第二届

前端开发者年度大会

FEDAY

欢聚时代 YY Inc. · QINIU · upyun · CODING CLOUD DEVELOPMENT · 友好速搭 品牌电商服务平台 · TURING 图灵教育 · Broadview www.broadview.com.cn · 100offer 程序员拍卖 · 无忧精英

Universal Applications

Stepan Parunashvili

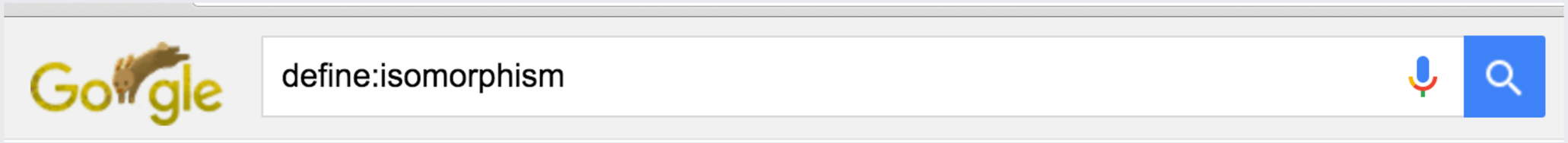
潘蒂文

@stopachka

www.stepanp.com

Universal Applications

Isomorphic?


Google
define:isomorphism

.... Formally, an **isomorphism** is bijective morphism. Informally, an **isomorphism** is a map that preserves sets and relations among elements.


MOVIECLIPS.com

~~Isomorphic~~

Universal

...2015

...2014

...2013

...2012

...2011

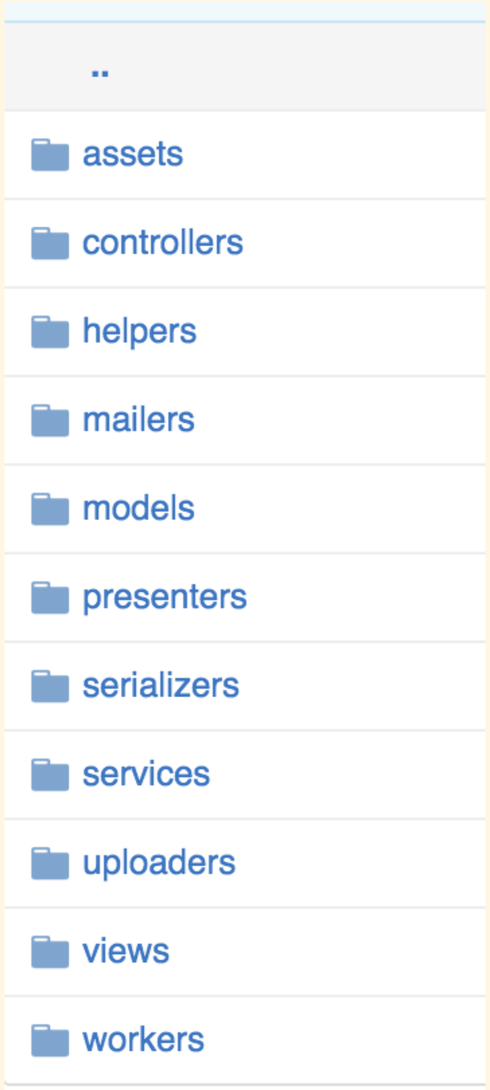
...2010

...2009

2008

Coldplay - Viva La Vida - YouTube
https://www.youtube.com/watch?v=dvgZkm1xWPE

v2.2.2

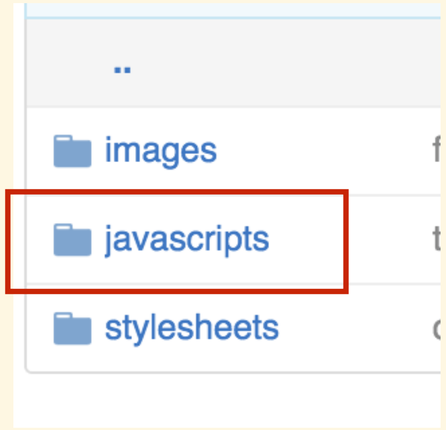

..
assets
controllers
helpers
mailers
models
presenters
serializers
services
uploaders
views
workers


..
assets
controllers
helpers
mailers
models
presenters
serializers
services
uploaders
views
workers

javascript?


..
assets
controllers
helpers
mailers
models
presenters
serializers
services
uploaders
views
workers

这里！




RAILS


JS

routing
validation
views

routing
validation
views

c00l
animations

routing
validation
views

c00l animations

(in some browsers...b^.^d**)**

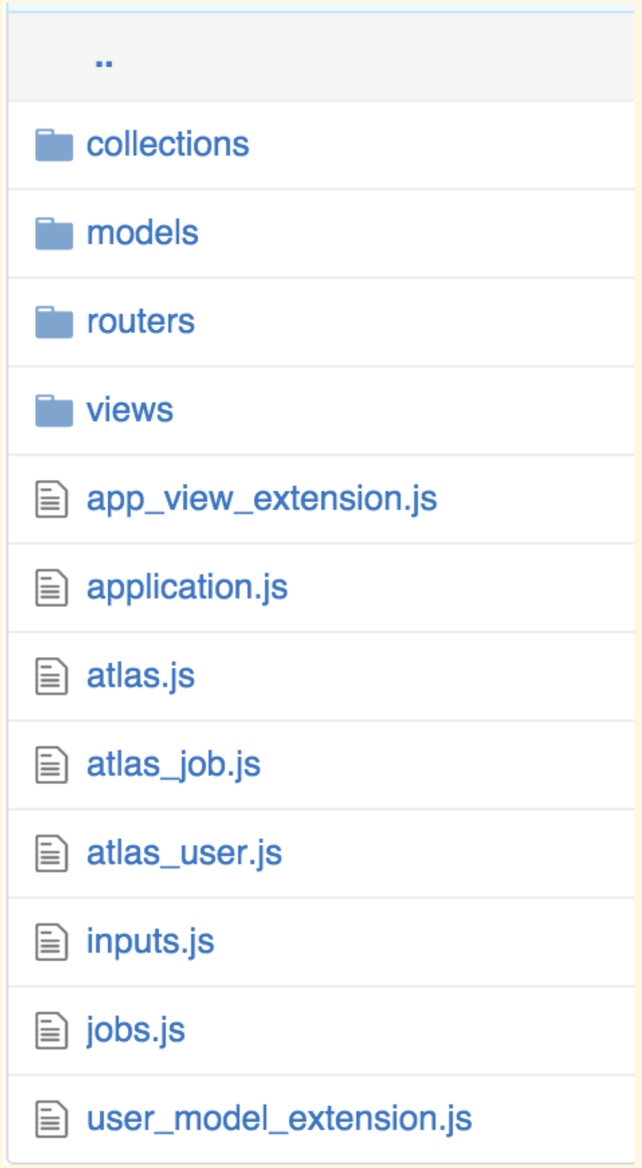
```
$(document).ready(function() {
  welcomeScreen.helloAnimation();
});
```

