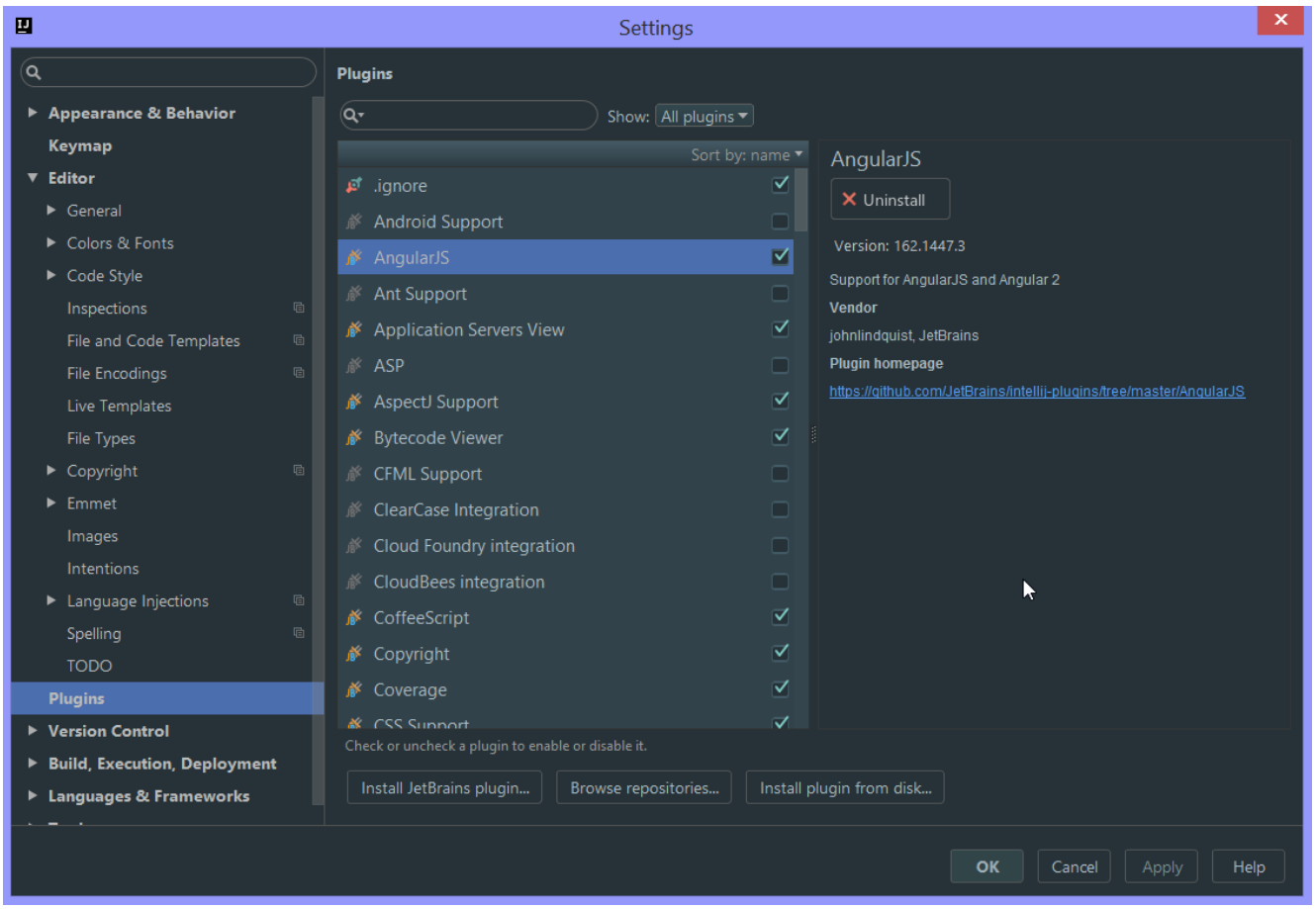
Einrichten der Entwicklungsumgebung (IntelliJ oder WebStorm)

