

# AssetGraph

Andreas Lind Petersen

June 16th, 2011

Bird's eye view

Data structures

Code

One.com build system

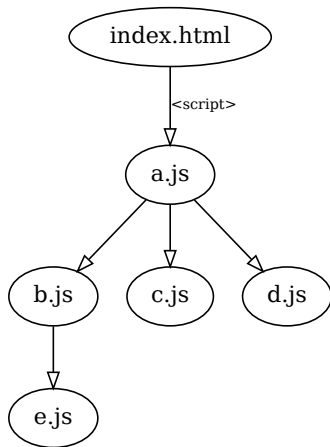
# Bird's eye view

# Developer friendly

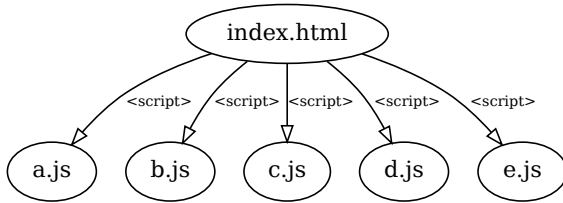
≠

# web friendly

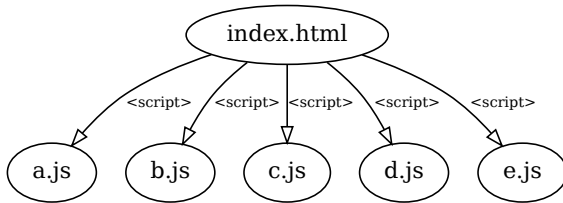
# Find and flatten dependencies



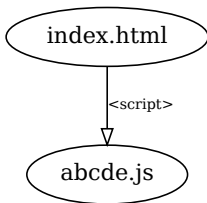
# Find and flatten dependencies



# Bundle JavaScript

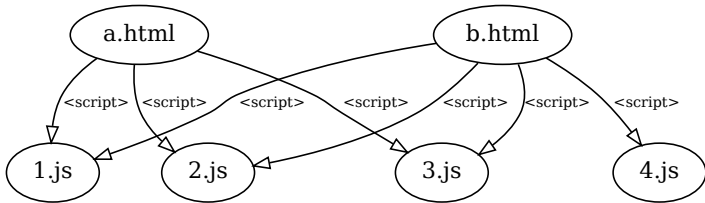


# Bundle JavaScript

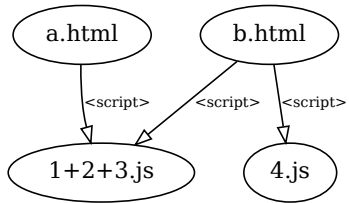




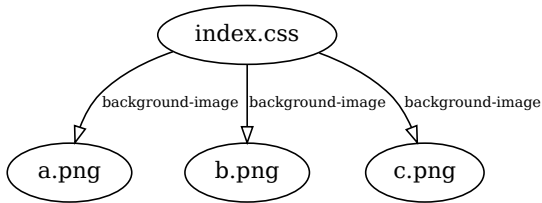
# Create shared bundles



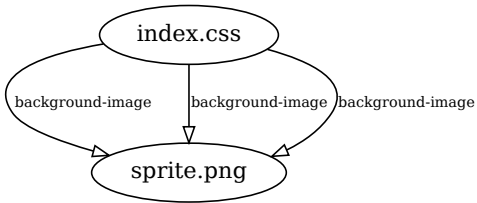
# Create shared bundles



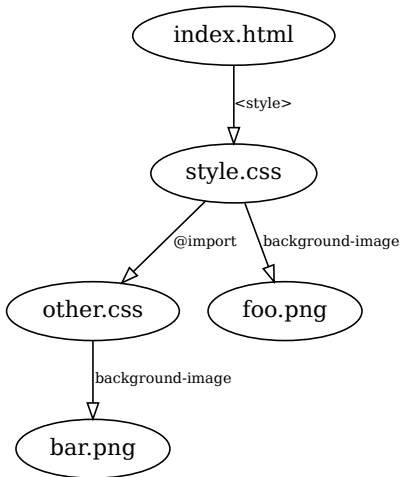
# Sprite background images



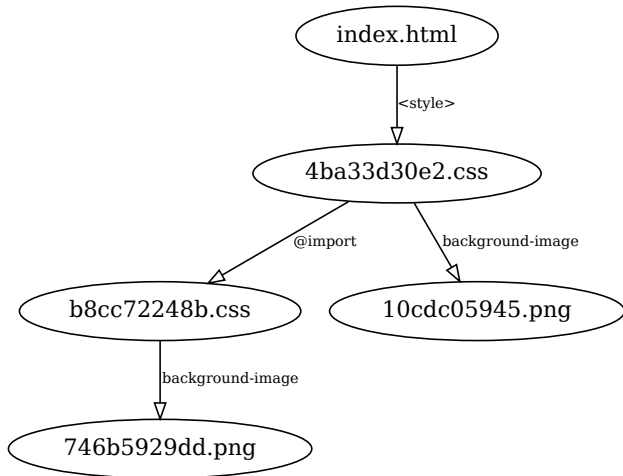
# Sprite background images



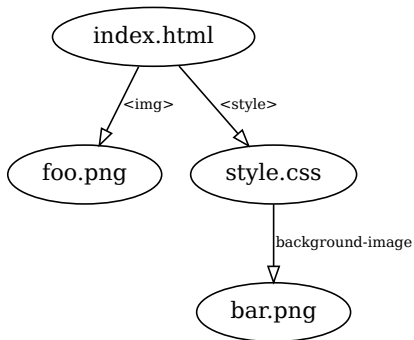
# Rename to hash of contents



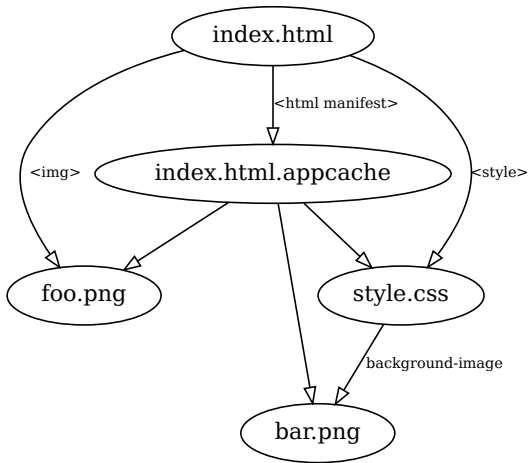
# Rename to hash of contents



# Add a cache manifest

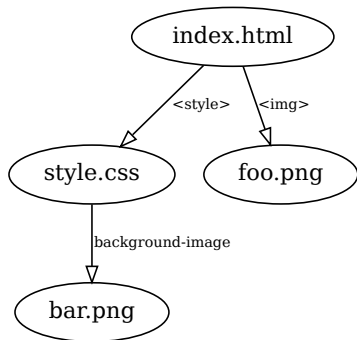


# Add a cache manifest

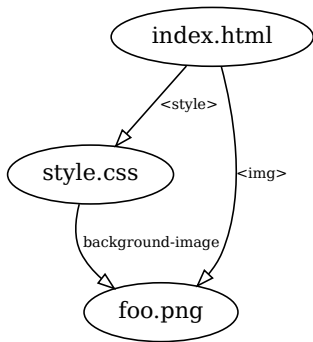




# Deduplication



# Deduplication



# Data structures

- Graph network

- Graph network
- Assets (=nodes)  
HTML, CSS, JavaScript, images, Json. . .

- Graph network
