MOVING AWAY FROM NODEJS TO A PURE PYTHON SOLUTION FOR ASSETS

Alessandro Molina
@__amol__
amol@turbogears.org

Why?

- NodeJS has a rich toolkit, but using it has some serious drawbacks.
- People really don't know there are very good or even better alternatives.
- Having to cope with complexity and idiosyncrasies of two languages is bad

Let’s talk about a common scenario…

Proudly Python!

- You wrote your app in Python
  - Django, TurboGears, Pyramid, Flask, Bottle...
- You run it using Python
  - Circus, Supervised, mod_wsgi, uwsgi, gevent...
- You deploy it through Python
  - Ansible, Salt, DockerCompose
- You monitor its state with Python
  - Datadog, BackLash, Sentry, NewRelic, OpBeat

Then one day...


WHEN DID A NEW LANGUAGE AND PACKAGE MANAGER
BECAME DEPENDENCIES OF MY PROJECT?!

Probably Assets!

- NodeJS has a great set of tools to:
  - Perform CSS/JS minification
  - Transpile Languages
  - Automated WebApps Testing
  - Perform Automatic Tasks

- Grunt & Gulp have a huge set of plugins to manage those tasks automatically

But now...

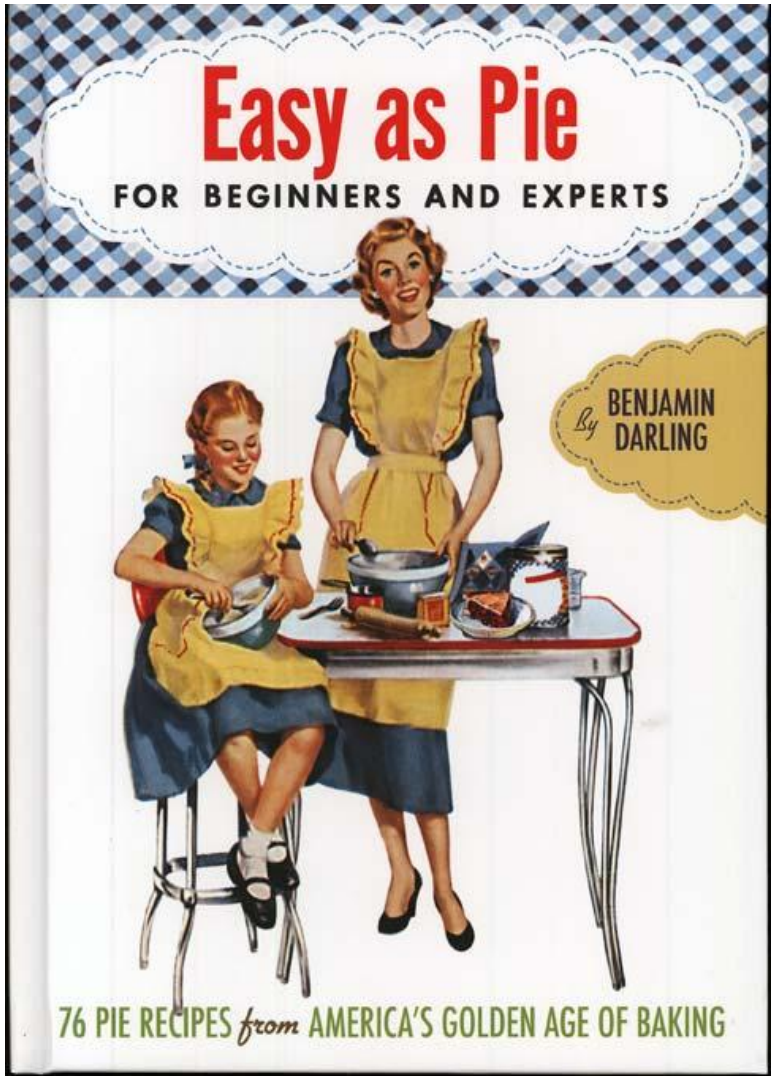
- You need a package manager to manage the package managers
- You have two places where dependencies are tracked.
- How long before you will introduce a third solution to manage the other two?

