

# VS Code Tooling

## Visual Studio Code



**Entwickelt von:** Microsoft

**Lizenz:** MIT

**Geschrieben in:** TypeScript, JavaScript, CSS

**Plattform:** Electron

# Gründe für den Wechsel

## Wechsel weg von IntelliJ

- fühlt sich aufgebläht an
- Oft Speicherprobleme
- Customizing recht kompliziert
- Abstürze / Freezes
- Java Applikation
  - da liegt auch der Fokus von JetBrains

Trotz schlechter Performance Erfahrung

**Dracula Theme FTW!**

# Wieso Visual Studio Code?

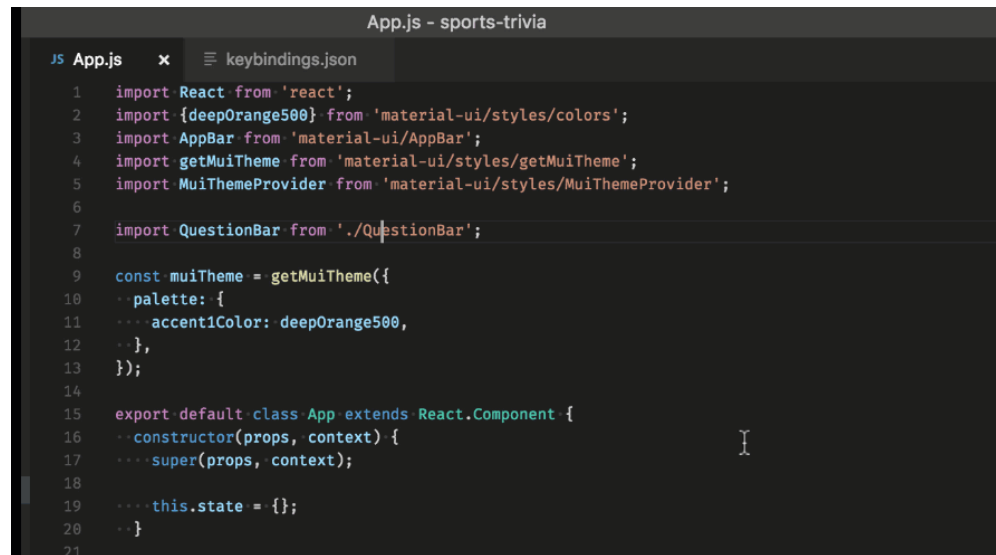
- große Community
- viele Erweiterungen
- ähnlicher Technologiestack
- geringe Einarbeitungszeit
- leichtgewichtig

## Schattenseiten :(

- nicht so ausgereift
- Qualität der Community Tools of mangelhaft
  - Überschneiden sich mit Core Funktionalität
- keine merge conflict Vergleichsansicht (großer Minuspunkt)
- Fühlt sich mehr wie ein Texteditor als IDE an
  - Texteditor on Steroids :)

# Killerfeatures

- Command Palette



The screenshot shows a code editor window titled 'App.js - sports-trivia'. The editor has two tabs: 'App.js' and 'keybindings.json'. The 'App.js' tab is active, displaying the following code:

```
1  import React from 'react';
2  import {deepOrange500} from 'material-ui/styles/colors';
3  import AppBar from 'material-ui/AppBar';
4  import getMuiTheme from 'material-ui/styles/getMuiTheme';
5  import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
6
7  import QuestionBar from './QuestionBar';
8
9  const muiTheme = getMuiTheme({
10    palette: {
11      accent1Color: deepOrange500,
12    },
13  });
14
15  export default class App extends React.Component {
16    constructor(props, context) {
17      super(props, context);
18
19      this.state = {};
20    }
21  }
```

- Settings JSON
- Tasks

# Persönliches VS Code Setup

## Theme / UI

Font: [JetBrains Mono](#)

Icons: [Material Icon Theme](#)

Keybindings: [IntelliJ Keybindings](#)

Theme: [Darcula IntelliJ Theme](#)

