TypeFox

Towards a TypeScript native DSL framework with Langium 

Mark Sujew

@mark_sujew @msujew

Introductions

Mark Sujew



- Software Engineer
- Mainly working on Open-Source Software
- Working on my Masters degree

TypeFox

- Software Consulting Company
- Specialized in building DSLs and IDEs

Language Engineering Tools

- Create a formal language
- Provide editor support:

Syntax Highlighting

```
package foo.bar {
    datatype Complex
}
```

Validation

```
Could not resolve reference to 'Sting'.
View Problem    No quick fixes available
name: Sting
```

Code Completion

```
entity E3 {
    that: E4
    other:
    nested:
}
```

E1 Entity
E3
E4

Language Engineering Tools



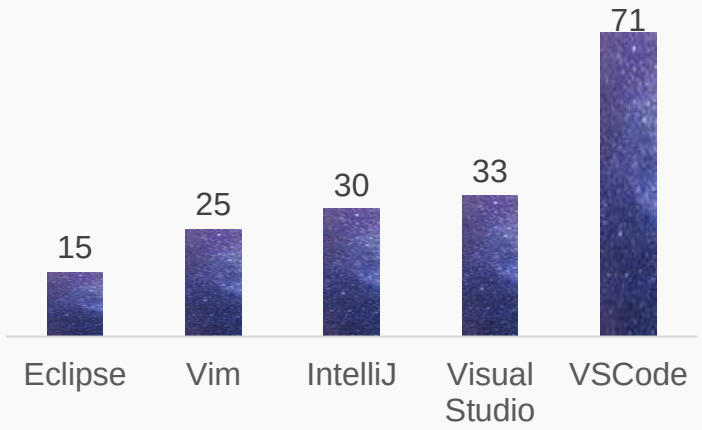
xtext

{S} spoofax

- Deeply integrated in selected platforms
- All running on the JVM

A Changing Industry

- The industry is moving towards the Web-Stack
- The IDE space has become fragmented
- VSCode has become the #1 Developer Tool



Source: StackOverflow Survey 2021

Langium 

- Key Principle: Simplicity
  - Relies on the Language-Server-Protocol (LSP)
  - Main target VSCode
  - Built on the lessons learned working with Xtext

Excursion: LSP

- IDE independent protocol for language support
- Supported by all modern IDEs
- Language Servers exist for all common languages

Setting up Langium

```
gitpod /workspace/test $ yo langium
```




```
        Copyright 2021 TypeFox GmbH

? Your extension name: hello-world
? Your language name: Hello World
? File extensions of your language, separated by commas: .hello
```

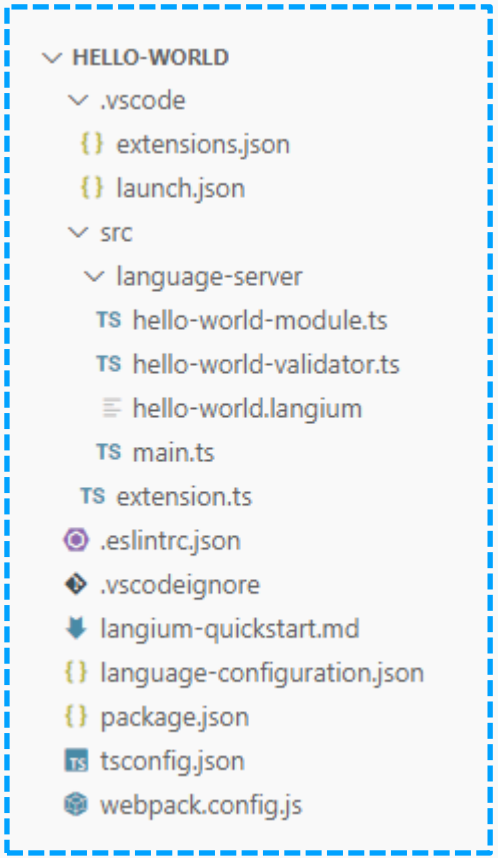
Setting up Langium

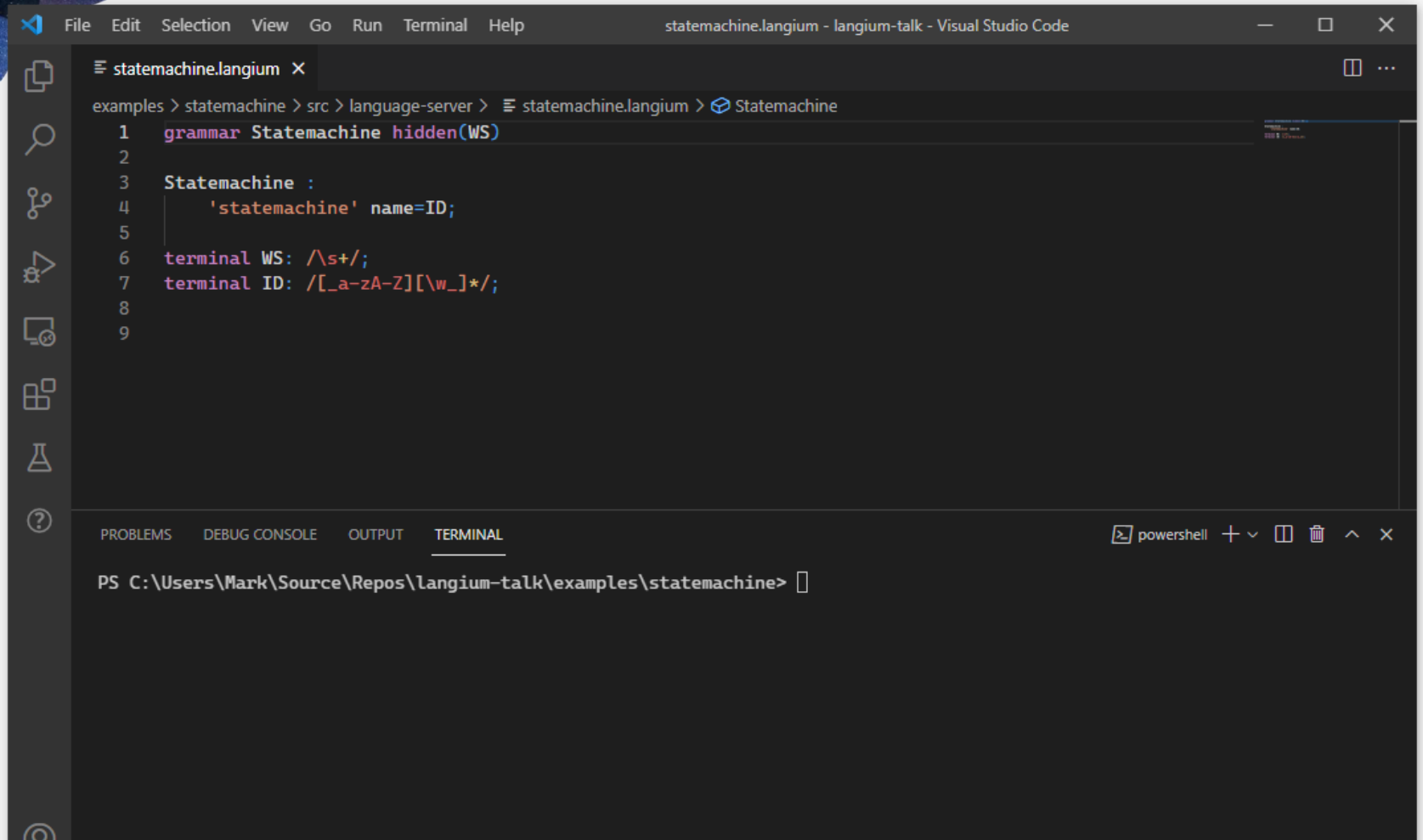
```
gitpod /workspace/test $ yo langium
```