WebAssets to the rescue

- Can replace Grunt or Gulp in managing your assets transformation pipeline
- Tools are available as Python distributions, only track dependencies in setup.py or requirements.txt
- Works with any WSGI framework
- Built-in cache busting for free

Define Assets Bundles and Filters

```
bundles:
    style:
        filters: cssutils
        output: build/css/style.css
        contents:
            - myapp/css/bootstrap.min.css
            - myapp/css/c3.css
            - myapp/css/bootstrap-datepicker3.standalone.min.css
            - contents:
                - myapp/css/style.scss
                - myapp/css/devices.scss
                - myapp/css/consumption.scss
              filters: libsass
    jsall:
        filters: jsmin
        output: build/js/dependencies.js
        contents:
            - myapp/config.js
            - myapp/js/browser-polyfill.min.js
            - myapp/js/jquery-2.1.4.min.js
            - myapp/js/bootstrap.min.js
            - myapp/js/ractive.js
            - myapp/js/utils.js
            - myapp/js/controllers/devices.js
            - myapp/js/controllers/consumption.js
            - myapp/js/app.js
```



Just use them

- Add style bundle

```
<link py:for="asset_url in g.webassets.style.urls()"
      rel="stylesheet" type="text/css" media="screen"
      href="$asset_url" />
```

- Add jsall bundle

```
<script py:for="asset_url in g.webassets.jsall.urls()"
        src="$asset_url"></script>
```

Yeah... Cool... But...

Javascript cross-compilers

class **webassets.filter.babel.Babel**(***kwargs*)

Processes ES6+ code into ES5 friendly code using Babel.

Requires the babel executable to be available externally. To install it, you might be able to do:

```
$ npm install --global babel-cli
```

You probably also want some presets:

```
$ npm install --global babel-preset-es2015
```

Example python bundle:

```
es2015 = get_filter('babel', presets='es2015')
bundle = Bundle('**/*.js', filters=es2015)
```

Example YAML bundle:

```
es5-bundle:
    output: dist/es5.js
    config:
        BABEL_PRESETS: es2015
    filters: babel
    contents:
        - file1.js
        - file2.js
```

We replaced Grunt and Gulp, but...

more advanced filters
still rely on NodeJS and npm

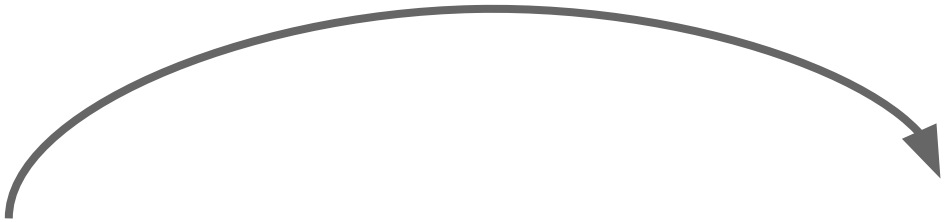
That's why DukPy was created!

- DukPy can replace NodeJS in asset management pipelines
- `pip install dukpy` or add it to your setup.py and you are ready to go.

DukPy

- No external dependencies apart from a working C compiler.
- Other solutions like PyExecJS rely on Spidermonkey, V8 and so on… which are really hard to build.
- Tailored explicitly for Web Development
- Cames with built-in WebAsset filters

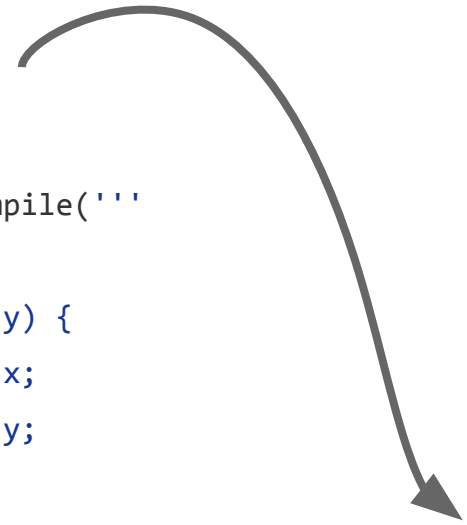
CoffeeScript



```
>>> import dukpy
>>> s = dukpy.coffee_compile('''
...     fill = (container, liquid = "coffee") ->
...         "Filling the #{container} with #{liquid}..."
... ''')
```

```
>>> print s
(function() {
  var fill = function*(container, liquid) {
    if (liquid == null) {
      liquid = "coffee";
    }
    return "Filling the " + container + "
with " + liquid + "...";
  };
}).call(this);
```

