Semantic Highlighting

History

In 1.43 we enabled Semantic Highlighting as a new feature for all themes.

Due to feedback, since 1.43.1 only the built-in themes and custom themes that opt-in get semantic highlighting.

Semantic tokens are available for TypeScript and JavaScript. Support for Java, C++ and more is in the works.

FAQ

What is the difference between syntax and semantic highlighting?

Syntax highlighting colors the text based on lexical rules. In VS Code the lexical rules are expressed as regular expressions contained in a TextMate grammar.

Semantic highlighting enriches the syntax coloring based on symbol information from a language service that has the full understanding of the project. Based on this understanding each identifier gets colored & styled with the color of the symbol it resolves to. A constant variable name is rendered as constant throughout the file, not just in its declaration. Same for parameter names, property names, class names and so on.

Why do the semantic highlighting changes comes in with some delay?

The server takes a while to load and analyze the project, that's why the highlighting comes in delayed depending on the project size.

Which languages offer semantic highlighting

Currently semantic highlighting is only offered by TypeScript, JavaScript as well as JavaScript in HTML. More languages will adopt. The semantic token provider API is now available for all since 1.44.

If you are a language extension and want to implement a semantic token provider, please check out the sample.

Which themes offer semantic highlighting

In 1.43.1, out-of-the-box, only built-in themes show semantic highlighting. Other themes need to opt-in to semantic highlighting by a new property in the theme file:

```
"semanticHighlighting": true
```

Users can override that setting in the user settings:

```
"editor.semanticTokenColorCustomizations": {
    "[Atom One Dark]": {
        "enabled": true
    }
}
```

To see the feature in action, open a file for a language that has a semantic token provider available (currently TypeScript & JavaScript) and you will see that each identifier in the code now gets the color of the symbol it references (consts, parameters, types..).

As a theme author, do I need to change my theme to make it work with semantic highlighting?

Yes, only built-in themes show semantic highlighting out-of the box. All other themes need to opt-in to semantic highlighting by a new property in the theme file:

```
"semanticHighlighting": true
```

When set, language extensions like TypeScript start reporting semantic tokens.

Each semantic token is described by a token type, any number of token modifiers and a language. There's a standardized set of semantic types and modifiers, but languages can also define new and derived types and modifiers.

Color themes can write rules directly against these semantic token types, modifiers and language.

Alternatively, if a theme does not contain a semantic theming rule for a token, VSCode will use a mapping from semantic token to a TextMate scopes and look up the color in the themes TextMate rules.

Here's an example how theme can write theming rules for semantic token types and modifiers:

```
"semanticTokenColors": {
    "variable.readonly": "#ff0000",
}
```

`variable.readonly` is called a selector and has the form `(*|type)(.modifier)*(:language)?`

This rule colors all token of type `variable` and contain modifier `readonly`, red.

Here are other examples of rules:

- `"*.declaration": { "fontStyle": "bold" }` // all declarations are bold
- `"class:java": { "foreground": "#00ff00" "fontStyle": "bold" }` // token of type `class` in `java` files

