Use RxJS to build a VSCode Extension in 30 Lines!

Dean Radcliffe, @deaniusaur

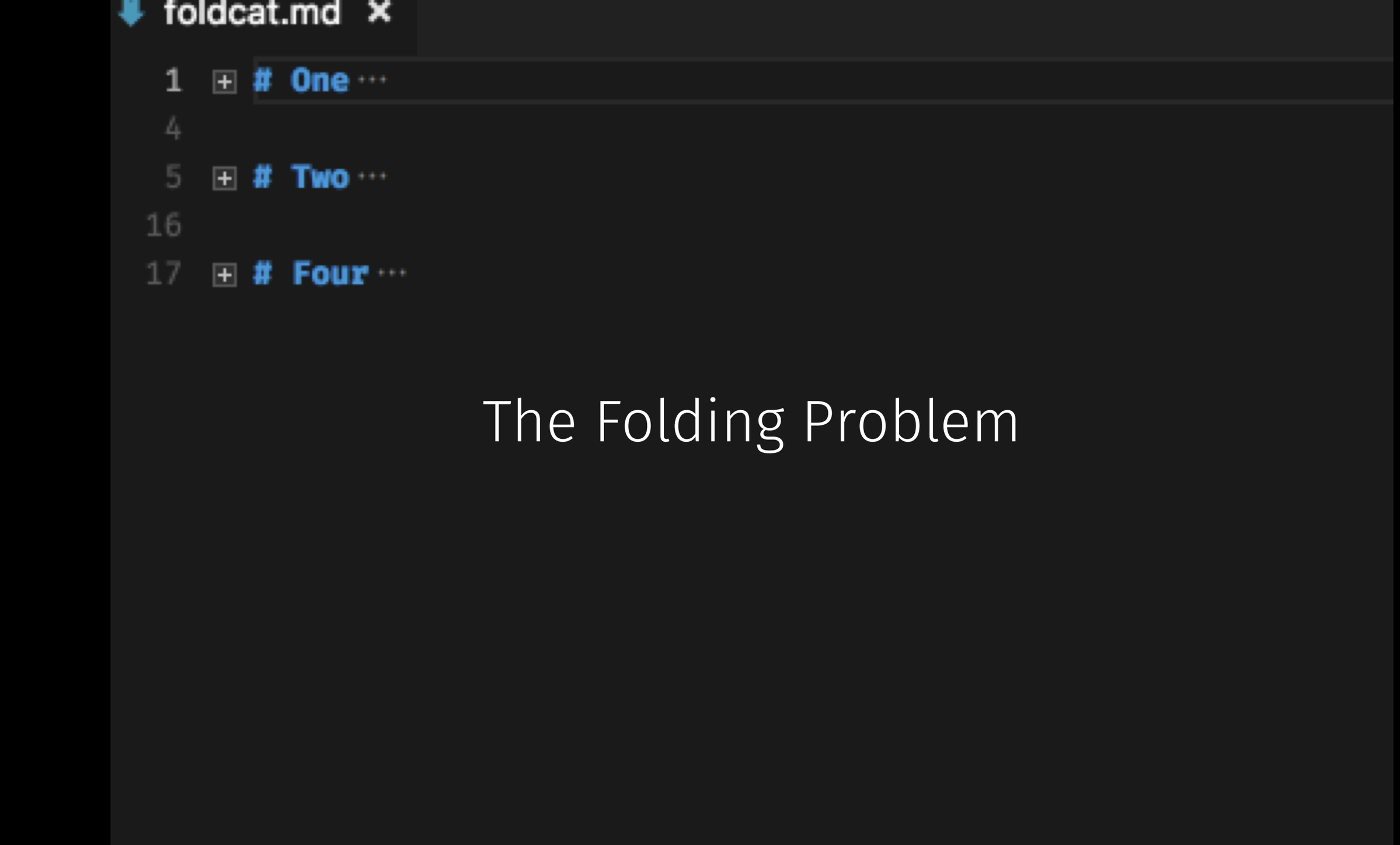
Deckset Presentation Software

→ Mac Based, Plain-Text (Markdown)

→ Bring Your Own Editor (VSCode)

→ Markdown conventions (--- for slide divides)

→ Used for this Presentation — *of course:*



The Folding Problem


I had a Dream

That code could be this expressive

```
const foldingRanges = numberedLines
  .scan(lineMarkingFunction)
  .filter(line => !line.drop)
  .thenDetermineRanges();

return foldingRanges;
```

That code could be this expressive

```
const foldingRanges = numberedLines
  .scan(lineMarkingFunction)
  .filter(line => !line.drop)
  .thenDetermineRanges();

return foldingRanges;
```

// yeah, sure, right:

RxJS - Marbles on Steroids

http://rxmarbles.com/#scan
http://rxmarbles.com/#distinctUntilChanged

RxJS Scan: a reducer across a stream

```
// The initial value, then each line becomes `prev`
numberedLines.scan((prev, incoming) => {
  return {
    ...prev
  };
});
```

RxJS Scan: a reducer across a stream

```
// The initial value, then each line becomes `prev`
numberedLines.scan((prev, incoming) => {
  return {
    ...prev
  };
}, {}); //initial value
```

You could call it a state machine.