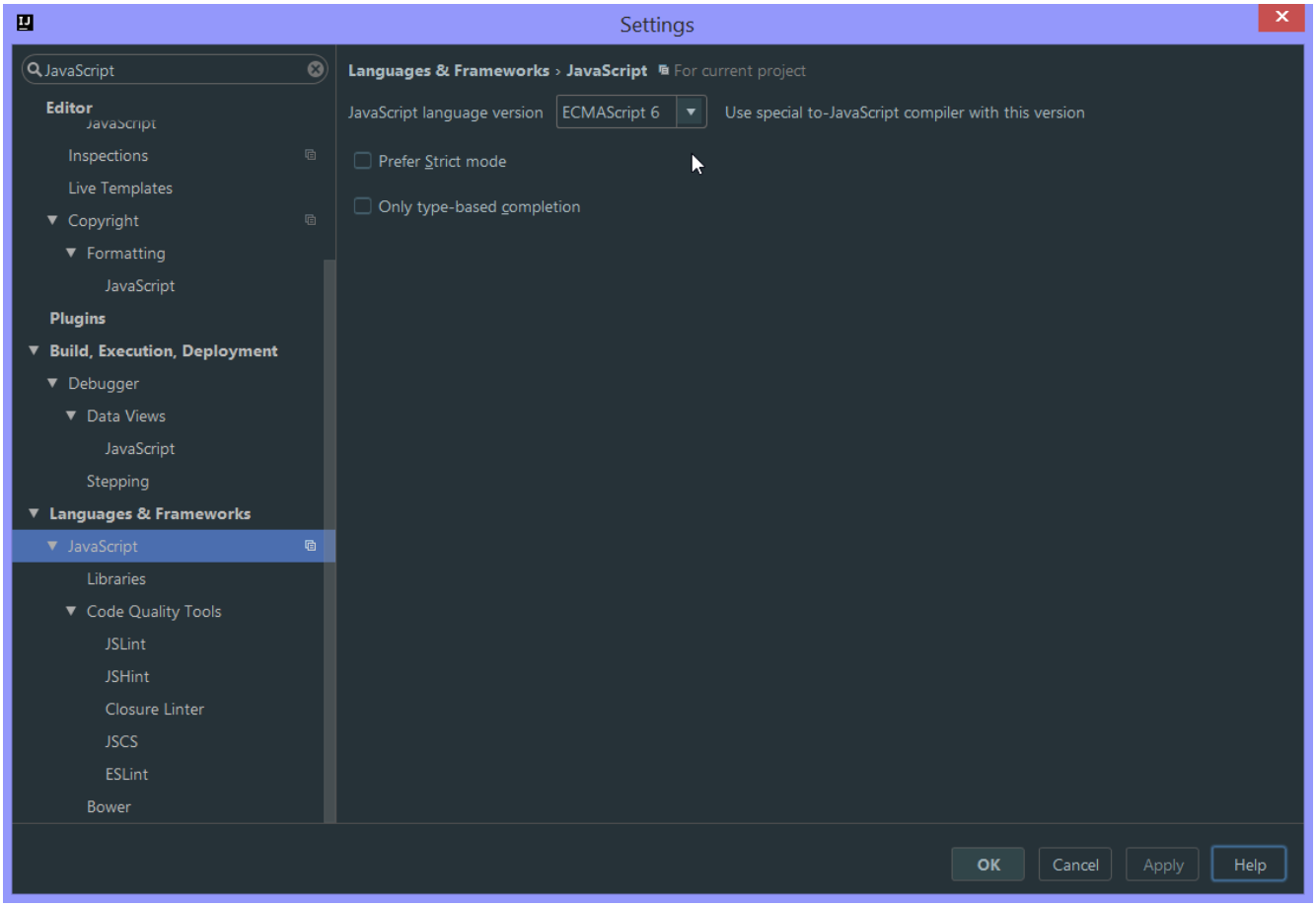
Um die neuen AngularJS 2 und TypeScript 2 Features nutzen zu können, muss die letzte IntelliJ oder WebStorm Version installiert werden. Die letzte Version von IntelliJ IDEA ist z.Z. **2016.2.2**.

Alle Einstellungen werden über den Menüpunkt File → Settings vorgenommen.

Als erstes muss das Plugin für AngularJS installiert sein, falls dies noch nicht geschehen ist.