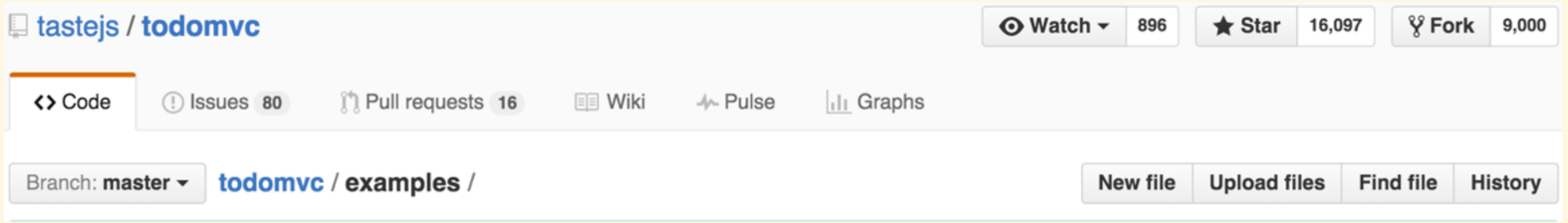
2009...

2010

Justin Bieber - Somebody To Love Remix ft. Usher - YouTube
https://www.youtube.com/watch?v=SOI4OF7iIr4

v0.1.0


..
collections
models
routers
views
app_view_extension.js
application.js
atlas.js
atlas_job.js
atlas_user.js
inputs.js
jobs.js
user_model_extension.js


tastejs / todomvc
Watch 896
Star 16,097
Fork 9,000
Code
Issues 80
Pull requests 16
Wiki
Pulse
Graphs
Branch: master
todomvc / examples /
New file
Upload files
Find file
History

ampersand	Fixes Long Task Overflow	3 months ago
angular-dart	Fixes Long Task Overflow	3 months ago
angular2	Added myself as an author!	2 months ago
angularjs-perf	angularjs-perf: Removed todo-escaped directive	2 months ago
angularjs	Fixes Long Task Overflow	3 months ago
angularjs_require	Fixes Long Task Overflow	3 months ago
ariatemplates	Fixes Long Task Overflow	3 months ago
atmajs	Fixes Long Task Overflow	3 months ago
aurelia	Aurelia and Mozart: rename `README.md` to `readme.md`	9 months ago
backbone	Backbone example: add id attr to input	2 months ago
backbone_marionette	Fixes Long Task Overflow	3 months ago
backbone_require	Fixes Long Task Overflow	3 months ago
canjs	Fixes Long Task Overflow	3 months ago
canjs_require	Fixes Long Task Overflow	3 months ago
chaplin-brunch	Updating the Marionette, Chaplin Comparision Link	14 days ago
closure	Fixes Long Task Overflow	3 months ago
componentjs	Componentjs: remove count from clear completed button	8 months ago
cujo	Cujo: drop transition	8 months ago
dijon	Fixes Long Task Overflow	3 months ago
dojo	Fixes Long Task Overflow	3 months ago
duel	Regenerated static files.	6 months ago
durandal	Typo fix in `durandal` example	8 months ago
elm	Fixes Long Task Overflow	3 months ago
emberjs	Fixes Long Task Overflow	3 months ago
emberjs_require	The Big Examples Move 🚚	a year ago
enyo_backbone	Fixes Long Task Overflow	3 months ago
exoskeleton	Fixes Long Task Overflow	3 months ago
extjs_deftjs	Fixes Long Task Overflow	3 months ago
firebase-angular	firebase-angular: Correct directive name	2 months ago
flight	Fixes Long Task Overflow	3 months ago
foam	Fixes Long Task Overflow	3 months ago
gwt	Fixes Long Task Overflow	3 months ago
humble	Fixes Long Task Overflow	3 months ago
jquery	Fixes Long Task Overflow	3 months ago
js_of_ocaml	Fixes Long Task Overflow	3 months ago
jsblocks	Fixes Long Task Overflow	3 months ago
kendo	Kendo: remove count from clear completed button	11 months ago
knockback	Fixes Long Task Overflow	3 months ago
knockoutjs	Fixes Long Task Overflow	3 months ago
knockoutjs_require	Fixes Long Task Overflow	3 months ago
lavaca_require	Fixes Long Task Overflow	3 months ago
meteor	Fix file paths in app readme files	9 months ago
mithril	Fixes Long Task Overflow	3 months ago
olives	Fixes Long Task Overflow	3 months ago
polymer	Update to polymer 1.2.3	4 months ago
puremvc	Fixes Long Task Overflow	3 months ago
ractive	Fixes Long Task Overflow	3 months ago
rappidjs	Fixes Long Task Overflow	3 months ago
react-alt	Fixes Long Task Overflow	3 months ago
react-backbone	Fixes Long Task Overflow	3 months ago
react	Fixes Long Task Overflow	3 months ago

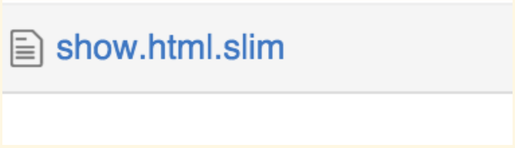

RAILS

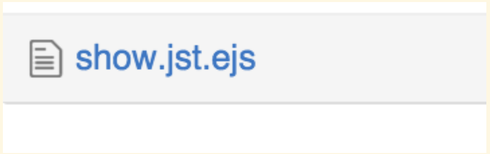

JS

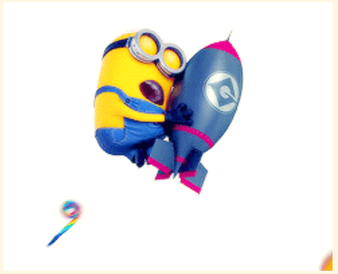
routing
validation
views

...routing
...validation
...views

routing
validation
views


show.html.slim


show.jst.ejs



is it just the way things are?


RAILS


JS


JS


JS

routing
validation
views

```
$('#root').prepend('<div>hello!</div>')
```

what if we made our own representation of the dom...

...React!

win 1:
code sharing

win 2:
perceived performance

win 3:
SEO

Can we build this?

Can we build this?