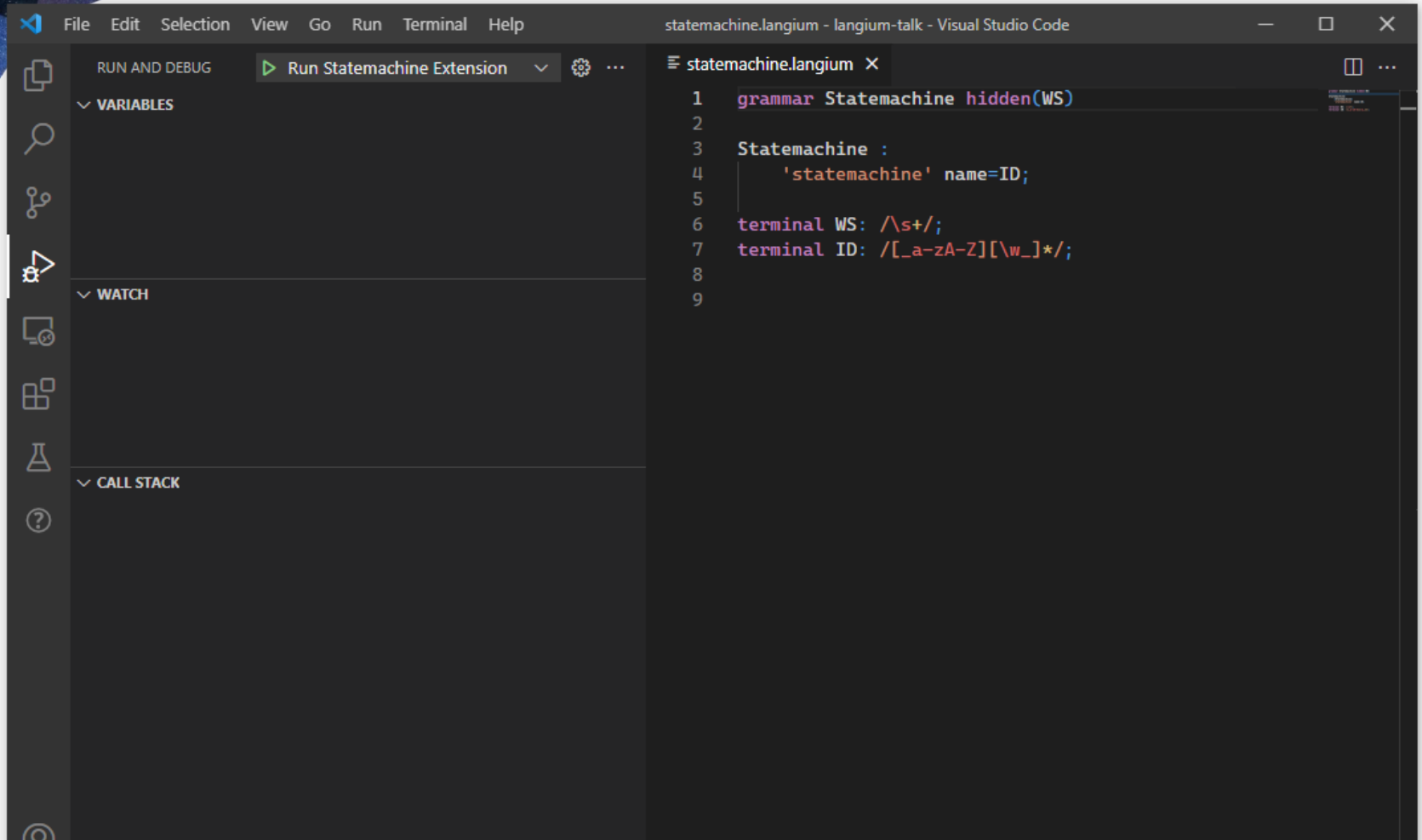


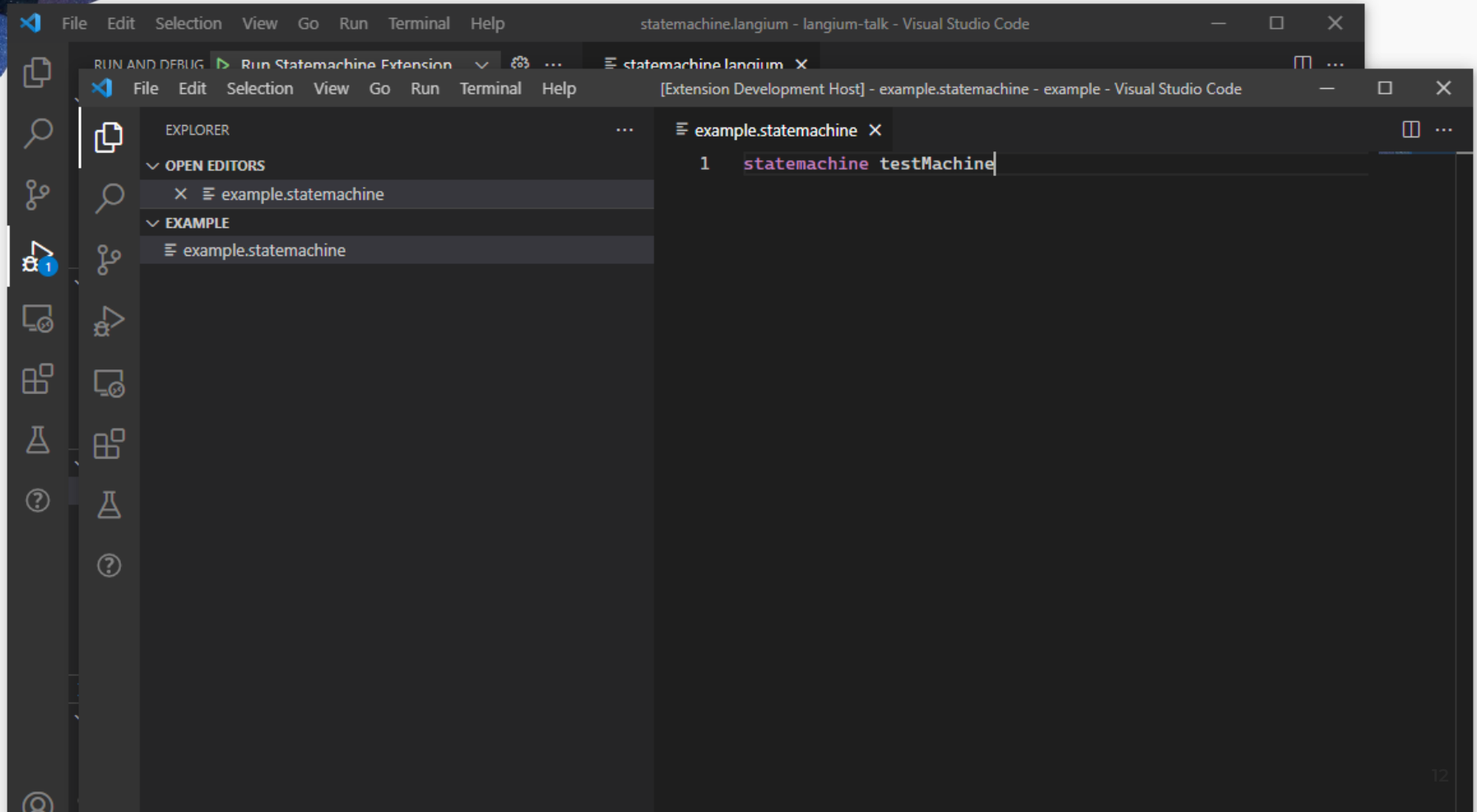

```
Copyright 2021 TypeFox GmbH

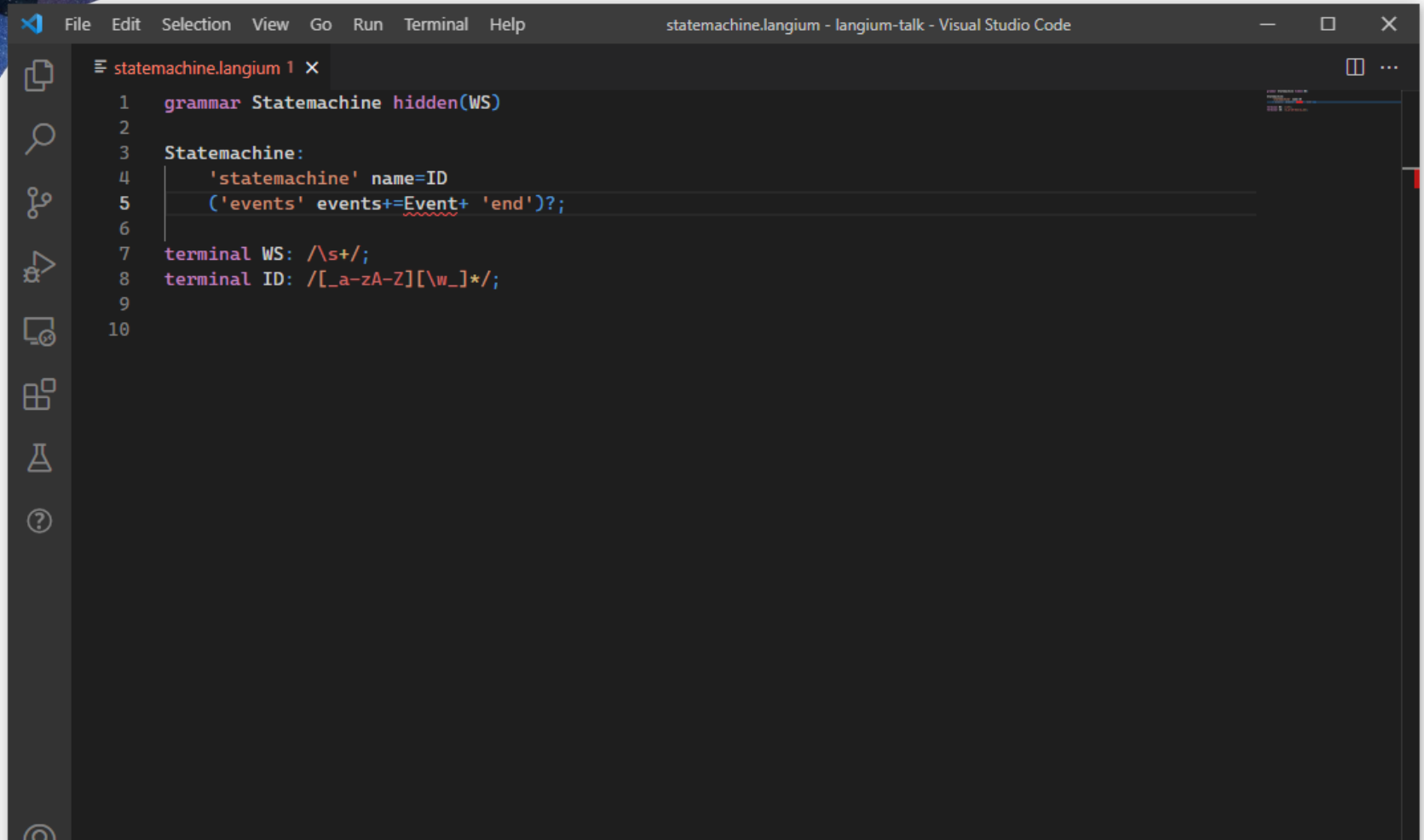
? Your extension name: hello-world
? Your language name: Hello World
```

