Javascript tools and workflow

POWERED BY:

Tools used in modern Javascript development workflow

- Source control tool - git
- Package manager - npm, yarn
- Code transpiler - babel, typescript
- Task runners and code bundlers - grunt, webpack

Source control: git

→ source control gives ability to save and manage changes of your code over time

→ it enables creation of alternate copies of work (branches) on which you can work without affection other branches

→ it allows to check any previous state of working tree on any branch

git: commands

- git init - initialize git repository
  - git init directory-name - initialize git repository in sub directory
- git add - staged files for commit
  - git add . - stage all changed files
  - git add file - name - stage one file
- git commit - commit added files

POWERED BY: nsoft sportradar

SPARK. SCHOOL

Zadatak

- Initialize repository “js-tools-and-workflow”

Package manger: npm

- npm is package (module) manager
- packages can contain code that can be used in your projects or command line tools
- it allows installation of packages locally in project or globally
- it has its own package registry registry.npmjs.org from which packages can be installed

POWERED BY: nsoft sportradar

SPARK. SCHOOL

npm - commands

→ npm init - creates new package.json

→ npm install - installs new package to project or install all project packages
  - --save
  - --save-dev

→ npm update - updates packages to newer version respecting npm semver

→ npm run - runs on of the scripts from scripts field in package.json

→ npm start - runs “start” script from package.json

POWERED BY: nsoft sportradar

SPARK. SCHOOL

Zadatak

- Initialize new package.json in “js-tools-and-workflow” folder
- add it and commit it with git

Code transpiler: babel

- babel enables conversion of ES2015+ code into code compatible with our desired runtime environment
- babel uses modular presets and plugins that are installed separately as needed

babel - installation to project BABEL

→ install it using npm
  - npm i @babel/cli @babel/core

→ @babel/cli - cli tool that can be used for direct transpilation of code but more often @babel/core is used as part of some another build tool

→ @babel/cli usage
  - babel `source file or directory` -d `destination file or directory`

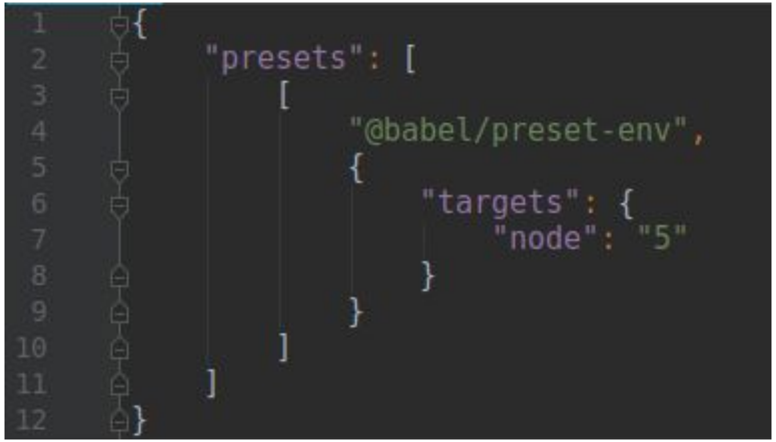
→ install desired preset
  - in our case it will be @babel/preset-env which will enable traspilation of code for desired environment

POWERED BY: nsoft sportradar

SPARK. SCHOOL

Example

BABEL

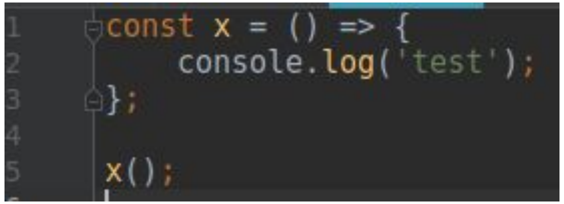
- install @babel/cli @babel/core @babel/preset-env as dev dependencies
- create .babelrc file
- add @babel/preset-env as presets with node target
- 

```
{
    "presets": [
        [
            "@babel/preset-env",
            {
                "targets": {
                    "node": "5"
                }
            }
        ]
    ]
}
```