准备了?

server.js

```
import express from 'express';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send('Hello!');
});
```

server.js

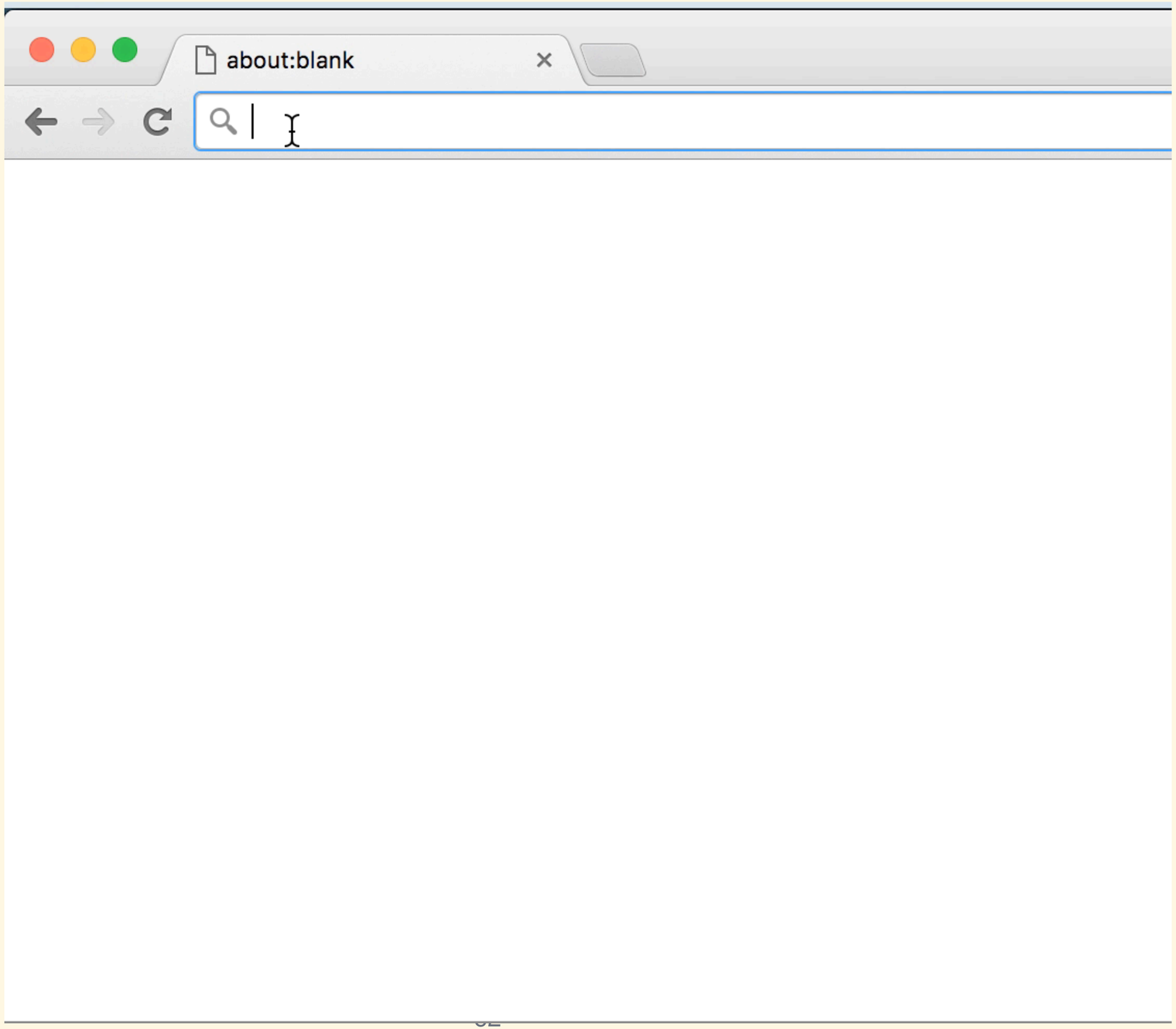
```
import express from 'express';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send('Hello!');
});
```

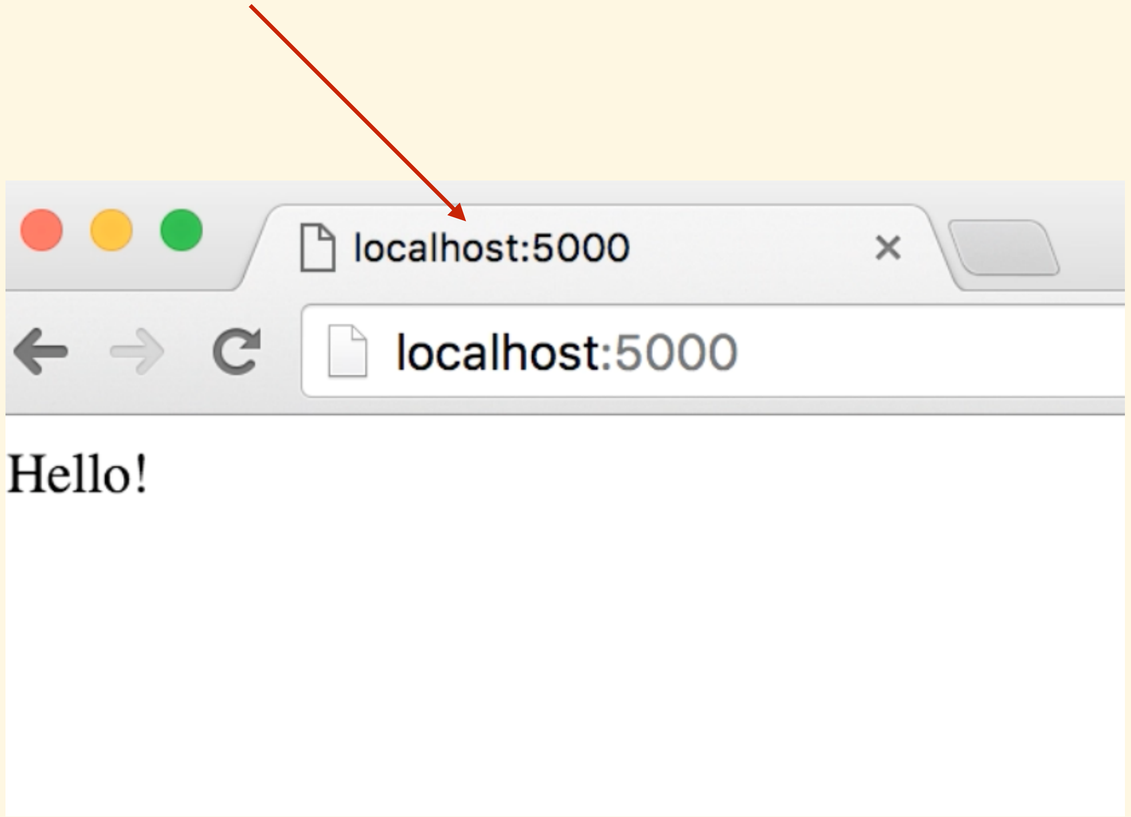
server.js

```
import express from 'express';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send('Hello!');
});
```

server.js

```
import express from 'express';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send('Hello!');
});
```


about:blank


localhost:5000
localhost:5000
Hello!