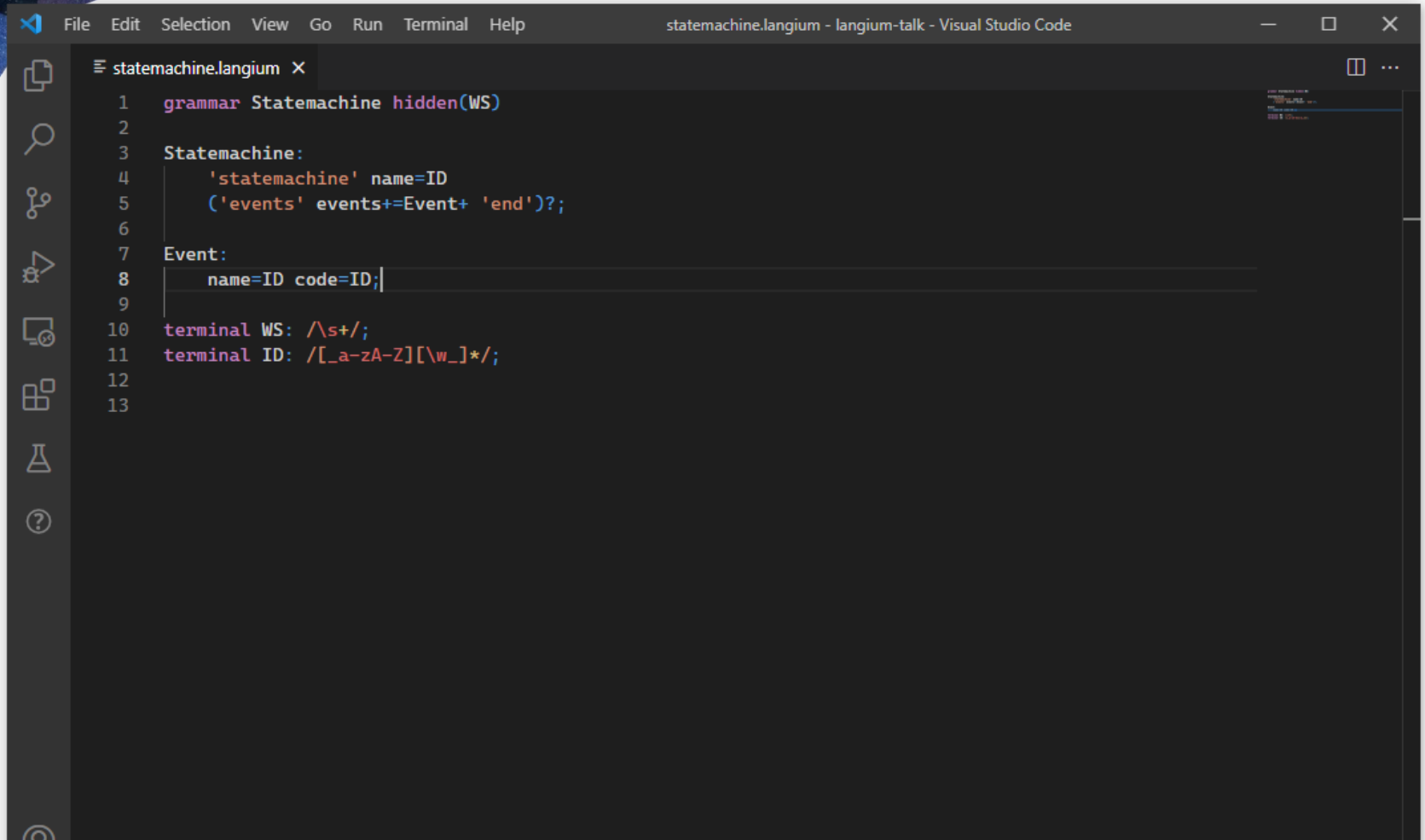


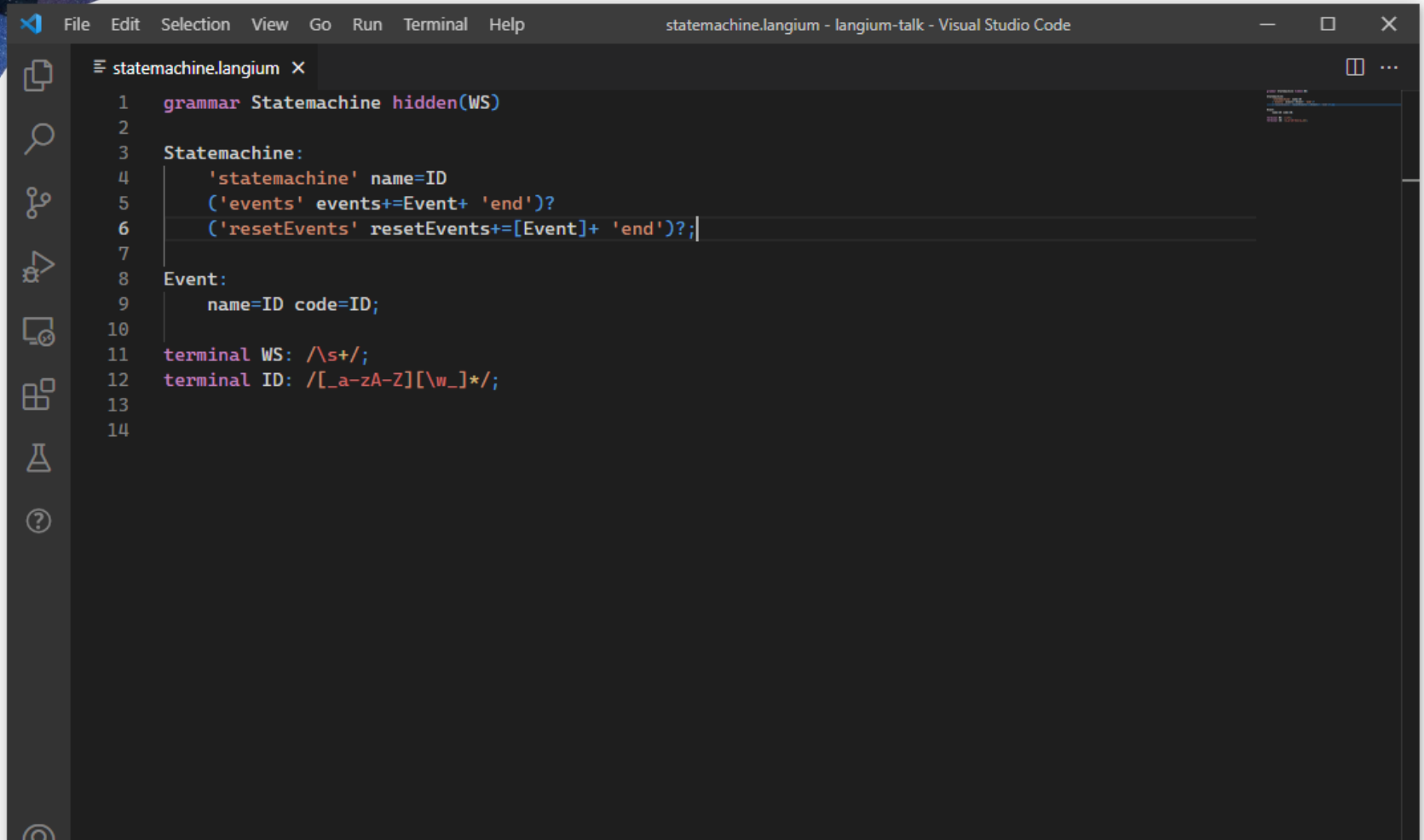

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
statemachine.langium
examples > statemachine > src > language-server > statemachine.langium > Statemachine
1 grammar Statemachine hidden(WS)
2
3 Statemachine :
4 'statemachine' name=ID;
5
6 terminal WS: /\s+/;
7 terminal ID: /[_a-zA-Z][\w_]*/;
8
9
PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL
powershell
PS C:\Users\Mark\Source\Repos\langium-talk\examples\statemachine>