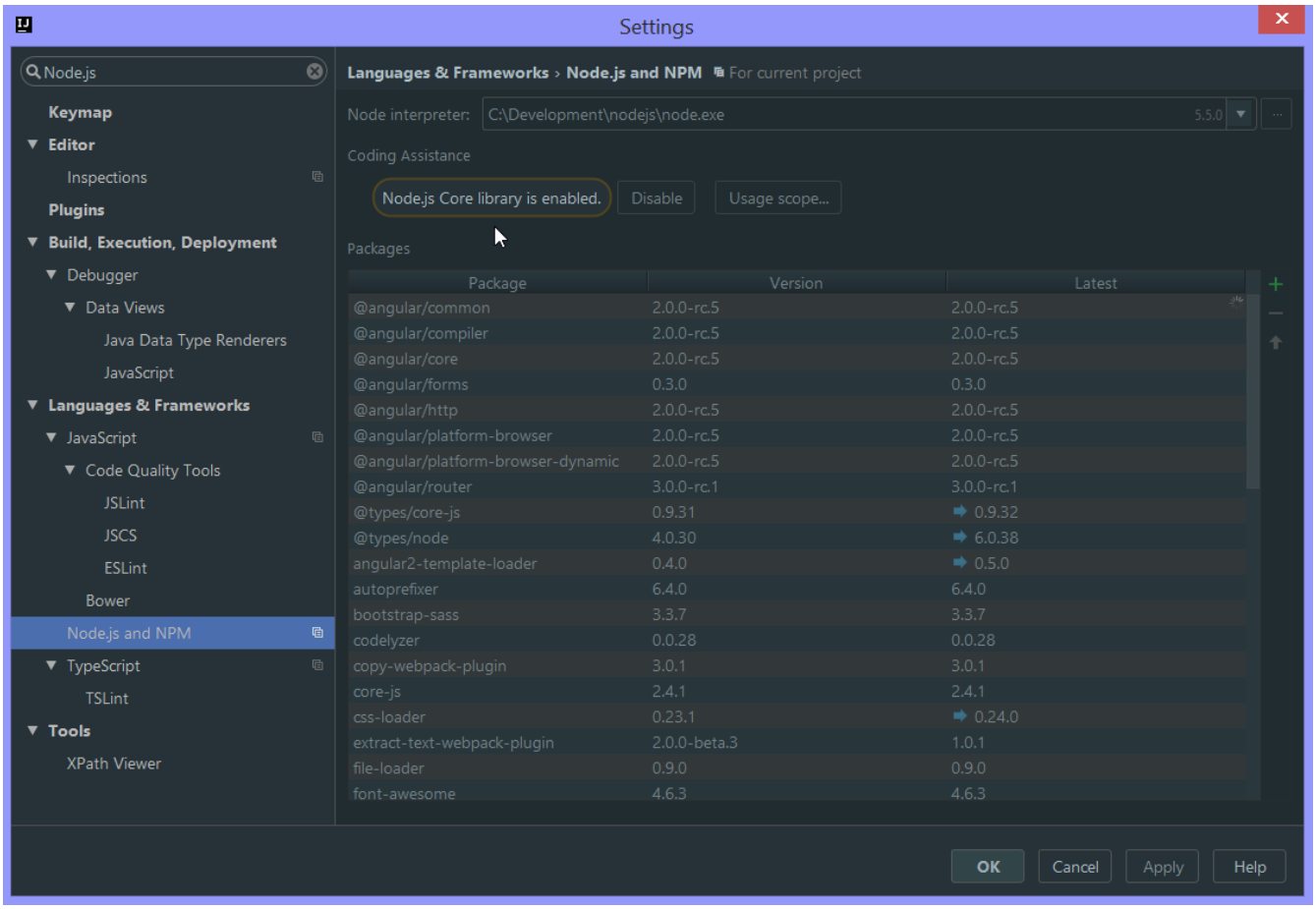


Auf der Language-Ebene muss ECMAScript 6 für JavaScript ausgewählt werden. Damit werden neue Konstrukte unterstützt, die auch in TypeScript zulässig sind (TypeScript ist ein Superset von ECMAScript 6).

