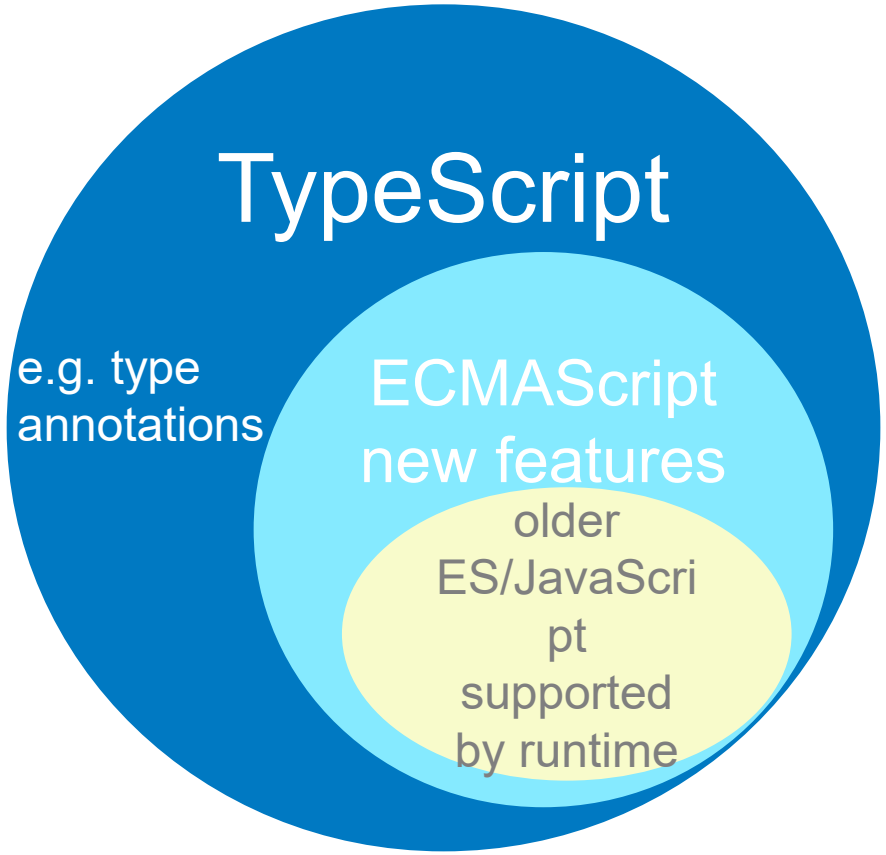
TypeScript usage explained

How TypeScript is related to JavaScript/ECMAScript and what steps are needed

2.10.2024

Juhani Välimäki

Principles as a picture



Installing TypeScript to your computer

- Assuming you have Node.js already installed to your computer, continue by installing TypeScript, e.g. this might be the correct command:

```
> npm i --save -g typescript @types/node
```

Starting to turn JavaScript project to TypeScript

- …

Initialize the (former JS?) project as TS project

```
> tsc init
```

- Creates the tsconfig.json file to the project root

tsconfig.json file:

- E.g.
- source folder (for TS files)
- Output/dist folder (for compiled JS files)
- How strict TypeScript required/followed?
- What version of ES should the output be? ES 2020?
- The module mechanism to use? e.g. export/import instead of module.exports/require

Package.json npm run/build etc scripts changed

- …scripts won’t be using JS tools anymore, but use TS tools like tsc compiler
- Or e.g. ts-watch could look for changing .ts files and compile them automatically to .js files

Renaming source files from .js to .ts

- … and start using TypeScript features (e.g. according to the list on course materials)

Run some checker that forces to 1. use TS features and 2. to use them correctly

This seems to be correct way to run **biome** in Windows computers, **crlf** : (Linux & Mac: change crlf to **cr**)

1. First you might need to fix and **rewrite formatting** of files for your enviroment (indentation and line-endings)
2. Second you can just **check** (for **other problems**/hints than formatting)
3. Third would also apply = **write those changes** to the files

```
npx @biomejs/biome format --write --max-diagnostics=200 --line-ending=crlf ./src

npx @biomejs/biome check  --max-diagnostics=200 --line-ending=crlf ./src

npx @biomejs/biome check --apply --max-diagnostics=200 --line-ending=crlf ./src
```

200 here means it will each time only notice/fix first 200 probs!

Install the TS versions of libraries, with their type definition modules

- E.g.

```
> npm i --save express @types/node @types/react @types/react-dom @types/jest
```

Understand compilation-time vs run-time

- - 1. Compilation time: tsc (TypeScript build) .ts->.j
- - 2. Runtime: run the .js, e.g. with node, nodemon, np2 or so

- See also how all TS tools are in DevDependencies in package.json, where as JS tools and modules are for running time

tsc compilation as the image