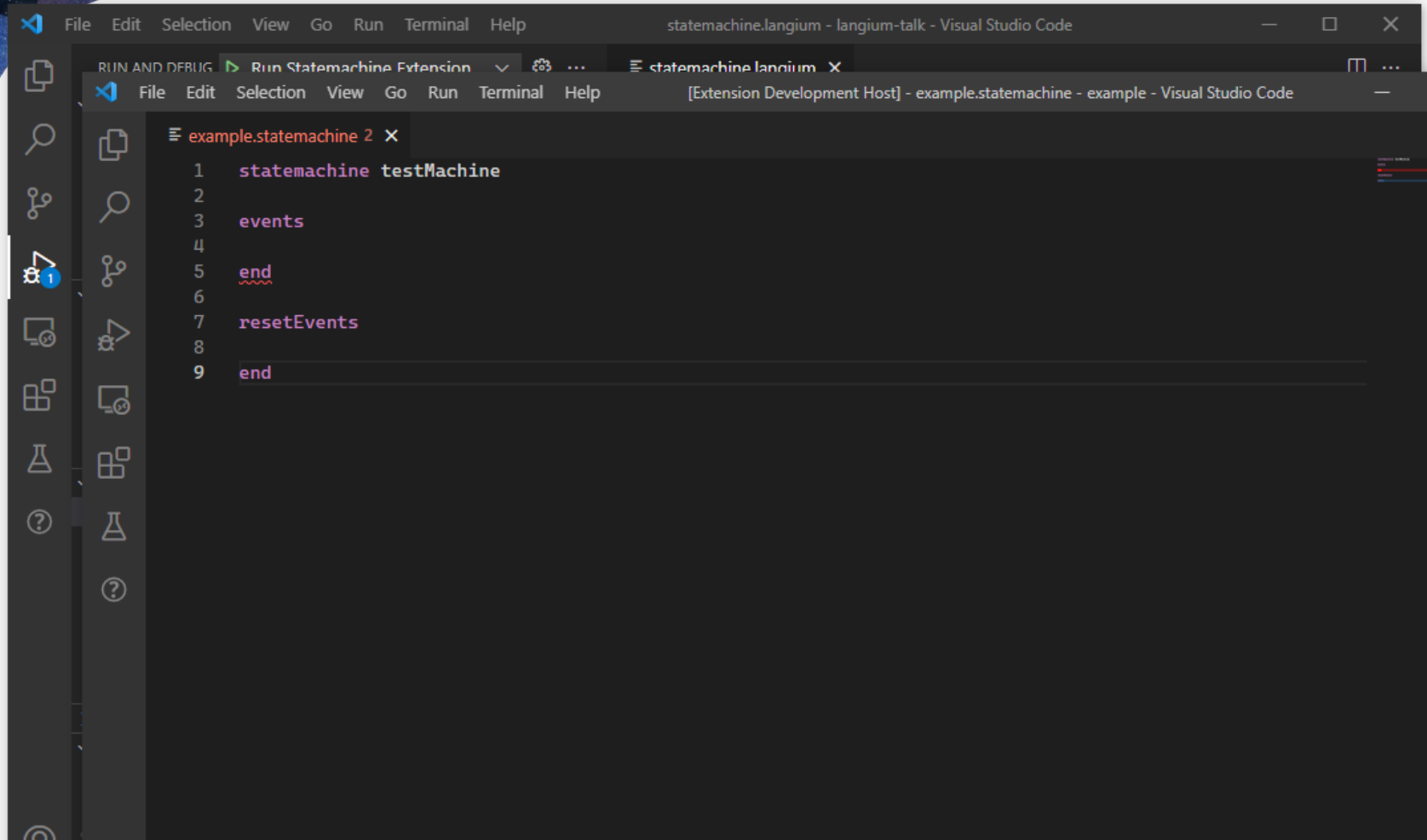

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
VARIABLES
WATCH
CALL STACK
statemachine.langium
1 grammar Statemachine hidden(WS)
2
3 Statemachine :
4 'statemachine' name=ID;
5
6 terminal WS: /\s+/;
7 terminal ID: /[_a-zA-Z][\w_]*/;
8
9

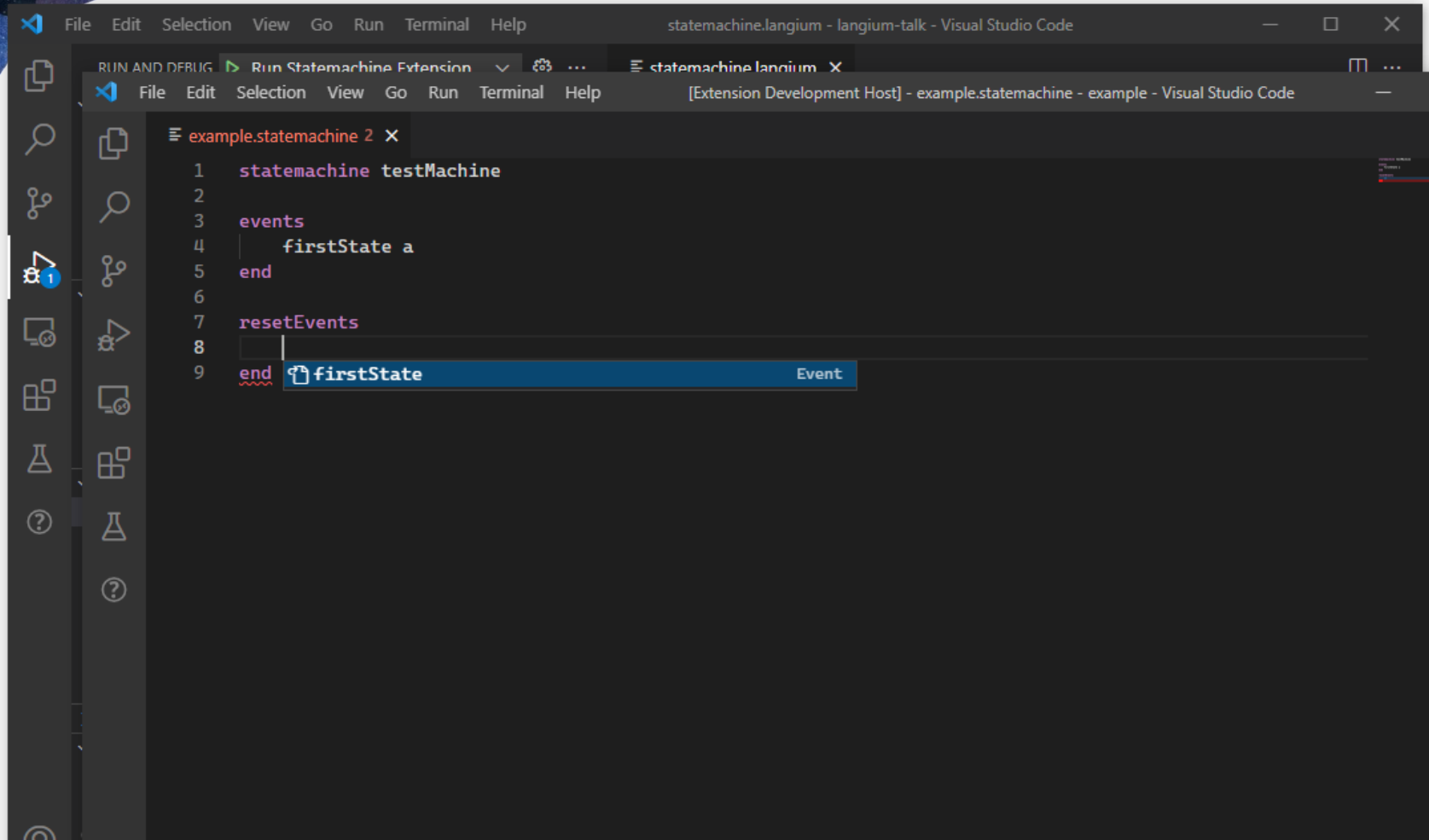

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - example.statemachine - example - Visual Studio Code
EXPLORER
OPEN EDITORS
example.statemachine
EXAMPLE
example.statemachine
example.statemachine
1 statemachine testMachine

