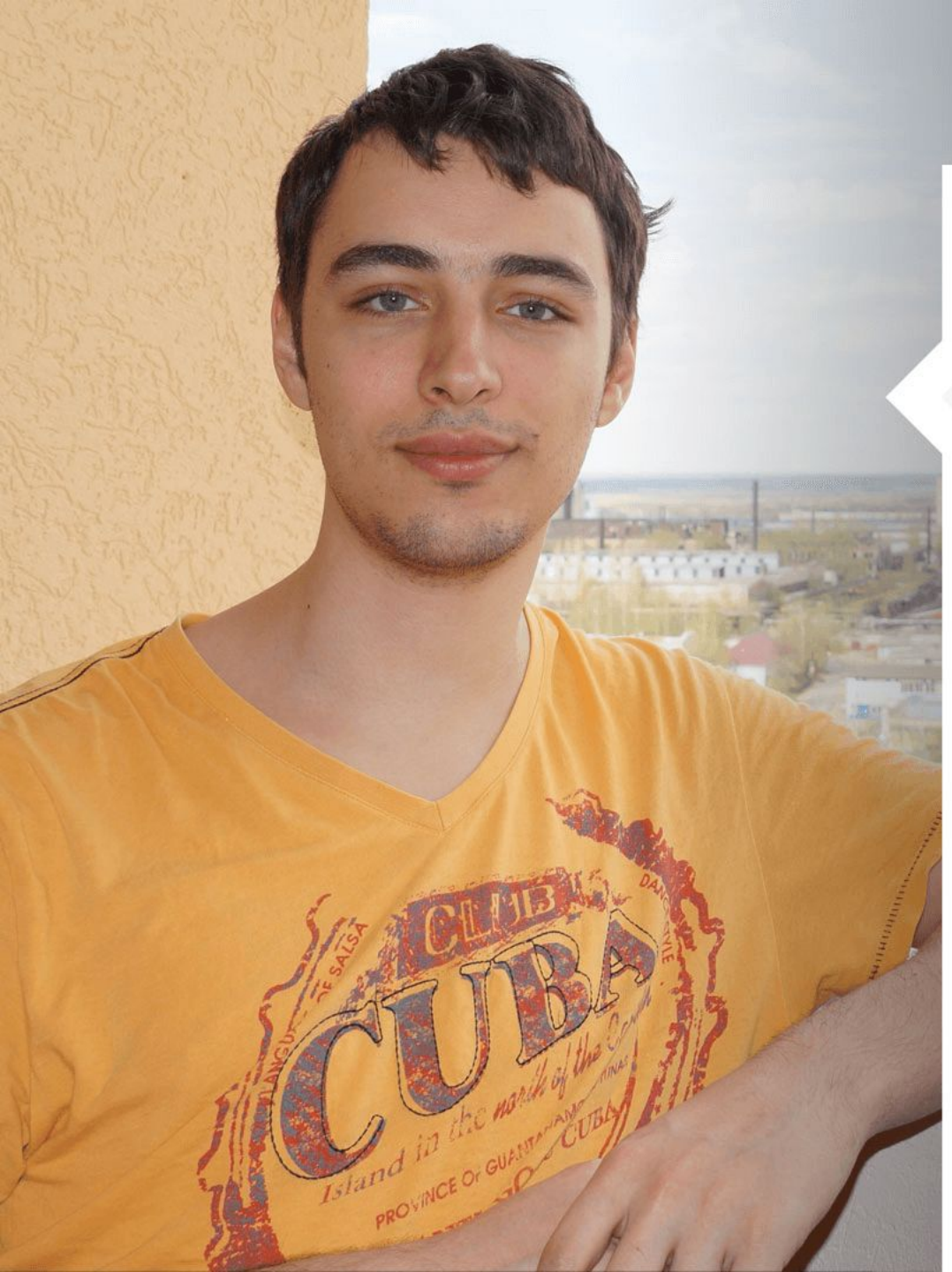

Внутри webpack

FRONTEND

Егор Баранов
руководитель направления
JavaScript-разработки

ВЕЩАЕТ

Егор Баранов

Инфраструктура фронтенда. API Blueprint.

Специалист по нагрузочному тестированию вентиляторов. Эксперт по проверке наличия бекапов production БД.

СВЯЗАТЬСЯ СО МНОЙ

@e.baranov

egor-baranov

Что такое webpack?

- Большой проект с открытым исходным кодом.
- Сборщик большинства фронтенд-проектов в компании.
- Расширяемая платформа.

Организационные решения

Минимум бюрократии

- Линтер решает, хорош код или нет.
- История коммитов – средство, а не цель.

Автоматизация

- CI.
- Тесты.

Большое сообщество

- StackOverflow.
- Gitter.
- Документация.