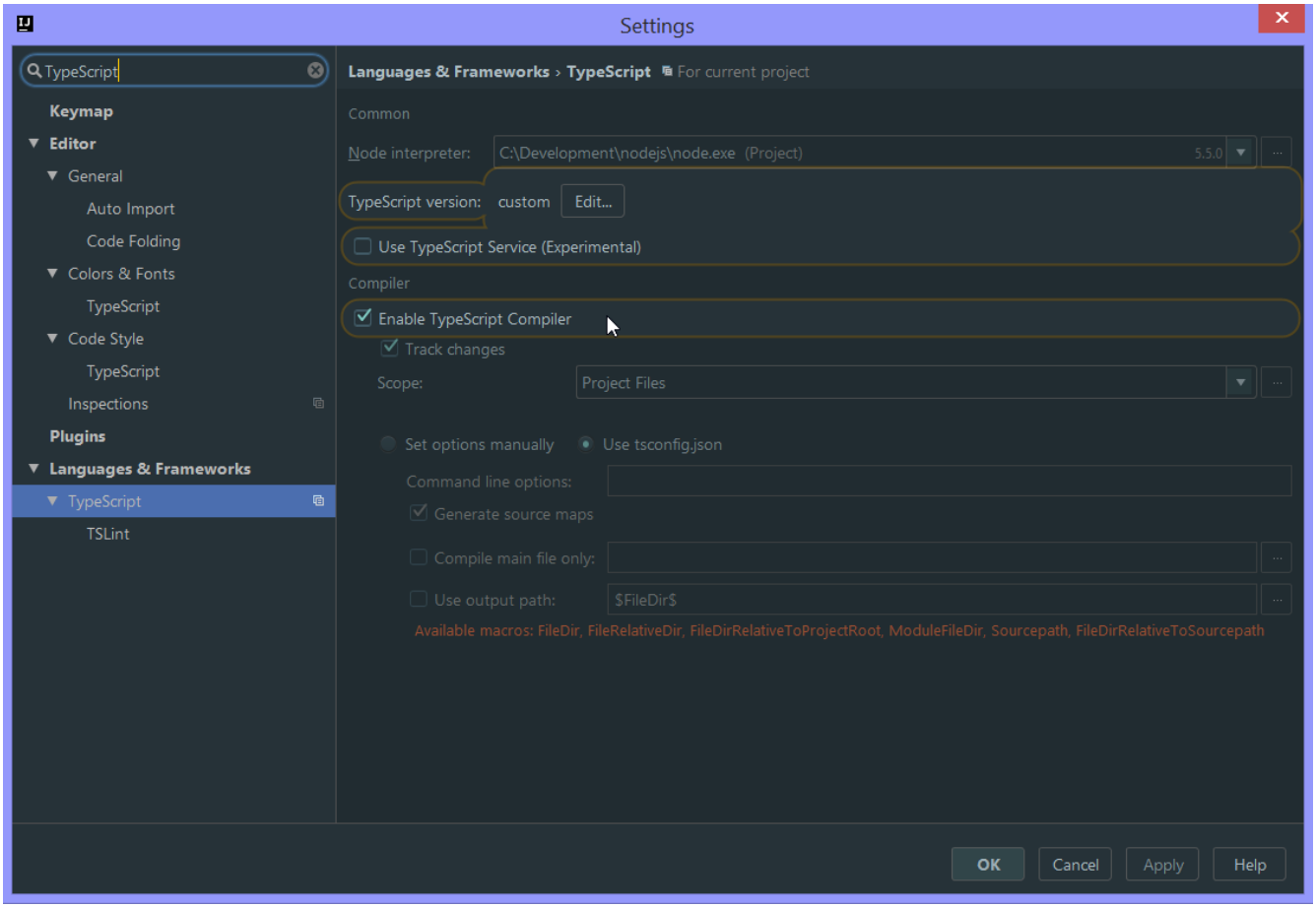


Bei Node.js and NPM muss „Node.js Core library“ enabled werden. Damit wird das Projekt als ein Node.js basiertes JavaScript-Projekt erkannt und z.B. „require“ und andere Node.js Anweisungen unterstützt (inkl. Navigation mit der Strg+Mausklick in die Anweisungen hinein).

Auch der Pfad zum Node.js Interpreter muss richtig gesetzt werden (geschieht normalerweise automatisch, wenn Node.js installiert ist).