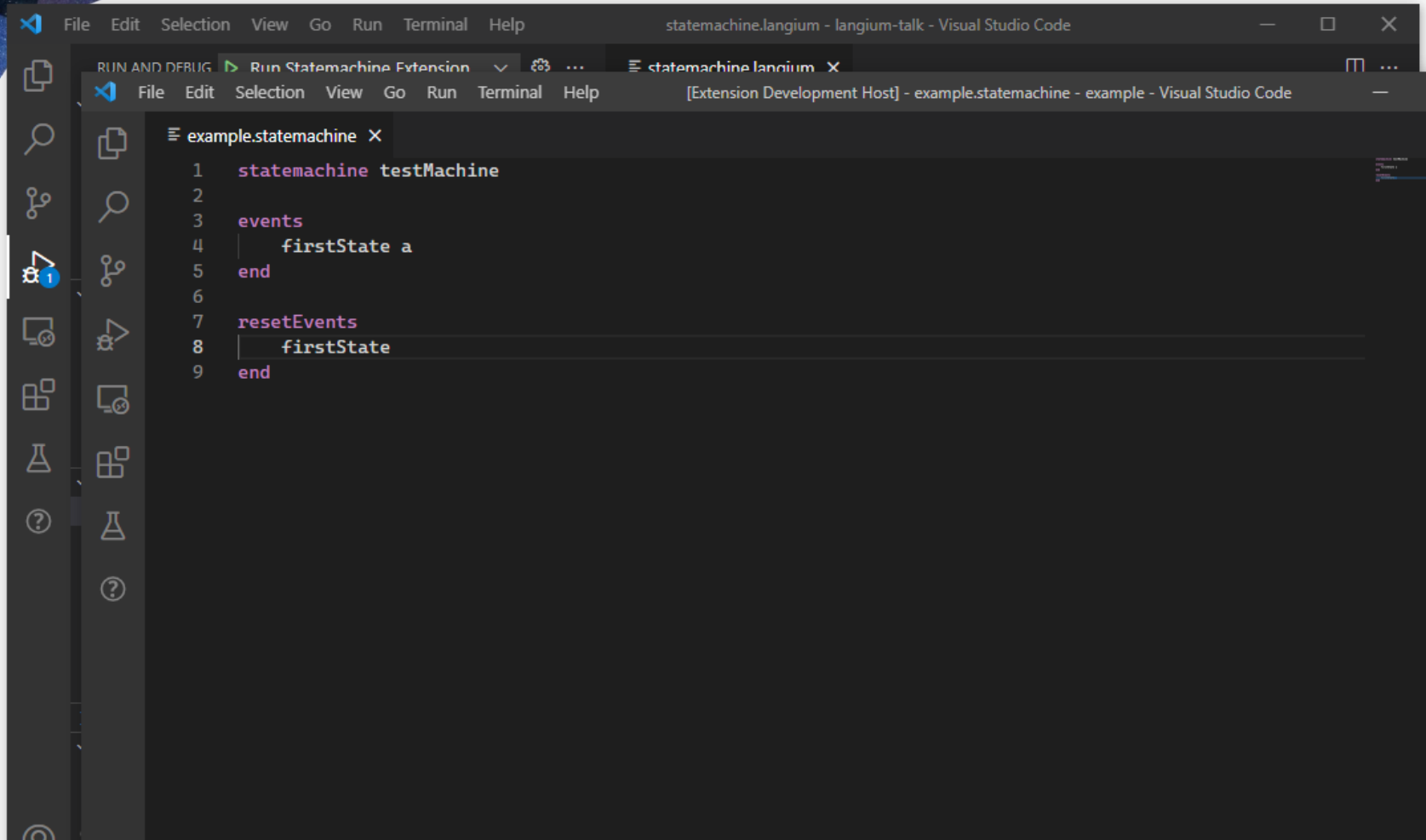

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
statemachine.langium 1
grammar Statemachine hidden(WS)
Statemachine:
'statemachine' name=ID
('events' events+=Event+ 'end')?;
terminal WS: /\s+/;
terminal ID: /[_a-zA-Z][\w_]*/;

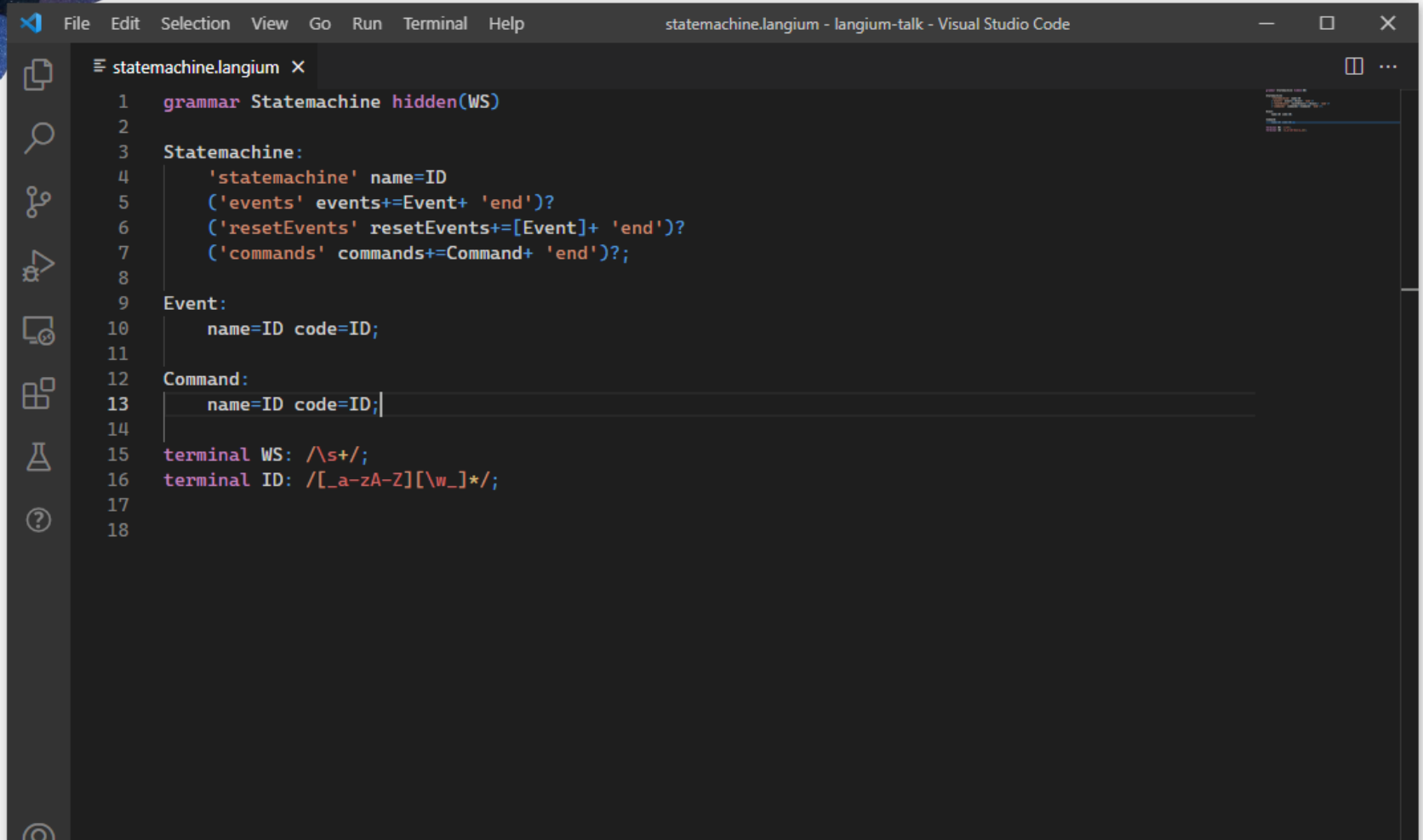

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
statemachine.langium
1 grammar Statemachine hidden(WS)
2
3 Statemachine:
4 'statemachine' name=ID
5 ('events' events+=Event+ 'end')?;
6
7 Event:
8 name=ID code=ID;
9
10 terminal WS: /\s+/;
11 terminal ID: /[_a-zA-Z][\w_]*/;
12
13