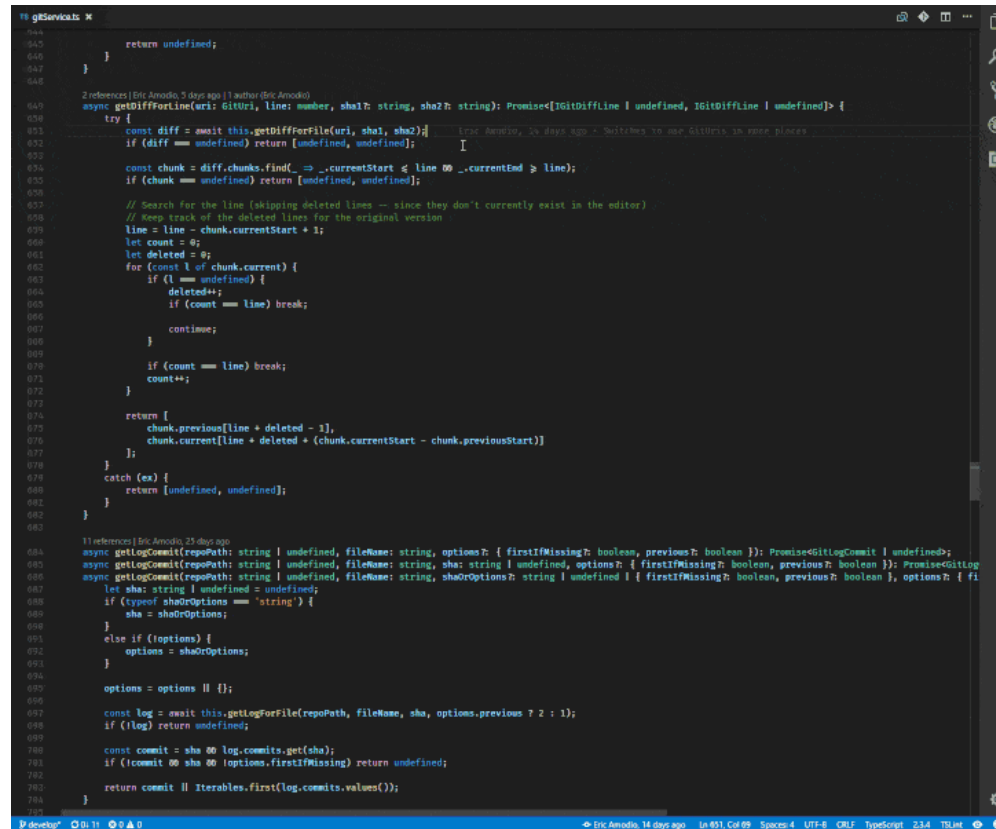
Highlighting: [Bracket Pair Colorizer](#)

```
22  public updateDocument(document: TextDocument) {  
23      const documentDecoration = this.getDocumentDecorations(document);  
24      if (documentDecoration) {  
25          documentDecoration.triggerUpdateDecorations();  
26      }  
27  }
```

