

```javascript
var Ractive = require("ractive"),
    $ = require("jquery"),
    superagent = require("superagent"),
    scroll = require('scroll'),
    tagsInput = require('tags-input'),
    allCountries = require('certsimple-countries'),
    paymentTemplateHelpers = require('payment-template-helpers');

// https://github.com/stripe/jquery.payment/issues/152
window.$ = $;
require("jquery.payment");
require('selectize');
require('browser-debug-mode');
require('es6-shim');

// Only enable debug when unminified
Ractive.DEBUG = /unminified/.test(function() {/*unminified*/});

var SPACE = 32,
    PAGE_UP = 33
```
How to bundle templates?

```json
package.json
{
  "name": "certsimple-frontend",
  "version": "1.0.0",
  "private": true,
  "browserify": {
    "transform": [ "brfs"
  ]
},
```

```js
// Note this can't be merged into another var statement, due to the way brfs works.
var fs = require("fs");

var evCertificatePartial = fs.readFileSync(__dirname + '/ev-certificate.mustache', 'utf8');
var signupStepsPartial = fs.readFileSync(__dirname + '/signup-steps.mustache', 'utf8');
```
Analysing bundle size:
Deploying

- package.json only specified version for your immediate dependencies
  - a version 1.0.0 - only use b version 1.2.0
    - b version 1.2.0 - use version c 1.0.0 or newer
      - c - could be any version
  - package.json alone isn’t enough to ensure C has a consistent version
npm shrinkwrap

- Provides **consistent deploys**
- Specifies the full tree, all the way down
- Built into npm
- Avoids overhead of committing node modules
THANKS!
Links!

Module counts
http://www.modulecounts.com/

Discify
http://hughsk.io/disc/

Sublime prioritise private modules
https://gist.github.com/mikemaccana/fe5c1a83bcbdc0157534

Browserify + gulp
https://github.com/gulpjs/gulp/blob/master/docs/recipes/browserify-uglify-sourcemap.md

Webpack vs Browserify
https://github.com/k88hudson/browserify-webpack#results