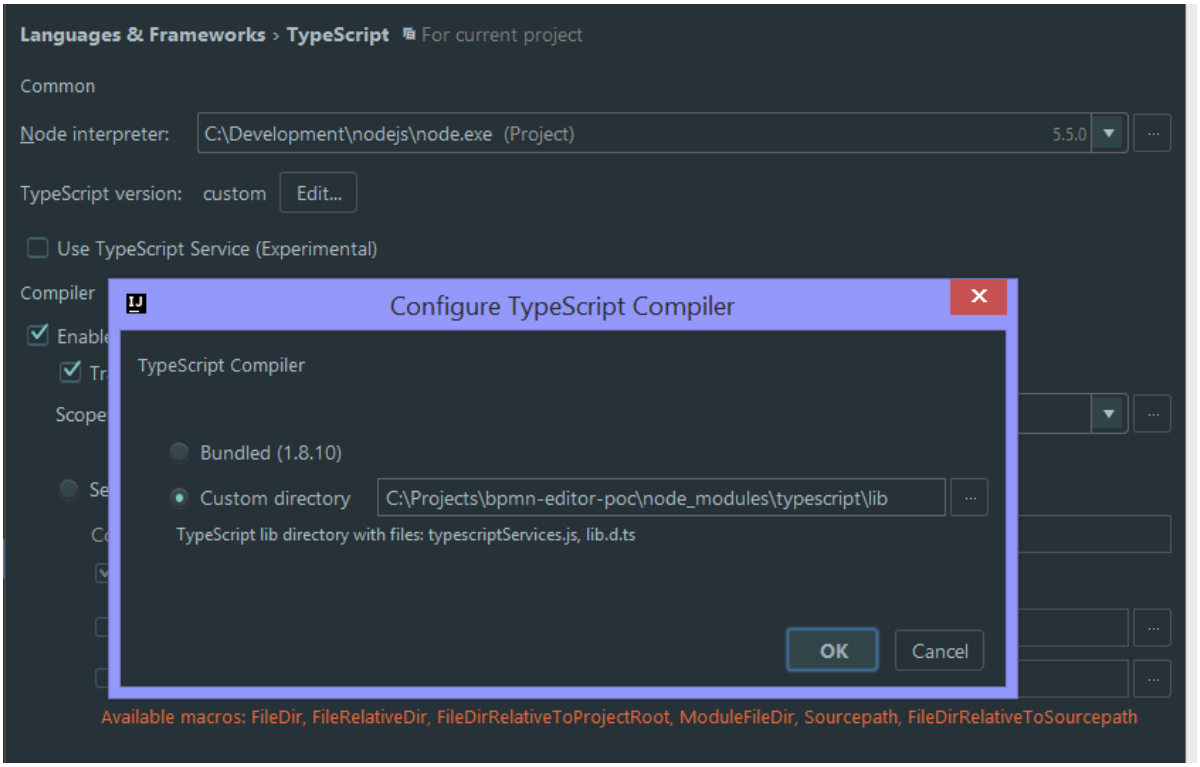


Bei „Languages & Frameworks“ → „TypeScript“ muss der TypeScript Compiler enabled werden. Dazu muss die Checkbox „Enable TypeScript Compiler“ angewählt werden.

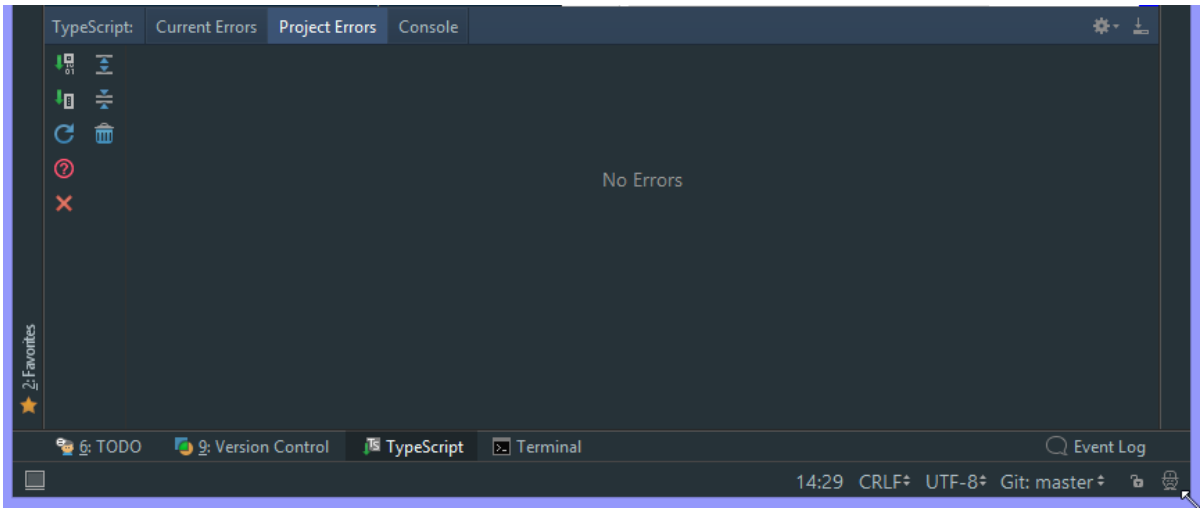


Hier müssen wir auch den Radio-Button „Use tsconfig.json“ wählen, damit die projektspezifische `tsconfig.json` Datei für Compiler-Settings genommen wird.