# Tools

## Git:

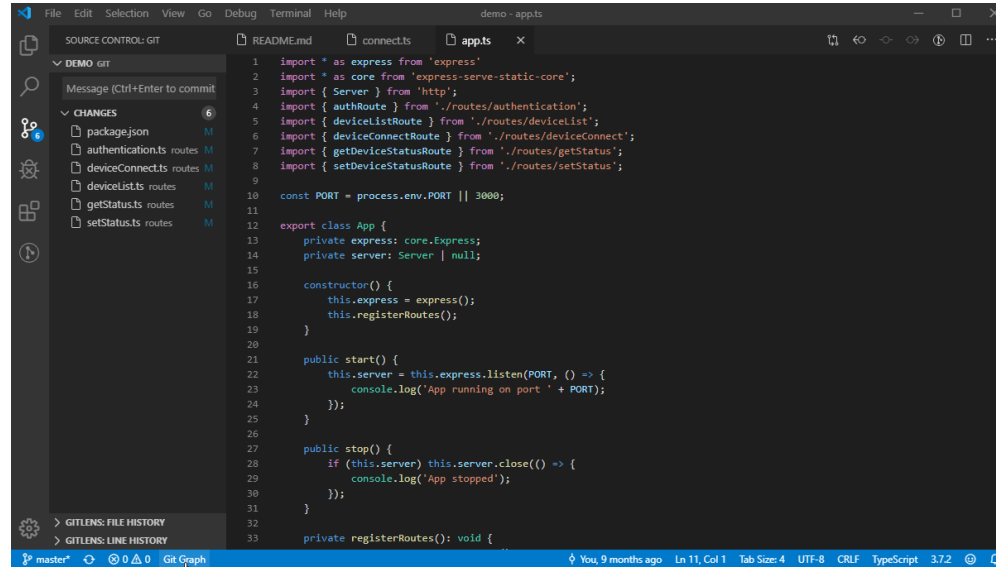
- **Git Lens** - Git blame, History



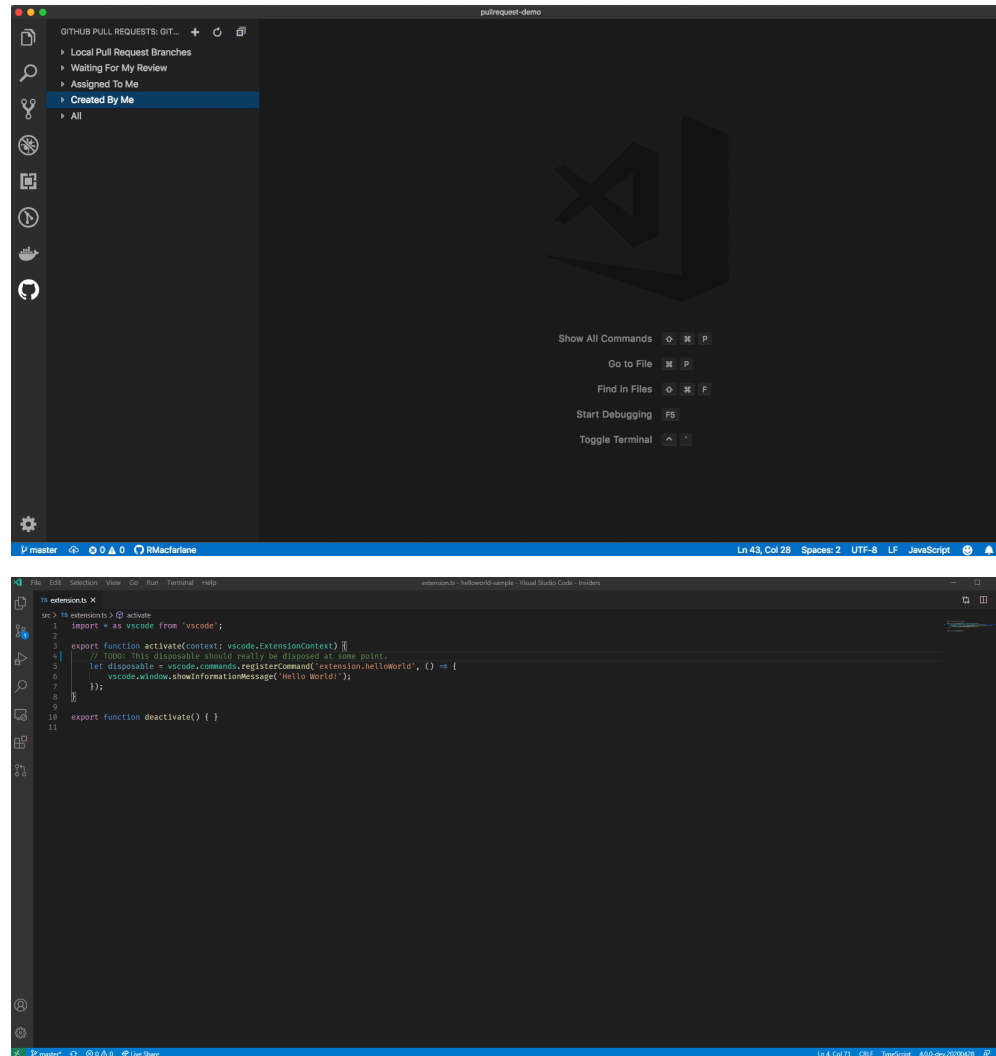
```
644     return undefined;
645   }
646 }
647
648 // References: [src/Amoeba, 5 days ago] editor [src/Amoeba]
649 async getDiffForLine(uri: GitUri, line: number, sha1: string, sha2: string): Promise<[IGitDiffLine | undefined, IGitDiffLine | undefined]> {
650   try {
651     const diff = await this.getDiffForFile(uri, sha1, sha2);
652     if (diff === undefined) return [undefined, undefined];
653     const chunk = diff.chunks.find(c => c.currentStart <= line && c.currentEnd >= line);
654     if (chunk === undefined) return [undefined, undefined];
655     // Search for the line (skipping deleted lines - since they don't currently exist in the editor)
656     // Keep track of the deleted lines for the original version
657     line = line - chunk.currentStart + 1;
658     let count = 0;
659     let deleted = 0;
660     for (const l of chunk.current) {
661       if (l === undefined) {
662         deleted++;
663         if (count === line) break;
664         continue;
665       }
666       if (count === line) break;
667       count++;
668     }
669     return [
670       chunk.previous[line + deleted - 1],
671       chunk.current[line + deleted + (chunk.currentStart - chunk.previousStart)]
672     ];
673   } catch (ex) {
674     return [undefined, undefined];
675   }
676 }
677
678 // References: [src/Amoeba, 23 days ago]
679 async getLogCommit(repoPath: string | undefined, fileName: string, options?: { firstIfMissing?: boolean, previous?: boolean }): Promise<GitLogCommit | undefined> {
680   async getLogCommit(repoPath: string | undefined, fileName: string, sha: string | undefined, options?: { firstIfMissing?: boolean, previous?: boolean }): Promise<GitLogCommit | undefined> {
681     let sha: string | undefined = undefined;
682     if (typeof shaOrOptions === 'string') {
683       sha = shaOrOptions;
684     } else if (options) {
685       options = shaOrOptions;
686     }
687     options = options || {};
688     const log = await this.getLogForFile(repoPath, fileName, sha, options.previous ? 2 : 1);
689     if (!log) return undefined;
690     const commit = sha && log.commits.get(sha);
691     if (!commit && sha && options.firstIfMissing) return undefined;
692     return commit || Iterables.first(log.commits.values());
693   }
694 }
```