server.js

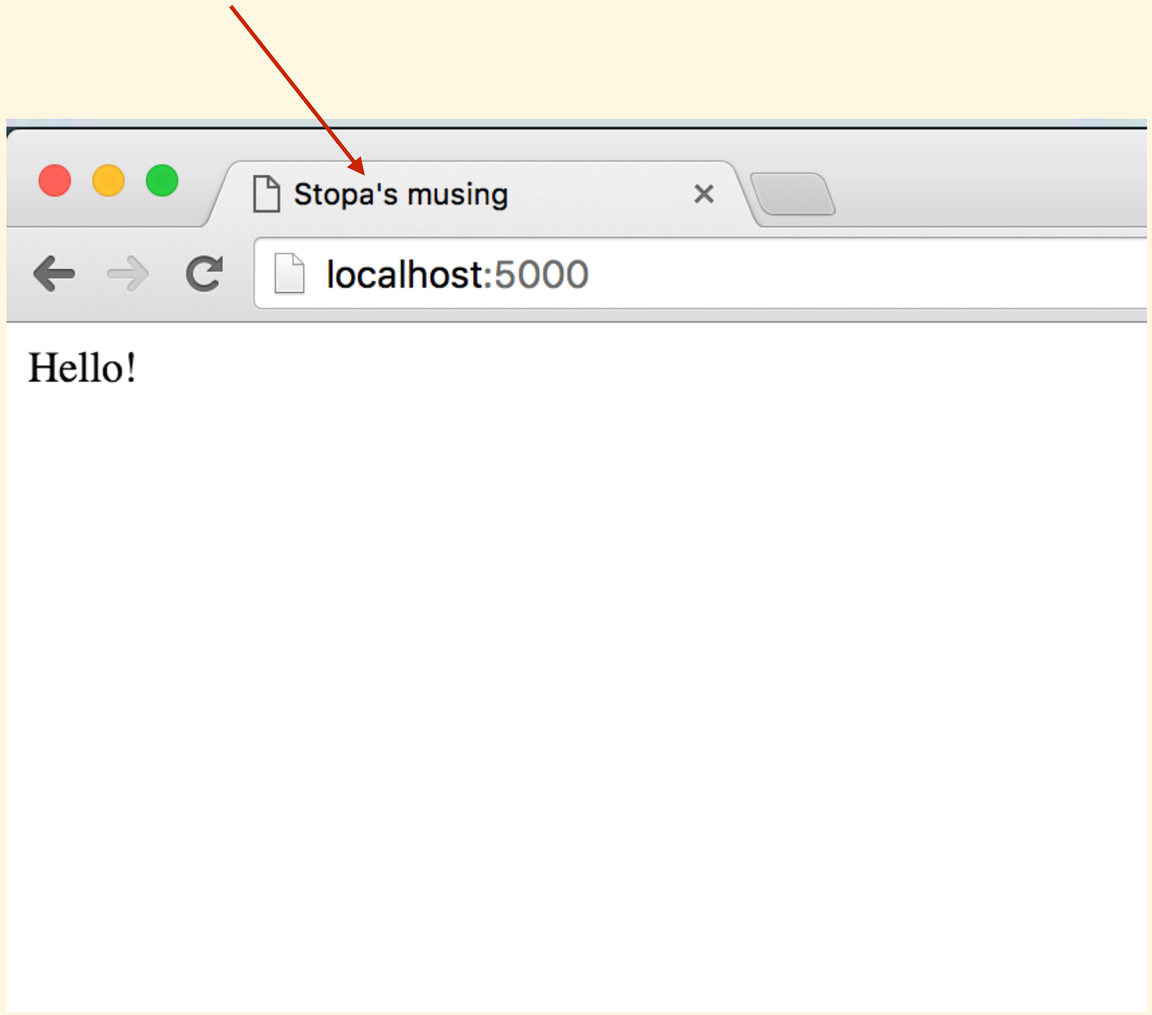
```
function template(body) {
  return `
    <!doctype html>
    <html>
      <head>
        <title>Stopa's musing</title>
      </head>
      <body>
        ${body}
      </body>
    </html>
  `
}
```

server.js

```
import express from 'express';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template('Hello!'));
});
```

server.js

```
import express from 'express';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template('Hello!'));
});
```


Stopa's musing
localhost:5000
Hello!

Time to react

components.js

```
class App extends React.Component {
  render() {
    return <h2>Welcome to Stopa's blog</h2>;
  }
}
```

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template('Hello!'));
});
```

server.js

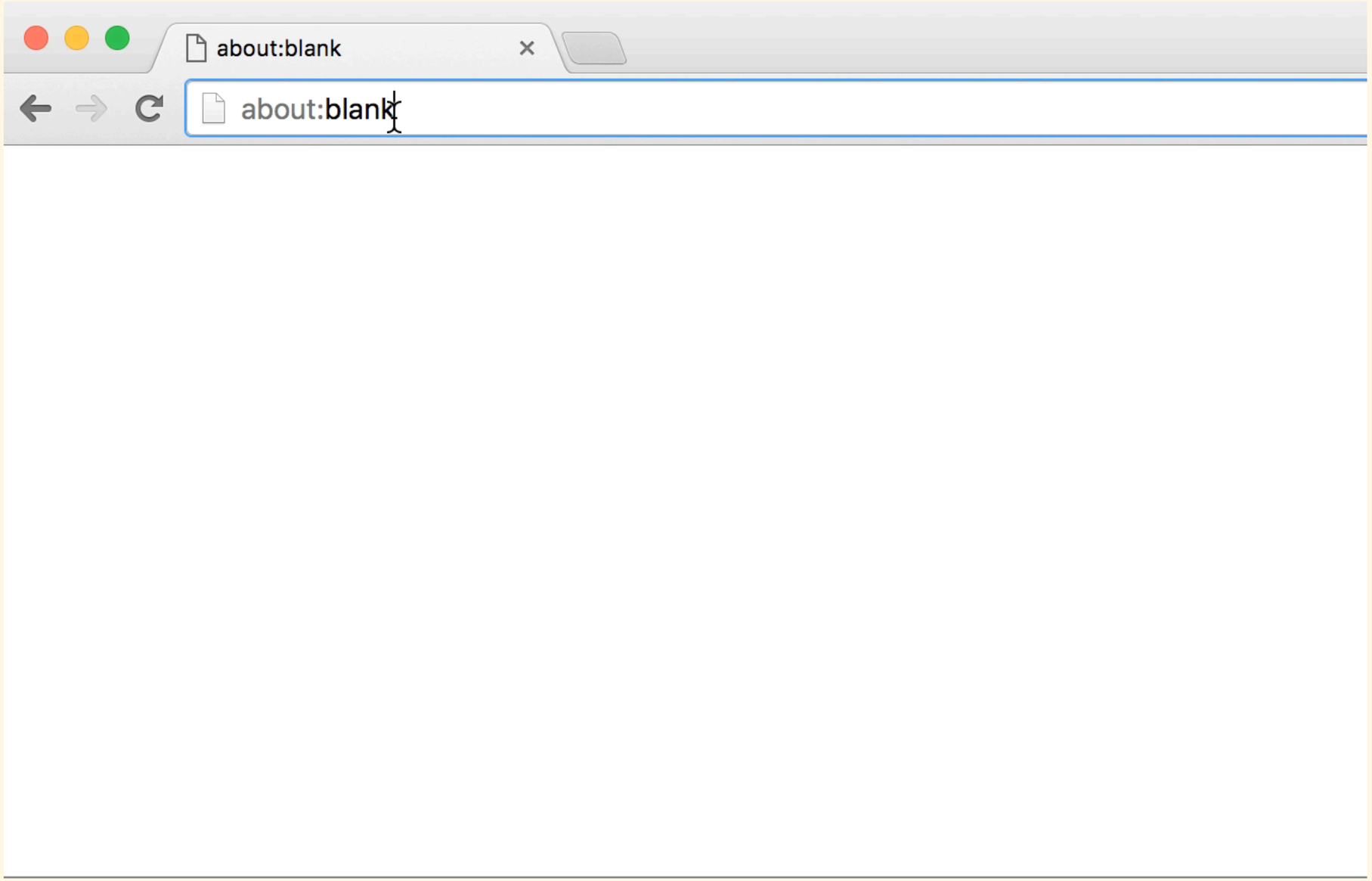
```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template('Hello!'));
});
```

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template(
    renderToString(<App />),
  ));
});
```