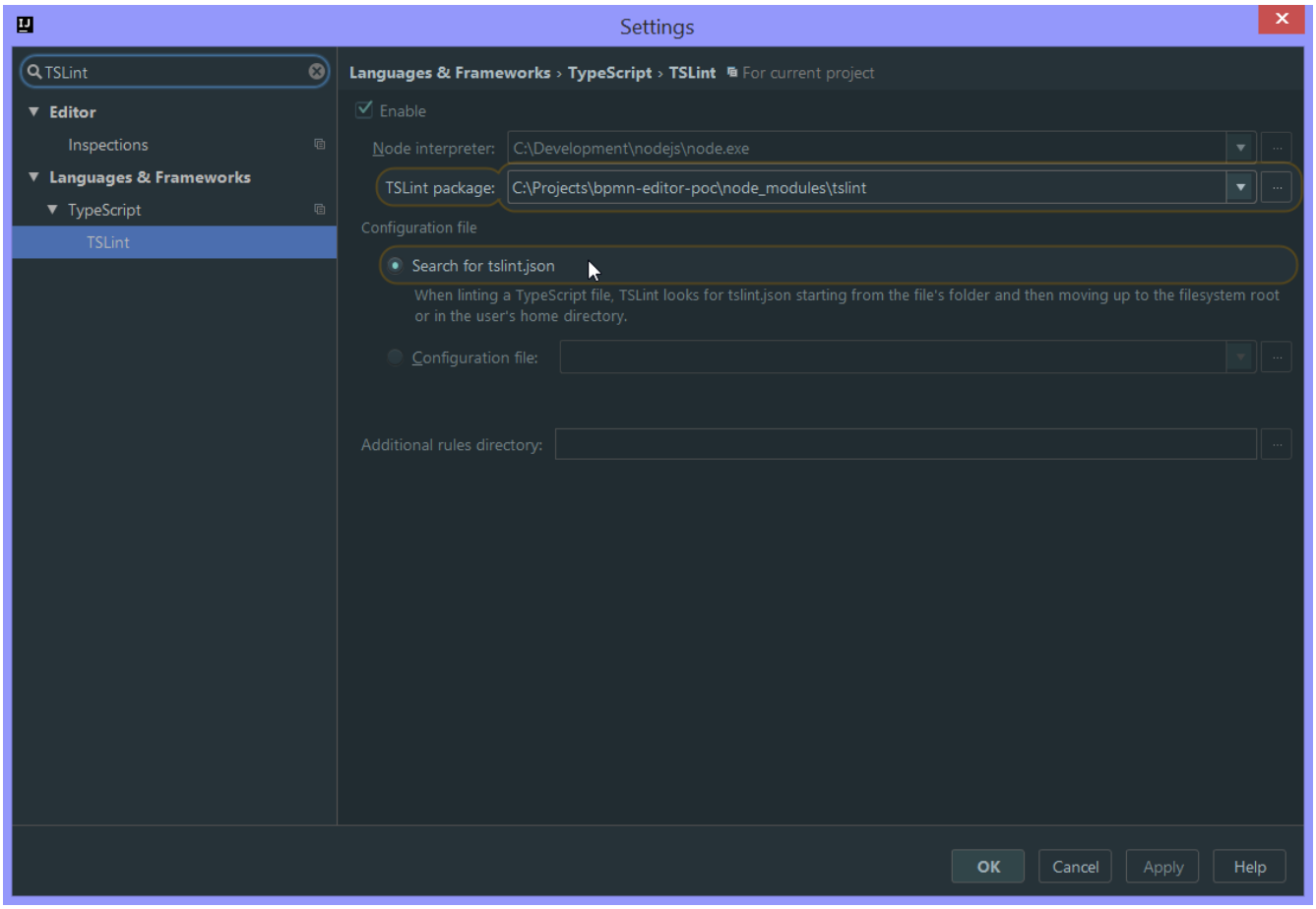
Die Entwicklungsumgebung wird mit der TypeScript Version 1.8.x mitgeliefert. Da wir die letzte TypeScript Version 2.x verwenden wollen, müssen wir bei der „TypeScript version“ auf „Edit“ klicken. Dort wählen wir „Custom directory“ und das Verzeichnis `<project root>/node_modules/typescript/lib` aus. Anschließend wird das mit OK übernommen.



