Webpack to tzw. module bundler, który pozwala nam tworzyć kod podzielony na wiele modułów, które później Webpack łączy.

Uruchamianie webpack

**webpack –config webpack.config.js**

Ważne jest określenie tzw. entry point, dzięki czemu Webpack wie, który plik jest głównym i gdzie powinny zacząć się zależności.

w folderze projektu stworzyć plik package.json:

npm init

dodać Webpack do pliku zależności package.json:

npm install webpack -- save-dev

stworzyć plik konfiguracyjny (webpack.config.js), dla przykładu:

module.exports = { entry: './file.js', output: { path: \_\_dirname, filename: 'bundle.js' } };

Plik main.js powinien być punktem wejściowym systemu modułów AMD lub CommonJS. Istnieje też możliwość skorzystania z ES6, na przykład za pomocą odpowiedniego “loadera” Babel’a - wtedy system modułów ES6 również będzie obsługiwany. Na temat “loaderów” napiszę za moment.

webpack allows you to write your modules in CommonJS (or another) format and bundle them together for production use. It also allows you to easily use modules published on NPM.

import greeter = require('./greeter');

import $ = require('jquery');

$(() => {

$(document.body).html(greeter("World"));

});

TSCONFIG.JSON

{

"compilerOptions": {

"module": "commonjs"

},

"files": [

"app.ts",

"typings/tsd.d.ts"

]

}

First we need to install webpack:

**npm install -g webpack**

Next we need the TypeScript loader which webpack will use to understand TypeScript:

**npm install ts-loader --save**

webpack.config.js

module.exports = {

entry: './app.ts',

output: {

filename: 'bundle.js'

},

resolve: {

extensions: ['', '.webpack.js', '.web.js', '.ts', '.js']

},

module: {

loaders: [

{ test: /\.ts$/, loader: 'ts-loader' }

]

}

}

Lastly we've told webpack to use ts-loader for all files that have the .ts extension. This is the crucial part that enables webpack to understand and bundle TypeScript files. ts-loader will respect the settings we've already defined in our tsconfig.json file so webpack, the IDE, and the tsc command line all use the same settings without duplication.

npm install jquery --save

webpack

First let's modify our webpack.config.js to add minification and turn on sourcemaps.

var webpack = require('webpack');

module.exports = {

entry: './app.ts',

output: {

filename: 'bundle.js'

},

// Turn on sourcemaps

devtool: 'source-map',

resolve: {

extensions: ['', '.webpack.js', '.web.js', '.ts', '.js']

},

// Add minification

plugins: [

new webpack.optimize.UglifyJsPlugin()

],

module: {

loaders: [

{ test: /\.ts$/, loader: 'ts' }

]

}

}

Next we need to tell TypeScript to generate sourcemaps. We do that by modifying tsconfig.json.

{

"compilerOptions": {

"module": "commonjs",

"sourceMap": true

},

"files": [

...

]

}