Chapter 68: Transpiling

Transpiling is the process of interpreting certain programming languages and translating it to a specific target language. In this context, transpiling will take compile-to-JS languages and translate them into the **target** language of JavaScript.

Section 68.1: Introduction to Transpiling

Examples

ES6/ES2015 to ES5 (via Babel):

This ES2015 syntax

```
// ES2015 arrow function syntax
[1,2,3].map(n => n + 1);
```

is interpreted and translated to this ES5 syntax:

```
// Conventional ES5 anonymous function syntax
[1,2,3].map(function(n) {
    return n + 1;
});
```

CoffeeScript to JavaScript (via built-in CoffeeScript compiler):

This CoffeeScript

```
# Existence:
alert "I knew it!" if elvis?
```

is interpreted and translated to JavaScript:

```
if (typeof elvis !== "undefined" && elvis !== null) {
  alert("I knew it!");
}
```

How do I transpile?

Most compile-to-JavaScript languages have a transpiler **built-in** (like in CoffeeScript or TypeScript). In this case, you may just need to enable the language's transpiler via config settings or a checkbox. Advanced settings can also be set in relation to the transpiler.

For **ES6/ES2016-to-ES5 transpiling**, the most prominent transpiler being used is Babel.

Why should I transpile?

The most cited benefits include:

- The ability to use newer syntax reliably
- Compatibility among most, if not all browsers
- Usage of missing/not yet native features to JavaScript via languages like CoffeeScript or TypeScript

Section 68.2: Start using ES6/7 with Babel

Browser support for ES6 is growing, but to be sure your code will work on environments that don't fully support it, you can use Babel, the ES6/7 to ES5 transpiler, try it out!

If you would like to use ES6/7 in your projects without having to worry about compatibility, you can use Node and Babel CLI

Quick setup of a project with Babel for ES6/7 support

1. Download and install Node
2. Go to a folder and create a project using your favourite command line tool

```
~ npm init
```

3. Install Babel CLI

```
~ npm install --save-dev babel-cli
~ npm install --save-dev babel-preset-es2015
```

4. Create a `scripts` folder to store your `.js` files, and then a `dist/scripts` folder where the transpiled fully compatible files will be stored.
5. Create a `.babelrc` file in the root folder of your project, and write this on it

```
{
    "presets": ["es2015"]
}
```

6. Edit your `package.json` file (created when you ran `npm init`) and add the `build` script to the `scripts` property:

```
{
    ...
    "scripts": {
    ... ,
    "build": "babel scripts --out-dir dist/scripts"
    },
    ...
}
```

7. Enjoy programming in ES6/7
8. Run the following to transpile all your files to ES5

```
~ npm run build
```

For more complex projects you might want to take a look at Gulp or Webpack