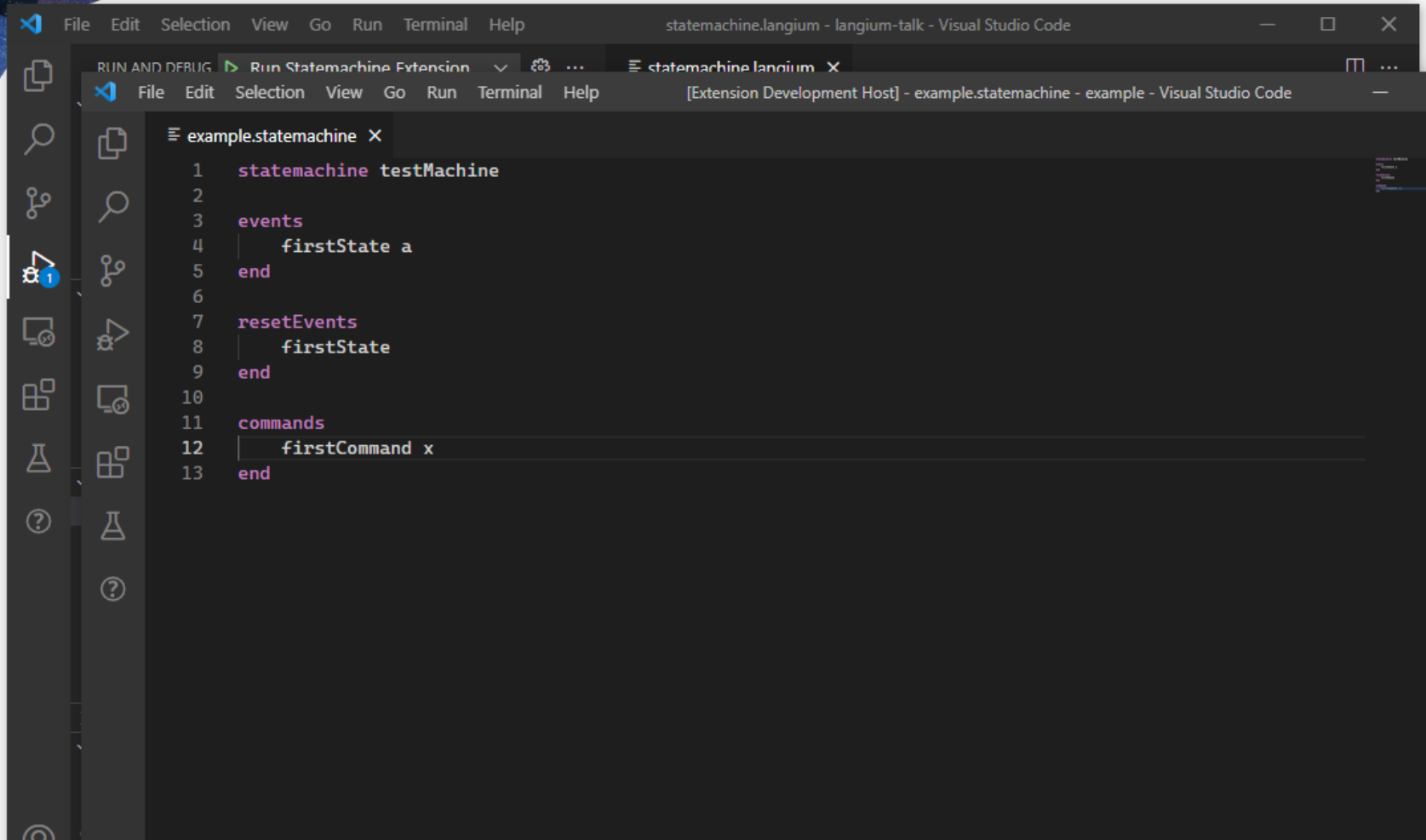

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
statemachine.langium
1 grammar Statemachine hidden(WS)
2
3 Statemachine:
4 'statemachine' name=ID
5 ('events' events+=Event+ 'end')?
6 ('resetEvents' resetEvents+=[Event]+ 'end')?;
7
8 Event:
9 name=ID code=ID;
10
11 terminal WS: /\s+/;
12 terminal ID: /[_a-zA-Z][\w_]*/;
13
14

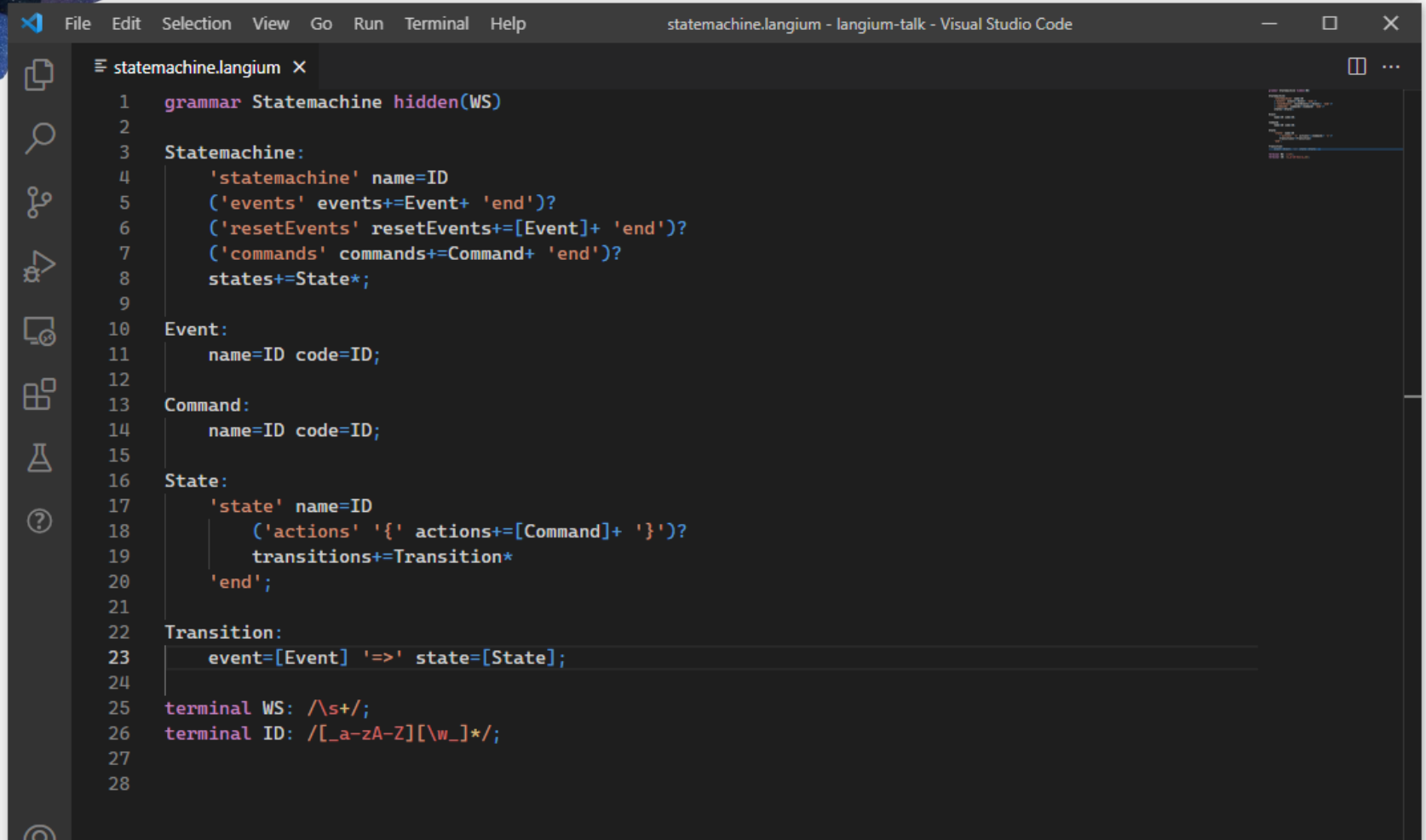

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - example.statemachine - example - Visual Studio Code
example.statemachine 2
1 statemachine testMachine
2
3 events
4
5 end
6
7 resetEvents
8
9 end