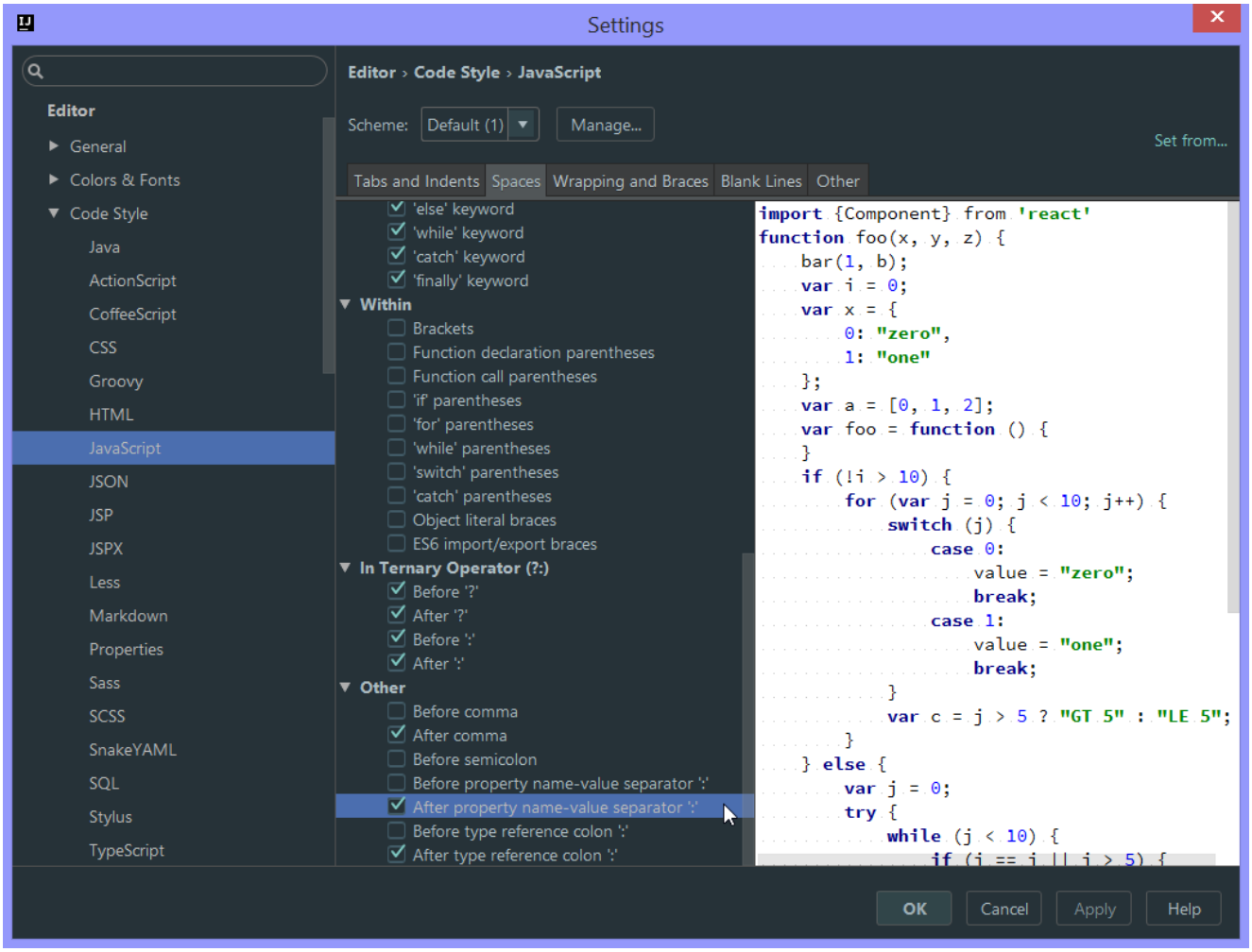
Beim Verzeichnis `<project root>/node_modules/typescript/lib` handelt es sich um das Verzeichnis wo die TypeScript-Dependency aus `package.json` installiert werden. Jetzt verwendet die IDE auch die letzte TypeScript Version und gibt entsprechende Meldungen im TypeScript-Tab aus (Inkrementelle Kompilierung).



TSLint muss enabled werden und das „TSLint package“ muss auf das installierte `<project root>/node_modules/tslint` Verzeichnis verweisen. Als Configuration file soll `tslint.json` ausgewählt werden (Radio-Button „Search for tslint.json“). Das ist eine projektspezifische Konfiguration für TSLint.

