TypeScript - Build Tools

Build Tools are utilities that help automate the transformation and bundling of your code into a single file. Most JavaScript projects use these build tools to automate the build process.

There are several common build tools available that can be integrated with TypeScript. We will take a look at how to integrate TypeScript with some of these tools:

* Browserify
* Grunt
* Gulp
* Webpack

Browserify

Use Browserify plugin [Tsify](https://github.com/smrq/tsify" \t "_blank) for compiling TypeScript files.

Install Tsify using NPM (Node.js Package Manager) using the following command:

npm install tsify

Use the following command to compile your code and output the result in a file named bundle.js.

browserify main.ts -p [ tsify --noImplicitAny ] > bundle.js

Grunt

Use [grunt-ts](https://github.com/TypeStrong/grunt-ts) plugin from Grunt for TypeScript.

Install Grunt-ts using the following NPM command:

npm install grunt-ts

You will need to include the Grunt config file:

module.exports = function(grunt) {

grunt.initConfig({

ts: {

default : {

src: ["\*\*/\*.ts", "!node\_modules/\*\*/\*.ts"]

}

}

});

grunt.loadNpmTasks("grunt-ts");

grunt.registerTask("default", ["ts"]);

};

Gulp

Use [gulp-typescript](https://github.com/ivogabe/gulp-typescript) plugin for TypeScript.

Install it using the following NPM command:

npm install gulp-typescript

You will need to include the gulp config file:

var gulp = require("gulp");

var ts = require("gulp-typescript");

gulp.task("default", function () {

var tsResult = gulp.src("src/\*.ts")

.pipe(ts({

noImplicitAny: true,

out: "output.js"

}));

return tsResult.js.pipe(gulp.dest("built/local"));

});

Webpack

Use [ts-loader](https://www.npmjs.com/package/ts-loader" \t "_blank) plugin for TypeScript.

npm install ts-loader â€“save-dev

OR

npm install awesome-typescript-loader

You will need to include the webpack.config.js config file:

module.exports = {

entry: "./src/index.tsx",

output: {

filename: "bundle.js"

},

resolve: {

// Add '.ts' and '.tsx' as a resolvable extension.

extensions: ["", ".webpack.js", ".web.js", ".ts", ".tsx", ".js"]

},

module: {

loaders: [

// all files with a '.ts' or '.tsx' extension will be handled by 'ts-loader'

{ test: /\.tsx?$/, loader: "ts-loader" } // replace with your plugin of choice

]

}