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template(
    renderToString(<App />),
  ));
});
```


about:blank
about:blank

components.js

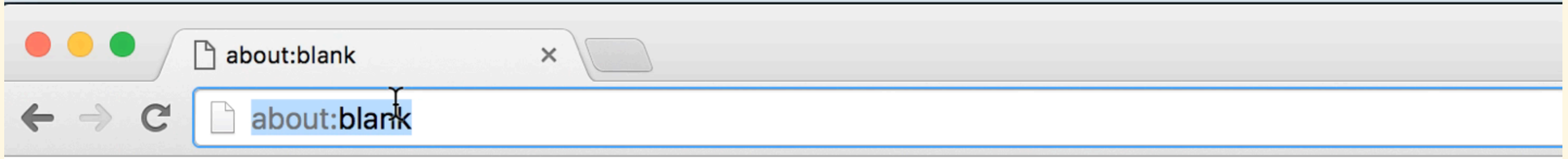
```
class App extends React.Component {
  render() {
    return <h2>Welcome to Stopa's blog</h2>;
  }
}
```

components.js

```
class App extends React.Component {
  render() {
    return (
      <h2 onClick={() => alert('hello!')}>
        Welcome to Stopa's blog
      </h2>
    );
  }
}
```

components.js

```
class App extends React.Component {
  render() {
    return (
      <h2 onClick={() => alert('hello!')}>
        Welcome to Stopa's blog
      </h2>
    );
  }
}
```


about:blank
about:blank

server.js

```
function template(body) {
  return `
    <!doctype html>
    <html>
      <head>
        <title>Stopa's musing</title>
      </head>
      <body>
        ${body}
      </body>
    </html>
  `
}
```

server.js

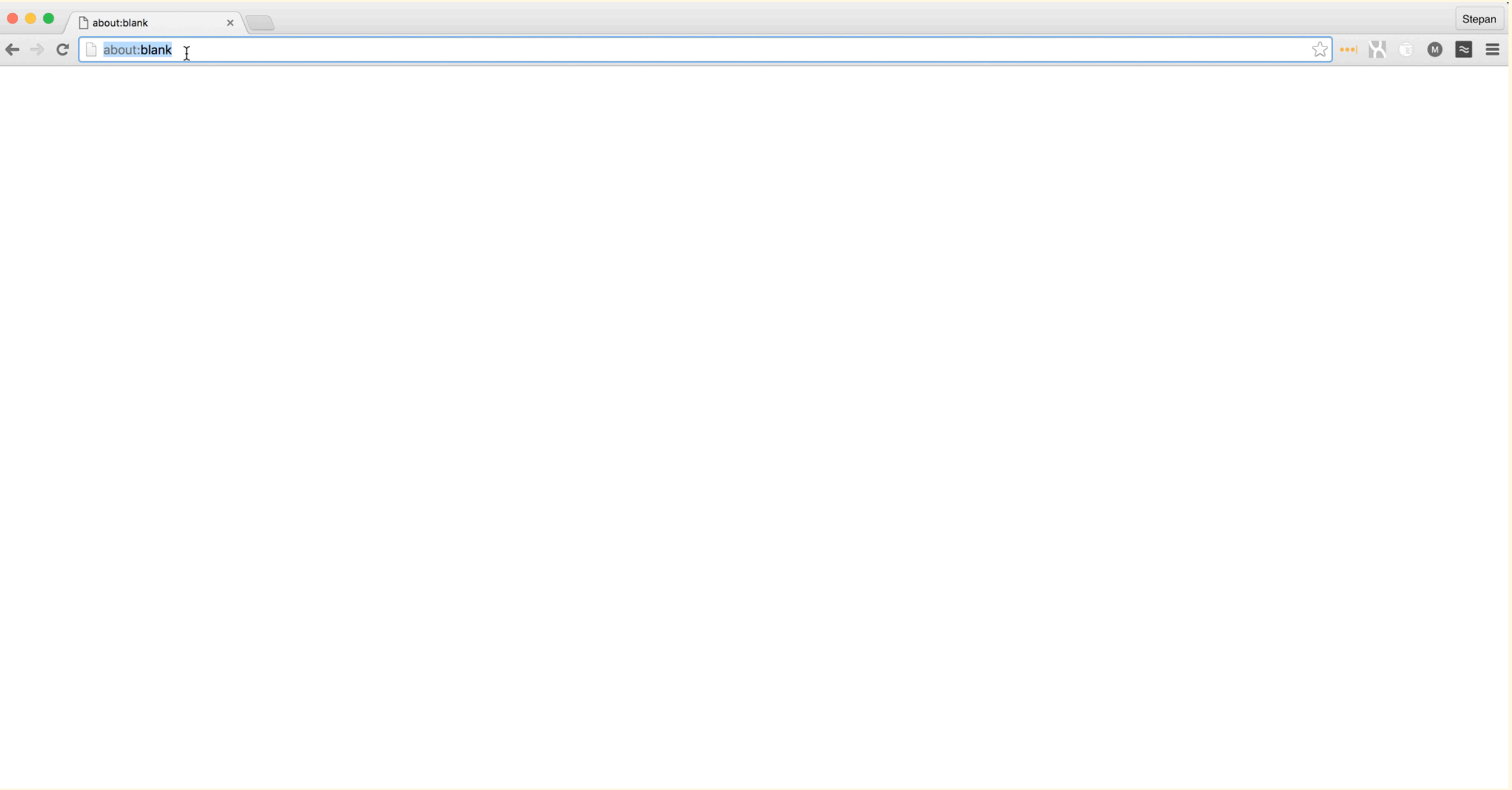
```
function template(body) {
  return `
    <!doctype html>
    <html>
      <head>
        <title>Stopa's musing</title>
      </head>
      <body>
        <div id="root">${body}</div>
        <script src="client.js"></script>
      </body>
    </html>
  `;
}
```

server.js

```
function template(body) {
  return `
    <!doctype html>
    <html>
      <head>
        <title>Stopa's musing</title>
      </head>
      <body>
        <div id="root">${body}</div>
        <script src="client.js"></script>
      </body>
    </html>
  `;
}
```

client.js

```
import React from 'react';
import ReactDOM from 'react-dom';

ReactDOM.render(
  <App />,
  document.getElementById('root'),
);
```


about:blank
Stepan
about:blank

...Now the views are shared

What's the catch?

build tools

(webpack, babel & family)

Time to route

client.js

