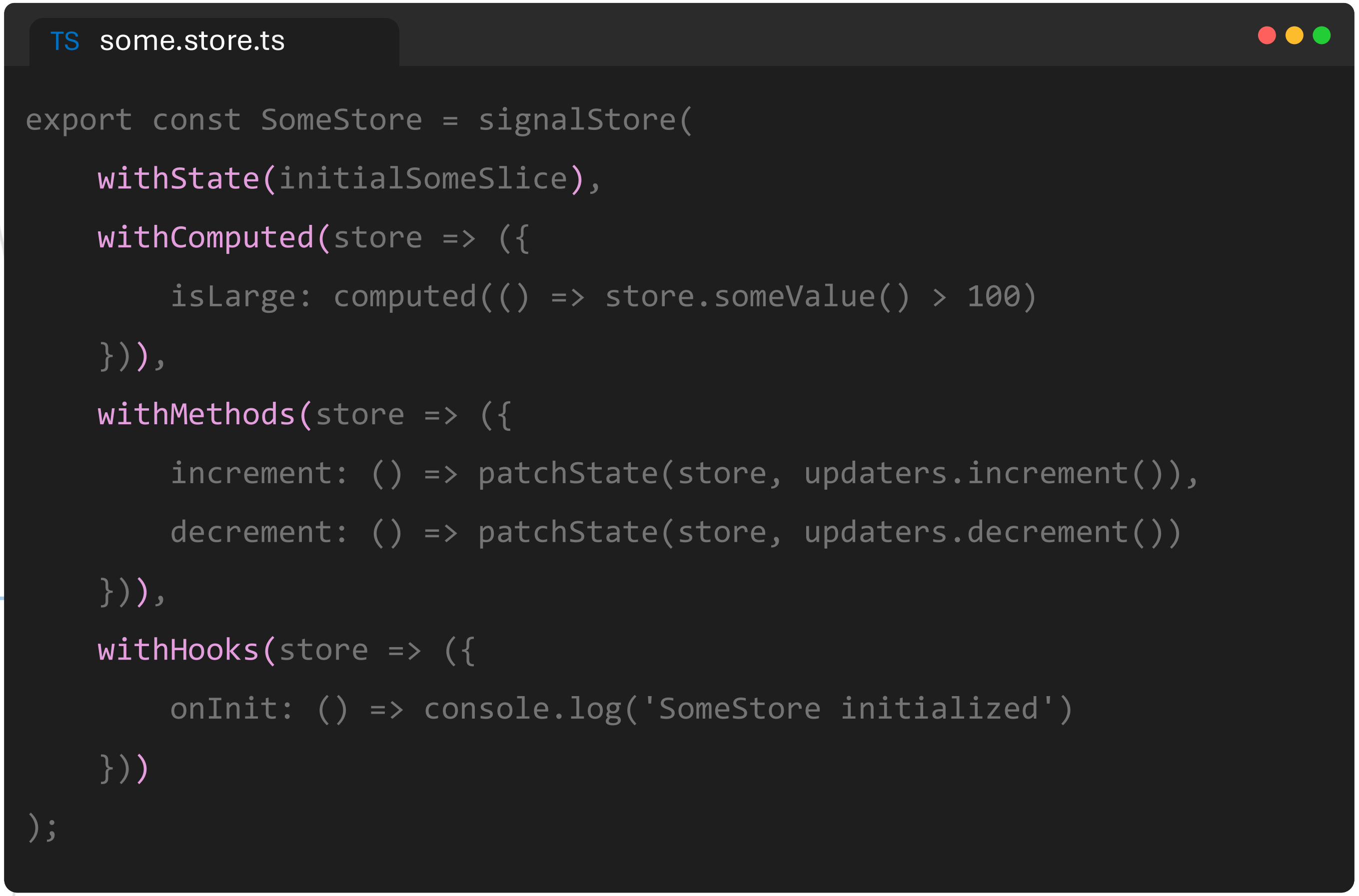
Function Categories

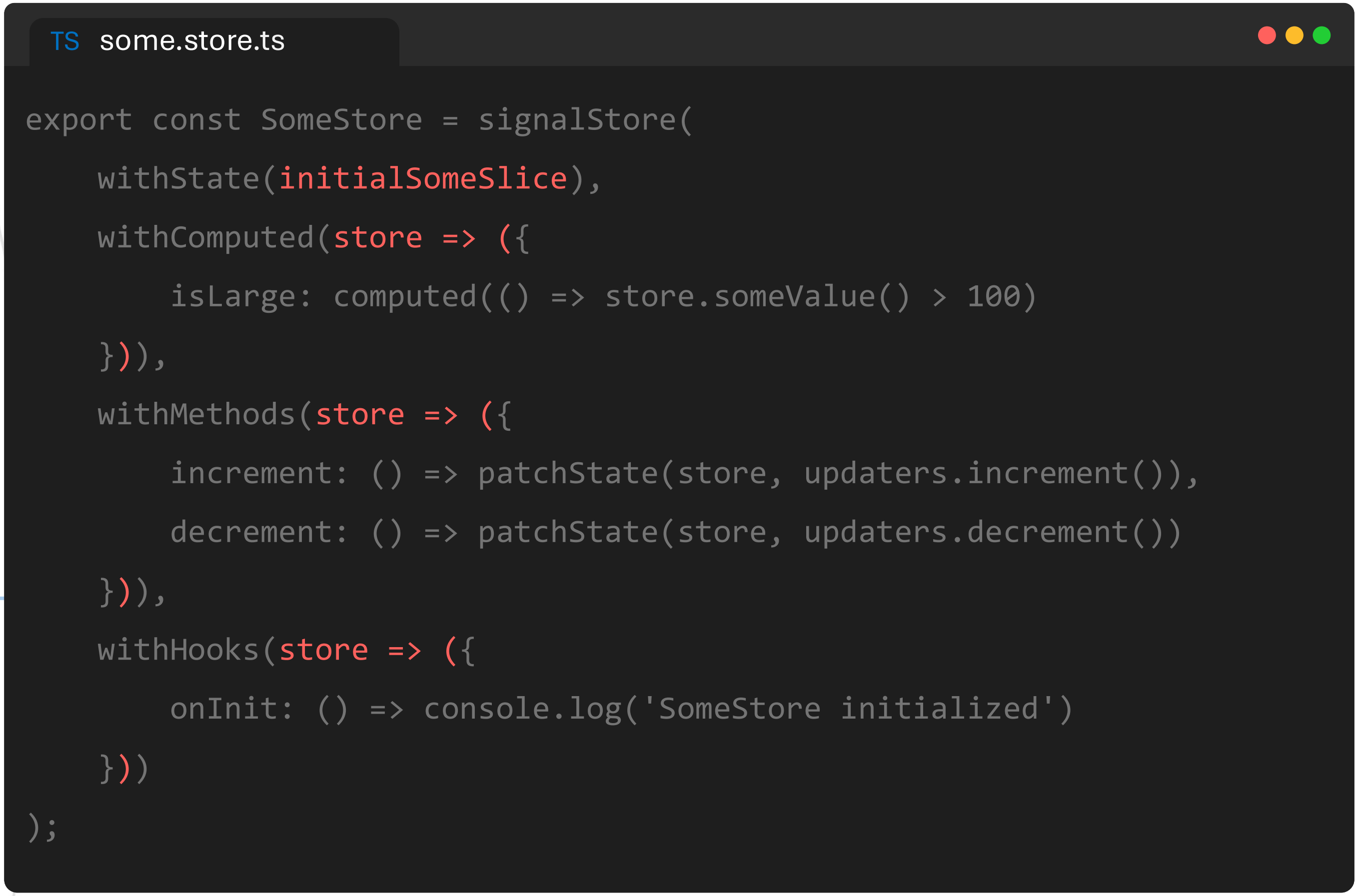
- signalStore
- Feature creators
- Feature parameters
- Computed expressions
- Method implementation
- Hooks



```typescript
export const SomeStore = signalStore(
    withState(initialSomeSlice),
    withComputed(store => ({
        isLarge: computed(() => store.someValue() > 100)
    })),
    withMethods(store => ({
        increment: () => patchState(store, updaters.increment()),
        decrement: () => patchState(store, updaters.decrement())
    })),
    withHooks(store => ({
        onInit: () => console.log('SomeStore initialized')