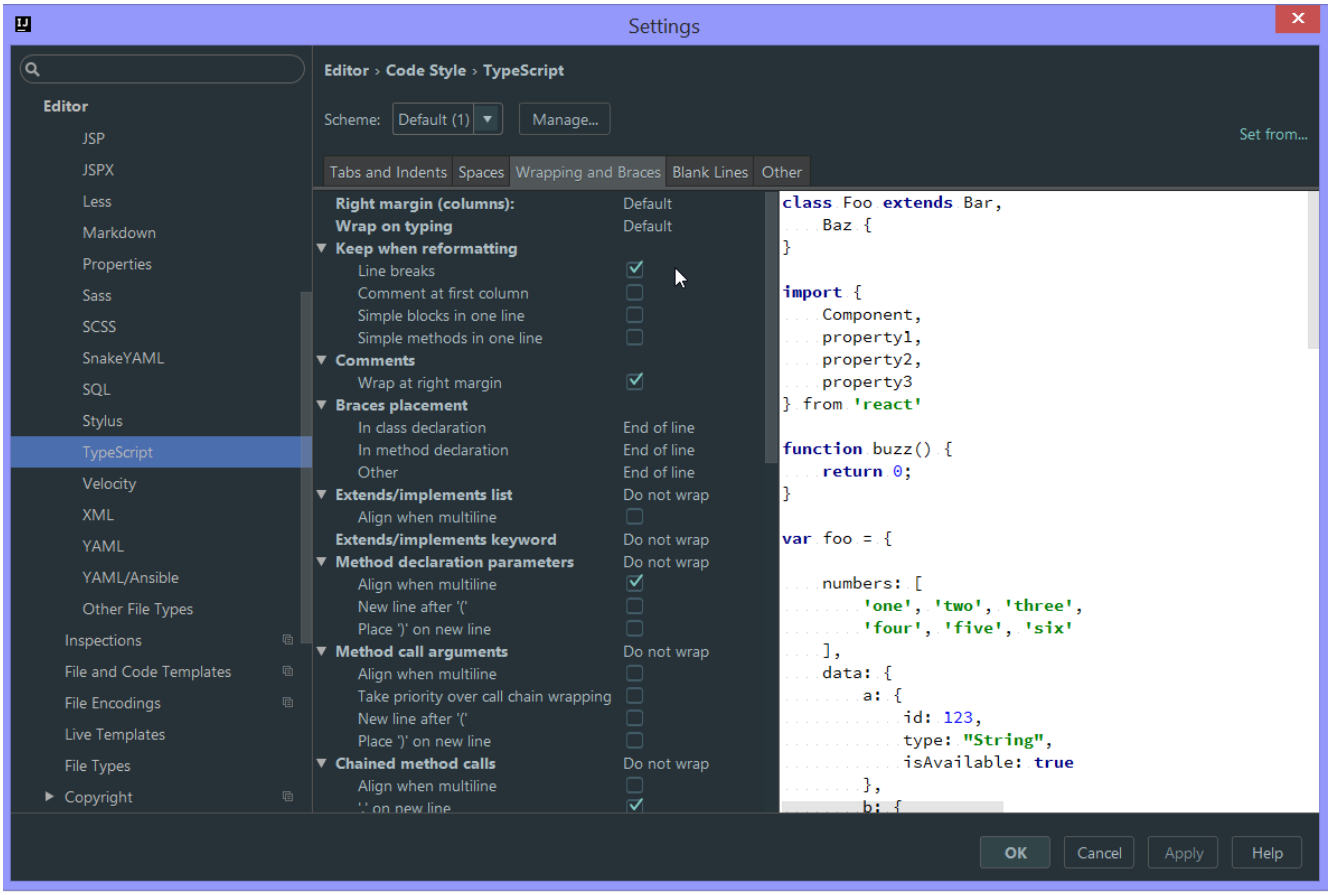


Code Styles können unter Editor → Code Style → JavaScript angepasst werden. Besonders wichtig wäre z.B. die Checkbox Spaces → After property name-value separator ‘:‘.



Den Source Code sollte immer formatiert werden. Die Tastenkombination für das Formatieren ist betriebssystemabgängig. Bei Windows wäre dies z.B. Shift+Strg+L.

Damit die Formatierung durch die IDE die manuellen Zeilenumbrüche nicht wieder neu formatiert, ist es anzuraten, die Checkbox „Keep when reformatting“ → „Line breaks“ anzukreuzen.



Mit dieser Einstellung bleibt z.B. die Formatierung eines Enums erhalten (jede Enum-Ausprägung in einer eigenen Zeile anstatt alles in einer Zeile).

```
export enum PaletteMode {
    Select,
    Marquee
}
```

Debuggen

Theoretisch wäre es möglich, den Source-Code aus der IDE heraus zu debuggen. Dazu muss zuerst ein Plugin (Extension) für Chrome installieren werden. Siehe https://www.jetbrains.com/help/webstorm/2016.1/using-jetbrains-chrome-extension.html

Praktisch geht das Debuggen leichter und schneller im Browser. Man kann den Source-Code direkt in Chrome Dev Console, Firebug oder IE Dev Tools debuggen (F12 zum Öffnen). Die meisten Entwickler machen das auch so. Wenn man Source Maps hat (und wir lassen die Source Maps durch WebPack beim Bauen generieren), lässt sich der TypeScript-Code in jedem Browser debuggen.