```
const path = window.location.pathname;
```

ReactRouter

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template(
    renderToString(<App />),
  ));
});
```

let's disable universal rendering for now

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template(
    renderToString(<App />),
  ));
});
```

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template(''));
});
```

components.js

```
class App extends React.Component {
  render() {
    return (
      <h2 onClick={() => alert('hello!')}>
        Welcome to Stopa's blog
      </h2>
    );
  }
}
```

components.js

```
class App extends React.Component {
  render() {
    return (
      <div>
        <Header />
        {this.props.children}
      </div>
    );
  }
}

class Header extends React.Component {...}
class About extends React.Component {...}
class Contact extends React.Component {...}
```

routes.js

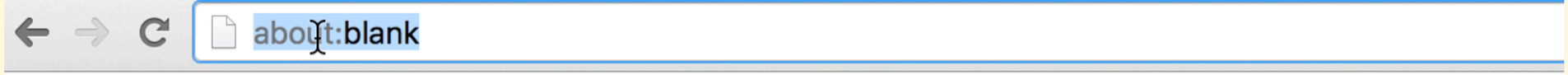
```
import {Route, IndexRoute} from 'react-router';
import React from 'react';

import {App, About, Contact} from './components';

export default (
  <Route path="/" component={App}>
    <IndexRoute component={About} />
    <Route path="contact" component={Contact} />
  </Route>
);
```

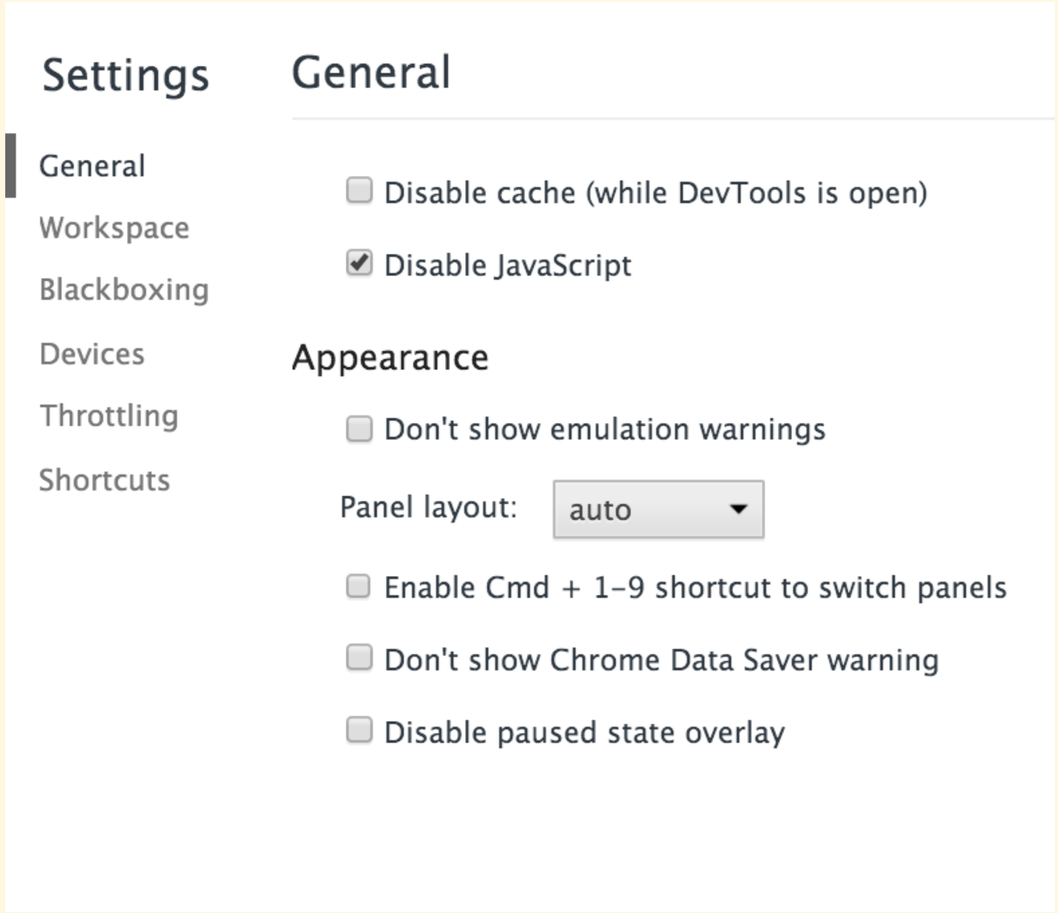
client.js

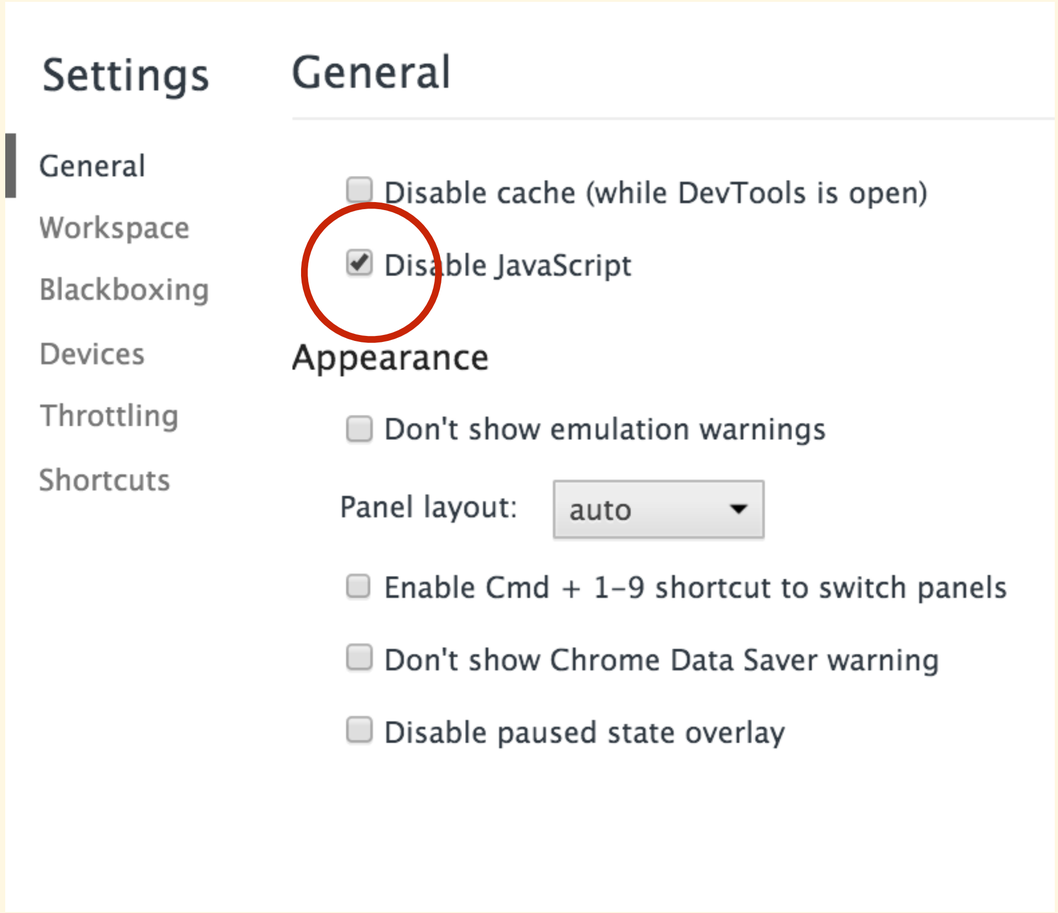
```
import React from 'react';
import ReactDOM from 'react-dom';

ReactDOM.render(
  <App />,
  document.getElementById('root'),
);
```