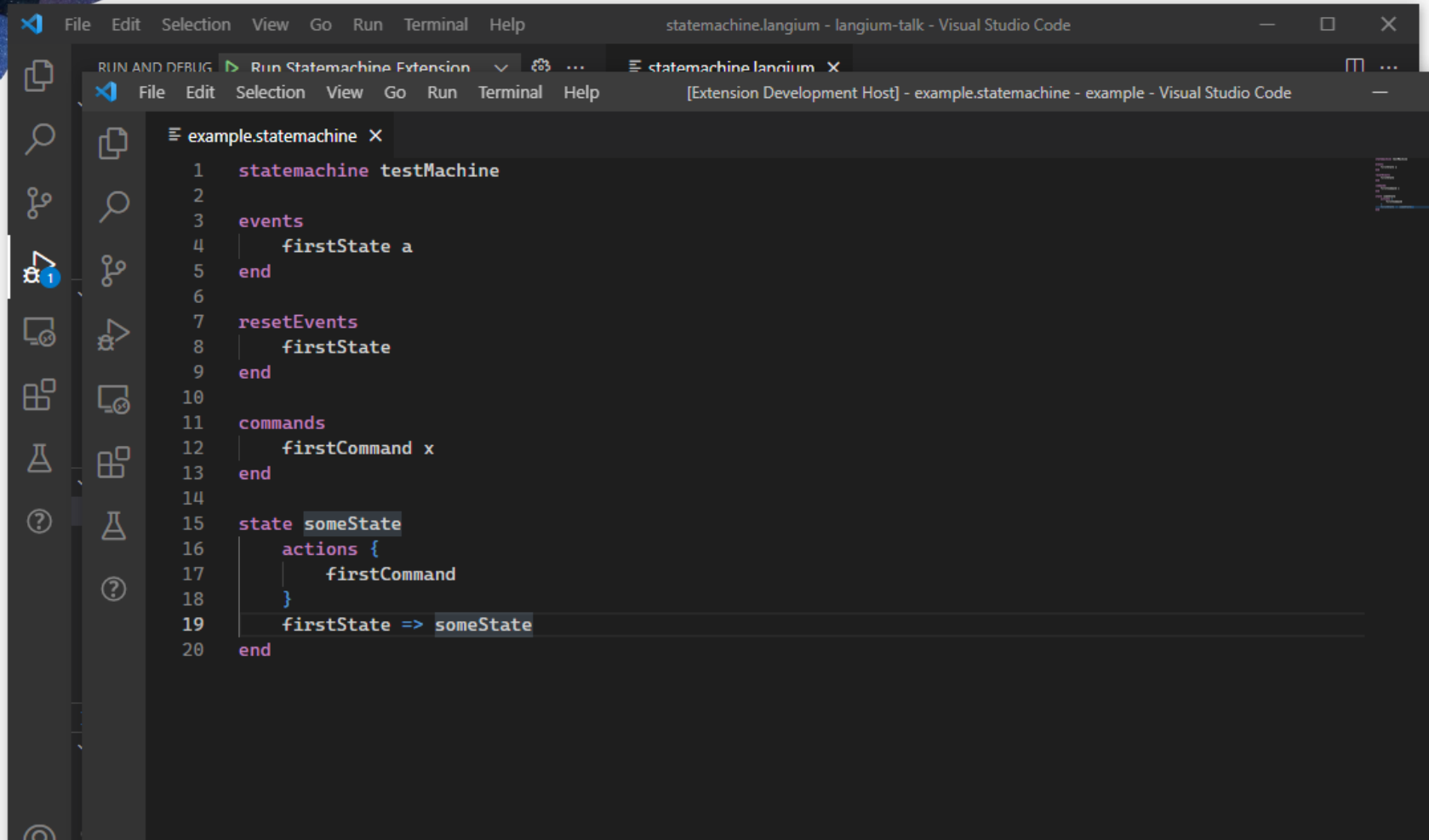

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - example.statemachine - example - Visual Studio Code
example.statemachine 2
1 statemachine testMachine
2
3 events
4 firstState a
5 end
6
7 resetEvents
8
9 end
firstState
Event

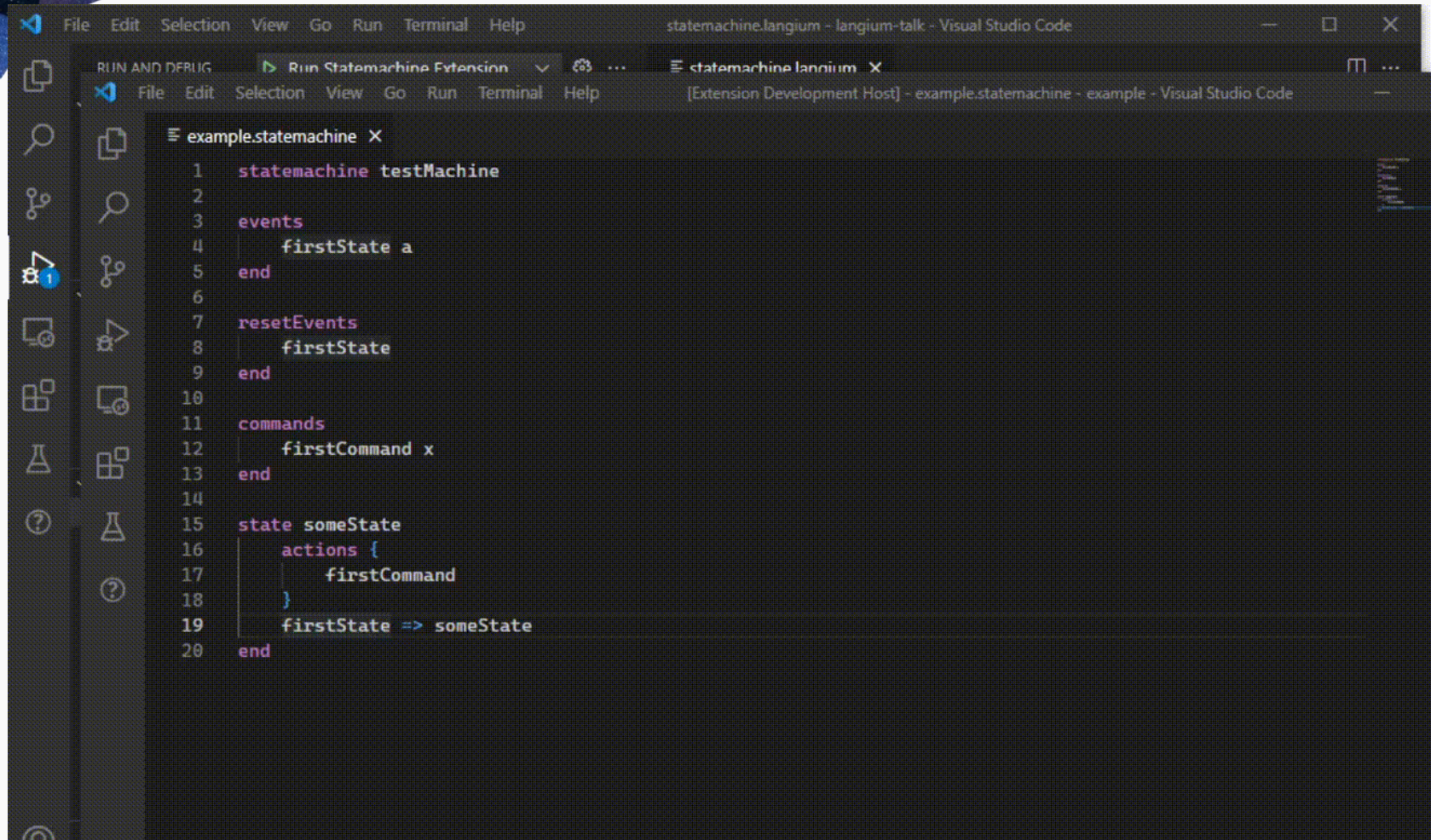

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - example.statemachine - example - Visual Studio Code
example.statemachine
1 statemachine testMachine
2
3 events
4 firstState a
5 end
6
7 resetEvents
8 firstState
9 end

