codecademy

Browser Compatibility and Transpilation

Running scripts with npm

Command line shortcuts can be defined in a **package.json** file in a *"scripts"* object. This is used to combine multiple command line instructions in one command. These shortcuts can be executed from the terminal with the `npm run` command. For example, a script command named `"build"` can be executed with `npm run build`.

Babel Package Installation

The `babel-cli` JavaScript package contains Babel command line tools. The `babel-preset-env` JavaScript package is used to transpile *ES6* JavaScript code to *ES5*. They should be installed with the flag `-D` in the command line instruction like `npm install -D babel-preset-env` to be installed as development dependency.

ES5 & ES6 Compatibility

ECMAScript (ES) is a scripting language specification standardized by Ecma International for Javascript. ES5 is the older JavaScript version which is supported by all web browsers while the ES6 version, released in 2015, is not supported by all web browsers.

Installing Development JavaScript Packages

When a `-D` flag is used to install a JavaScript package using the command line, this adds the package under the property called *devDependencies* in a **package.json** file for the project. Whenever the developer runs `npm install`, all the listed packages and their dependencies will be installed.

ES6 JavaScript backwards compatibility

ES6 is a version of JavaScript that was released in 2015 and is backward compatible. One example of how to do this is the JavaScript *Babel* library which *transpiles ES6* JavaScript to *ES5*. Transpilation is the process of converting one programming language to another.

Babel configuration file

Babel uses the **.babelrc** file as a configuration file to determine the JavaScript file's presets, plugins and more. It can be created with the command `touch .babelrc` .

Babel build process

Babel uses the `npm run build` commend to covert all the JavaScript *ES6* files in the **src** folder of the project to *ES5*. This conversion makes the JavaScript code compatible on all modern browsers. The *ES5* code is written in a new file named **main.js** within the folder called **lib**

Installing JavaScript Packages

The *npm install* command installs JavaScript packages to the project directory. It creates a folder called **node_modules** and copies the JavaScript package files to it. This command also installs all the dependencies for the given package.

Node Package Manager

The *node package manager*, or *npm*, is a package manager for JavaScript which is used to access and manage Node packages - modules that contain JavaScript code written by other developers that are meant to provide helpful tools, reduce duplication of work, and reduce bugs.

Initiate JavaScript project

The command line instruction *npm init* is used to set up a new JavaScript project. A **package.json** file is created by the *npm init* command and contains a list of key-value pairs that provides information about the JavaScript project, such as the project name, version, description, a list of node packages required for the project, command line scripts, and more.

Caniuse.com for checking browser support

The website `caniuse.com` is useful for looking up the percent of browsers that support certain features in HTML, CSS, JavaScript, and their libraries.
For instance, you can find out what percent of browsers support specific features in ES6, as covered in the "Browser compatibility and transpilition" lesson.

Reasons to update JavaScript

Ecma international updated JavaScript from *ES5* to *ES6* in 2015 to make JavaScript syntax more similar to other languages, make JavaScript syntax more efficient and easier to read, and address *ES5* bugs.