— Some nerd, somewhere

Build It Up

```
numberedLines.scan((prev, current) => {
  // what's returned becomes `prev`
  return {
    ...prev
  };
}, {});
```

```
// numberedLines: [
//   {text: "# Header", line: 1},
//   {text: "<!--", line: 2},
//     ...
// ]
numberedLines.scan((prev, current) => {
  return {
    ...prev,
    startsComment: current.text.includes("<!--"),
    dropping: startsComment
  };
}, {});
```

```
numberedLines.scan((prev, current) => {
  return {
    ...prev,
    startsComment: current.text.includes("<!--"),
    endsComment: current.text.includes("-->"),
    dropping: startsComment || (prev.dropping && !endsComment)
  };
});
```

```
numberedLines.scan((prev, current) => {
  return {
    ...prev,
    startsComment: current.text.includes("<!--"),
    endsComment: current.text.includes("-->"),
    dropping: startsComment || (prev.dropping && !endsComment),
    drop: dropping || endsComment
  };
});
```

```
numberedLines.scan((prev, current) => {
  return {
    ...prev,
    startsComment: current.text.includes("<!--"),
    endsComment: current.text.includes("-->"),
    dropping: startsComment || (prev.dropping && !endsComment),
    drop: dropping || endsComment
  };
});
```

Do this first, actually..

dropping							
text	∅	`#foo`	`<!--`	`---`	`-->`	`---`	`#bar`
dropping	*false*	*false*	👍🏽	👍🏽	*false*	*false*	*false*
startsComment			👍🏽				
endsComment					👍🏽		
drop?			👍🏽	👍🏽	👍🏽		
folding range						`0..4`	`5..6`

Boom! 💡💡💡

```
notDroppedLines = numberedLines.scan((prev, current) => {
  return {
    ...prev,
    startsComment: current.text.includes("<!--"),
    endsComment: current.text.includes("-->"),
    dropping: startsComment || (prev.dropping && !endsComment),
    drop: dropping || endsComment
  };
}).filter(x => !x.drop);
```

Thank you Church Lady May I Have Another?

```
const slideDividers = notDroppedLines
  ...
```

```
const slideDividers = notDroppedLines
  .filter(line => line.startsSlide)
  .map(line => line.lineNum)
```

```
const slideDividers = notDroppedLines
  ...
  .startWith(0)
  .endWith(document.lineCount-1)
  .bufferCount(2, 1) // [0,4], [4,7], [7, 15]
```

```
const slideDividers = notDroppedLines
  ...
  .map(lastTwo => {
    const [start, end] = lastTwo
    return end && new FoldingRange(start, end)
  })
```

```
const slideDividers = notDroppedLines
  ...
  .toArray()
```

```
const slideDividers = notDroppedLines
  .filter(line => line.startsSlide)
  .map(line => line.lineNum)
  .startWith(0)
  .endWith(document.lineCount-1)
  .bufferCount(2, 1)
  .map(lastTwo => {
        const [start, end] = lastTwo
    return end && new FoldingRange(start, end)
  })
  .filter(x => !!x)
  .toArray()
```

How Is This Possible ?!

Let's Drive it Home 

```
provideFoldingRanges(): FoldingRange[] {
    let ranges : FoldingRange[] = []

    slideDividers = // obtain our built-up procedure

    // Similar to Promise.then(fn), get the FoldingRange[]
    slideDividers.subscribe(dividers => {
      ranges = dividers
    })

  // Unlike Promises, the callback has run synchronously
    return ranges
}
```

Synchronously or async?
Don't sweat it.

— Someone, once

Observables are Pure Awesome.

— My paraphrase of Ben Lesh

Visual Studio Code > Programming Languages > Deckset

Deckset

Deanius Solutions | 16 installs | ★★★★★ (0)

Make Deckset files in VSCode splendiferously.

Install Trouble Installing?

Now with *Snippets*!

Thank You!