ES6



```
>>> import dukpy
>>> s = dukpy.babel_compile('''
... class Point {
...     constructor(x, y) {
...           this.x = x;
...           this.y = y;
...     }
...     toString() {
...          return '('+this.x+', '+this.y+')';
...     }
... }
... ''')
```

```
>>> print s
"use strict";
var _prototypeProperties = ...
var _classCallCheck = ...

var Point = (function () {
    function Point(x, y) {
        _classCallCheck(this, Point);
        this.x = x;
        this.y = y;
    }
    _prototypeProperties(Point, null, {
        toString: {
            value: function toString() {
                return "("+this.x+", "+this.y+")";
            },
            writable: true,
            configurable: true
        }
    });
    return Point;
})();
```

TypeScript

```
>>> import dukpy
>>> dukpy.typescript_compile('''
... class Greeter {
...     constructor(public greeting: string) { }
...     greet() {
...         return "<h1>"+this.greeting+"</h1>";
...     }
... };
... var greeter = new Greeter("Hello, world!");
... ''')
```

```
>>> print s
var Greeter = (function () {
    function Greeter(greeting) {
        this.greeting = greeting;
    }
    Greeter.prototype.greet = function () {
        return "<h1>"+this.greeting+"</h1>";
    };
    return Greeter;
})();
;
var greeter = new Greeter("Hello, world!");
```

Built-in as WebAssets filters

```
from webassets.filter import
register_filter
```

```
from dukpy.webassets import BabelJS
register_filter(BabelJS)
```

```
from dukpy.webassets import TypeScript
register_filter(TypeScript)
```

```
jsapp:
  filters: babeljs
  output: build/js/app.js
  contents:
    - app/js/data_abstraction_layer.js
    - app/js/utils.js
    - app/js/controllers/devices.js
    - app/js/controllers/home.js
    - app/js/controllers/history.js
    - app/js/app.js
```

Manage your Javascript Deps

- `dukpy.install_jspackage` able to install nodejs packages in your python projects

```
>>> import dukpy
>>> dukpy.install_jspackage('react', '0.14.8', './js_vendor')
Downloading https://registry.npmjs.org/react/-/react-0.14.8.tgz
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
....
Extracting...
Installed in ./js_vendor/react
```

React ServerSide Rendering

- `dukpy.jsx_compile` and `require()`

```
>>> code = '''
... var React = require('react/react'),
... ReactDOM = require('react/react-dom-server');
... var HelloWorld = React.createClass({
...  render: function() {
...    return (
...      <div className="helloworld">
...        Hello {this.props.data.name}
...      </div>
...    );
...  }
... });
... ReactDOM.renderToStaticMarkup(<HelloWorld data={dukpy.data}/>, null);
... '''
>>> jsx = dukpy.jsx_compile(code)
>>> dukpy.evaljs(jsx, data={'id': 1, 'name': "Alessandro"})
u'<div class="helloworld">Hello Alessandro</div>'
```

Run Python code from JS

- `dukpy.JSInterpreter.export_function` to export python functions to JS

```
>>> jsi = dukpy.JSInterpreter()
>>> jsi.export_function('sort_numbers', sorted)
>>> jsi.evaljs("var nums=[5,4,3,2,1]; call_python('sort_numbers', nums)")
[1, 2, 3, 4, 5]
```

Python all the way down!

Feel free to try it!

- Python 2.6, 2.7, 3.2, 3.3, 3.4 and 3.5
- pip install dukpy
- Tested with 100% coverage

  https://travis-ci.org/amol-/dukpy
- Come and try it!

  https://github.com/amol-/dukpy

Questions?