    }))
);
```

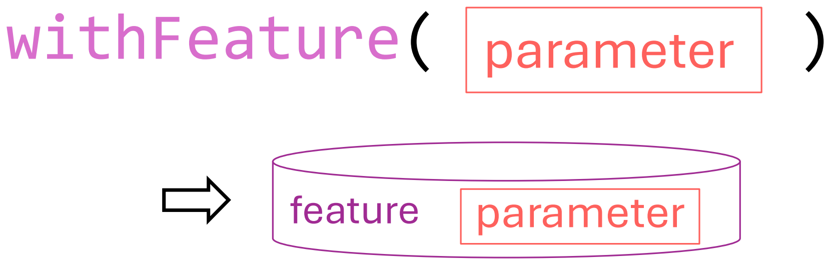
Function Categories

- signalStore
- Feature creators
- Feature parameters
- Computed expressions
- Method implementation
- Hooks



```ts
export const SomeStore = signalStore(
    withState(initialSomeSlice),
    withComputed(store => ({
        isLarge: computed(() => store.someValue() > 100)
    })),
    withMethods(store => ({
        increment: () => patchState(store, updaters.increment()),
        decrement: () => patchState(store, updaters.decrement())
    })),
    withHooks(store => ({
        onInit: () => console.log('SomeStore initialized')
    }))
);
```