- **Git Graph** - Commit, Branch, etc. Übersicht



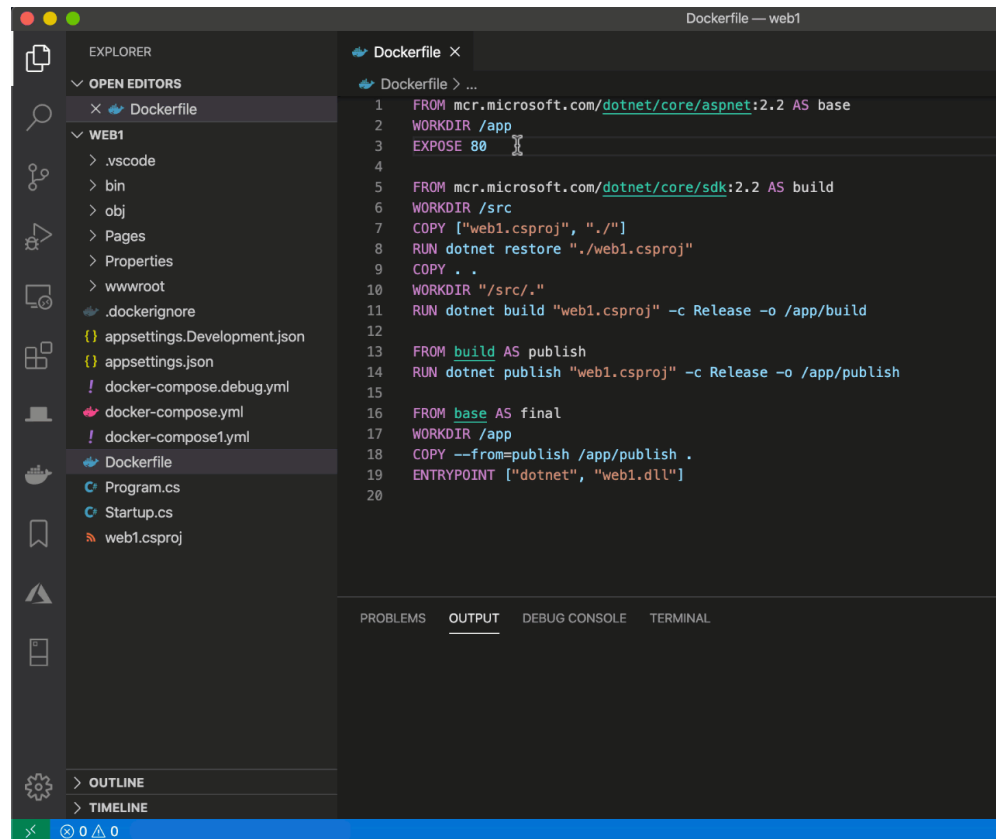
- GitHub Pull Requests and Issues



- Jira and Bitbucket (Official)

DevOps:

- Docker



The screenshot shows the Visual Studio Code interface with a Dockerfile open in the editor. The Explorer sidebar on the left shows the project structure, including a 'WEB1' folder and various configuration files. The Dockerfile content is as follows:

```
1 FROM mcr.microsoft.com/dotnet/core/aspnet:2.2 AS base
2 WORKDIR /app
3 EXPOSE 80
4
5 FROM mcr.microsoft.com/dotnet/core/sdk:2.2 AS build
6 WORKDIR /src
7 COPY ["web1.csproj", "."]
8 RUN dotnet restore "./web1.csproj"
9 COPY . .
10 WORKDIR "/src/."
11 RUN dotnet build "web1.csproj" -c Release -o /app/build
12
13 FROM build AS publish
14 RUN dotnet publish "web1.csproj" -c Release -o /app/publish
15
16 FROM base AS final
17 WORKDIR /app
18 COPY --from=publish /app/publish .
19 ENTRYPOINT ["dotnet", "web1.dll"]
20
```

The bottom of the interface shows the 'OUTPUT' tab, which is currently empty.

## IntelliSense:

- [Npm Intellisense](#) - Autovervollständigung für npm Module
- [Path Intellisense](#) - Autovervollständigung für Pfade
- [Visual Studio IntelliCode](#) - Intelligent Code Completion

## Sync:

- [Settings Sync](#)

## Scratchpad:

- Quokka.js

```
import delay from './delay'
import people from './people'

const add = async function(a, b) {
  delay(10)

  if (a < 0) throw new Error('not supported')

  return a + b
}

const result = await add(2, 3)

result 5

people.map(p => p.age)
```

- Scratchpads