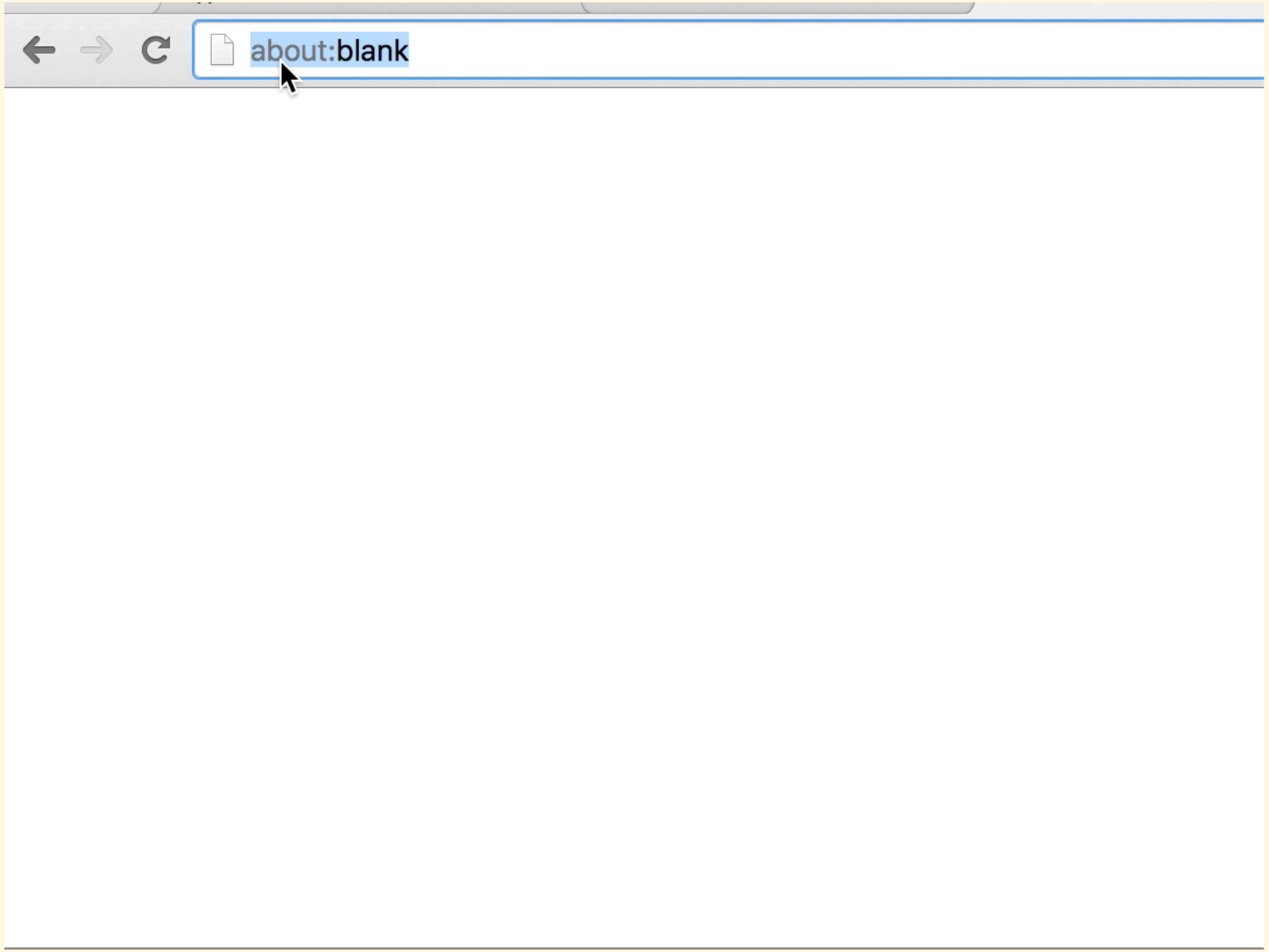
client.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Router, browserHistory} from 'react-router';

ReactDOM.render(
  <Router history={browserHistory}>
    {routes}
  </Router>,
  document.getElementById('root'),
);
```


about:blank


Settings
General
Workspace
Blackboxing
Devices
Throttling
Shortcuts
General
Disable cache (while DevTools is open)
Disable JavaScript
Appearance
Don't show emulation warnings
Panel layout:
auto
Enable Cmd + 1-9 shortcut to switch panels
Don't show Chrome Data Saver warning
Disable paused state overlay


Settings
General
Workspace
Blackboxing
Devices
Throttling
Shortcuts
General
Disable cache (while DevTools is open)
Disable JavaScript
Appearance
Don't show emulation warnings
Panel layout:
auto
Enable Cmd + 1-9 shortcut to switch panels
Don't show Chrome Data Saver warning
Disable paused state overlay


about:blank

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';

const app = express();

app.get('/:params?*', function(req, res) {
  return res.status(200).send(template(''));
});
```

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';
import routes from './routes';
import {match, RouterContext} from 'react-router';

const app = express();

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    res.status(200).send(template(
      renderToString(<RouterContext {...props} />),
    ));
  });
});
```

server.js

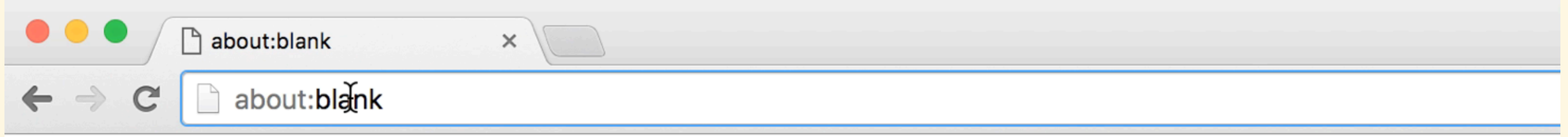
```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';
import routes from './routes';
import {match, RouterContext} from 'react-router';

const app = express();

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    res.status(200).send(template(
      renderToString(<RouterContext {...props} />),
    ));
  });
});
```

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';
import routes from './routes';
import {match, RouterContext} from 'react-router';

const app = express();

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    res.status(200).send(template(
      renderToString(<RouterContext {...props} />),
    ));
  });
});
```

server.js

```
import {renderToString} from 'react-dom/server';
import express from 'express';
import React from 'react';
import routes from './routes';
import {match, RouterContext} from 'react-router';

const app = express();

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    res.status(200).send(template(
      renderToString(<RouterContext {...props} />),
    ));
  });
});
```


about:blank
about:blank

How about stores?