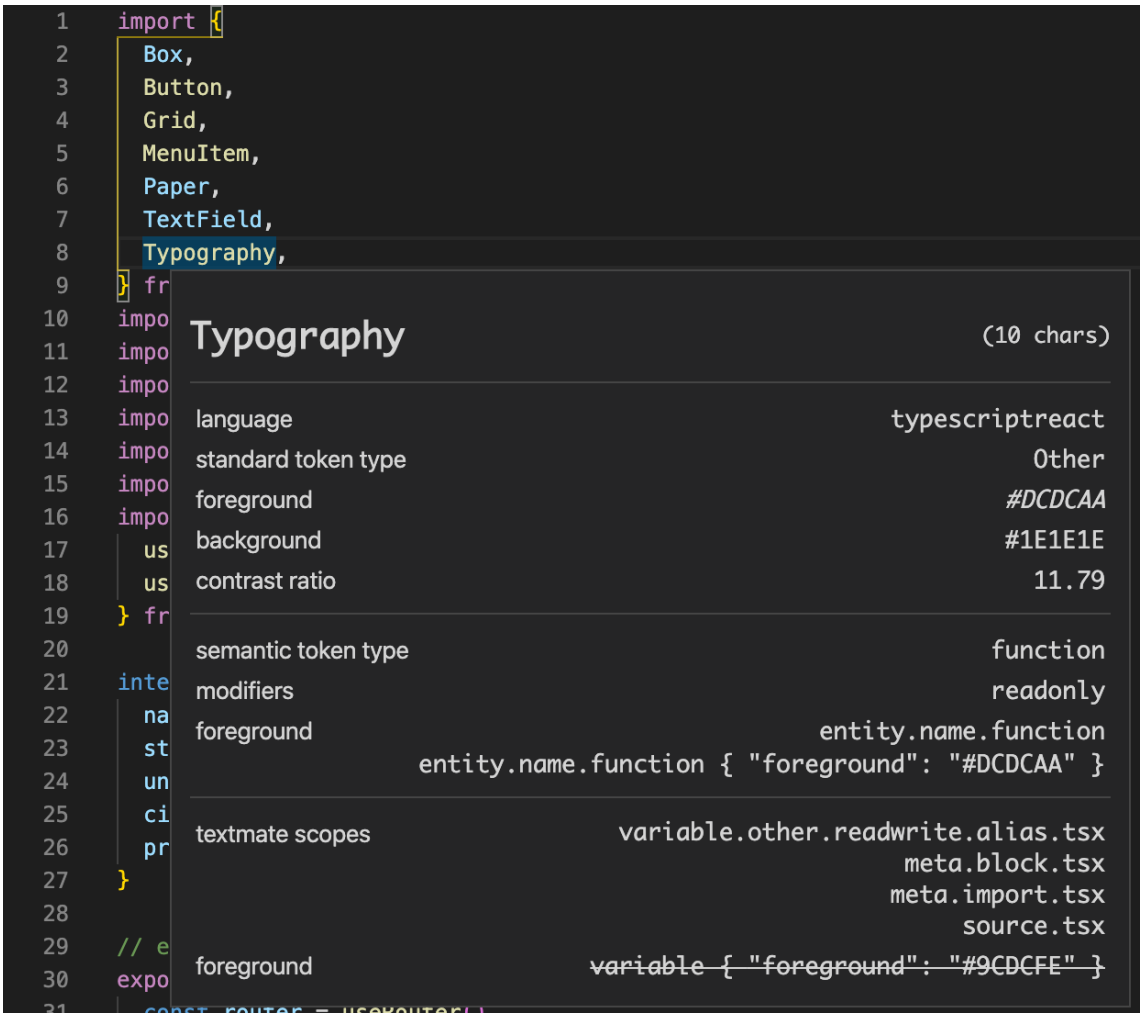
It's also possible to define semantic theming rules in the user settings:

```
"editor.semanticTokenColorCustomizations": {
    "enabled": true, // enable for all themes
    "rules": {
            "property.readonly": {
                "foreground": "#35166d"
            },
            "*.defaultLibrary": {
                "fontStyle": "underline"
            }
    }
}
```

How can I find out what semantic tokens are available and how they are themed?

Open a file for a language that has a semantic token provider available (currently TypeScript & JavaScript).

Set the cursor to the symbol to inspect and run the `Developer: Inspect Editor Tokens and Scopes` command.



`Semantic token type` and `modifiers` show the classification of the given symbol and the semantic theming rule or the TextMate theme rule that was used to style the token.

This readme describes the token types and modifiers that the TypeScript / JavaScript semantic highlighter produces, along with a list of known issues. If you feel that a classification is wrong, please file an issue against that repo. Please add a small code sample to reproduce along what classification you expect.

Links:

- API: vscode.d.ts
- Latest standard token types and modifiers: tokenClassificationRegistry.ts#L364
- Semantic highlighting for JS in HTML: javascriptSemanticTokens.ts
- Sample: semantic-tokens-sample