Технические решения

Технические решения

Контекст для Loader-ов

```
require('./index.html');     // html-loader
require('./styles.css');     // css-loader
require('./template.pug');   // pug-loader
```

```
module.exports = function(content) {
  return `module.exports = "${transformContent(content)}";`;
}
```

```
module.exports = function(content) {
  const callback = this.async();
  someAsyncOperation(content, function(err, result) {
    if (err) return callback(err);
    callback(null, result);
  });
};
```

Технические решения

Как передать данные в функцию?

Параметры

```
function sum(a, b) {
  return a + b;
}

console.log(sum(5, 10));
```

Глобальные переменные

```
let a = 5;
let b = 10;
function sum() {
  return a + b;
}

console.log(sum());
```

React Hooks

```
import React, { useState } from 'react';

function Counter() {
  const [count, setCount] = useState(0);
  return (
    <div>
      <p>You clicked {count} times</p>
        <button onClick={() => setCount(count + 1)}>
          Click me
        </button>
    </div>
  );
}
```

Технические решения

Контекст This

```
function handleClick() {
  console.log(this.innerText);
}

document.getElementById('btn1').onclick = handleClick;
document.getElementById('btn2').onclick = handleClick;
```

```
function User(name, position) {
  this.name = name;
  this.position = position;
}

const user = new User('Egor', 'developer');
```

```
$("li").each(function(index) {
  console.log(index + ": " + $(this).text());
});
```

```
function sum() {
  return this.a + this.b;
}

const context = {
  a: 5,
  b: 10
};
console.log(sum.apply(context));
```

```
context.async = function async() { // здесь фомируется контекст this для loader
  if(isDone) {
    if(reportedError) return; // ignore
    throw new Error("async(): The callback was already called.");
  }
  isSync = false;
  return innerCallback;
};

// ...

var result = (function LOADER_EXECUTION() {
  return fn.apply(context, args); // fn - вызываемый loader
}());
```

Технические решения

Tapable

Tapable

- Разделение логики на основную и дополнительную.
- Необходимость добавления дополнительной логики из плагинов.

```
compiler.hooks.emit.tapAsync('HtmlWebpackPlugin',
  (compilation, callback) => {
    // ...
  }
)
```

```
class RequestProcessor {
  constructor() {
    this.hooks = { processRequest: new SyncHook("request") };
  }

  async processRequests() {
    while (true) {
      const request = await this.getRequest();
      logRequest(request);

      this.hooks.processRequest.call(request);
    }
  }
}
```

```
const requestProcessor = new RequestProcessor();

requestProcessor.hooks.processRequest.tap("email", (request) => {
  // send email
});

requestProcessor.hooks.processRequest.tap("db", (request) => {
  // save info to db
});

requestProcessor.processRequests();
```

Технические решения

~~Закат солнца~~

Рекурсия вручную

```
const result = sum(1, 2);
console.log(result);

function sum(a, b) {
  return a + b;
}
```

```
const result = sum(1, 2); // <-- сохраняем адрес возврата
console.log(result);

function sum(a, b) {
  return a + b;
}
```

```
const result = sum(1, 2);
console.log(result);

function sum(a, b) {
  return a + b; // <-- выполняем функцию
}
```

```
const result = sum(1, 2); // <-- возвращаемся обратно
console.log(result);

function sum(a, b) {
  return a + b;
}
```

Call Stack

baz (/tmp/test.js:12:15)

bar (/tmp/test.js:8:3)

foo (/tmp/test.js:4:3)

Object. (/Users/ebaranov/test.js:1:63)

Module._compile (internal/modules/cjs/loader.js:689:30)

Module.load (internal/modules/cjs/loader.js:599:32)

tryModuleLoad (internal/modules/cjs/loader.js:538:12)

Function.Module._load (internal/modules/cjs/loader.js:530:3)

Function.Module.runMain (internal/modules/cjs/loader.js:742:12)

Размер Call Stack ограничен

```
let i = 0;

setTimeout(function() {
  console.log(i);
}, 0);

function inc() {
  i++;
  inc();
}
inc();
```

Технические решения

Рекурсия

```
const graph = {
  value: 10,
  children: [
    {
      value: 5,
      children: [
        {
          value: 15,
          children: []
        }
      ]
    }
  ]
};
```

```
function findMax(node) {
  let maxValue = node.value;

  findMaxIter(node);
  return maxValue;

  function findMaxIter(node) {
    maxValue = Math.max(maxValue, node.value);
    node.children.forEach(child => {
      findMaxIter(child);
    });
  }
}
```

```
function findMax(node) {
  let maxValue = node.value;
  const queue = [node];

  while (queue.length) {
    const queueItem = queue.pop();

    maxValue = Math.max(maxValue, queueItem.value);

    queueItem.children.forEach(child => queue.push(child));
  }

  return maxValue;
}
```

Полезные ссылки

webpack.js.org

github.com/webpack/webpack

0e39bf7b.io

Спасибо
за внимание!