client.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Router, browserHistory} from 'react-router';

ReactDOM.render(
  <Router history={browserHistory}>{routes}</Router>,
  document.getElementById('root'),
);
```

client.js

```
import createStore from './create-store';
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Provider} from 'react-redux';
import {Router, browserHistory} from 'react-router';

ReactDOM.render(
  <Router history={browserHistory}>{routes}</Router>,
  document.getElementById('root'),
);
```

client.js

```
import createStore from './create-store';
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Provider} from 'react-redux';
import {Router, browserHistory} from 'react-router';

ReactDOM.render(
  <Router history={browserHistory}>{routes}</Router>,
  document.getElementById('root'),
);
```

client.js

```
import createStore from './create-store';
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Provider} from 'react-redux';
import {Router, browserHistory} from 'react-router';

const store = createStore();

ReactDOM.render(
  <Router history={browserHistory}>{routes}</Router>,
  document.getElementById('root'),
);
```

client.js

```
import createStore from './create-store';
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Provider} from 'react-redux';
import {Router, browserHistory} from 'react-router';

const store = createStore();

ReactDOM.render(
  <Router history={browserHistory}>{routes}</Router>,
  document.getElementById('root'),
);
```

client.js

```
import createStore from './create-store';
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Provider} from 'react-redux';
import {Router, browserHistory} from 'react-router';

const store = createStore();

ReactDOM.render(
  <Provider store={store}>
    <Router history={browserHistory}>{routes}</Router>,
  </Provider>
  document.getElementById('root'),
);
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    res.status(200).send(template(
      renderToString(<RouterContext {...props} />)
    ));
  });
});
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    const store = createStore();
    res.status(200).send(template(
      renderToString(
        <Provider store={store}
          <RouterContext {...props} />
        </Provider>
      ),
    ));
  });
});
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    const store = createStore();
    res.status(200).send(template(
      renderToString(
        <Provider store={store}
          <RouterContext {...props} />
        </Provider>
      ),
    ));
  });
});
```

initial data 呢?

client.js

```
import createStore from './create-store';
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Provider} from 'react-redux';
import {Router, browserHistory} from 'react-router';

const store = createStore();

ReactDOM.render(
  <Provider store={store}>
    <Router history={browserHistory}>{routes}</Router>,
  </Provider>
  document.getElementById('root'),
);
```

client.js

```
import createStore from './create-store';
import React from 'react';
import ReactDOM from 'react-dom';
import routes from './routes';
import {Provider} from 'react-redux';
import {Router, browserHistory} from 'react-router';

const store = createStore(window.__DATA__);

ReactDOM.render(
  <Provider store={store}>
    <Router history={browserHistory}>{routes}</Router>,
  </Provider>
  document.getElementById('root'),
);
```

server.js

```
function template(body) {
  return `
    <!doctype html>
    <html>
      <head>
        <title>Stopa's musing</title>
      </head>
      <body>
        <div id="root">${body}</div>
        <script src="client.js"></script>
      </body>
    </html>
  `
}
```

server.js

```
function template(body, data) {
  return `
    <!doctype html>
    <html>
      <head>
        <title>Stopa's musing</title>
      </head>
      <body>
        <div id="root">${body}</div>
        <script>window.__DATA__ = ${JSON.stringify(data)}</script>
        <script src="client.js"></script>
      </body>
    </html>
  `
}
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    fetchAllData(props).then(data => {
      const store = createStore();
      res.status(200).send(template(
        renderToString(
          <Provider store={store}
            <RouterContext {...props} />
          </Provider>
        ),
      ));
    });
  });
});
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    fetchAllData(props).then(data => {
      const store = createStore(data);
      res.status(200).send(template(
        renderToString(
          <Provider store={store}
            <RouterContext {...props} />
          </Provider>
        ),
      ));
    });
  });
});
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    fetchAllData(props).then(data => {
      const store = createStore(data);
      res.status(200).send(template(
        renderToString(
          <Provider store={store}
            <RouterContext {...props} />
          </Provider>
        ),
        data,
      ));
    });
  });
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    fetchAllData(props).then(data => {
      const store = createStore(data);
      res.status(200).send(template(
        renderToString(
          <Provider store={store}
            <RouterContext {...props} />
          </Provider>
        ),
        data,
      ));
    });
  });
```

ok, ok....可是...

How about data fetching?

api.js

```
function getPosts() {
  return $.ajax.get(`${API_BASE}/posts`);
}
```

api.js

```
import fetch from 'isomorphic-fetch';

function getPosts() {
  return fetch(`${API_BASE}/posts`);
}
```

components.js

```
class PostsIndex extends React.Component {
  render() {
    return this.props.posts.map(
      post => <Post key={post.id} post={post} />
    );
  },
}
```

components.js

```
class PostsIndex extends React.Component {
  fetchData({page}) {
    return api.getPosts(page);
  }

  render() {
    return this.props.posts.map(
      post => <Post key={post.id} post={post} />
    );
  },
}
```

server.js

```
...

app.get('/:params?*', function(req, res) {
  match({routes, location: req.url}, (err, redirect, props) => {
    fetchAllData(props).then(data => {
      const store = createStore(data);
      res.status(200).send(template(
        renderToString(
          <Provider store={store}
            <RouterContext {...props} />
          </Provider>
        ),
        data,
      ));
    });
  });
```

server.js

```
async function fetchAllData(props) {
  return Promise.all(
    props
      .components
      .filter(x => x.fetchData)
      .map(x => x.fetchData(props))
  );
}
```

server.js

```
async function fetchAllData(props) {
  return Promise.all(
    props
      .components
      .filter(x => x.fetchData)
      .map(x => x.fetchData(props))
  );
}
```

server.js

```
async function fetchAllData(props) {
  return Promise.all(
    props
      .components
      .filter(x => x.fetchData)
      .map(x => x.fetchData(props))
  );
}
```

/home -> /page/2 ?

client.js

```
...

ReactDOM.render(
  <Provider store={store}>
    <Router
      history={browserHistory}>
      {routes}
    </Router>,
  </Provider>
  document.getElementById('root'),
);
```

client.js

```
...

ReactDOM.render(
  <Provider store={store}>
    <Router
      createElement={createElement}
      history={browserHistory}>
      {routes}
    </Router>,
  </Provider>
  document.getElementById('root'),
);
```

client.js

```
function createElement(Component, props) {
  if (Component.fetchData) {
    Component.fetchData(props);
  }
  return <Component {...props} />;
}
```

...but there's a library for that

react-resolver

is it all rosy?

not really

but the payoff?


RAILS


JS


RAILS


JS


JS

JS

JS

JS




routing
validation
views

wait...
single-threaded-backend?

ClojureScript.

ClojureScript. Boom

so..

Build Declaratively

Question assumptions

Questions?