- Assets (=nodes)  
HTML, CSS, JavaScript, images, Json...
- Relations (=edges)  
<link>, <script>, <img>, <iframe>, @import,  
background-image...

- Graph network
- Assets (=nodes)  
HTML, CSS, JavaScript, images, Json...
- Relations (=edges)  
<link>, <script>, <img>, <iframe>, @import,  
background-image...
- Easy manipulation

- Graph network
- Assets (=nodes)  
HTML, CSS, JavaScript, images, Json...
- Relations (=edges)  
<link>, <script>, <img>, <iframe>, @import,  
background-image...
- Easy manipulation
- Other core features:
  - Inline relations
  - Move assets
  - Get relative urls right



# Code

# Querying the graph

# Querying the graph

```
assetGraph.findAssets({type: "Html"})
```

# Querying the graph

```
assetGraph.findAssets({type: "Html"})
```

```
assetGraph.findRelations({  
  type: "HtmlAnchor",  
  to: {type: "Png"}  
}).forEach(function (htmlAnchor) {  
  console.log(htmlAnchor.to.url);  
});
```

# High level operations (“transforms”)

# High level operations (“transforms”)

- Inline all relations matching a query

```
transforms.inlineRelations({type: 'HtmlScript'})
```

# High level operations (“transforms”)

- Inline all relations matching a query

```
transforms.inlineRelations({type: 'HtmlScript'})
```