POWERED BY:

Example

→ create index.js file



```
const x = () => {
      console.log('test');
};

x();
```

→ run babel index.js -d dist

→ check code in dist/index.js file

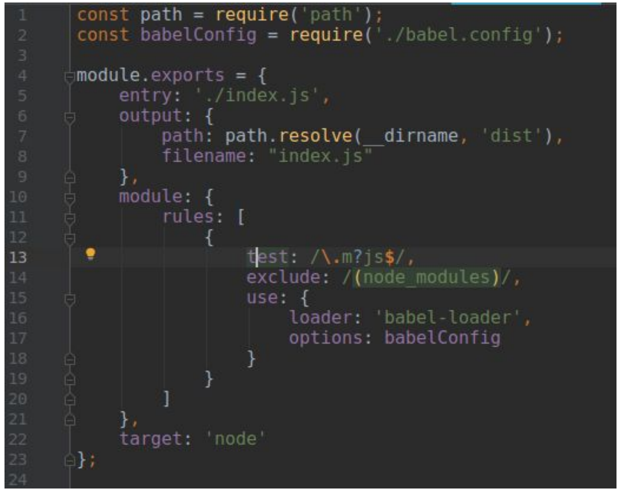
Code bundler: webpack

- webpack is module bundler for modern backend and frontend javascript applications
- it doesn’t require configuration file by default but it’s almost always used with configuration file

POWERED BY:

Webpack - configuration

- basic configuration consists of - entry file/s that will be used for bundling other files
- output file/s - bundled files

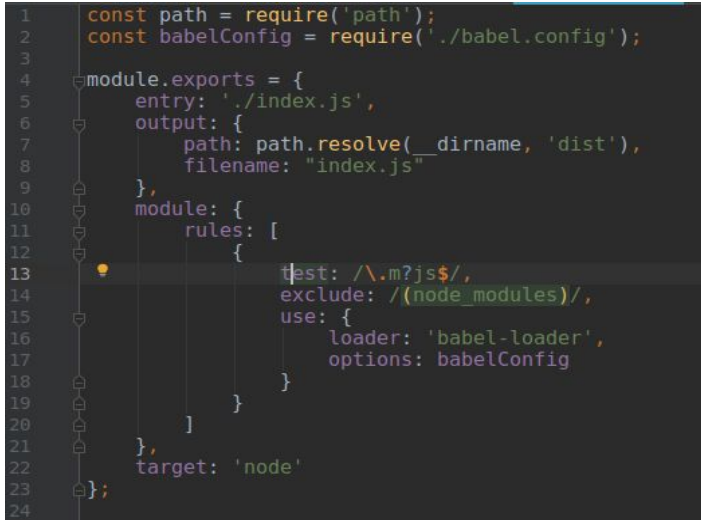


```
const path = require('path');
const babelConfig = require('./babel.config');

module.exports = {
    entry: './index.js',
    output: {
        path: path.resolve(__dirname, 'dist'),
        filename: "index.js"
    },
    module: {
        rules: [
            {
                test: /\.m?js$/,
                exclude: /(node_modules)/,
                use: {
                    loader: 'babel-loader',
                    options: babelConfig
                }
            }
        ]
    },
    target: 'node'
};
```

Example

- → install webpack and webpack-cli with npm as dev dependency
- → create webpack config file - webpack.config.js
- → add “build”: “webpack” script to “package.json”



```
const path = require('path');
const babelConfig = require('./babel.config');

module.exports = {
    entry: './index.js',
    output: {
        path: path.resolve(__dirname, 'dist'),
        filename: "index.js"
    },
    module: {
        rules: [
            {
                test: /\.m?js$/,
                exclude: /(node_modules)/,
                use: {
                    loader: 'babel-loader',
                    options: babelConfig
                }
            }
        ]
    },
    target: 'node'
};
```

POWERED BY: nsoft sportradar

SPARK. SCHOOL

Any questions ?

POWERED BY: nsoft sportradar

SPARK. SCHOOL

THANK YOU

for attention