Creating project folder and package.json file

Create folder

C:\>md ec6

C:\>cd ec6

C:\ec6>npm init

Creates a package.json file in ec6 folder

Install babel command line tool

|  |  |
| --- | --- |
|  | C:\ec6> npm install --save-dev babel-cli |

Edit package.json file and edit scripts section

"scripts": {

    "build": "babel src -d dist"

}

Create **src** and **dist**  two folders in ec6

In src folder create app.js

var nums = [1,2,3];

var doubleNums = nums.map((e) => e \* 2);

C:\ec6> npm run build

You will see the following output:

|  |  |
| --- | --- |
| 1  2  3  4 | > es6@1.0.0 build D:\projects\es6-demo  > babel src -d dist    src\app.js -> dist\app.js |

Now Babel configuration:

C:\ec6>npm install babel-preset-latest --save-dev

Create a file in ec6 folder with name .babelrc  and add the content

{

  "presets": ["latest"]

}

Now again run the build

C:\ec6>npm run build

Now observe the difference in build folders app.js file

=====================================================

First, you create a directory to store the project files e.g., d:\projects\es6-demo.

Then, you issue the following command to create the package.json file for your project:

|  |  |
| --- | --- |
| 1 | > npm init |

The command line will ask you some basic information, just enter it as follows:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | name: (es6-demo) es6  version: (1.0.0) 1.0.0  description: ES6 demo  entry point: (index.js)  test command:  git repository:  keywords: es6  author: jstut  license: (ISC)  About to write to D:\projects\es6-demo\package.json:    {    "name": "es6",    "version": "1.0.0",    "description": "ES6 demo",    "main": "index.js",    "scripts": {      "test": "echo \"Error: no test specified\" && exit 1"    },    "keywords": [      "es6"    ],    "author": "jstut",    "license": "ISC"  } |

Check the d:\projects\es6-demo\ folder, you will see the new file with the name package.json created.

Installing Babel CLI tool

To install Babel CLI tool, you use the following command:

|  |  |
| --- | --- |
| 1 | > npm install --save-dev babel-cli |

It will install Babel CLI tool locally in your project folder.

Once completed, you will see the new section in the package.json file:

|  |  |
| --- | --- |
| 1  2  3 | "devDependencies": {      "babel-cli": "^6.18.0"  } |

Note that your version may be newer.

Setting up folders for JavaScript codes

For organizing JavaScript files, you create two new folders src and dist. The src folder contains the JavaScript code in the latest version. Babel CLI will take the code from the src folder, convert it to the older version, and save the code files in the dist folder.

Instead of running Babel from the command line directly, you can place commands in the scripts of the package.json section as follows:

|  |  |
| --- | --- |
| 1  2  3  4 | "scripts": {      "build": "babel src -d dist"  } |

Now, you can create a JavaScript file named app.js in the src folder:

|  |  |
| --- | --- |
| 1  2 | var nums = [1,2,3];  var doubleNums = nums.map((e) => e \* 2); |

And issue the following command in the command line:

|  |  |
| --- | --- |
| 1 | > npm run build |

You will see the following output:

|  |  |
| --- | --- |
| 1  2  3  4 | > es6@1.0.0 build D:\projects\es6-demo  > babel src -d dist    src\app.js -> dist\app.js |

Now, check the dist folder you will see the app.js file created. However, the content of the app.jsfile in the dist folder is the same as the one in the src folder.

To transform the JavaScript code, you need to instruct Babel to do it.

Configuring Babel

To enable the transformation of JavaScript using Babel, you need to install the latest presets:

|  |  |
| --- | --- |
| 1 | > npm install babel-preset-latest --save-dev |

To use a preset, you create a new file named .babelrc in the projects folder d:\projects\es6-demo\with the following content:

|  |  |
| --- | --- |
| 1  2  3 | {    "presets": ["latest"]  } |

Now, you can run the build command again:

|  |  |
| --- | --- |
| 1 | > npm run build |

and check the app.js in the  dist folder:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | "use strict";    var nums = [1, 2, 3];  var doubleNums = nums.map(function (e) {    return e \* 2;  }); |

As you can see, Babel converts the JavaScript ES6 version into ES5 version.