Feature Creators

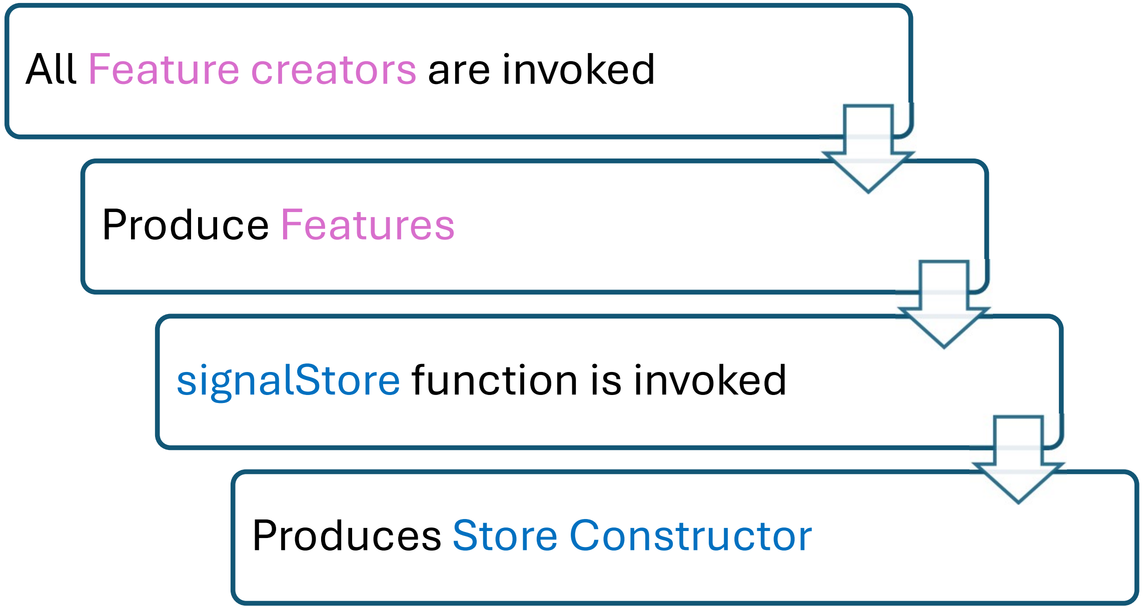


```
withFeature( parameter )
```

⇨ feature parameter

Type Creation Phase

- Happens Once
- When the JS is loaded
- Feature parameters are **stored**, but not yet **invoked**



All Feature creators are invoked

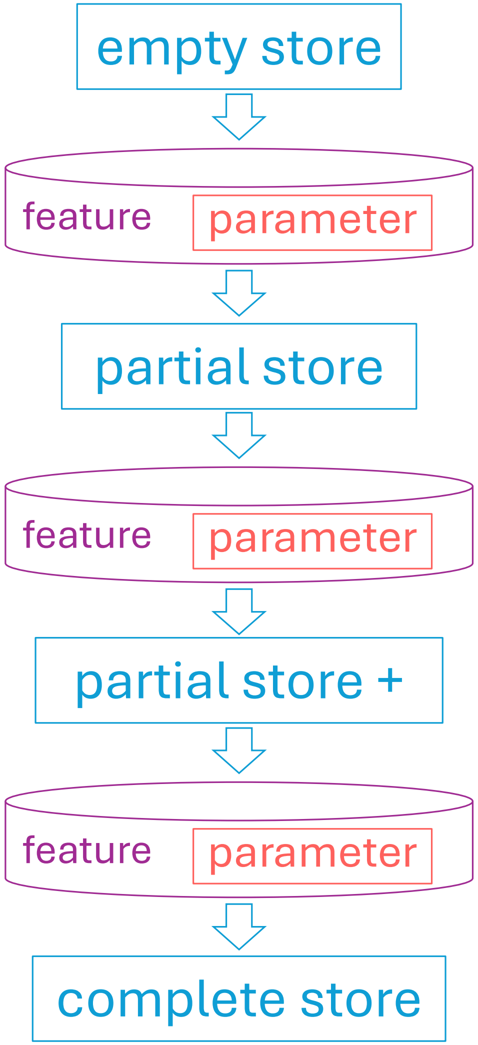
Produce Features

signalStore function is invoked

Produces Store Constructor

Store Construction Phase

- Happens Once per store instance
- Occurs in **Injection Context**


empty store

feature parameter

partial store

feature parameter

partial store +

feature parameter

complete store

Feature Parameter Functions

- Are called once per instance
- Run in injection context
- Return functions
  (computed expressions, methods, or hooks)
- Are, therefore, higher order functions
- Can Inject
- Can use closures