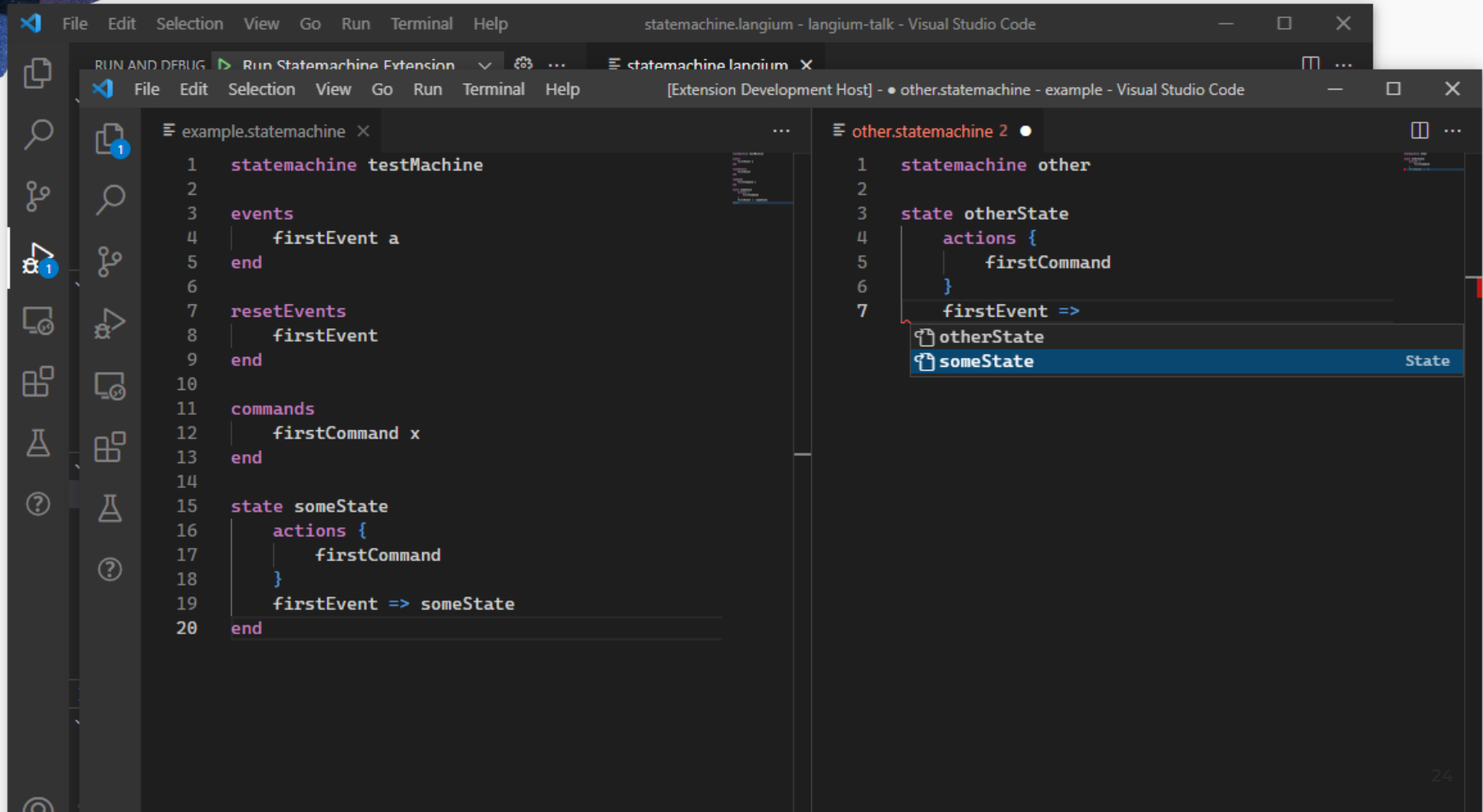

File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
statemachine.langium
1 grammar Statemachine hidden(WS)
2
3 Statemachine:
4 'statemachine' name=ID
5 ('events' events+=Event+ 'end')?
6 ('resetEvents' resetEvents+=[Event]+ 'end')?
7 ('commands' commands+=Command+ 'end')?;
8
9 Event:
10 name=ID code=ID;
11
12 Command:
13 name=ID code=ID;
14
15 terminal WS: /\s+/;
16 terminal ID: /[_a-zA-Z][\w_]*/;
17
18


File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - example.statemachine - example - Visual Studio Code
example.statemachine
1 statemachine testMachine
2
3 events
4 firstState a
5 end
6
7 resetEvents
8 firstState
9 end
10
11 commands
12 firstCommand x
13 end


File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
statemachine.langium
grammar Statemachine hidden(WS)
Statemachine:
'statemachine' name=ID
('events' events+=Event+ 'end')?
('resetEvents' resetEvents+=[Event]+ 'end')?
('commands' commands+=Command+ 'end')?
states+=State*;
Event:
name=ID code=ID;
Command:
name=ID code=ID;
State:
'state' name=ID
('actions' '{' actions+=[Command]+ '}')?
transitions+=Transition*
'end';
Transition:
event=[Event] '=>' state=[State];
terminal WS: /\s+/;
terminal ID: /[_a-zA-Z][\w_]*/;


File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - example.statemachine - example - Visual Studio Code
example.statemachine
statemachine testMachine
events
firstState a
end
resetEvents
firstState
end
commands
firstCommand x
end
state someState
actions {
firstCommand
}
firstState => someState
end


File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - example.statemachine - example - Visual Studio Code
example.statemachine
statemachine testMachine
events
firstState a
end
resetEvents
firstState
end
commands
firstCommand x
end
state someState
actions {
firstCommand
}
firstState => someState
end


File Edit Selection View Go Run Terminal Help
statemachine.langium - langium-talk - Visual Studio Code
RUN AND DEBUG
Run Statemachine Extension
statemachine.langium
File Edit Selection View Go Run Terminal Help
[Extension Development Host] - ● other.statemachine - example - Visual Studio Code
example.statemachine
1 statemachine testMachine
2
3 events
4 firstEvent a
5 end
6
7 resetEvents
8 firstEvent
9 end
10
11 commands
12 firstCommand x
13 end
14
15 state someState
16 actions {
17 firstCommand
18 }
19 firstEvent => someState
20 end
other.statemachine 2 ●
1 statemachine other
2
3 state otherState
4 actions {
5 firstCommand
6 }
7 firstEvent =>
otherState
someState State

Langium's Stack









Deriving Types

```
State:
    'state' name=ID
        ('actions' '{' actions+=[Command]+ '}')?
        transitions+=Transition*
    'end';
```

```
export interface State extends AstNode {
    readonly $container: Statemachine;
    actions: Array<Reference<Command>>
    name: string
    transitions: Array<Transition>
}
```

Chevrotain

- LL(k) Parser
- Internal parser DSL
- Embedded Actions
- *Really fast*

Library	Ops/sec	Relative Speed	?
Chevrotain	**2457.66** ±2.55%	100%	☑
Chevrotain-PT	**1539.22** ±3.43%	62.63%	☑
HandBuilt	**1226.53** ±1.86%	49.91%	☑
Parsimmon	**689.98** ±1.57%	28.07%	☑
ANTLR4	**612.13** ±1.85%	24.91%	☑
ANTLR4-PT	**518.84** ±2.02%	21.11%	☑
Myna	**512.97** ±0.23%	20.87%	☑
PEG.js	**493.34** ±1.53%	20.07%	☑
Nearley	**260.51** ±3.81%	10.60%	☑

Source: chevrotain.io

Dependency Injection

- Common for extensible frameworks
  - Overriding default behavior of services
  - Template Methods
  - Adding custom services

Dependency Injection

```ts
export type LangiumDefaultServices = {
    parser: {
        GrammarConfig: GrammarConfig
        ValueConverter: ValueConverter
        LangiumParser: LangiumParser
        TokenBuilder: TokenBuilder
    }
    documents: {
        DocumentBuilder: DocumentBuilder
        LangiumDocuments: LangiumDocuments
        LangiumDocumentFactory: LangiumDocumentFactory
        TextDocuments: TextDocuments<TextDocument>
        TextDocumentFactory: TextDocumentFactory
    }
```

```ts
export class DefaultLangiumDocumentFactory
    implements LangiumDocumentFactory {

    protected readonly parser: LangiumParser;

    constructor(services: LangiumServices) {
        this.parser = services.parser.LangiumParser;
    }
```

Langium speaks LSP

```
export interface GoToResolver {
    goToDefinition(document: LangiumDocument, position: TextDocumentPositionParams): LocationLink[]
}

export interface FoldingRangeProvider {
    getFoldingRanges(document: LangiumDocument): FoldingRange[]
}

export interface RenameHandler {
    renameElement(document: LangiumDocument, params: RenameParams): WorkspaceEdit | undefined
    prepareRename(document: LangiumDocument, params: TextDocumentPositionParams): Range | undefined
}
```

Other Services

- Scoping
- Workspace Indexing
- Code Generation
- Testing Support

Thank You

Backlog

Dependency Injection

```
export function createDefaultModule(): Module<LangiumServices, LangiumDefaultServices> {
    return {
        parser: {
            GrammarConfig: () => ({
                multilineCommentRules: ['ML_COMMENT']
            }),
            LangiumParser: (injector) => createLangiumParser(injector),
            ValueConverter: () => new DefaultValueConverter(),
            TokenBuilder: () => new DefaultTokenBuilder()
        },
        documents: {
            LangiumDocuments: (injector) => new DefaultLangiumDocuments(injector),
            LangiumDocumentFactory: (injector) => new DefaultLangiumDocumentFactory(injector),
            DocumentBuilder: (injector) => new DefaultDocumentBuilder(injector),
            TextDocuments: () => new TextDocuments(TextDocument),
            TextDocumentFactory: (injector) => new DefaultTextDocumentFactory(injector),
        },
```