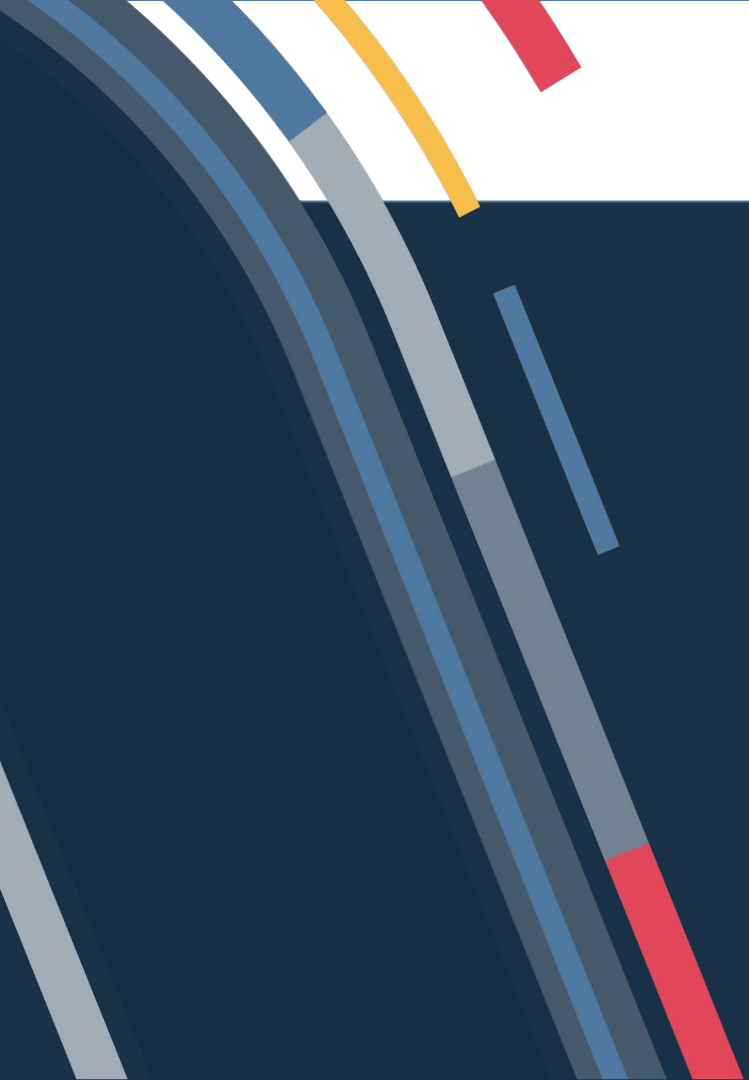

Front-end development course

JavaScript

Lecture #7

Node.js, npm and modern JS toolbox

Node.js

Node.js - is an interpreter and runtime for JavaScript language written in C++, so it takes your JavaScript code, interprets and runs it.

Node can run arbitrary script or work in REPL.

npm

npm - is a package manager for node.js started in 2009, now it is default package manager for node and it comes as a part of node.

Package - is just an archive (zip or tar) that contains JavaScript code.

npm has a concept of registries - private or public place where you can store your packages.

npm packages can be installed locally or globally. Packages use release approach called Semantic Versioning

npm can store all dependencies of your project (to make build reproducible) in project descriptor file package.json

There are some alternatives to npm:

- yarn
- pnpm

npm

List of common npm commands:

- npm init https://docs.npmjs.com/cli/install
- npm install (-g) https://docs.npmjs.com/cli/install
- npm update https://docs.npmjs.com/cli/update
- npm run (some_script) https://docs.npmjs.com/misc/scripts
- npm publish

Automation with npm scripts article

To understand what is a package and what is a module see an article https://docs.npmjs.com/getting-started/packages

Modules in details

Using modules is essential in front-end engineering nowadays. It gives you many benefits like: easier maintainability, namespacing and reusability.

Native ES6 module creates immutable binding when imported.

Native ES6 modules loads asynchronously in browser and creates a scope, so variables inside module doesn't become global.

CommonJS and ES6 modules comparison: http://jsmodules.io/cjs.html

Module bundlers

Module bundling is simply the process of stitching together (concatenating) a group of modules (and their dependencies) into a single file (or group of files) in the correct order.

Module bundling have many advantages: less transport overhead for clients and more abilities to cache and zipping.

Bundling can involve many step apart from just concatenating files: transpiling code, minification, uglification (UglifyJS)

Most popular module bundlers:

- webpack
- rollup
- brunch
- parcel

Webpack

Webpack is relatively new tool, it was designed to be agnostic to the module system you use, allowing developers to use CommonJS, AMD, or ES6 as appropriate.

Webpack supports: code splitting, tree shaking, dead code elimination

Webpack core concepts:

- Entry
- Output
- Loader
- Plugins

Transpilers, Babel

Babel (originally named 5to6) - is a tool to transpile your next version JS to ES5

Babel itself doesn’t do anything, it just code to code transpiler.

Babel is very pluggable and support a lot of syntaxes and works well with webpack.

Source maps

The JavaScript sources executed by the browser are often transformed in some way from the original sources created by a developer.

For example:

- sources are often combined and minified to make delivering them from the server more efficient.
- JavaScript running in a page is often machine-generated, as when compiled from a language like CoffeeScript or TypeScript.

In these situations, it's much easier to debug the original source, rather than the source in the transformed state that the browser has downloaded. A source map is a file that maps from the transformed source to the original source, enabling the browser to reconstruct the original source and present the reconstructed original in the debugger. See full article .

Q&A

Thank You