# Build Tools

For every Build Tool create a separate folder and run “**npm init**” in it.

## Grunt

Install Grunt **CLI**:

**npm install -g grunt-cli**

Then the **Grunt**:

**npm install grunt --save-dev**

Then install all grunt **plugins** you are going use:

After that create a “**Gruntfile.js**” in the root of your directory (where is “**package.json**” file). There will be

**npm install grunt-contrib-concat --save-dev**

**npm install grunt-contrib-uglify --save-dev**

**npm install grunt-eslint --save-dev**

module.exports = (grunt) => {  
 grunt.initConfig({  
 **pkg**: grunt.**file**.readJSON(**'package.json'**),**concat**: {  
 **build**: {  
 **src**: [**'src/jquery-3.1.1.js'**, **'src/main.js'**,],  
 **dest**: **'build/bundle.js'**,  
 },  
 },  
 **uglify**: {  
 **build**: {  
 **src**: [**'build/bundle.js'**],  
 **dest**: **'build/bundle.min.js'** }  
 },

**eslint**: {  
 **options**: {  
 **configFile**: **'.eslintrc.json'**,  
 },  
 **target**: [**'src/main.js'**]  
 }  
 });

grunt.loadNpmTasks(**'grunt-contrib-concat'**);  
 grunt.loadNpmTasks(**'grunt-contrib-uglify'**);

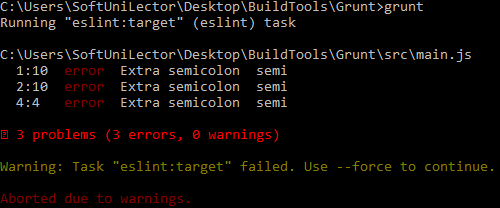
grunt.loadNpmTasks(**'grunt-eslint'**);

some configuration in it:

After that in your project create **two** **folders**: “**src**” – main source of pure scripts and “**build**” – main destination for minified/uglified scripts.

In the “**src**” – create several scripts, which we can later on concatenate and minify later on. In the “build” do nothing. After that again in the root of your project create new "**.eslintrc.json**". It should contain:

{  
 **"env"**: {  
 **"browser"**: **true**,  
 **"es6"**: **true**,  
 **"node"**: **true** },  
 **"extends"**: **"eslint:recommended"**,  
 **"parserOptions"**: {  
 **"ecmaFeatures"**: {  
 **"experimentalObjectRestSpread"**: **true**,  
 **"jsx"**: **true** },  
 **"sourceType"**: **"module"** },  
 **"rules"**: {  
 **"indent"**: [  
 **"error"**,  
 **"tab"** ],  
 **"linebreak-style"**: [  
 **"error"**,  
 **"windows"** ],  
 **"quotes"**: [  
 **"error"**,  
 **"single"** ],  
 **"semi"**: [  
 **"error"**,  
 **"never"** ]  
 }  
}

Open new command prompt (if you have closed it) and just type “**grunt**”. Something like this may appear:

This warning message says that there are 3 **errors** in "**src/main.js**" file. The **file** looks like this:

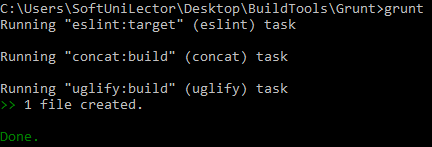
**var *a*** = 3;  
**var *b*** = 9;

***a***+***b***;

After removing all "unecessary" semicolons looks like this:

**var *a*** = 3  
**var *b*** = 9  
***a***+***b***

So the errors are fixed and next time when "**grunt**" is called in command prompt, the following should display:

**Note** that the code you want to minify **must** be in **ES5** because ES6 is not supported by the current build tool. And the order here is important because you want first to concatenate files then to “uglify” the merged file.

## Gulp

Install Gulp **CLI**:

**npm install gulp --save-dev**

**npm install -g gulp-cli**

Then the Gulp:

**npm install gulp-concat --save-dev**

**npm install gulp-uglify --save-dev**

**npm install gulp-eslint --save-dev**

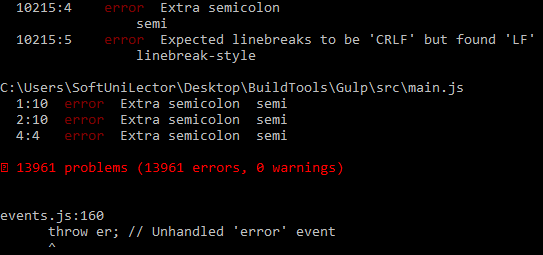
Then install all gulp **plugins** you are going use:

After that create a “**gulpfile.js**” in the root of your directory (where is “**package.json**” file). There will be some configuration in it:

**let** gulp = require(**'gulp'**);  
**let** concat = require(**'gulp-concat'**);  
**let** uglify = require(**'gulp-uglify'**);  
**let** *eslint* = require(**'gulp-eslint'**);  
  
gulp.**task**(**'default'**, () => {  
 **return** gulp.**src**([**'src/jquery-3.1.1.js'**, **'src/main.js'**])  
 .**pipe**(*eslint*())  
 .**pipe**(*eslint*.format()) // To display linting errors  
 .**pipe**(*eslint*.failAfterError()) // To stop if any lint error occurs  
 .**pipe**(concat(**'bundle.min.js'**))  
 .**pipe**(uglify())  
 .**pipe**(gulp.**dest**(**'./build/'**));  
});

After that in your project create **two** **folders**: “**src**” – main source of pure scripts and “**build**” – main destination for minified/uglified scripts.

In the “**src**” – create several scripts, which we can later on concatenate and minify later on. In the “build” do nothing.

Open new command prompt (if you have closed it) and just type “**gulp**”. Something like this may appear:

Because **JQuery** is included again there are a lot of errors. You may create a **separate** task in (**gulpfile.js**) to lint through the code which is not third-party.

**Note** that the code you want to minify **must** be in **ES5** because ES6 is not supported by the current build tool. And the order here is important because you want first to concatenate files then to “uglify” the merged file.

# Lodash

Create a new project (or use one of the previous). Run "**npm init**" (if the project is new). Then run:

**npm i -g npm**

**npm i --save lodash**

Then in your code you have to require it, so you can use the library:

**let** \_ = require(**'lodash'**);  
  
**let** last =\_.*findLast*([1, 2, 3, 4], (n) => {  
 **return** n % 2 === 0  
});  
  
**console**.log(last); *// 4*

# ESLint

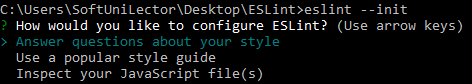
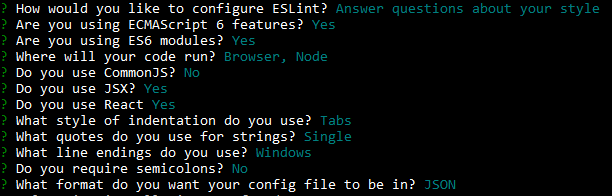
Create a new project (or use one of the previous). Run "**npm init**" (if the project is new). Then run:

**eslint --init**

**npm install -g eslint**

This will install globally **ESLint**. Then we need to initialize a **configuration** file in our current directory:

After that the terminal will ask you about some configuration information (feel free to choose whatever you like):

Note that these rules come from the team you are working with. You should follow the conventions that your team is using as well. Here is how the first option will look like:



After that you should have “**.eslintrc**” or “**.eslintrc.json**” file:

Here are the basics rules. If you want to try some specific rule see [here](http://eslint.org/docs/rules/).

# Electron.js

**npm init**

Take the code from this project. We will use Electron.js to make that web application to desktop. Run "**npm init**":

**npm install electron-prebuilt**

First install electron in root of the folder:

{  
 **"name"**: **"name"**,  
 **"version"**: **"1.0.0"**,  
 **"description"**: **""**,  
 **"main"**: **"index.js"**,  
 **"scripts"**: {  
 **"start"**: **"electron ."**,  
 **"test"**: **"echo \"Error: no test specified\" && exit 1"** },  
 **"author"**: **""**,  
 **"license"**: **"MIT"**,  
 **"devDependencies"**: {  
 **"electron-prebuilt"**: **"^1.4.10"** }  
}

Later on, we add a "**script**" command to our "**package.json**" file. It may look like this:

In the **root** of the project create "**index.js**" file:

**const** {app, BrowserWindow} = require(**'electron'**);  
app.on(**'ready'**, () => {  
 **let** mainWindow = **new** BrowserWindow({  
 **width**: 800,  
 **height**: 600  
 });  
  
 mainWindow.loadURL(**`file://**${\_\_dirname}**/index.html`**);  
  
 mainWindow.on(**'closed'**, () => {  
 mainWindow = **null** })  
});

Go to "**index.html**" file and **fix** the paths to the scripts:

<**head**>  
 <**title**>Prodavachnik</**title**>  
 <**script src="./scripts/jquery-3.1.1.min.js"**></**script**>  
 <**script src="./scripts/prodavachnik.js"**></**script**>  
 <**link rel="stylesheet" type="text/css" href="./styles/prodavachnik.css"** />  
 <**link href="https://fonts.googleapis.com/css?family=Lato|Open+Sans" rel="stylesheet"**>  
</**head**>

Then in "**prodavachnik.js**" on the **top** of the file add the following:

**let** $ = require(\_\_dirname + **'/scripts/jquery-3.1.1.min.js'**)

Your app is ready to go, run "**npm start**":