- Move all assets matching a query to a new root

```
transforms.moveAssetsToNewRoot({  
  type: ['Png', 'Gif', 'Jpeg', 'Ico', 'Css', ' ←  
        JavaScript']  
}, 'http://static.one.com/')
```

# Example workflow: Inline background images

```
var AssetGraph = require('assetgraph'),
    t = AssetGraph.transforms;

new AssetGraph({root: '/path/to/files/'}).queue(
  t.loadAssets('*.*css'),
  t.populate({followRelations: {to: 'CssImage'}}),
  t.inlineRelations({type: 'CssImage'}),
  t.writeAssetsToDisc()
).run(function (err) {
  // CSS now written back with inlined images.
});
```



# Example: Load HTML + related assets recursively

```
var AssetGraph = require('assetgraph'),
    transforms = AssetGraph.transforms;

AssetGraph({root: '.'}).queue(
  transforms.loadAssets('*.*html'),
  transforms.populate({
    followRelations: {to: {url: /^file:/}},
    onError: function (err) {
      console.error(err.message);
      process.exit(1);
    }
  })
).run();
```

# One.com's build system



- Very opinionated

- Very opinionated
- buildDevelopment and buildProduction

- Very opinionated
- `buildDevelopment` and `buildProduction`
- Single-page web apps

- Very opinionated
- `buildDevelopment` and `buildProduction`
- Single-page web apps
- Steve Souders' shopping list

- Very opinionated
- `buildDevelopment` and `buildProduction`
- Single-page web apps
- Steve Souders' shopping list
- Self-contained components



- Very opinionated
- `buildDevelopment` and `buildProduction`
- Single-page web apps
- Steve Souders' shopping list
- Self-contained components
- Relative urls

- Very opinionated
- `buildDevelopment` and `buildProduction`
- Single-page web apps
- Steve Souders' shopping list
- Self-contained components
- Relative urls
- Far-future Cache-Control/Expires on everything

- Very opinionated
- `buildDevelopment` and `buildProduction`
- Single-page web apps
- Steve Souders' shopping list
- Self-contained components
- Relative urls
- Far-future Cache-Control/Expires on everything — except entry-point HTML and cache manifests

- Very opinionated
- `buildDevelopment` and `buildProduction`
- Single-page web apps
- Steve Souders' shopping list
- Self-contained components
- Relative urls
- Far-future Cache-Control/Expires on everything — except entry-point HTML and cache manifests
- New opinions formed every day

# Internationalization

# Internationalization

- namespace/Component.js:

```
alert(one.tr("namespace.Component.Greeting"));
```

- namespace/Component.i18n:

```
{  
  "namespace.Component.Greeting": {  
    "en": "Hello",  
    "en_AU": "Howdy",  
    "da": "Goddag"  
  }  
}
```

# Internationalization

- namespace/Component.js:

```
alert(one.tr("namespace.Component.Greeting"));
```

- namespace/Component.i18n:

```
{  
  "namespace.Component.Greeting": {  
    "en": "Hello",  
    "en_AU": "Howdy",  
    "da": "Goddag"  
  }  
}
```

- Produces three separate builds in production:

```
alert("Hello"); // index.en.html  
alert("Howdy"); // index.en_AU.html  
alert("Goddag"); // index.da.html
```

# Urls in JavaScript code

```
div.innerHTML = '<img src="' +  
    one.getStaticUrl('images/blank.gif') +  
    '>';
```



# Urls in JavaScript code

```
div.innerHTML = '<img src="' +  
    one.getStaticUrl('images/blank.gif') +  
    '>';  
  
$.ajax(one.getStaticUrl('myStuff.json'), {  
    success: function () {...}  
});
```

# TODO

# TODO

- Documentation :)

# TODO

- Documentation :)
- Lazy loading

# TODO

- Documentation :)
- Lazy loading
- More script loaders, Dojo, YUI, require.js, LABjs, Control.js

# TODO

- Documentation :)
- Lazy loading
- More script loaders, Dojo, YUI, require.js, LABjs, Control.js
- Your project?

# Getting it

# Getting it

- Install Ubuntu



# Getting it

- Install Ubuntu
- Install node.js and npm

# Getting it

- Install Ubuntu
- Install node.js and npm
- Run:

```
$ sudo apt-get install libcairo2-dev libjpeg8-dev \  
                        optipng pngcrush pngquant
```

```
$ npm install assetgraph
```